Vision Neural Architecture Designs for Improved Robustness and Accuracy

YANXI LI

Doctor of Philosophy

Supervisor: Associate Professor Chang Xu

A thesis submitted in fulfilment of
the requirements for the degree of
Doctor of Philosophy

School of Computer Science
Faculty of Engineering
The University of Sydney
Australia

28 February 2025

Statement of Originality

This is to certify that, to the best of my knowledge, the content of this thesis is my own work. This thesis has not been submitted for any degree or other purposes.

I certify that the intellectual content of this thesis is the product of my own work and that all the assistance received in preparing this thesis and sources have been acknowledged.

Student Name: Yanxi Li

Signature:

Authorship Attribution Statement

I designed the study and wrote the drafts of the papers that constitute parts of the thesis. Chapter 3 was published as "*Neural Architecture Dilation for Adversarial Robustness*" (Li et al., 2021b). Chapter 4 was published as "*Trade-Off Between Robustness and Accuracy of Vision Transformers*" (Li and Xu, 2023). Chapter 5 was published as "*Harnessing Edge Information for Improved Robustness in Vision Transformers*" (Li et al., 2024). Chapter 6 was published as "*Adapting Neural Architectures Between Domains*" (Li et al., 2020b). In addition to the statements above, in instances where I am not the corresponding author of a published work, the corresponding author has granted permission to include the material.

Student Name: Yanxi Li
Signature:

As the supervisor of the candidate for whom this thesis is submitted, I confirm that the authorship attribution statements above are accurate.

Supervisor Name: Chang Xu
Signature:

Copyright Page

List of Research Outcome

Research Outcomes Covered in This Thesis

(1) Yanxi Li, Zhaohui Yang, Yunhe Wang, and Chang Xu. Adapting neural architectures between domains. *Advances in neural information processing systems*, 33:789–798, 2020b.

(2) Yanxi Li, Zhaohui Yang, Yunhe Wang, and Chang Xu. Neural architecture dilation for adversarial robustness. *Advances in Neural Information Processing Systems*, 34: 29578–29589, 2021b.

(3) Yanxi Li and Chang Xu. Trade-off between robustness and accuracy of vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7558–7568, 2023.

(4) Yanxi Li, Chengbin Du, and Chang Xu. Harnessing edge information for improved robustness in vision transformers. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(4):3252–3260, 2024.

Other Peer-Reviewed Research Outcomes

(1) Yanxi Li, Minjing Dong, Yunhe Wang, and Chang Xu. Neural architecture search in a proxy validation loss landscape. In *International Conference on Machine Learning*, pages 5853–5862. PMLR, 2020a.

(2) Yanxi Li, Zean Wen, Yunhe Wang, and Chang Xu. One-shot graph neural architecture search with dynamic search space. *Proceedings of the AAAI conference on artificial intelligence*, 35(10):8510–8517, 2021a.

(3) Xiu Su, Tao Huang, Yanxi Li, Shan You, Fei Wang, Chen Qian, Changshui Zhang, and Chang Xu. Prioritized architecture sampling with monto-carlo tree search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10968–10977, 2021.

(4) Yanxi Li, Xinghao Chen, Minjing Dong, Yehui Tang, Yunhe Wang, and Chang Xu. Spatial-channel token distillation for vision mlps. In *International Conference on Machine Learning*, pages 12685–12695. PMLR, 2022a.

(5) Yanxi Li, Minjing Dong, Yunhe Wang, and Chang Xu. Neural architecture search via proxy validation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(6):7595–7610, 2022b.

(6) Yanxi Li, Minjing Dong, Yixing Xu, Yunhe Wang, and Chang Xu. Neural architecture tuning with policy adaptation. *Neurocomputing*, 485:196–204, 2022c.

(7) Yehui Tang, Kai Han, Jianyuan Guo, Chang Xu, Yanxi Li, Chao Xu, and Yunhe Wang. An image patch is a wave: Phase-aware vision mlp. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10935–10944, 2022.

(8) Zhi Cheng, Yanxi Li, Minjing Dong, Xiu Su, Shan You, and Chang Xu. Neural architecture search for wide spectrum adversarial robustness. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(1):442–451, 2023.

(9) Chengbin Du, Yanxi Li, Zhongwei Qiu, and Chang Xu. Stable diffusion is unstable. *Advances in Neural Information Processing Systems*, 36, 2024.

(10) Xiaohuan Pei, Yanxi Li, Minjing Dong, and Chang Xu. Neural architecture retrieval. In *The Twelfth International Conference on Learning Representations*, 2024.

(11) Yanxi Li and Chengbin Du. Optimizing quantized diffusion models via distillation with cross-timestep error correction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 18530–18538, 2025.

(12) Minjing Dong, Yanxi Li, Yunhe Wang, and Chang Xu. Adversarially robust neural architectures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

(13) Yunke Wang, Yanxi Li, and Chang Xu. Position: Ai scaling: From up to down and out. In *International Conference on Machine Learning Position Track*, 2025.

Abstract

Deep neural networks (DNNs), including Convolutional Neural Networks (CNNs) and Vision Transformers (ViTs), have achieved remarkable performance in computer vision tasks. However, their vulnerability to shifts in inputs remains a critical challenge. These shifts can arise from intentional adversarial attacks or from natural distribution shifts. As a result, the vulnerability compromise the trustworthiness and reliability of DNNs and hinder their deployment in real-world scenarios. Existing methods attempt to mitigate these vulnerabilities through adversarial training, which improves resistance to adversarial attacks by optimizing DNNs with perturbed training examples, and distributionally robust optimization (DRO), which seeks to enhance generalization across diverse data distributions. However, these approaches often suffer from a substantial trade-off between robustness and accuracy: adversarial training tends to reduce standard accuracy, while methods targeting distributional robustness struggle to balance in-distribution (ID) accuracy with out-of-distribution (OOD) robustness. This trade-off remains a fundamental challenge in deep learning, which necessitates new approaches that can improve robustness while preserving accuracy.

Firstly, we propose Neural Architecture Dilation (NAD), a framework that improves robustness by integrating searched dilation architectures into pre-trained neural backbones. Building on this, we introduce Neural Architecture Dilation for Adversarial Robustness (NADAR), which formulates the dilation process as a constrained optimization problem. By leveraging theoretical standard and adversarial error bounds, NADAR ensures that robustness improvements do not degrade clean accuracy or introduce excessive computational overhead. Secondly, we introduce TORA-ViTs, a transformer-based architecture that explicitly disentangles robust and predictive features through lightweight adapters. These robust and predictive adapters extract complementary information, while an attention-based gated fusion mechanism dynamically balances their contributions based on input conditions. To optimize this fusion, we implement a two-phase training strategy, ensuring the model effectively

integrates robust and predictive features. Thirdly, we propose EdgeNet, a plug-and-play module designed to enhance robustness by incorporating shape-based edge features into vision transformers. Utilizing a "sandwich" architecture with zero convolutions, EdgeNet introduces robust structural representations while preserving the predictive power of pre-trained backbones. This structured integration ensures improved robustness and accuracy with minimal parameter overhead. Lastly, we introduce AdaptNAS, a Neural Architecture Search (NAS) framework that explicitly optimizes architectures for both ID accuracy and OOD robustness. AdaptNAS leverages domain adaptation principles and adversarial learning to reduce OOD generalization gap, ensuring architectures that generalize more effectively while maintaining computational efficiency.

Acknowledgements

I would like to express my deepest gratitude to everyone who contributed to the successful completion of my PhD research and this thesis.

First and foremost, I am sincerely grateful to my supervisor, Associate Professor Chang Xu, for his invaluable guidance, insightful feedback, and continuous encouragement throughout my research. I deeply admire his academic expertise, creativity in research, and profound insights into future directions. His support extended far beyond professional supervision. Whenever I felt upset by failures or lost about the future, his gentle advice helped me overcome the challenges. It has been an honor to learn from him.

Special thanks go to my colleague and friend, Assistant Professor Minjing Dong. Having joined our lab a year before me, he played a crucial role as a senior colleague in guiding me through the early stages of my research. I still remember how he patiently helped me revise every equation and sentence in my first manuscript. I also enjoyed the time spent with his pet cat, Ram, whose companionship provided great comfort during the isolation of the COVID-19 period when we had to work from home.

I am genuinely grateful to my research collaborators. I would like to thank Dr. Yunhe Wang and Dr. Xinghao Chen for their professional advice and the valuable resources they generously provided. I appreciate Mr. Chengbin Du, Mr. Xiaohuan Pei, Dr. Xiu Su, Dr. Yunke Wang, and Dr. Zhaohui Yang for their insightful discussions, collaboration, and support. I am also thankful to all my peers and friends in our lab for their kindness and encouragement.

I deeply appreciate the financial support from the Faculty of Engineering Research Scholarship from the University of Sydney, which made my research possible.

Finally, I would like to express my heartfelt gratitude to my family and friends for their endless love and unwavering support.

To everyone who has been part of this journey, I sincerely thank you.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

In the past few decades, Deep Neural Networks (DNNs), including Convolutional Neural Networks (CNNs; LeCun et al., 1989, 1995; Krizhevsky et al., 2012; He et al., 2016; Zagoruyko and Komodakis, 2016) and Vision Transformers (ViTs; Parmar et al., 2018; Hu et al., 2019; Zhao et al., 2020; Dosovitskiy et al., 2020; Touvron et al., 2021; Naseer et al., 2021; He et al., 2022), have been well developed to achieve or even surpass the performance of humans on computer vision tasks, which encourages their applications in real-world scenarios. Real-world applications demand DNNs that can maintain reliable performance despite diverse and often unpredictable input variations. However, a fatal drawback of DNNs is that they are vulnerable to shifts in input data (Goodfellow et al., 2014; Hendrycks and Gimpel, 2017), which will cause a dramatic drop in their accuracy. These shifts may be intentionally introduced by malevolent third parties through **adversarial attacks** (Goodfellow et al., 2014; Madry et al., 2017) or may naturally arise due to **distribution shifts** (Hendrycks and Dietterich, 2019; Hendrycks et al., 2021a,b), where test data deviate from the training distribution. This vulnerability notably reduces the reliability and usability of DNNs in practical applications. Consequently, developing methods to increase their robustness against shifts in input data has attracted particular attention from researchers.

In defending against adversarial attacks, adversarial training is one of the most straightforward strategies, which augments the training data with adversarial examples. These adversarial examples are often generated using attack methods such as the Fast Gradient Sign Method (FGSM; Goodfellow et al., 2014) and Projected Gradient Descent (PGD; Madry et al., 2017). Beyond them, Tramèr et al. (2017) investigate the adversarial examples produced by multiple pre-trained models and introduce ensemble adversarial training. Meanwhile, Wong and Kolter

(2018) propose a provably robust model by focusing on the worst-case loss over a convex outer region. Curriculum Adversarial Training (CAT; Cai et al., 2018b) improves adversarial robustness by gradually increasing attack strength during training, mitigating overfitting, and enhancing generalization compared to traditional adversarial training.

Beyond adversarial robustness, it is also crucial to consider robustness against naturally occurring distribution shifts, which is also known as out-of-distribution (OOD) robustness. These shifts arise due to variations in the environment, domain, or task. Several approaches have been proposed to improve OOD robustness, including unsupervised representation learning, supervised model learning, and optimization-based methods. Both unsupervised representation learning techniques and supervised model learning methods aim to improve OOD generalization by learning representations that remain stable across varying data distributions. Unsupervised representation learning techniques (He et al., 2022; Higgins et al., 2017) focus on discovering domain-invariant or disentangled features without requiring explicit labels or environment information. In contrast, supervised model learning methods (Arjovsky et al., 2019; Yang et al., 2021a) incorporate additional information, such as causal relationships or environment labels, to enforce invariance explicitly. Additionally, optimization strategies such as distributionally robust optimization (DRO; Duchi and Namkoong, 2021) aim to minimize the worst-case risk under distributional shifts. Adversarial training (Sinha et al., 2017; Liu et al., 2022a) is also a widely used method in DRO for identifying and mitigating worst-case risk to improve OOD robustness.

However, there is often a trade-off between robustness and accuracy. In the context of adversarial robustness, Tsipras et al. (2018) demonstrate that there exists a trade-off between accuracy and robustness in adversarial training. When DNNs are trained to defend against adversarial attacks, their performance on natural image classification can be negatively influenced. The authors attribute this trade-off to the fact that robust classifiers learn fundamentally different features compared to standard classifiers. Standard training makes DNNs leverage all available features, including weakly correlated ones that contribute to prediction accuracy but are vulnerable to adversarial manipulation. Attackers can exploit these weak features, leading to misclassification. In contrast, adversarial training forces the model to rely only on

strongly correlated features that are more robust to perturbations. This constraint limits the features the model can use, ultimately reducing standard accuracy.

A similar trade-off exists in OOD robustness. While early studies often assumed that in-distribution (ID) and OOD performance improve together, Wenzel et al. (2022) highlight the limitations of prior evaluations and suggest that the correlation between ID accuracy and OOD robustness can be positive, negative, or negligible, depending on the dataset. Teney et al. (2023) further provide both empirical evidence and theoretical analysis and demonstrate that in real-world datasets, this correlation can be negative. They also identify spurious correlations, i.e., features that improve ID accuracy but degrade OOD robustness, as a key factor contributing to this negative correlation, which aligns with findings on the trade-off between adversarial robustness and standard accuracy by Tsipras et al. (2018).

Following them, several studies have explored this trade-off. Zhang et al. (2019) provide a theoretical framework by decomposing the robust error and establishing an upper bound. They propose the TRADES defense algorithm that explicitly balances accuracy and robustness. Wang et al. (2020) propose Once-for-All Adversarial Training (OAT), which allows in-situ calibration of a trained model to navigate this trade-off efficiently without retraining. Their method leverages dual batch normalization to separate adversarial and standard feature statistics. Rade and Moosavi-Dezfooli (2021b) analyze decision boundary shifts in adversarial training. Based on the analysis, a Helper-based Adversarial Training (HAT) method is proposed, which mitigates excessive margin expansion. Föll et al. (2022) introduces Gated Domain Units (GDUs) to balance the trade-off between ID and OOD performance using invariant elementary distributions and a weighted ensemble approach. GDUs adaptively assign weights to learned elementary distributions based on test-time similarity.

Although these methods have made notable progress in balancing robustness and accuracy, they do not fully resolve the robustness-accuracy dilemma. By focusing primarily on the training process and representation learning, these approaches often overlook the role of network architecture in shaping the trade-off. When the network architecture is fixed, the expressive capacity of the model is inherently constrained. Firstly, a fixed architecture limits the model's ability to capture complex, high-level representations, which are essential for

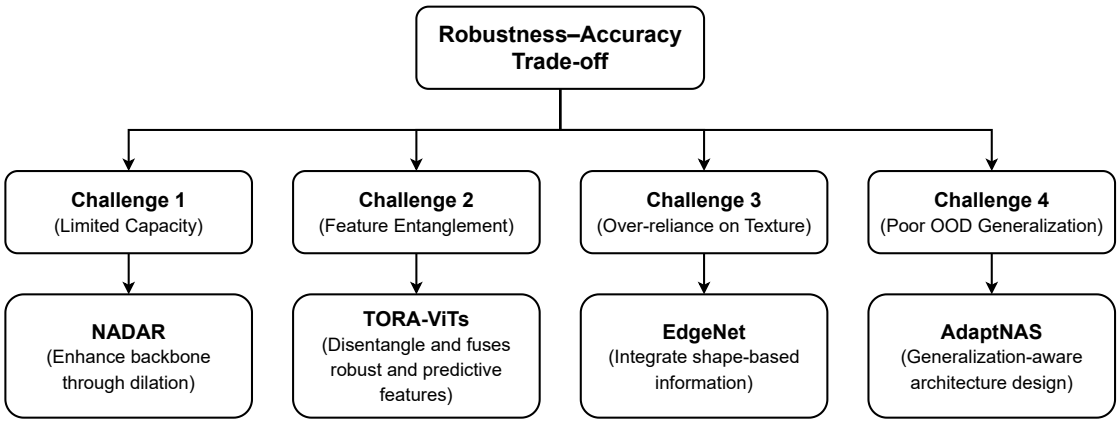


FIGURE 1.1. The overall structure of this thesis.

distinguishing relevant and domain-invariant features from irrelevant and domain-specific features. Additionally, a fixed architecture constrains the model's VC dimension, thereby restricting the range of functions it can learn. This limitation may prevent the model from forming robust decision boundaries that generalize well under input shifts. Consequently, focusing solely on training hinders the development of models that achieve both robustness and accuracy.

In contrast, architectural innovations can provide additional degrees of freedom, which enables the model to better separate informative features from perturbations and domain-specific features and to utilize them for improved robustness and accuracy. This flexibility allows the model to learn more nuanced representations and decision boundaries, ultimately addressing the robustness-accuracy dilemma. This thesis explores this novel paradigm that enhances DNNs from an architectural perspective. We introduce four novel architectural approaches, namely NADAR, TORA-ViTs, EdgeNet, and AdaptNAS, that enhance both robustness and accuracy from an architectural perspective by leveraging insights from theoretical analysis and empirical evaluations. The following chapters will detail our proposed methodologies and their contributions to robust and accurate deep neural network architectures. The overall structure of this thesis is shown in Fig. 1.1.

1.1 Research Questions

RQ1: How does the capacity of neural architectures influence the trade-off between robustness and accuracy? Traditional training and optimization-based approaches primarily focus on balancing robustness and accuracy but fail to fully resolve the robustness-accuracy dilemma. When the architecture of a DNN is fixed, its expressive capacity is inherently constrained, which limits its ability to capture features that support both robustness and accuracy. Understanding the relationship between neural network capacity and this trade-off is crucial for designing architectures that can achieve both high robustness and strong predictivity.

RQ2: How can DNNs be designed to distinguish, disentangle, and adaptively fuse robust and predictive features based on input characteristics? Deep neural networks often struggle to balance robust and predictive features effectively. When the network architecture is fixed, its ability to capture complex, high-level representations is limited, which restricts its ability to distinguish and leverage robust features when needed. It is important to explore architectural modifications that enable networks to disentangle and adaptively fuse robust and predictive features based on input characteristics.

RQ3: How can add-on architectural modules be designed to extract and integrate input characteristics that are both task-relevant and resilient to perturbations? Most DNNs for image classification rely solely on RGB image data, making them sensitive to perturbations, particularly those that target texture-based features. Integrating additional feature modalities that are resilient to perturbations, such as shape-based edge information, has the potential to improve robustness. Therefore, it is important to explore how lightweight architectural modules can be designed to extract and incorporate such features while maintaining computational efficiency and preserving accuracy.

RQ4: How can neural architectures be adapted to improve their OOD robustness and cross-domain generalization? Architectures designed and evaluated within a single domain often lack OOD robustness when applied to new, unseen distributions. Without explicit architectural adaptations, these models struggle with OOD generalization. This

question investigates how neural architectures can be designed to enhance OOD robustness and cross-domain performance while ensuring competitive ID performance.

1.2 Thesis Contributions

In Chapters 3, 4, 5, and 6, we introduce four different methods for enhancing robustness and accuracy from an architectural perspective.

1.2.1 Contribution of NADAR

Designing new robust architectures from scratch is costly, and their accuracy is not guaranteed. To address this challenge, we propose a novel paradigm called **Neural Architecture Dilation** (NAD). Unlike traditional Neural Architecture Search (NAS) methods that search for architectures from scratch, NAD leverages existing neural network backbones. It dilates the backbone network by integrating searched dilation architectures into fixed, pre-trained backbone models, resulting in dilated networks.

Building on our NAD paradigm, we introduce **Neural Architecture Dilation for Adversarial Robustness** (NADAR), which employs this dilation approach to significantly increase adversarial robustness while guaranteeing competitive accuracy on clean data. To search the dilation architectures efficiently and effectively, we analyze the theoretical bounds of standard and adversarial errors in the dilated networks. We propose two key error bounds: a **standard error bound**, which ensures that the dilated architectures do not degrade accuracy on clean samples, and an **adversarial error bound**, which demonstrates that the dilation architectures can correct misclassifications induced by adversarial perturbations. These bounds enable the design of dilation networks that enhance robustness without compromising accuracy.

Based on these theoretical insights, we formulate the neural architecture dilation problem as a constrained optimization problem, which ensures robustness improvements do not come at the expense of standard accuracy or excessive computational overhead. The primary objective is to minimize adversarial loss by optimizing the dilation architecture to defend

against adversarial attacks. To guarantee accuracy, we introduce a **standard performance constraint**, inspired by our theoretical analysis, which ensures that the hybrid network does not significantly degrade the accuracy of the original backbone on clean data. Additionally, to ensure computational efficiency, we incorporate an **FLOPs constraint**, which controls network complexity and prevents excessive computational overhead.

1.2.2 Contribution of TORA-ViTs

Inspired by the theorem that there exist robust, non-predictive features and predictive, non-robust features, we propose a novel framework that explicitly disentangles and fuses these features at the architectural level. To achieve this, we leverage the inherent strengths of ViTs, which utilize attention layers to model contextual relationships and dependencies and MLP layers to capture patterns and store knowledge. We enhance standard ViTs with lightweight adapters, which are small MLP layers inserted into each Transformer block. Specifically, we design two types of adapters: **robust adapters**, which capture adversarially robust features, and **predictive adapters**, which extract features crucial for achieving high accuracy on clean data. These adapters introduce only a minimal increase in trainable parameters while significantly improving both robustness and accuracy.

Another key contribution of our approach is the **attention-based gated fusion** mechanism, which dynamically balances the contribution of predictive and robust features. This module leverages cross-attention mechanisms to compare features extracted from ViT blocks with features from both adapters. It then applies a modified softmax function to control their weighted contributions at each spatial location. By doing so, TORA-ViTs enable adaptive feature selection, allowing the model to prioritize accuracy or robustness based on task requirements. If the features extracted from ViT blocks are likely to originate from clean data, the model can assign greater weight to predictive features; conversely, when adversarial perturbations are detected, it can emphasize robust features.

To optimize this process, we introduce a **two-phase training strategy**. In the first phase, the predictive and robust adapters are independently trained alongside the fusion module to ensure

each learns its respective feature representation effectively. In the second phase, the adapters are frozen while the fusion module is fine-tuned to control the fusion of robustness and accuracy. This optimization strategy ensures the model maintains competitive performance under both clean and adversarial conditions.

1.2.3 Contribution of EdgeNet

Conventional DNNs often rely heavily on texture and background information for accurate classification. However, this information is highly sensitive to perturbations, making DNNs vulnerable. In contrast, shape-related information provides a stronger semantic foundation for object recognition and is inherently more robust. However, conventional vision models often underutilize this information, as their architectural designs and optimization strategies tend to prioritize information that maximizes accuracy on clean datasets. To mitigate this gap, we introduce **EdgeNet**, a plug-and-play module that explicitly extracts and integrates shape-related edge information into ViTs. EdgeNet encourages models to rely on structural cues rather than solely on non-robust textures. Because EdgeNet learns edge-related features in a separate branch while the backbone continues to focus on predictive texture features, and since EdgeNet integrates its features into the backbone gradually, this approach ensures enhanced adversarial robustness while maintaining high accuracy.

We introduce a **"sandwich" architecture** into EdgeNet to seamlessly integrate edge-based features into ViTs. Each EdgeNet block consists of two zero convolutions, one at the input and one at the output, enclosing a standard ViT block. The input zero convolution acts as a filter, selectively allowing edge features that enhance robustness. In contrast, the output zero convolution ensures that the injected information starts from a zero point, preventing any disruptive changes to the pre-trained backbone. This structured integration enables EdgeNet to gradually introduce robust features without interfering with the backbone's learned representations, thereby avoiding overfitting or catastrophic forgetting. By leveraging this approach, EdgeNet effectively strengthens the model's ability to defend against adversarial attacks and other perturbations, all while maintaining high accuracy on clean images. Additionally,

its lightweight design ensures minimal computational overhead, making it a practical and scalable solution for improving robustness in vision models.

1.2.4 Contribution of AdaptNAS

Neural Architecture Search (NAS) methods have demonstrated impressive success in designing architectures that achieve high ID accuracy but often fail to ensure OOD robustness. Moreover, NAS typically searches for optimal architectures using small-scale ID datasets, such as CIFAR-10, and then applies them to more complex tasks, like ImageNet-1K. This transfer often results in unstable generalization and suboptimal performance. The root of the problem lies in the optimization objective of conventional NAS methods, which prioritize ID validation accuracy. While NAS allows for flexible architecture design, the resulting models remain vulnerable to distribution shifts. One possible workaround is to conduct NAS directly on large-scale datasets, which eliminates the need for generalization from small-scale searches. However, this approach comes with a significant drawback: searching on large-scale datasets increases computational costs by approximately 28$\sim$52 times, which makes it impractical for most scenarios.

To address these challenges, we propose AdaptNAS, an efficient, generalization-aware NAS framework that explicitly optimizes architectures for both ID and OOD performance. Our theoretical analysis establishes a formal **generalization bound** that links an architecture's empirical ID validation error in the source domain to its expected OOD error in the target domain. To achieve this, we leverage domain adaptation principles and use A-distance as a measure of domain discrepancy. In this way, we can still optimize the architecture by minimizing the ID validation error while simultaneously minimizing the generalization bound, which ensures strong performance both ID and OOD. Building on this analysis, AdaptNAS incorporates domain adaptation constraints into NAS to explicitly minimize the generalization gap. It employs **adversarial learning** with a domain discriminator to reduce distribution shifts and optimizes architectures based on a **hybrid validation** loss, which integrates a small subset of unlabeled target domain data. This principled approach enables

AdaptNAS to discover neural architectures that generalize more effectively across datasets while maintaining computational efficiency.

1.3 Thesis Outline

The following structured outline provides a comprehensive roadmap for our research on enhancing DNNs through novel architectural paradigms to achieve both robustness and accuracy.

In Chapter 1, we introduce the motivation behind improving DNNs from an architectural perspective by analyzing the need for robustness against perturbations while maintaining high accuracy. We discuss the limitations of existing approaches, set the stage for the novel paradigms introduced in this thesis, and outline our proposed solutions.

In Chapter 2, we review related work, including adversarial attacks and defense mechanisms, recent advances in CNNs and ViTs design, traditional Neural Architecture Search (NAS) methods, and perturbations beyond adversarial attacks.

In Chapter 3, we propose Neural Architecture Dilation (NAD), a novel framework for enhancing robustness by integrating dilation architectures into pre-trained backbones. We propose Neural Architecture Dilation for Adversarial Robustness (NADAR) and establish the theoretical foundations for its standard and adversarial error bounds. We also formulate the neural architecture dilation problem as a constrained optimization problem and discuss our FLOPs and standard performance constraints, ensuring efficient and effective robustness improvements.

In Chapter 4, we propose TORA-ViTs, a novel transformer-based architecture that explicitly disentangles and fuses robust and predictive features. We introduce robust and predictive adapters as lightweight architectural modifications to ViTs and develop an attention-based gated fusion mechanism for dynamically balancing these features. We also outline our two-phase training strategy, which ensures effective optimization of robustness and accuracy.

In Chapter 5, we propose EdgeNet, a plug-and-play module that enhances adversarial robustness by leveraging shape-based edge features. We propose the "sandwich" architecture, which uses zero convolutions to seamlessly integrate EdgeNet into ViTs without disrupting the pre-trained backbone. We analyze its effectiveness in strengthening robustness while maintaining accuracy and computational efficiency.

In Chapter 6, we propose AdaptNAS, a novel NAS framework designed to enhance both ID accuracy and OOD robustness. We establish a formal generalization bound, integrate domain adaptation constraints into NAS, and employ adversarial learning with a domain discriminator to mitigate distribution shifts. By incorporating a hybrid validation loss, AdaptNAS is capable of discovering architectures that achieve superior ID accuracy and OOD robustness.

In Chapter 7, we conclude the thesis by summarizing our key contributions and discussing future research directions.

CHAPTER 2

Literature Review

2.1 Adversarial Attacks

Adversarial examples are carefully crafted perturbations that cause neural networks to misclassify inputs. The Fast Gradient Sign Method (FGSM; Goodfellow et al., 2014) demonstrated that neural networks are vulnerable due to their linear nature rather than overfitting or nonlinearity. This method introduced a simple and efficient way to generate adversarial examples using a one-step gradient ascent. The Projected Gradient Descent (PGD; Madry et al., 2017) expanded upon FGSM by introducing a multi-step iterative attack method, establishing it as a universal first-order adversary. PGD-based adversarial training remains one of the strongest empirical defences against adversarial attacks. The Carlini and Wagner (C&W; Carlini and Wagner, 2017) attack formulates adversarial attacks as a constrained optimization problem. The Momentum Iterative FGSM (MI-FGSM; Dong et al., 2017, 2018) integrates momentum into iterative gradient-based attacks to stabilize update directions and enhance transferability. This method further improves attack success rates, particularly in black-box settings. The AutoAttack (Croce and Hein, 2020) introduces a robust, parameter-free ensemble of attacks designed to overcome common pitfalls in adversarial robustness evaluation.

2.2 Adversarial Training

A straightforward method to improve the robustness of DNNs is to leverage the aforementioned attack methods, usually FGSM or PGD, for adversarial training. Beyond simply using

one attack method for training, Tramèr et al. (2017) introduce Ensemble Adversarial Training to improve robustness against adversarial attacks by incorporating adversarial examples generated from multiple pre-trained models rather than just the model being trained. This approach addresses the issue of gradient masking. Wong and Kolter (2018) propose a provably robust defence against adversarial attacks by optimizing a convex outer approximation of the adversarial polytope. Their method minimizes the worst-case loss within this bound using a linear program (LP). This results in certified robustness. Curriculum Adversarial Training (CAT; Cai et al., 2018b) proposes to enhance adversarial robustness by gradually increasing the strength of adversarial attacks during training. Unlike traditional adversarial training, which directly uses strong attacks and risk overfitting, CAT starts with weak adversarial examples and progressively increases attack strength, helping the model generalize better.

2.3 Adversarial Purification

Denoiser-based methods for adversarial defence commonly utilize generative models to transform adversarial examples back into their clean versions prior to classification. This strategy is often referred to as adversarial purification (AP). MagNet (Meng and Chen, 2017) introduces a two-pronged defence using detector and reformer networks to separate and recover adversarial inputs by approximating the manifold of clean data. PixelDefend (Song et al., 2017) purifies adversarial examples through a generative model that enforces proximity to high-probability regions of the data distribution. Defense-GAN (Samangouei et al., 2018) similarly uses a generative model to project inputs onto the data manifold before classification. High-level representation guided denoiser (HGD) (Liao et al., 2018) addresses error amplification by aligning denoised and clean activations of a target model. More recently, score-based generative models (Yoon et al., 2021) and diffusion models (Nie et al., 2022; Chen et al., 2023; Lee and Kim, 2023) have advanced adversarial purification, offering strong robustness by reversing noisy diffusion processes or directly modelling class-conditional distributions. Diffusion-based defences, such as DiffPure (Nie et al., 2022) and the Robust Diffusion Classifier (Chen et al., 2023), demonstrate state-of-the-art robustness against adaptive attacks without requiring adversarial training. Complementing these, DDAD

(Zhang et al., 2025) integrates distributional discrepancy detection with denoising to retain clean data and enhance robustness, which underscores the efficacy of hybrid strategies that combine detection and purification mechanisms.

2.4 The Trade-Off in Adversarial Training

While adversarial training is an effective defence mechanism, it introduces a trade-off between adversarial robustness and standard accuracy. Tsipras et al. (2018) demonstrated that the optimal robust classifier learns different feature representations compared to a standard classifier, leading to an inevitable accuracy drop on clean inputs. Building on the trade-off identified by Tsipras et al. (2018), several studies have proposed methods to balance robustness and standard accuracy. TRADES (Zhang et al., 2019) provided a theoretical analysis of this trade-off by introducing a boundary error metric that quantifies the gap between standard and adversarial accuracy. To mitigate this, they proposed a tuning parameter λ that adjusts the balance between these two objectives. Friendly Adversarial Training (FAT; Zhang et al., 2020) further refines this approach by selecting adversarial examples that maintain minimal classification loss, thereby improving model robustness without excessive accuracy degradation. Wang et al. (2020) introduce Once-for-All Adversarial Training (OAT), which enables efficient in-situ calibration of a trained model without retraining by leveraging dual batch normalization to separate adversarial and standard feature statistics. Similarly, Rade and Moosavi-Dezfooli (2021b) investigate decision boundary shifts in adversarial training and propose Helper-based Adversarial Training (HAT) to mitigate excessive margin expansion, improving robustness without sacrificing accuracy.

2.5 Perturbations Beyond Adversarial Attacks

Recently, perturbations beyond adversarial attacks have been gaining increasing interest, including real-world perturbations such as noise, occlusion, and distribution shifts. ImageNet-C (Hendrycks and Dietterich, 2019) considers common corruptions, which applies a series of 19 common visual corruptions in 5 categories to images. This benchmark standardizes

and expands on the topic of corruption robustness, aiming to show which classifiers are preferable in safety-critical applications. ImageNet-A (Hendrycks et al., 2021b) considers natural adversarial examples, which place objects in unusual contexts or orientations. By using a simple adversarial filtration technique, the dataset ensures that real-world, unmodified examples transfer to various unseen models reliably, highlighting shared weaknesses in computer vision models. ImageNet-R (Hendrycks et al., 2021a) considers out-of-distribution data, which contains abstract or rendered versions of objects. The authors critically evaluate previously proposed methods for improving out-of-distribution robustness, revealing that larger models and artificial data augmentations can enhance real-world robustness. Contrary to some claims in prior work, the findings emphasize that these techniques are effective and that improvements in artificial robustness benchmarks can indeed transfer to real-world distribution shifts.

2.6 Convolutional Neural Networks (CNNs)

CNNs have significantly advanced image recognition tasks over the past few decades. Early work by LeCun et al. (1989) demonstrated the application of backpropagation in CNNs for handwritten zip code recognition, achieving high accuracy without extensive preprocessing. Building upon this, LeCun et al. (1995) compared various learning algorithms for handwritten digit classification, highlighting the superior performance of CNNs in capturing spatial hierarchies in image data. The field experienced a substantial leap with Krizhevsky et al. (2012) introduction of AlexNet, a deep CNN that utilized GPU acceleration to win the ImageNet Large Scale Visual Recognition Challenge, significantly reducing error rates and popularizing deep learning in computer vision. Further advancements were made by He et al. (2016) with the development of Deep Residual Networks (ResNets), which addressed the degradation problem in deep networks by introducing residual learning, enabling the training of substantially deeper models. Zagoruyko and Komodakis (2016) expanded on this by proposing Wide Residual Networks, which demonstrated that increasing the width of residual networks could achieve comparable or better performance with reduced depth, simplifying the training process.

2.7 Vision Transformers (ViTs)

Inspired by the success of multi-head self-attention (MSA, Vaswani et al., 2017) in Natural Language Processing (NLP), attention-based Transformer architectures have been adapted for vision tasks. Image Transformer (Parmar et al., 2018) applies self-attention to image generation, while Hu et al. (2019); Zhao et al. (2020) develops local self-attention for classification tasks. Vision Transformers (ViTs; Dosovitskiy et al., 2020) introduce a novel image patch embedding method but require large-scale pertaining. However, ViTs require extensive pretraining on large-scale datasets such as ImageNet-21K (Deng et al., 2009) and JFT-300M (Sun et al., 2017) to achieve competitive performance.

Efficient training methods such as DeiT (Touvron et al., 2021) introduced knowledge distillation with a distillation token to improve sample efficiency. Naseer et al. (2021) modify DeiT and introduce a shape token to encode shape information. He et al. (2022) proposes a masked autoencoder (MAE) built with ViTs for self-supervised learning, which masks random patches of input images. To reduce the fine-tuning cost of Transformers, Houlsby et al. (2019) propose the adapter, which adds and fine-tunes only a few trainable parameters to pre-trained BERT Transformers (Devlin et al., 2018). This enables efficient transfer learning among a large number of diverse text classification tasks. To support multi-task learning (MTL), AdapterFusion (Pfeiffer et al., 2020) uses multiple adapters in parallel in a two-stage manner, which consists of a knowledge extraction stage and a knowledge composition stage.

Robustness of ViTs. Empirical studies (Bhojanapalli et al., 2021; Mahmood et al., 2021; Paul and Chen, 2022) indicate that ViTs exhibit improved robustness against various perturbations compared to CNNs. Robust Vision Transformer (RVT; Mao et al., 2022) introduced position-aware attention scaling and patch-wise augmentation to enhance robustness. Pyramid Adversarial Training (PyramidAT; Herrmann et al., 2022) proposed a novel adversarial attack methodology that perturbs input images at multiple scales instead of modifying the network itself. FAN (Zhou et al., 2022) explored the role of self-attention in ViTs and introduced fully

attentional networks (FANs) to enhance mid-level feature robustness. To address vulnerabilities to patch-wise perturbations, Gu et al. (2022) proposed a temperature-scaling method that improves ViT robustness against adversarial patches.

2.8 Neural Architecture Search (NAS)

NAS automates the design of neural networks, optimizing their architectures for improved performance. Early NAS methods (Baker et al., 2016; Zoph et al., 2018; Liu et al., 2017; Real et al., 2017) were computationally expensive, requiring thousands of GPU hours due to the need to train and evaluate numerous candidate architectures. Differentiable NAS approaches such as DARTS (Liu et al., 2018) and PC-DARTS (Xu et al., 2019) significantly reduced the computational overhead by using a supernetwork and optimizing architecture parameters with gradient descent. However, these methods introduced memory constraints due to parallel training of multiple architectures. To address this, PC-DARTS proposed a partial channel connection technique that samples sub-channels for processing, reducing memory usage. Performance-estimating NAS techniques such as CARS (Yang et al., 2020) and PVLL-NAS (Li et al., 2020a) further improved efficiency by reducing reliance on exhaustive architecture evaluations.

NAS for Robustness. Robustness considerations in NAS have led to architectures specifically designed for adversarial resistance. RACL (Dong et al., 2020) imposed constraints on architecture parameters to minimize the Lipschitz constant, improving robustness. Prior studies (Cisse et al., 2017; Weng et al., 2018) support the claim that networks with smaller Lipschitz constants exhibit greater robustness. It is, therefore, effective to improve the robustness of neural architectures by constraining their Lipschitz constant. RobNet (Guo et al., 2020) introduced adversarial training into the NAS optimization process, ensuring that selected architectures inherently resist adversarial perturbations.

CHAPTER 3

Dilating Neural Architectures with Theoretical Guarantees

3.1 Motivation

In the past few decades, Convolutional Neural Networks (CNNs; LeCun et al., 1989, 1995; Krizhevsky et al., 2012; He et al., 2016; Zagoruyko and Komodakis, 2016) have achieved remarkable success through novel architecture designs and network scale expansion, often surpassing human performance in certain tasks. However, they remain vulnerable to adversarial attacks (Goodfellow et al., 2014), where small perturbations in input images can mislead predictions, reducing their reliability in practical applications. Adversarial training, a key defense strategy, augments training data with adversarial examples generated using methods like the Fast Gradient Sign Method (FGSM; Goodfellow et al., 2014) and Projected Gradient Descent (PGD; Madry et al., 2017). However, a fundamental challenge lies in the trade-off between standard accuracy and adversarial robustness (Tsipras et al., 2018).

Although numerous efforts have been made to balance this trade-off by carefully designing various training methods for neural networks (Zhang et al., 2019; Wang et al., 2020; Rade and Moosavi-Dezfooli, 2021b), less attention has been given to the fact that the neural architecture itself inherently limits the network's performance. The capacity of deep neural network has been demonstrated to be critical to its adversarial robustness (Madry et al., 2017; Tsipras et al., 2018; Zhang et al., 2021). Madry et al. (2017) find capacity plays an important role in adversarial robustness, and networks require a larger capacity for adversarial than standard tasks. Tsipras et al. (2018) suggests simple classifiers for standard tasks do not have the capability of reaching good performance on adversarial tasks. However, how to use

the minimal increase of network capacity in exchange for maximum adversarial robustness remains an open question.

Recently, there have been a few attempts to automatically design robust neural architectures from scratch. For example, RACL (Dong et al., 2020) applies the Lipschitz constraint on architecture parameters in one-shot NAS to reduce the Lipschitz constant and improve the robustness. RobNet (Guo et al., 2020) search for adversarially robust network architectures directly with adversarial training. Despite these studies, a deeper understanding of the accuracy and robustness trade-off from the architectural perspective is still largely missing. Despite their success in designing robust architectures, they cannot guarantee the accuracy of them. Once again, the trade-off becomes a challenge.

In this chapter, we focus on designing neural network architectures that are sufficient for both standard and adversarial classification from the architecture perspective. We propose **n**eural **a**rchitecture **d**ilation for **a**dversarial **r**obustness (NADAR). Beginning with the backbone network of satisfactory accuracy over the natural data, we search for a dilation architecture to pursue a maximal robustness gain while preserving a minimal accuracy drop. Besides, we also apply a FLOPs-aware approach to optimize the architecture, which can prevent the architecture from increasing the computation cost of the network too much. We theoretically analyze our dilation framework and prove that our constrained optimization objectives can effectively achieve our motivations. Experimental results on benchmark datasets demonstrate the significance of studying the adversarial robustness from the architecture perspective and the effectiveness of the proposed algorithm.

3.2 Methodology

The adversarial training can be considered as a minimax problem, where the adversarial perturbations are generated to attack the network by maximizing the classification loss, and the network is optimized to defend against such attacks

$$\min_{f} \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}} \left[\max_{\boldsymbol{x}'\in\mathcal{B}_p(\boldsymbol{x},\varepsilon)} \ell(f(\boldsymbol{x}'), y) \right], \tag{3.1}$$

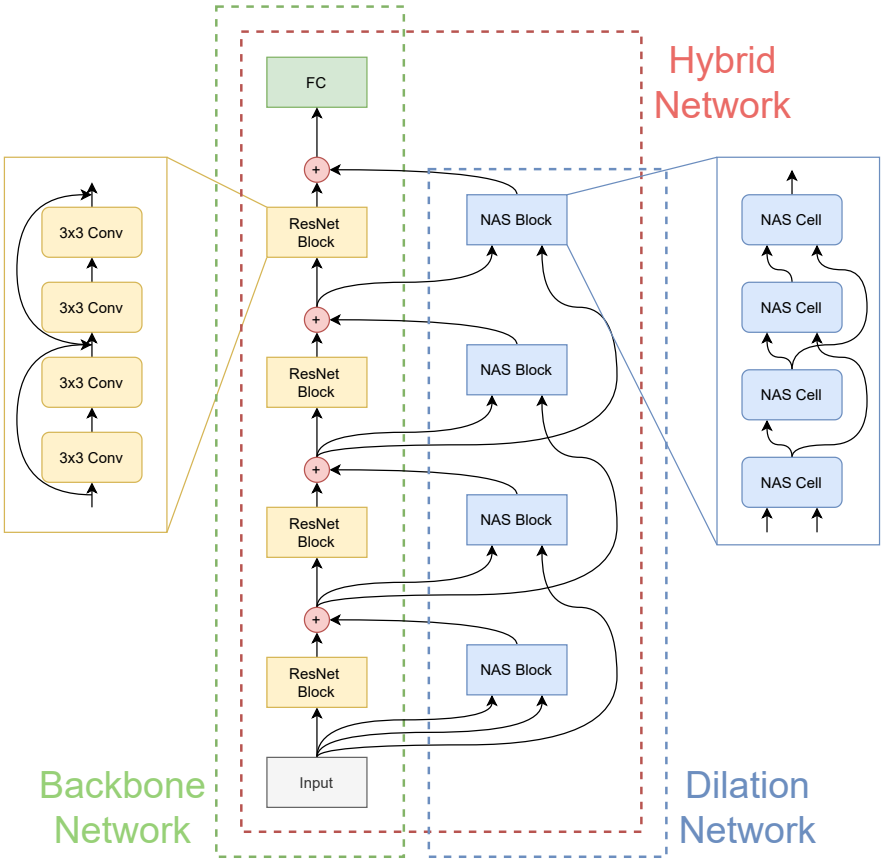


FIGURE 3.1. The overall structure of a NADAR hybrid network.

where $\mathcal{D}$ is the distribution of the natural examples $\boldsymbol{x}$ and the labels y, $\mathcal{B}_p(\boldsymbol{x}, \varepsilon) = \{\boldsymbol{x}' : \|\boldsymbol{x} - \boldsymbol{x}'\|_p \leq \varepsilon\}$ defines the set of allowed adversarial examples $\boldsymbol{x}'$ within the scale ε of small perturbations under l_p normalization, and f is the network under attack.

3.2.1 Robust Architecture Dilation

Suppose that we have a backbone network f_b that can achieve satisfactory accuracy with the natural data. To strengthen its adversarial robustness without hurting the standard accuracy, we propose to increase the capacity of this backbone network f_b by dilating it with a network f_d, whose architecture and parameters will be optimized within the adversarial training.

The backbone network f_b is split into blocks. A block $f_b^{(l)}$ is defined as a set of successive layers in the backbone with the same resolution. For a backbone with L blocks, i.e., $f_b = \{f_b^{(l)}, l \in 1, \ldots, L\}$, we attach a cell $f_d^{(l)}$ of the dilation network to each block $f_b^{(l)}$. Therefore,

the dilation network also has L cells, i.e., $f_d = \{f_d^{(l)}, l \in 1, \ldots, L\}$. For the dilation architecture, we search for cells within a NASNet-like (Zoph et al., 2018) search space. In a NASNet-like search space, each cell takes two previous outputs as its inputs. The backbone and the dilation network are further aggregated by element-wise sum. The overall structure of a NADAR hybrid network is as shown in Fig. 3.1. Formally, the hybrid network for the adversarial training is defined as

$$f_{\texttt{hyb}}(\boldsymbol{x}) = h\left(\odot_{l=1,\ldots,L}\left(f_b^{(l)}(\boldsymbol{z}_{\texttt{hyb}}^{(l-1)}) + f_d^{(l)}(\boldsymbol{z}_{\texttt{hyb}}^{(l-1)}, \boldsymbol{z}_{\texttt{hyb}}^{(l-2)})\right)\right), \tag{3.2}$$

where $\boldsymbol{z}_{\texttt{hyb}}^{(l)} = f_b^{(l)}(\boldsymbol{z}_{\texttt{hyb}}^{(l-1)}) + f_d^{(l)}(\boldsymbol{z}_{\texttt{hyb}}^{(l-1)}, \boldsymbol{z}_{\texttt{hyb}}^{(l-2)})$ is the latent feature extracted by the backbone block and the dilation block, and $\odot$ represents functional composition. We also define a classification hypothesis $h : \boldsymbol{z}_{\texttt{hyb}}^{(L)} \to \hat{y}$, where $\boldsymbol{z}_{\texttt{hyb}}^{(L)}$ is the latent representation extracted by the last convolutional layer L, and $\hat{y}$ is the predicted label.

During the search, the backbone network f_b has a fixed architecture and is parameterized by network weights $\boldsymbol{\theta}_b$. The dilation network f_d is parameterized by not only network weights $\boldsymbol{\theta}_d$ but also the architecture parameter $\boldsymbol{\alpha}_d$. The objective of robust architecture dilation is to optimize $\boldsymbol{\alpha}_d$ for the minimal adversarial loss

$$\min_{\boldsymbol{\alpha}_d} \quad \mathcal{L}_{\texttt{valid}}^{(\texttt{adv})}(f_{\texttt{hyb}}; \boldsymbol{\theta}_d^*(\boldsymbol{\alpha}_d)), \tag{3.3}$$

$$\textbf{s.t.} \quad \boldsymbol{\theta}_d^*(\boldsymbol{\alpha}_d) = \arg\min_{\boldsymbol{\theta}_d} \mathcal{L}_{\texttt{train}}^{(\texttt{adv})}(f_{\texttt{hyb}}), \tag{3.4}$$

where $\mathcal{L}_{\texttt{train}}^{(\texttt{adv})}(f_{\texttt{hyb}})$ and $\mathcal{L}_{\texttt{valid}}^{(\texttt{adv})}(f_{\texttt{hyb}}; \boldsymbol{\theta}_d^*(\boldsymbol{\alpha}_d))$ are the adversarial losses of $f_{\texttt{hyb}}$ on the training set $\mathcal{D}_{\texttt{train}}$ and the validation set $\mathcal{D}_{\texttt{valid}}$, respectively, and $\boldsymbol{\theta}_d^*(\boldsymbol{\alpha}_d)$ is the optimal network weights of f_d depending on the current dilation architecture $\boldsymbol{\alpha}_d$.

3.2.2 Standard Performance Constraint

Existing works on adversarial robustness often fix the network capacity, and the increase of adversarial robustness is accompanied by the standard accuracy drop (Tsipras et al., 2018; Zhang et al., 2019). However, in our method, we increase the capacity with dilation, which allows us to increase the robustness while maintaining a competitive standard accuracy. We

reach that with a standard performance constraint on the dilation architecture. The constraint is achieved by comparing the standard performance of the hybrid network $f_{\texttt{hyb}}$ to the standard performance of the backbone. We denote the network using the backbone only as $f_{\texttt{bck}}$, which can be formally defined as

$$f_{\texttt{bck}}(\boldsymbol{x}) = h\left(\odot_{l=1,\dots,L} f_b^{(l)}\left(\boldsymbol{z}_{\texttt{bck}}^{(l-1)}\right)\right), \tag{3.5}$$

where $\boldsymbol{z}_{\texttt{bck}}^{(l)} = f_b^{(l)}(\boldsymbol{z}_{\texttt{bck}}^{(l-1)})$ is the latent feature extracted by the backbone block. The standard model is optimized with natural examples by

$$\min_{\boldsymbol{\theta}_b} \quad \mathcal{L}^{(\texttt{std})}(f_{\texttt{bck}}) = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}\left[\ell(f_{\texttt{bck}}(\boldsymbol{x}), y)\right]. \tag{3.6}$$

where $\mathcal{L}^{(\texttt{std})}$ is the standard loss. Similarly, we can define the standard loss $\mathcal{L}^{(\texttt{std})}(f_{\texttt{hyb}})$ for the hybrid network $f_{\texttt{hyb}}$. In this way, we can compare the two networks by the difference in their losses and constrain the standard loss of the hybrid network to be equal to or lower than the standard loss of the standard network

$$\mathcal{L}^{(\texttt{std})}(f_{\texttt{hyb}}) - \mathcal{L}^{(\texttt{std})}(f_{\texttt{bck}}) \leq 0. \tag{3.7}$$

We do not directly optimize the dilation architecture on the standard task because it is introduced to capture the difference between the standard and adversarial tasks to improve the robustness of the standard-trained backbone. It is unnecessary to let both the backbone network and the dilation network learn the standard task.

3.2.3 FLOPs-Aware Architecture Optimization

By enlarging the capacity of networks, we can improve the robustness, but a drawback is that the model size and computation cost are raised. We want to obtain the largest robustness improvement with the lowest computation overhead. Therefore, a computation budget constraint on architecture search is applied. As we are not targeting any specific platform, the number of **fl**oating point **op**eration**s** (FLOPs) in the architecture instead of the inference latency is considered. The FLOPs are calculated by counting the number of multi-add operations in the network.

We use a differentiable manner to optimize the dilation architecture. In differentiable NAS, a directed acyclic graph (DAG) is constructed as the supernetwork, whose nodes are latent representations and edges are operations. Given that the adversarial training is computationally intensive, to reduce the search cost, a partial channel connections technique proposed by Xu et al. (2019) is utilized.

During the search, operation candidates for each edge are weighted summed with a softmax distribution of the architecture parameter $\boldsymbol{\alpha}$

$$\bar{o}^{(i,j)}(\boldsymbol{x}_i) = (1 - S_{i,j}) * \boldsymbol{x}_i + \sum_{o \in \mathcal{O}} \left(\frac{\exp(\boldsymbol{\alpha}_{i,j}^{(o)})}{\sum_{o' \in \mathcal{O}} \exp(\boldsymbol{\alpha}_{i,j}^{(o')})} \cdot o(S_{i,j} * \boldsymbol{x}_i) \right), \tag{3.8}$$

where $\mathcal{O}$ is a set of operation candidates, $\boldsymbol{x}_i$ is the output of the i-th node, and $S_{i,j}$ is binary mask on edge (i, j) for partial channel connections. The binary mask $S_{i,j}$ is set to 1 or 0 to let the channel be selected or bypassed, respectively. Besides the architecture parameter $\boldsymbol{\alpha}$, the partial channel connections technique also introduces an edge normalization weight $\boldsymbol{\beta}$

$$\boldsymbol{I}^{(j)} = \sum_{i<j} \left(\frac{\exp(\boldsymbol{\beta}_{i,j})}{\sum_{i'<j} \exp(\boldsymbol{\beta}_{i',j})} \cdot \bar{o}^{(i,j)}(\boldsymbol{x}_i) \right), \tag{3.9}$$

where $\boldsymbol{I}^{(j)}$ is the j-th node. The edge normalization can stabilize differentiable NAS by reducing fluctuation in edge selection after search.

Considering Eqs. 3.8 and 3.9, the expected FLOPs of the finally obtained discrete architectures from the one-shot supernetwork can be estimated according to $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. We calculate the weighted sum of FLOPs of the operation candidates with the identical softmax distributions in Eqs. 3.8 and 3.9, which can naturally lead to an expectation. Therefore, the expected FLOPs of node $\boldsymbol{I}^{(j)}$ can be calculated by

$$\mathrm{FLOPs}(\boldsymbol{I}^{(j)}) = \sum_{i<j} \frac{\exp(\boldsymbol{\beta}_{i,j})}{\sum_{i'<j} \exp(\boldsymbol{\beta}_{i',j})} \cdot \sum_{o \in \mathcal{O}} \frac{\exp(\boldsymbol{\alpha}_{i,j}^{(o)})}{\sum_{o' \in \mathcal{O}} \exp(\boldsymbol{\alpha}_{i,j}^{(o')})} \cdot \mathrm{FLOPs}(o). \tag{3.10}$$

After that, the FLOPs of the dilation network $\mathrm{FLOPs}(f_d)$ can be estimated by taking the sum of the FLOPs of all the nodes and cells. The objective function in Eq. 3.3 can be rewritten

with the FLOPs constraint as

$$\min_{\boldsymbol{\alpha}_d} \quad \gamma \log(\text{FLOPs}(f_d))^\tau \cdot \mathcal{L}^{(\texttt{adv})}_{\texttt{valid}}(f_{\texttt{hyb}}), \tag{3.11}$$

where γ and τ are two coefficient terms. τ controls the sensitivity of the objective function to the FLOPs constraint, and γ scales the constraint to a reasonable range (e.g., around 1.0).

3.2.4 Optimization

We reformulate the bi-level form optimization problem defined in Eqs 3.3 and 3.4 into a constrained optimization form. Combining with the standard performance constraint in Eq. 3.7 and the FLOPs-aware objectives in Eq. 3.11, we have

$$\min_{\boldsymbol{\alpha}_d} \quad \gamma \log(\text{FLOPs}(f_d))^\tau \cdot \mathcal{L}^{(\texttt{adv})}_{\texttt{valid}}(f_{\texttt{hyb}}; \boldsymbol{\theta}^*_d(\boldsymbol{\alpha}_d)), \tag{3.12}$$

$$\textbf{s.t.} \quad \mathcal{L}^{(\texttt{std})}_{\texttt{valid}}(f_{\texttt{hyb}}) - \mathcal{L}^{(\texttt{std})}_{\texttt{valid}}(f_{\texttt{bck}}) \leq 0, \tag{3.13}$$

$$\boldsymbol{\theta}^*_d(\boldsymbol{\alpha}_d) = \arg\min_{\boldsymbol{\theta}_d} \mathcal{L}^{(\texttt{adv})}_{\texttt{train}}(f_{\texttt{hyb}}), \quad \textbf{s.t.} \quad \mathcal{L}^{(\texttt{std})}_{\texttt{train}}(f_{\texttt{hyb}}) - \mathcal{L}^{(\texttt{std})}_{\texttt{train}}(f_{\texttt{bck}}) \leq 0. \tag{3.14}$$

To solve the constrained architecture optimization problem, we apply a common method for constrained optimization, namely the alternating direction method of multipliers (ADMM). To apply ADMM, the objective function needs to be reformulated as an augmented Lagrangian function. We first deal with the upper-level optimization of the architecture parameter $\boldsymbol{\alpha}_d$

$$L(\{\boldsymbol{\alpha}_d\}, \{\lambda_1\}) = \gamma \log(\text{FLOPs}(f_d))^\tau \cdot \mathcal{L}^{(\texttt{adv})}_{\texttt{valid}}(f_{\texttt{hyb}}) + \lambda_1 \cdot c_1 + \frac{\rho}{2} \| \max\{0, c_1\} \|_2^2 \tag{3.15}$$

$$\textbf{s.t.} \quad c1 = \mathcal{L}^{(\texttt{std})}_{\texttt{valid}}(f_{\texttt{hyb}}) - \mathcal{L}^{(\texttt{std})}_{\texttt{valid}}(f_{\texttt{bck}}), \tag{3.16}$$

where λ_1 is the Lagrangian multiplier, and $\rho \in \mathbb{R}_+$ is a positive number predefined in ADMM. We update $\boldsymbol{\alpha}_d$ and λ_1 alternately with

$$\boldsymbol{\alpha}_d^{(t+1)} \leftarrow \boldsymbol{\alpha}_d^{(t)} - \eta_1 \nabla L(\{\boldsymbol{\alpha}_d^{(t)}\}, \{\lambda_1^{(t)}\}) \tag{3.17}$$

$$\lambda_1^{(t+1)} \leftarrow \lambda_1^{(t)} + \rho \cdot c_1, \tag{3.18}$$

where η_1 is a learning rate for architecture. Similarly, the lower-level optimization problem of network weights $\boldsymbol{\theta}_d$ as an augmented Lagrangian function can be defined as

$$L(\{\boldsymbol{\theta}_d\}, \{\lambda_2\}) = \mathcal{L}^{(\texttt{adv})}_{\texttt{train}}(f_{\texttt{hyb}}) + \lambda_2 \cdot c_2 + \frac{\rho}{2}\| \max\{0, c_2\}\|_2^2 \tag{3.19}$$

$$\textbf{s.t.} \quad c2 = \mathcal{L}^{(\texttt{std})}_{\texttt{train}}(f_{\texttt{hyb}}) - \mathcal{L}^{(\texttt{std})}_{\texttt{train}}(f_{\texttt{bck}}), \tag{3.20}$$

where λ_2 is the Lagrangian multiplier. Similarly, we can update $\boldsymbol{\theta}_d$ and λ_2 with the same alternate manner

$$\boldsymbol{\theta}_d^{(t+1)} \leftarrow \boldsymbol{\theta}_d^{(t)} - \eta_2 \nabla L(\{\boldsymbol{\theta}_d^{(t)}\}, \{\lambda_2^{(t)}\}), \tag{3.21}$$

$$\lambda_2^{(t+1)} \leftarrow \lambda_2^{(t)} + \rho \cdot c_2, \tag{3.22}$$

where η_2 is the learning rate for network weights.

3.3 Theoretical Analysis

In this section, we provide a theoretical analysis of our proposed NADAR. As there are two major goals in our optimization problem, i.e., the standard performance constraint and the adversarial robustness, this analysis is also twofold. Firstly, a standard error bound of NADAR is analyzed. We demonstrate that the standard error of the dilated adversarial network can be bounded by the standard error of the backbone network and our standard performance constraint. Secondly, we compare the adversarial error of the dilated adversarial network and the standard error of the backbone standard network. We demonstrate that the adversarial performance can be improved by adding a dilation architecture to the backbone, even if the backbone is fixed. These two error bounds can naturally motivate the optimization problem in Eqs. 3.12 and 3.13. Detailed proofs are provided in Appendix A4. Besides, through this analysis, we want to reveal two **remarks**: (1) enlarging the backbone network with dilation can improve its performance, which proves the validity of our neural architecture dilation; (2) the dilation architecture should be consistent with the backbone of clean samples and samples that are insensitive to attacks, which directly inspires our standard performance constraint.

We discuss the binary classification case for simplification, where the label space is $\mathcal{Y} = \{-1, +1\}$. The obtained theoretical results can also be generalized to the multi-class classification case. A binary classification *hypothesis* $h \in \mathcal{H}$ is defined as a mapping $h : \mathcal{X} \mapsto \mathbb{R}$, where $\mathcal{H}$ is a hypothesis space, and $\mathcal{X}$ is an input space of natural examples. The output of the hypothesis is a real value score. The predicted label can be obtained from the score by applying the sign function $\text{sign}(\cdot)$ on it. Denote the backbone hypothesis as h_b. By further investigating the influence of the dilation architecture, the hypothesis of the resulting hybrid network can be defined as $h_{\texttt{hyb}}(\boldsymbol{x}) = h_b(\boldsymbol{x}) + h_d(\boldsymbol{x})$, where h_d stands for the change resulting from the dilation architecture. The standard model corresponds to a hypothesis $h_{\texttt{bck}}(\boldsymbol{x}) = h_b(\boldsymbol{x})$. We further define the *standard error* of a hypothesis h as

$$R_{\texttt{std}}(h) := \mathbb{E}\left[\mathbf{1}\{\text{sign}(h(\boldsymbol{x})) \neq y\}\right], \tag{3.23}$$

and the *adversarial error* of it as

$$R_{\texttt{adv}}(h) := \mathbb{E}\left[\mathbf{1}\{\exists \boldsymbol{x}' \in \mathcal{B}_p(x, \varepsilon),\ \textbf{s.t.}\ \text{sign}(h(\boldsymbol{x}')) \neq y\}\right], \tag{3.24}$$

where $\mathbf{1}\{\cdot\}$ denotes the indicator function.

3.3.1 Standard Error Bound

To compare the error of two different hypotheses, we first slightly modify the error function. Eq. 3.23 checks the condition that $\text{sign}(h(\boldsymbol{x})) \neq y$. Because the label space is binary and the output space of h is real value, we can remove the sign function by replacing the condition with $yh(\boldsymbol{x}) \leq 0$. Then, by applying a simple inequality $\mathbf{1}\{yh(\boldsymbol{x}) \leq 0\} \leq e^{-yh(\boldsymbol{x})}$, we have a very useful inequality about the standard error

$$R_{\texttt{std}}(h) \leq \mathbb{E}\left[e^{-yh(\boldsymbol{x})}\right]. \tag{3.25}$$

Eq. 3.25 can lead to our standard error bound in Theorem 3.3.1.

THEOREM 3.3.1. *Let $h_{\texttt{bck}}(\boldsymbol{x}) = h_b(\boldsymbol{x})$ be a standard hypothesis, $h_{\texttt{hyb}}(\boldsymbol{x}) = h_b(\boldsymbol{x}) + h_d(\boldsymbol{x})$ be a hybrid hypothesis, and $\mathcal{R}_{\texttt{std}}(h_{\texttt{bck}})$ and $\mathcal{R}_{\texttt{std}}(h_{\texttt{hyb}})$ be the standard error of $h_{\texttt{bck}}$ and $h_{\texttt{hyb}}$,*

respectively. For any mapping $h_b, h_d : \mathcal{X} \mapsto \mathbb{R}$*, we have*

$$\mathcal{R}_{\texttt{std}}(h_{\texttt{hyb}}) \leq \mathcal{R}_{\texttt{std}}(h_{\texttt{bck}}) + \mathbb{E}\left[e^{-h_b(\boldsymbol{x})h_d(\boldsymbol{x})}\right], \tag{3.26}$$

where $\boldsymbol{x} \in \mathcal{X}$ *is the input.*

Theorem 3.3.1 illustrates that the standard performance of the hybrid network is bounded by the standard performance of the backbone network and the sign disagreement between $h_b(\boldsymbol{x})$ and $h_d(\boldsymbol{x})$. This reflects our **remark (2)**. If the backbone accurately predicts the label of the natural data x, $h_d(\boldsymbol{x})$ shall make the same category prediction, which implies that the prediction by the hybrid hypothesis $h_{\texttt{hyb}}(\boldsymbol{x}) = h_b(\boldsymbol{x}) + h_d(\boldsymbol{x})$ can be strengthened and would not lead to a worse result than that of the standard hypothesis. To reach such an objective, it naturally links with the standard performance constraint proposed in Eq. 3.7 and applied in Eq. 3.13.

3.3.2 Adversarial Error Bound

Similar to Eq. 3.25, we can have an inequality about the adversarial error

$$R_{\texttt{adv}}(h) \leq \mathbb{E}\left[\max_{\boldsymbol{x}' \in \mathcal{B}_p(x,\varepsilon)} e^{-yh(\boldsymbol{x}')}\right], \tag{3.27}$$

based on which we can derive the following Lemma 1.

LEMMA 1. *For any mapping* $h : \mathcal{X} \mapsto \mathbb{R}$*, we have*

$$\mathbb{E}\left[\max_{\boldsymbol{x}' \in \mathcal{B}_p(x,\varepsilon)} e^{-yh(\boldsymbol{x}')}\right] \leq \mathbb{E}\left[\max_{\boldsymbol{x}' \in \mathcal{B}_p(x,\varepsilon)} e^{-yh(\boldsymbol{x})} e^{-h(\boldsymbol{x})h(\boldsymbol{x}')}\right], \tag{3.28}$$

where $\boldsymbol{x} \in \mathcal{X}$ *is the input,* $y \in \{-1, +1\}$ *is the corresponding label, and* ε *is the bound of allowed adversarial perturbation.*

Lemma 1 is an inherent feature of a single hypothesis. We generalize it to the case of dilating $h_{\texttt{bck}}$ to $h_{\texttt{hyb}}$ with a dilation hypothesis h_d.

THEOREM 3.3.2. *Let* $h_{\texttt{bck}}(\boldsymbol{x}) = h_b(\boldsymbol{x})$ *be a standard hypothesis,* $h_{\texttt{hyb}}(\boldsymbol{x}) = h_b(\boldsymbol{x}) + h_d(\boldsymbol{x})$ *be a dilated hypothesis,* $\mathcal{R}_{\texttt{std}}(h_{\texttt{bck}})$ *be the standard error of* $h_{\texttt{bck}}$*, and* $\mathcal{R}_{\texttt{adv}}(h_{\texttt{hyb}})$ *be the*

adversarial error of $h_{\texttt{hyb}}$. *For any mapping* $h_b, h_d : \mathcal{X} \mapsto \mathbb{R}$, *we have*

$$\mathcal{R}_{\texttt{adv}}(h_{\texttt{hyb}}) \leq \mathcal{R}_{\texttt{std}}(h_{\texttt{bck}}) + \mathbb{E}\left[\max_{\boldsymbol{x}'\mathcal{B}_p(x,\varepsilon)} e^{-yh_b(\boldsymbol{x})}\left(e^{-h_b(\boldsymbol{x})h_b(\boldsymbol{x}')}e^{-yh_d(\boldsymbol{x}')} - 1\right)\right]. \tag{3.29}$$

where $\boldsymbol{x} \in \mathcal{X}$ *is the input,* $y \in \{-1, +1\}$ *is the corresponding label, and* ε *is the bound of allowed adversarial perturbation.*

By minimizing $e^{-yh_b(\boldsymbol{x})}$ in Theorem 3.3.2, we expect the backbone network to have a satisfactory accuracy on the natural data, which is a prerequisite of the proposed algorithm. As the backbone network has been fixed in our method, the term $e^{-h_b(\boldsymbol{x})h_b(\boldsymbol{x}')}$ will not be influenced by the algorithm. The remaining term $e^{-yh_d(\boldsymbol{x}')}$ implies that even if the backbone network makes wrong prediction on the adversarial example $\boldsymbol{x}'$, there is still a chance for the dilation network h_d to correct the misclassification and improve the overall adversarial accuracy of the hybrid network $h_{\texttt{hyb}}$. This capability of dilation reflects our **remark (1)**. In another case, if h_b makes a correct prediction, h_d should agree with it, which reflects our **remark (2)** is also applied to the adversarial error.

3.4 Experiments

We perform extensive experiments to demonstrate that NADAR can improve the adversarial robustness of neural networks by dilating the neural architecture. In this section, we first compare both the standard and adversarial accuracy of our hybrid network to various baseline methods. Then, we perform experiments to analyze the impact of each component in the NADAR framework, including the dilation-based training approach and the standard performance constraint. Finally, we explore the sufficient scale of dilation and the effect of FLOPs constraint. More results on other datasets under various attacking manners with different backbones are also available in Appendix A.

3.4.1 Experiment Setting

We use a similar pipeline to previous NAS works (Liu et al., 2018; Xu et al., 2019; Dong et al., 2020; Guo et al., 2020). Firstly, we optimize the dilation architecture in a one-shot model. Then, a discrete architecture is derived according to the architecture parameters α and β. Finally, a discrete network is constructed and retrained for validation. During the dilation phase, the training set is split into two equal parts. One is used as the training set for network weights optimization, and the other one is used as the validation set for architecture parameter optimization. During the retraining and validation phases, the entire training set is used for training, and the trained network is validated on the original validation set.

We perform dilation under white-box attacks on CIAFR-10/100 (Krizhevsky et al., 2009) and ImageNet (Russakovsky et al., 2015) and under black-box attacks on CIFATR-10. The NADAR framework requires a backbone to be dilated. Following previous works (Madry et al., 2017; Shafahi et al., 2019; Zhang et al., 2020, 2019), we use the 10 times wider variant of ResNet, i.e., the Wide ResNet 34-10 (WRN34-10; Zagoruyko and Komodakis, 2016), on both CIFAR dataset, and use ResNet-50 (He et al., 2016) on ImageNet. The search space for dilated architecture and the searched dilated architectures are illustrated in Appendix A1.

Considering both the optimization of neural architecture and the generation of adversarial examples are computationally intensive, we apply methods to reduce the computational overhead during the dilation phase. As aforementioned, we utilize partial channel connections to reduce the cost of architecture optimization. As for the adversarial training during the search, we use FreeAT (Shafahi et al., 2019), which recycles gradients during training for the generation of adversarial examples and reduces the training cost.

3.4.2 Defense Against White-box Attacks

CIFAR-10. We compare the hybrid network with 4 categories of baseline methods, including standard training, adversarial training, standard NAS, and robust NAS. The standard training method and all the adversarial training methods use the WRN34-10. For the standard NAS methods, the architecture is the best architecture searched with standard training as

Category	Method	Params (M)		+× (G)		Valid Acc. Against (%)				
		Back.	Arch.	Back.	Arch.	Natural	FGSM	PGD-20	PGD-100	MI-FGSM
Standard	Standard	46.2	-	6.7	-	95.01	0.00	0.00	0.00	0.00
Adversarial Training	PGD-7 (Madry et al., 2017)	46.2	-	6.7	-	87.25	56.10	45.84	45.29	-
	FAT (Zhang et al., 2020)	46.2	-	6.7	-	89.34	65.52	46.13	46.82	-
	FreeAT-8 (Shafahi et al., 2019)	46.2	-	6.7	-	85.96	-	46.82	46.19	-
	TRADES-1 (Zhang et al., 2019)	46.2	-	6.7	-	88.64	-	48.90	-	51.26
	TRADES-6 (Zhang et al., 2019)	46.2	-	6.7	-	84.92	-	56.43	-	57.95
Standard NAS	AmoebaNet (Real et al., 2019)	-	3.2	-	0.5	83.41	56.40	39.47	-	47.60
	NASNet (Zoph et al., 2018)	-	3.8	-	0.6	83.66	55.67	48.02	-	53.05
	DARTS (Liu et al., 2018)	-	3.3	-	0.5	83.75	55.75	44.91	-	51.63
	PC-DARTS (Xu et al., 2019)	-	3.6	-	0.6	83.94	52.67	41.92	-	49.09
Robust NAS	RobNet-small (Guo et al., 2020)	-	4.4	-	N/A	78.05	53.93	48.32	48.08	48.98
	RobNet-medium (Guo et al., 2020)	-	5.7	-	N/A	78.33	54.55	49.13	48.96	49.34
	RobNet-large (Guo et al., 2020)	-	6.9	-	N/A	78.57	54.98	49.44	49.24	49.92
	RACL (Dong et al., 2020)	-	3.6	-	0.5	83.89	57.44	49.34	-	54.73
Dilation	NADAR-A (ours)	46.2	3.6	6.7	0.6	86.61	59.98	52.84	52.54	57.72
	NADAR-B (ours)	46.2	4.4	6.7	0.7	86.23	60.46	53.43	53.06	58.43

TABLE 3.1. The standard validation accuracy on natural images and adversarial validation accuracy under various attacks of NADAR compared to different baseline methods on CIFAR-10.

reported in their papers and is retrained with PGD-7 (Madry et al., 2017). For the adversarial NAS methods, we follow their original setting. We include two of the best dilation architectures obtained with our method, NADAR-A, and NADAR-B, dilated without and with the FLOPs constraint. The architectures are visualized in Appendix A1. Our architectures are also retrained with PGD-7. In Table 3.1, the standard accuracy on natural images and the adversarial accuracy under PGD-20 attack are reported.

Comparing NADAR-A and NADAR-B, the FLOPs constraint can obviously reduce the FLOPs number (reducing by 14.28%) as well as the parameters number (reducing by 18.19%) of the dilation architecture. The negative impact of it on the adversarial accuracy is marginal (only 0.59% under PGD-20 attack), and the standard accuracy can even be slightly improved.

As for adversarial training methods, we improve the adversarial performance by 7.59% with only a 1.02% standard performance drop compared to PGD-7 while we use the same training method but the hybrid network. This result illustrates that our dilation architecture can indeed improve the robustness of a network without modifying the training method. In the meantime, the standard accuracy is constrained to a competitive level. Compared to FreeAT-8, which is

their best adversarial setting, our method reaches both a lower standard accuracy drop and a higher adversarial accuracy gain.

A very different method to PGD and FreeAT, namely friendly adversarial training (FAT), aims to improve the standard accuracy of adversarially trained models by generating weaker adversarial examples than regular adversarial training. Although FAT can make significant improvements in standard accuracy with weak attacks, its adversarial accuracy gain against PGD-7 is marginal (only 0.29%). It even has lower adversarial accuracy than FreeAT-8 despite its higher standard accuracy. Unlike FAT, our method is dedicated to another direction of improvement, which improves the adversarial robustness without significantly affecting the standard performance. Even though there is still a trade-off between the standard and the adversarial accuracy, we can increase the ratio of the standard drop to the adversarial gain to $1 : 7.44$.

A previous work that focuses on the trade-off is TRADES, which introduces a tuning parameter (λ) to adjust the balance of the trade-off. Nevertheless, comparing the TRADES-1 ($1/\lambda = 1$) and TRADES-6 ($1/\lambda = 6$), their trade-off ratio is only $1 : 2.02$ (i.e., 3.72% standard accuracy drop for 7.53% adversarial accuracy gain). We can provide a better ratio of trade-offs than them. Besides, our standard performance is naturally constrained by Eq. 3.7. There are no hyperparameters in it that need to be adjusted, which leads to our better trade-off ratio of the standard drop to the adversarial gain and a more reasonable balance than TRADES.

Finally, compared to NAS methods, our hybrid network can outperform both the standard and robust NAS architectures. The standard NAS architectures are not optimized for adversarial robustness. Except for the NASNet, their adversarial accuracies are generally poor. Although they are optimized for standard tasks, their standard accuracy after adversarial training is significantly lower than the WRN34-10 trained with PGD-7. As for robust NAS methods, RobNet significantly sacrifices its standard accuracy for robustness, which has the lowest standard accuracy among all the works listed in Table 3.1. RACL has a better trade-off, but it can only reach the standard accuracy of standard NAS architecture, which is still lower than adversarially trained WRN34-10. This demonstrates that dilating a standard backbone

Method	Valid Acc. Against (%)	
	Natural	PGD-20
Standard	78.84	0.00
PGD-7 (Madry et al., 2017)	-	23.20
FreeAT-8 (Shafahi et al., 2019)	62.13	25.88
RobNet-large (Guo et al., 2020)	-	23.19
RACL (Dong et al., 2020)	-	27.80
NADAR-A (ours)	61.73	27.77
NADAR-B (ours)	62.56	28.40

TABLE 3.2. The standard and adversarial validation on CIFAR-100.

for both standard constraint and adversarial gain is more effective than designing a new architecture from scratch.

CIFAR-100. We adapt architectures dilated on CIFAR-10 to CIFAR-100 and report the results in Table 3.2. We consider two kinds of baselines, including traditional adversarial training methods (PGD-7 and FreeAT-8) and two robust NAS methods (RobNet and RACL). The results show that even with more categories, NADAR can still reach superior robustness under the PGD-20 attack. As for the standard validation accuracy, we can reach competitive performance compared to FreeAT-8. The other works do not report standard accuracy in their papers. As for the adversarial validation accuracy, our NADAR-B can outperform all the baselines, while NADAR-A is slightly lower than RACL but significantly better than the others.

Tiny-ImageNet. We also adapt our architectures to a larger dataset, namely Tiny-ImageNet. For efficient training on Tiny-ImageNet, we compare our dilated architecture with PGD and two efficient adversarial training methods, i.e., FreeAT (Shafahi et al., 2019) and FastAT (Wong et al., 2020). We follow the ImageNet setting of Shafahi et al. (2019) and Wong et al. (2020), which uses ResNet-50 as the backbone and sets the clip size $\epsilon = 4$. For PGD and FreeAT, we set the number of steps $K = 4$ and the step size $\epsilon_S = 2$ The results are reported in Table 3.3. We also report the GPU days cost to train the networks with NVIDIA

Architecture	Training Method	GPU Days	Valid Acc. Against (%)			
			Natural	FGSM	PGD-10	PGD-20
Backbone	PGD-4	0.19	43.23	24.13	22.25	22.16
Dilation (ours)	PGD-4	0.45	44.37	24.33	22.69	22.70
Backbone	FreeAT-4	0.05	42.73	24.10	22.67	22.58
Dilation (ours)	FreeAT-4	0.12	44.68	24.66	22.88	22.78
Backbone	FastAT	0.12	45.92	23.53	20.66	20.54
Dilation (ours)	FastAT	0.22	46.22	23.90	21.21	21.14

TABLE 3.3. The standard and adversarial validation accuracy on Tiny-ImageNet with ResNet-50 as the backbone.

Category	Method	Valid Acc. Against (%)				
		APGD_{CE}	$\text{APGD}^{\text{T}}_{\text{DLR}}$	FAB^{T}	Square	AA
Adversarial Training	PGD-7 (Madry et al., 2017)	44.75	44.28	44.75	53.10	44.04
	FastAT (Wong et al., 2020)	45.90	43.22	43.74	53.32	43.21
	FreeAT-8 (Shafahi et al., 2019)	43.66	41.64	43.44	51.95	41.47
Dilation	NADAR-A (ours)	52.27	50.00	50.00	58.69	49.83
	NADAR-B (ours)	52.64	50.45	50.88	59.33	50.44

TABLE 3.4. The adversarial validation accuracy of NADAR compared to different baseline methods under AutoAttack on CIFAR-10.

V100 GPU. Although NADAR consumes approximately 1.8$\sim$2.4$\times$ GPU days, our method can consistently outperform the baselines in terms of both natural and adversarial accuracy.

3.4.3 Defense Against AutoAttack

Besides the traditional attack methods, we also consider a novel and promising parameter-free evaluation method, namely AutoAttack (Croce and Hein, 2020). We use the standard setting of AutoAttack, including four individual attacks: APGD_{CE}, $\text{APGD}^{\text{T}}_{\text{DLR}}$, FAB^{T} and Square. The column AA is a combination of the four attacks. The validation accuracy is reported in Table 3.4. As can be seen, our method achieves superior performance compared to the baselines. For simplification, we only list the comparison to the best performance among PGD-7, FastAT, and FreeAT-8 as follows: under APGD_{CE} attack, we can outperform FastAT

Defense Network	Source Network	Valid Acc. (%)			
		FGSM	PGD-20	PGD-100	MI-FGSM
WRN34-10 + PGD-7	WRN34-10 + Natural	83.99	84.56	84.76	84.05
NADAR-B + PGD-7	WRN34-10 + Natural	85.94	86.59	86.51	85.95
WRN34-10 + PGD-7	WRN34-10 + FGSM	70.78	68.26	68.30	69.73
NADAR-B + PGD-7	WRN34-10 + FGSM	77.25	77.66	77.69	77.19
WRN34-10 + PGD-7	NADAR-B + PGD-7	69.33	67.08	67.11	68.26
NADAR-B + PGD-7	WRN34-10 + PGD-7	70.78	68.26	68.30	69.73

TABLE 3.5. The adversarial validation accuracy under black-box attacks on CIFAR-10.

by 6.74%; under $\text{APGD}^{\text{T}}_{\text{DLR}}$, we can outperform PGD-7 by 6.17%; under FAB^{T} attacks we can outperform PGD-7 by 6.13%; under Square attack, we can outperform FastAT by 6.01%.

3.4.4 Defense Against Black-box Attacks

We perform black-box attacks on CIFAR-10. We use different source networks to generate adversarial examples. For the source networks, we use the WRN34-10 backbone trained with natural images and adversarial images generated with FGSM and PGD-7. For the defense networks, we compare our best NADAR-B architecture with the plain WRN34-10 backbone. Both of them are trained with PGD-7. The results are reported in Table 3.5 grouped according to source networks. With the WRN34-10 source network trained with natural images and FGSM, NADAR-B can consistently outperform the backbone. We also use NADAR-B trained with PGD-7 and WRN34-10 trained with PGD-7 to attack each other. Our hybrid networks can consistently achieve superior performance.

3.4.5 NADAR Trained with Different Adversarial Training Methods

In previous experiments, our NADAR was trained with PGD-7, and other competitors were trained in their corresponding settings. To further investigate whether NADAR can work along with other stronger adversarial training methods than the PGD-7, we train the obtained architecture with various adversarial training methods, including FAT, TRADES-1, and

Model	AT Method	Valid Acc. Against (%)		
		Natural	PGD-20	AA
Backbone	PGD-7	87.25	45.84	44.04
NADAR (ours)	PGD-7	86.23	53.43$^{(+7.59)}$	50.44$^{(+6.4)}$
Backbone	FAT	89.34	46.13	N/A
NADAR (ours)	FAT	88.12	54.53$^{(+8.40)}$	51.37$^{(N/A)}$
Backbone	TRADES-1	88.64	48.90	43.01
NADAR (ours)	TRADES-1	89.77	55.13$^{(+6.23)}$	50.90$^{(+7.89)}$
Backbone	TRADES-6	84.92	56.43	53.08
NADAR (ours)	TRADES-6	83.94	57.43$^{(+1.00)}$	55.25$^{(+2.17)}$

TABLE 3.6. Comparison of test accuracy of NADAR and WRN34-10 backbone when using various AT methods for training.

TRADES-6. The results are shown in Table 3.6. As can be seen, our method can consistently outperform the backbone in terms of adversarial accuracy. Regarding natural accuracy, NADAR is competitive to the backbone (only 0.52% lower on average, which is a marginal drop, given the magnitude of robustness improvement). Especially under the TRADES-1 setting, which focuses on natural accuracy, NADAR can outperform its backbone on both natural accuracy (89.77% vs. 88.64%) and adversarial accuracy (55.13% vs. 48.90% under PGD-20 and 50.90% vs. 43.01% under AutoAttack). In the meantime, its adversarial accuracy (55.13%) is much closer to the one of TRADES-6 (56.43%), which focuses on robustness and has low (4.85% lower) natural accuracy, than to the one of TRADES-1 (48.90%), which has a similar natural accuracy.

3.4.6 Ablation Study of Dilation Method

We perform an ablation study on the dilation method. There are two crucial components in our method. Firstly, the separate optimization objectives of standard and adversarial tasks can ensure that the backbone focuses on clear images, and the dilation network learns to improve the robustness of the backbone. Secondly, the standard performance constraint prevents the dilation network from harming the standard performance of the backbone network. This experiment demonstrates that both of them make crucial contributions to the final results.

Separate Objectives	Standard Constraint	Valid Acc. Against (%)	
		Natural	PGD-20
No	N/A	84.19±0.32	45.97±0.18
Yes	No	84.79±0.55	48.53±0.33
Yes	Yes	85.97±0.26	53.18±0.25

TABLE 3.7. The standard and adversarial accuracy by retraining of various networks dilated with ablated manners.

Note that without the separate objectives, the hybrid network is trained as a whole. Therefore, there is also no standard constraint. We use the same settings as Section 3.4.2.

The standard and adversarial accuracy of the obtained networks by retraining is reported in Table 3.7. If there is no standard performance constraint, dilating together or separately has a similar standard performance. Although the backbone of the latter is trained with standard objective, it won't influence the retraining results too much (only 0.6% higher on average). The complete framework with standard constraint reaches the best standard accuracy after retraining. As for adversarial accuracy, dilating with separate objectives can consistently outperform dilating with a single adversarial objective.

3.5 Chapter Summary

The trade-off between accuracy and robustness is considered an inherent property of neural networks, which cannot be easily bypassed with adversarial training or robust NAS. In this chapter, we propose to dilate the architecture of neural networks to increase the adversarial robustness while maintaining a competitive standard accuracy with a straightforward constraint. The framework is called neural architecture dilation for adversarial robustness (NADAR). Extensive experiments demonstrate that NADAR can effectively improve the robustness of neural networks and can reach a better trade-off ratio than existing methods.

CHAPTER 4

A Gated Module for Feature Disentanglement and Adaptive Fusion

---

4.1 Motivation

A popular theory explaining the trade-off between robustness and accuracy is the existence of two kinds of different features (Tsipras et al., 2018; Kim et al., 2021; Yang et al., 2021b). The first kind of features is moderately correlated to the task and robust to attacks, while the second kind of features is weakly correlated to the task and, therefore, non-robust. Unfortunately, Tsipras et al. (2018) demonstrates that those moderately correlated and robust features only have limited contributions to accurate predictions (i.e., they are *robust* but *non-predictive*), and further improving the accuracy heavily relies on those weakly related and non-robust features (i.e., they are *predictive* but *non-robust*) Therefore, this trade-off is usually considered an inherent characteristic of DNNs. Although there are many efforts that aim to control or improve this trade-off (Zhang et al., 2019; Wang et al., 2020; Rade and Moosavi-Dezfooli, 2021a,b; Li et al., 2021b), it is still hard to efficiently and effectively improve it on large-scale datasets such as ImageNet (Deng et al., 2009).

Recently, a new family of vision models, namely Vision Transformers (ViTs; Dosovitskiy et al., 2020; Touvron et al., 2021; Steiner et al., 2021), has outperformed CNNs on various kinds of tasks. There are many subsequent works that discuss diverse variants of ViTs to improve their performance. TNT (Han et al., 2021) divides patches in ViTs into smaller sub-patches and applies a transformer-in-transformer architecture with an additional inner transformer. T2T-ViT (Yuan et al., 2021) introduces local feature aggregation to boost local information. Swin (Liu et al., 2021) performs local attention within various windows, and a shifted window partitioning approach is introduced for cross-window connections.

However, the aforementioned works mainly focus on the natural accuracy with clean data. Although empirical analyses have demonstrated that ViTs demonstrate robustness against various kinds of perturbations (Bhojanapalli et al., 2021; Mahmood et al., 2021; Paul and Chen, 2022), there are only a limited number of works (Mao et al., 2022; Herrmann et al., 2022; Zhou et al., 2022; Gu et al., 2022) focus on improving the robustness. Besides, existing methods have ignored how to boost a naturally pre-trained ViT for robustness. A pre-trained ViT has high utility because it can extract predictive features to ensure high accuracy on downstream tasks. Despite its high utility, it also has low reliability because its non-robust features are vulnerable to perturbations. Therefore, it is worth studying how to obtain a useful and reliable ViT.

Furthermore, fine-tuning of ViTs is very computationally intensive (Dosovitskiy et al., 2020). Considering the high overheads of adversarial training (Goodfellow et al., 2014; Madry et al., 2017; Zhang et al., 2019), it is even more expensive to train both accurate and robust ViT models on a large-scale dataset. To reduce the intractable cost of training and fine-tuning large-scale Transformer models on various tasks, adapter (Houlsby et al., 2019) proposes to add and fine-tune only a few parameters per task in the field of NLP. AdapterFusion (Pfeiffer et al., 2020) further supports multi-task transfer learning by using multiple adapters in parallel and combining their outputs with an attention-based gated fusion module.

In this chapter, we propose the Trade-off between Robustness and Accuracy of Vision Transformers (**TORA-ViTs**) for utility and reliability at the same time. TORA-ViTs transfer naturally pre-trained models with low computational demands to improve their adversarial robustness while maintaining competitive natural accuracy. Based on the theory of robust non-predictive and non-robust predictive features, we add two kinds of adapter modules after the MLP layer of an existing ViT block, including an accuracy one $\psi_{A,l}$ and a robust one $\psi_{R,l}$ to extract predictive and robust features, respectively. Then, a gated fusion module (ϕ_l) is introduced to combine the extracted features in a trade-off-aware manner utilizing the attention mechanism. In the gated fusion module, features extracted by pre-trained ViT blocks are used as queries, and features extracted by the newly added accuracy and robustness adapters are used as keys and values. Then, the softmax function is applied adapter-wise as

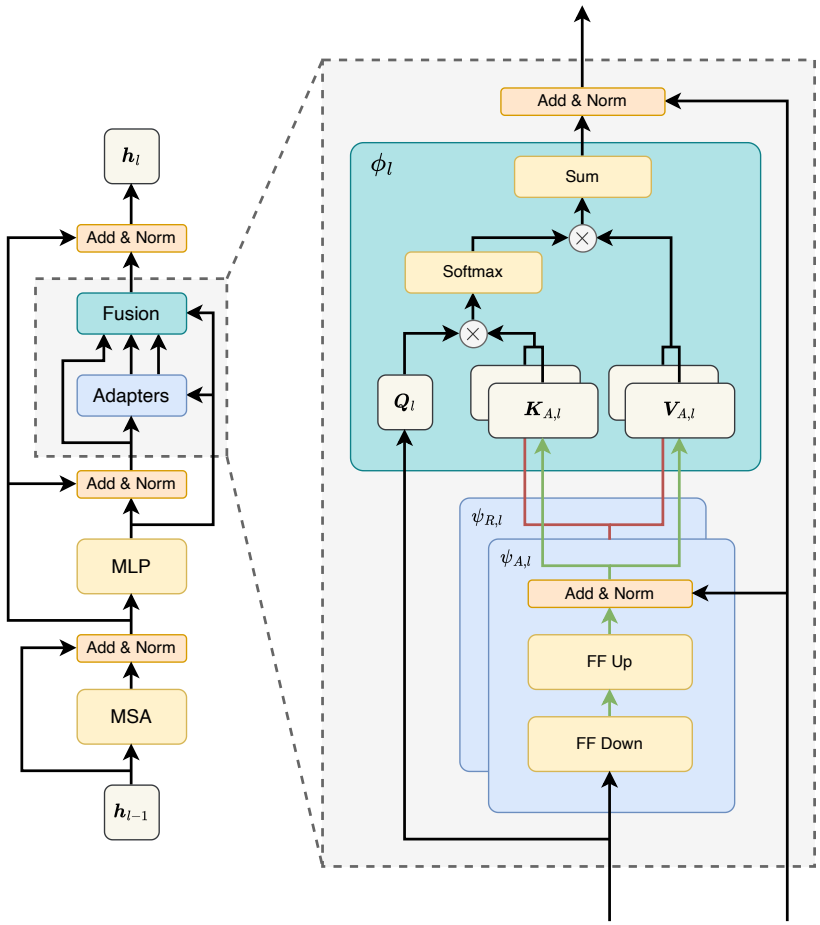


FIGURE 4.1. The overall architecture of our TORA-ViTs. The TORA-ViTs consist of two major components, including a pair of an accuracy adapter $\psi_{A,l}$ to extract predictive features and a robust adapter $\psi_{R,l}$ to extract robust features, and a gated fusion module to combine those features as inputs for next ViT block. The components are inserted after the MLP layer in each ViT block.

a gate to combine the two kinds of features. The overall architecture of our TORA-ViTs is shown in Fig. 4.1.

The TORA-ViTs are optimized in a two-phase manner. In the first phase, the accuracy and robustness adapters are optimized alternately along with the gated fusion module. When each of them reaches a proper performance, they are frozen, and the gated fusion module is optimized with a joint objective of accuracy and robustness with a trade-off ratio λ. Experiments on ImageNet with various robust benchmarks, including white-box adversarial

attacks (FGSM and PGD), natural adversarial example (ImageNet-A), out-of-distribution data (ImageNet-R), and common corruptions (ImageNet-C), show that our TORA-ViTs can efficiently improve the robustness of naturally pre-trained ViTs. Meanwhile, the natural accuracy is still competitive with or even better than the models pursuing accuracy. Our most balanced setting (TORA-ViTs with $\lambda = 0.5$) can maintain 83.7% accuracy on clean ImageNet and reach 54.7% and 38.0% accuracy under FGSM and PGD white-box attacks, respectively. In terms of various ImageNet variants, it can reach 39.2% and 56.3% accuracy on ImageNet-A and ImageNet-R and reach 34.4% mCE on ImageNet-C.

4.2 Methodology

4.2.1 Preliminary

Given an input image x and its relevant label y in the training set $\mathcal{D}$, a common supervised training objective of vision transformers can be written as

$$\mathcal{L}_{\text{ACC}}(f; \mathcal{D}) = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}[\ell\left(f(\boldsymbol{x}), y\right)], \tag{4.1}$$

where ℓ is a loss function such as the cross-entropy loss, and f stands for the vision transformer.

To improve the adversarial robustness of the model against perturbations on inputs, adversarial training is a common method, where perturbations are used to attack a target model, and the target model is optimized under such attacks. The perturbations are generated with gradient ascent to maximize the classification objective. An adversarial example $\boldsymbol{x}'$ with perturbations is typically limited in a l_p ball $\mathcal{B}_p(\boldsymbol{x}, \varepsilon) = \{\boldsymbol{x}' : \|\boldsymbol{x} - \boldsymbol{x}'\|_p \leq \varepsilon\}$ around its corresponding natural example $\boldsymbol{x}$, where ε defines the scale of allowed perturbations, and $\|\cdot\|_p$ is a l_p normalization. Then, adversarial training can be formed into a min-max problem, whose objective function is defined as

$$\mathcal{L}_{\text{ROB}}(f; \mathcal{D}) = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}\left[\max_{\boldsymbol{x}'\in\mathcal{B}_p(\boldsymbol{x},\varepsilon)} \ell\left(f(\boldsymbol{x}'), y\right)\right], \tag{4.2}$$

where f is a deep model, $\mathcal{D}$ is the distribution of the natural example $\boldsymbol{x}$ and the corresponding label y, and where ℓ is a loss function such as the cross-entropy loss.

4.2.2 Robustness and Accuracy Adapters

Off-the-shelf vision transformers (Dosovitskiy et al., 2020; Han et al., 2021; Yuan et al., 2021; Liu et al., 2021) are often well trained to pursue the natural accuracy through Eq. 4.1. To enhance the adversarial robustness of these well-trained vision transformers, a standard fine-tuning process can be executed by tuning the weights with the objective of Eq. 4.2. However, adversarial robustness as a new objective for vision transformers may lead to a completely different set of weights, which are far from the initialization and degrade the accuracy of vision transformers.

Taking both natural accuracy and adversarial robustness into consideration, we insert two adapters into an existing ViT block, including an accuracy adapter $\psi_{A,l}$ for predictive features and a robustness adapter $\psi_{R,l}$ for robust features. Given the feature $\boldsymbol{z}_l \in \mathbb{R}^{N\times d_{\mathrm{m}}}$ output by the MLP layer in the $1 \leq l \leq L$ block with N tokens of d_m dimensions, we have the accuracy adapter

$$\boldsymbol{a}_l = \psi_{A,l}(\boldsymbol{z}_l), \tag{4.3}$$

and the robust adapter

$$\boldsymbol{r}_l = \psi_{R,l}(\boldsymbol{z}_l), \tag{4.4}$$

where $\boldsymbol{a}_l, \boldsymbol{r}_l \in \mathbb{R}^{N\times d_{\mathrm{m}}}$ are the predictive and robust features, respectively.

For the architecture of adapters, we use two feed-forward layers with a bottleneck and a residual connection following Houlsby et al. (2019). We insert adapters right after the MLP layer of an existing ViT block and do not insert adapters after the multi-head attention (MSA). The overall architecture of a block in our TORA-ViTs is as shown in Fig. 4.1.

4.2.3 Attention-based Gated Fusion

To combine the predictive and robust features extracted by the accuracy and robustness adapters in a trade-off-aware manner, we propose an attention-based gated fusion module. We first calculate the dot-product attention score matrices between the features from the ViT blocks and adapters. Then, a softmax function is applied adapter-wise to the score matrices. The softmax results are used as a **weighted gate** to fuse the predictive and robust features.

The feature $\boldsymbol{z}_l$ output by the ViT block is used to generate the **query**, and the features $\boldsymbol{a}_l$ and $\boldsymbol{r}_l$ output by adapters are used to generate **keys**. The dot products between the two Q-K pairs are calculated as

$$\boldsymbol{s}_{A,l} = (\boldsymbol{z}_l \cdot \boldsymbol{w}_{Q,l}) \cdot (\boldsymbol{a}_l \cdot \boldsymbol{w}_{K,l})^\top, \tag{4.5}$$

$$\boldsymbol{s}_{R,l} = (\boldsymbol{z}_l \cdot \boldsymbol{w}_{Q,l}) \cdot (\boldsymbol{r}_l \cdot \boldsymbol{w}_{K,l})^\top, \tag{4.6}$$

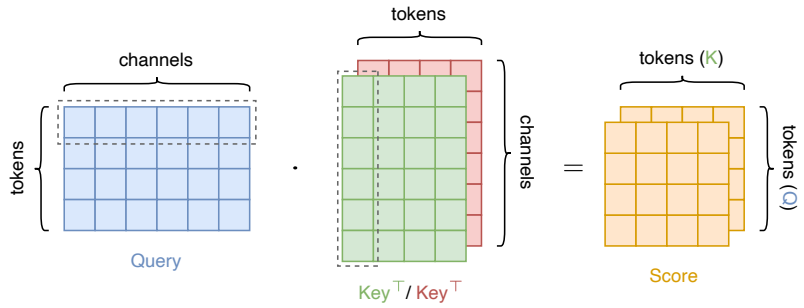
where $\boldsymbol{s}_{A,l}$, $\boldsymbol{s}_{R,l} \in \mathcal{R}^{N\times N}$ are the attention score matrices for the accuracy adapter and robustness adapter, and $\boldsymbol{w}_{Q,l}$, $\boldsymbol{w}_{K,l} \in \mathbb{R}^{d_\mathrm{m}\times d_\mathrm{q}}$ are projection parameters for query and key matrices. The projection parameters are shared among adapters.

Then, the $\mathrm{softmax}$ function is applied to the attention score matrices adapter-wise instead of token-wise by

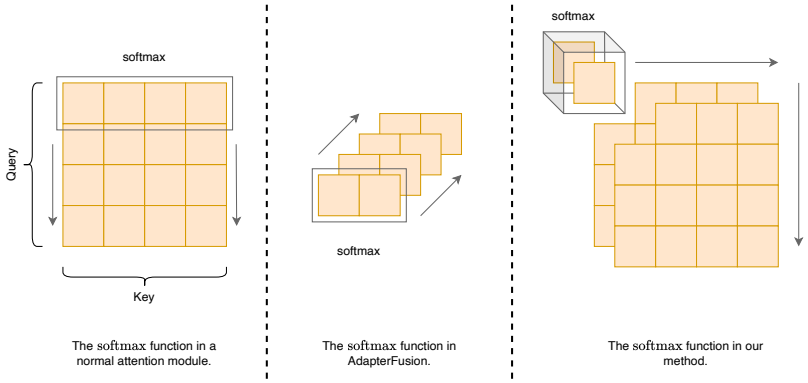
$$\boldsymbol{s}'_{A,l,m,n} = \frac{\exp(\boldsymbol{s}_{A,l,m,n})}{\sum_{k\in\{A,R\}} \exp(\boldsymbol{s}_{k,l,m,n})} \tag{4.7}$$

$$\boldsymbol{s}'_{R,l,m,n} = \frac{\exp(\boldsymbol{s}_{R,l,m,n})}{\sum_{k\in\{A,R\}} \exp(\boldsymbol{s}_{k,l,m,n})}. \tag{4.8}$$

In this manner, if a token in one adapter corresponds to a larger attention score than the other, it will be assigned a larger weight, and vice versa. It can act as a gate to select which kind of features can be forwarded to the next block at a larger scale. Similar to keys, the **values** are also generated from the features of adapters. By applying the weights, we can calculate the



(A) The dot products between query and keys.



(B) Comparison of various methods to apply the softmax function in the attention mechanism.

FIGURE 4.2. The dot-product attention and softmax function in our gated fusion module.

output of the attention module for each adapter as

$$\boldsymbol{o}_{A,l} = \boldsymbol{s}'^{\top}_{A,l} \cdot (\boldsymbol{a}_l \cdot \boldsymbol{w}_{V,l}) \tag{4.9}$$

$$\boldsymbol{o}_{R,l} = \boldsymbol{s}'^{\top}_{R,l} \cdot (\boldsymbol{r}_l \cdot \boldsymbol{w}_{V,l}), \tag{4.10}$$

where $\boldsymbol{o}_{A,l}, \boldsymbol{o}_{R,l} \in \mathbb{R}^{N \times d_m}$, and $\boldsymbol{w}_{V,l} \in \mathbb{R}^{d_\mathrm{m} \times d_\mathrm{v}}$ is the projection parameter for value matrices. The comparison of our method to other previous methods is shown in Fig. 4.2.

Because $\boldsymbol{o}_{A,l}$ and $\boldsymbol{o}_{R,l}$ has already been multiplied with weight matrices $\boldsymbol{s}'_{A,l}$ and $\boldsymbol{s}'_{R,l}$ from softmax function, we can directly sum them adapter-wise for the final output

$$\boldsymbol{o}_l = \sum_{k \in \{A,R\}} \boldsymbol{o}_{k,l}, \tag{4.11}$$

which ensures $\boldsymbol{o}_l$ to have same dimensions as $\boldsymbol{z}_l$. Finally, we add a residual connection from the output of the ViT block and have the final layer output $\boldsymbol{h}_l$ as

$$\boldsymbol{h}_l = \boldsymbol{z}_l + \mathrm{LN}(\boldsymbol{o}_l), \tag{4.12}$$

where $\mathrm{LN}(\cdot)$ is a layer norm.

4.2.4 Two-Phase Trade-off Training

Dosovitskiy et al. (2020) introduces an extra randomly initialized classification token [CLS] to the embedded patch tokens in ViTs following BERT. This token is later used for the classification task. Similarly, we add an accuracy token and a robustness token for our trade-off training. The original class token is at the first dimension of the output (i.e., [CLS] $:= \boldsymbol{z}_{l,1,:}$). To add our new tokens, we replace $\boldsymbol{z}_{l,1,:}$ with the accuracy token $[\mathrm{ACC}]_{l-1}$ and the robust token $[\mathrm{ROB}]_{l-1}$ to form the inputs to adapters. Then, Eqs. 4.3 and 4.4 become

$$\boldsymbol{a}_l = \psi_{A,l}\left(\mathrm{Concat}([\mathrm{ACC}]_{l-1}, \boldsymbol{z}_{l,2:,:})\right) \tag{4.13}$$

$$\boldsymbol{r}_l = \psi_{R,l}\left(\mathrm{Concat}([\mathrm{ROB}]_{l-1}, \boldsymbol{z}_{l,2:,:})\right). \tag{4.14}$$

After the adapter, we can have $[\mathrm{ACC}]_l = \boldsymbol{a}_{l,1,:}$ and $[\mathrm{ROB}]_l = \boldsymbol{r}_{l,1,:}$. To make the final classification, an accuracy classification head f_{ACC} and a robustness classification head f_{ROB} are added, and their predictions are averaged

$$\hat{y} = \frac{1}{2} f_{\mathrm{ACC}}([\mathrm{ACC}]_L) + \frac{1}{2} f_{\mathrm{ROB}}([\mathrm{ROB}]_L). \tag{4.15}$$

To optimize our TORA-ViTs, we use a two-phase training strategy. We first optimize each adapter independently with their specific objective. In this phase, the fusion module is also optimized. Then, the two adapters are frozen, and the fusion module is optimized with the joint robustness and accuracy objective. During the entire training process, the pre-trained ViT is always frozen. In the first phase, Eqs. 4.1 and 4.2 are used to optimize the corresponding

adapter along with the gated fusion

$$\min_{\Psi_R,\Phi} \quad \mathcal{L}_{\text{ROB}}(F;\mathcal{D}), \tag{4.16}$$

$$\min_{\Psi_A,\Phi} \quad \mathcal{L}_{\text{ACC}}(F;\mathcal{D}), \tag{4.17}$$

where $F = \{f, \Psi_R, \Psi_A, \Phi\}$ with $\Psi_R = \{\psi_{R,l} | 1 \leq l \leq L\}$, $\Psi_A = \{\psi_{A,l} | 1 \leq l \leq L\}$ and $\Phi = \{\phi_l | 1 \leq l \leq L\}$ represents the TORA-ViT model with adapters and gated fusion. In Eqs. 4.16 and 4.17, the trade-off ratio λ is temporarily omitted because each objective is optimized alternately. In the second phase, we use a joint objective to optimize Φ with λ

$$\min_{\Phi} \quad \lambda\mathcal{L}_{\text{ROB}}(F;\mathcal{D}) + (1-\lambda)\mathcal{L}_{\text{ACC}}(F;\mathcal{D}). \tag{4.18}$$

Because the fusion module Φ can be easily biased to the current object in the previous phase, this phase aims to adjust Φ with joint optimization and make the trade-off better correlated with the demand ratio λ.

4.3 Experiments

4.3.1 Settings

Pre-trained ViTs. We consider the vanilla ViT architecture proposed by Dosovitskiy et al. (2020) in our experiments. The ViT-B/16 with 224×224 input size, 16×16 patch size, 768-dimension embedding, and 12 layers are used. We initialize the network with pre-trained parameters provided by Steiner et al. (2021).

Training. The existing ViT blocks are frozen during training. The adapters are optimized with AdamW (Loshchilov and Hutter, 2019) optimizer on ImageNet-1K (Deng et al., 2009) with a 0.0001 initial learning rate and step decay with a rate of 0.97. For adversarial training, we use the single-step FGSM (Goodfellow et al., 2014) with $\varepsilon = 1/255$ to generate adversarial examples. The model is trained for 9 epochs in total. In the first 6 epochs, the alternate optimization in Eqs. 4.16 and 4.17 is performed, which means each objective is optimized for 3 epochs. In the last 3 epochs, the joint optimization in Eq. 4.18 is performed.

Evaluation. For white-box attacks, we use single-step FGSM (Goodfellow et al., 2014) and multi-step PGD (Madry et al., 2017) on ImageNet-1K. We follow Mao et al. (2022) and use $\varepsilon = 1/255$, PGD with 5 steps, and step size $0.5/255$. For natural adversarial examples, we use ImageNet-A (Hendrycks et al., 2021b), which places the ImageNet objects in unusual contexts or orientations. For out-of-distribution data, we use ImageNet-R (Hendrycks et al., 2021a), which contains abstract or rendered versions of the object. For common corruption, we use ImageNet-C (Hendrycks and Dietterich, 2019), which applies 19 common corruptions in 5 categories (e.g., motion blur, Gaussian noise, fog, JPEG compression, etc.).

4.3.2 Comparison to Baseline Methods

In Table 4.1, we compare our TORA-ViT (categorized as "robust adapters") to 4 categories of baseline methods, including naturally trained CNNs, robust CNNs, naturally trained ViTs, and robust ViTs. We report three different trade-offs of TORA-ViT in Table 4.1, including a most balanced setting, which outperforms all the baselines, a setting for good natural accuracy, and a setting for extremely high robustness against adversarial attacks. Other ratios are reported and discussed in more detail in Table 4.2.

Our method performs well on both clean and robust tasks with $\lambda = 0.5$. It outperforms previous works on all metrics. This is our **most balanced** setting. Compared to the previous best baseline methods, it improves natural accuracy on *clean data* by 0.3% than Swin-B; improves accuracy under *FGSM* by 1.7%, accuracy under *PGD* by 8.1% and accuracy on *ImageNet-R* by 7.6% than RVT-B*; improves accuracy on *ImageNet-A* by 2.8% than PyramidAT-384; and reduce mCE on *ImageNet-C* by 10.6% than PyramidAT.

The model mainly focuses on natural accuracy when $\lambda = 0.1$. Compared to naturally trained ViTs, it improves the natural accuracy by 0.7% compared to the previous best model, i.e., Swin-B. Besides, in terms of robustness, it also reaches better performance than Swin-B under PGD attack (2% higher accuracy) and on all ImageNet variants (10.7% and 11.0% higher accuracy on ImageNet-A and -R, respectively, and 22.7% lower mCE on ImageNet-C). It is only slightly lower than Swin-B under FGSM by 0.8%. Compared to robust ViTs, it is better

Categories	Models	Clean	Attacks		ImageNet Variants		
			FGSM	PGD	A	R	C(↓)
CNNs	ResNet-50 (He et al., 2016)	76.1	12.2	0.9	0.0	36.1	76.7
	ResNeXt50-32x4d (Xie et al., 2017)	79.8	34.7	13.5	10.7	41.5	64.7
	EfficientNet-B4 (Tan and Le, 2019)	83.0	**44.6**	**18.5**	26.3	47.1	71.1
	ConvNeXt-B (Liu et al., 2022b)	**83.8**	-	-	**36.7**	**51.3**	**46.8**
Robust CNNs	ANT (Rusak et al., 2020)	76.1	17.8	3.1	1.1	39.0	63.0
	AugMix (Hendrycks et al., 2019)	77.5	20.2	3.8	3.8	41.0	65.3
	Debiased CNN (Li et al., 2020c)	76.9	20.4	5.5	3.5	40.8	67.5
	DeepAugment (Hendrycks et al., 2021a)	75.8	27.1	9.5	3.9	**46.7**	**53.6**
	Anti-Aliased CNN (Zhang, 2019)	**79.3**	**32.9**	**13.5**	**8.2**	41.1	68.1
ViTs	ViT-B/16 (Dosovitskiy et al., 2020)	72.8	-	-	8.0	27.1	74.8
	ViT-B/16 + CutMix (Dosovitskiy et al., 2020)	75.5	-	-	14.8	28.5	64.1
	ViT-B/16 + MixUp (Dosovitskiy et al., 2020)	77.8	-	-	12.2	34.9	61.8
	ViT-B/16 + AugReg (Steiner et al., 2021)	79.9	-	-	17.5	38.2	52.5
	ViT-B/16-384 + AugReg (Steiner et al., 2021) †	81.4	-	-	26.2	38.2	58.2
	PVT-Large (Wang et al., 2021)	81.7	33.1	7.3	26.6	42.7	59.8
	ConViT-B (d'Ascoli et al., 2021)	82.4	45.4	20.8	29.0	**48.4**	**46.9**
	DeiT-B/16 (Touvron et al., 2021)	82.0	46.4	21.3	27.4	44.9	48.5
	T2T-ViT_t-24 (Yuan et al., 2021)	82.6	46.7	17.5	28.9	47.9	48.0
	Swin-B (Liu et al., 2021)	**83.4**	49.2	21.3	**35.8**	46.6	54.4
	PiT-B (Heo et al., 2021)	82.4	**49.3**	**23.7**	33.9	43.7	48.2
Robust ViTs	PyramidAT (Herrmann et al., 2022)	81.7	-	-	23.0	47.7	45.0
	PyramidAT-384 (Herrmann et al., 2022) †	83.3	-	-	**36.4**	46.7	47.8
	RVT-B (Mao et al., 2022)	82.5	52.3	27.4	27.7	48.2	47.3
	RVT-B* (Mao et al., 2022)	82.7	**53.0**	**29.9**	28.5	48.7	46.8
	MAE-ViT-B (He et al., 2022)	83.6	-	-	35.9	48.3	51.7
	FAN-L-ViT (Zhou et al., 2022)	**83.9**	-	-	34.2	**53.1**	**43.3**
Robust Adapters (ours)	TORA-ViT-B/16 ($\lambda = 0.1$)	**84.1**	48.4	23.3	**46.5**	**57.6**	**31.7**
	TORA-ViT-B/16 ($\lambda = 0.5$)	83.7	54.7	38.0	39.2	56.3	34.4
	TORA-ViT-B/16 ($\lambda = 0.9$)	80.3	**74.2**	**57.5**	22.2	53.7	41.6

TABLE 4.1. Performance on ImageNet-1K and variants. For performance on clean ImageNet-1K, under adversarial attacks, on ImageNet-A, and on ImageNet-R, the top-1 accuracy is reported. For performance on ImageNet-C, the mean Corruption Error (mCE) is reported, which is the smaller the better (marked by ↓).

†: "ViT-B/16-384 + AugReg" and "PyramidAT-384" use 384×384 inputs, and other models use 224×224 inputs.

than all of them in terms of natural accuracy and accuracy on ImageNet variants. Although the performance under adversarial attacks is lower than robust ViTs, considering this is a model trading robustness for accuracy, it is still remarkable to reach the best natural accuracy

and the best robustness on ImageNet variants among our settings, which also outperforms all the previous baseline methods, by sacrificing some robustness against adversarial attacks.

The adversarial robustness becomes the main target if we set $\lambda = 0.9$. This setting improves accuracy under FGSM by 21.2% and accuracy under PGD by 27.6% compared to the previously best RVT-B*, which is a surprisingly large improvement in robustness against adversarial attacks. This improvement sacrifices performance on clean data and ImageNet variants. Compared to the most balanced setting with $\lambda = 0.5$, its natural accuracy drops 3.4%, and its accuracy on ImageNet-A drops 17%. Although its robustness on ImageNet-R and -C is also the worst among our settings, it is still better than previous baseline methods.

Another interesting observation about our method is that *robustness on the three ImageNet variants have a positive correlation with natural accuracy and a negative correlation with robustness under adversarial attacks*. This phenomenon also exists for other baseline methods, although the correlations are not as strong as those in our method. For example, PiT reaches the best robustness against adversarial attacks among ViTs, but its performance under other kinds of perturbations is not the best; Anti-Aliased CNN reaches the best robustness against adversarial attacks among robust CNNs, but its robustness on ImageNet-R and -C is worse than DeepAugment. This also demonstrates the importance of controlling the trade-off when applying adversarial training because robustness to adversarial attacks is only one aspect of robustness, and the robustness to other kinds of perturbations is not always improved together with it.

4.3.3 Classification Heads and Trade-off Ratios

Our TORA-ViT uses two kinds of adapters and tokens to extract different features, and each token corresponds to a corresponding classification head. To decide on final predictions, we use their average outputs for joint prediction. To better understand the behaviors of the two kinds of heads, the performance of each head along with the joint prediction with different λ are reported in Table 4.2.

λ	Head	Clean	Attacks		ImageNet Variants		
			FGSM	PGD	A	R	C(↓)
0.1	Acc.	**84.15**	47.96	22.08	45.75	56.79	32.61
	Rob.	83.89	**48.54**	**24.89**	**46.33**	**57.38**	**31.89**
	Joint	*84.10*	*48.44*	*23.26*	*46.73*	*57.64*	*31.69*
0.3	Acc.	**83.79**	50.42	32.42	42.05	56.17	33.77
	Rob.	83.36	**53.73**	**35.62**	**42.32**	**56.49**	**33.19**
	Joint	*84.03*	*51.85*	*33.84*	*42.45*	*56.72*	*32.91*
0.5	Acc.	**83.38**	53.41	36.58	**38.93**	55.80	35.29
	Rob.	83.01	**56.19**	**39.78**	38.85	**56.12**	**34.73**
	Joint	*83.66*	*54.75*	*37.99*	*39.23*	*56.27*	*34.44*
0.7	Acc.	**80.80**	63.70	49.89	**23.64**	**54.09**	42.27
	Rob.	80.37	**67.37**	**52.23**	23.59	54.04	**42.13**
	Joint	*81.11*	*65.75*	*50.99*	*23.68*	*54.29*	*41.55*
0.9	Acc.	**80.66**	70.02	56.10	**22.69**	**53.64**	42.30
	Rob.	80.04	**74.24**	**58.34**	22.37	53.39	**42.11**
	Joint	*80.34*	*74.19*	*57.50*	*22.21*	*53.67*	*41.56*

TABLE 4.2. Performance of different heads and their joint prediction with different λ.

Firstly, the natural accuracy and robustness against adversarial attacks are well correlated to the trade-off ratio λ. Furthermore, the accuracy head and robustness head also perform consistently on these two kinds of metrics. To be specific, the accuracy head always outperforms the robustness head on clean data, and the robustness always outperforms the accuracy head under attacks.

However, the behaviors change under other perturbations. As aforementioned in Section 4.3.2, the robustness against other kinds of perturbations is not always positively correlated with robustness against adversarial attacks. When $\lambda < 0.5$, the robust head can still consistently outperform the accuracy head on all 5 kinds of perturbations. When $\lambda = 0.5$, the accuracy head outperforms the robust head on ImageNet-A (natural adversarial examples). When $\lambda > 0.5$, the accuracy head outperforms the robust head on ImageNet-A and ImageNet-R (out-of-distribution data). The robustness head consistently performs better than the accuracy head on ImageNet-C (common corruptions).

λ	Tuning	FLOPs (G)	Params (M)	GPU Hours	Clean	Attacks		ImageNet Variants		
						FGSM	PGD	A	R	C(↓)
0.1	Head only	17.6	88.1	15.55	80.2	41.1	15.5	22.1	42.0	56.9
	Single adapter	17.8	88.3	15.55	82.5	40.9	15.1	36.9	48.3	46.2
	AdapterFusion	24.9	111.2	19.63	82.2	46.2	22.6	36.4	52.2	35.5
	TORA-ViT	26.0	111.2	19.82	84.1	48.4	23.3	46.5	57.6	31.7
0.9	Head only	17.6	88.1	15.55	79.0	42.0	16.3	12.9	40.2	62.5
	Single adapter	17.8	88.3	15.55	72.3	53.1	30.1	3.1	21.4	78.7
	AdapterFusion	24.9	111.2	19.69	79.5	66.2	55.3	20.4	51.7	42.9
	TORA-ViT	26.0	111.2	19.83	80.3	74.2	57.5	22.2	53.7	41.6

TABLE 4.3. Comparison of different tuning methods.

Overall, we can conclude that when the natural accuracy is high, adversarial training indeed contributes more to the robustness against perturbations other than adversarial attacks. However, if the natural accuracy drops, the contribution of adversarial training to those kinds of robustness also reduces. Therefore, from this point of view, controlling the trade-off between robustness and accuracy is also crucial for the overall robustness against various kinds of perturbations.

4.3.4 Tuning Methods

As we design a new tuning method for ViTs, which considers the trade-off between robustness and accuracy via leveraging the robust non-predictive and predictive non-robust features, it is meaningful to compare our method with existing methods that are agnostic to this characteristic of features. In Table 4.3, we compare our TORA-ViT with three other tuning methods, including tuning a new classification head only, tuning a single new adapter for robustness, and tuning two new adapters with AdapterFusion. Our model has a similar number of FLOPs and parameters with AdapterFusion, and our method only requires 0.16 more GPU hours to train, which is approximately 10 minutes. Although the heads-only and single adapter tuning are very lightweight, their performance is not as good as our method and AdapterFusion.

In terms of performance, the weakest method is tuning a new head only. Although it is easy for the new head to maintain competitive accuracy, it is hard to improve its robustness. Because the entire model except the head is frozen, the extracted features cannot be changed. It is hard to train a robust classification head on top of non-robust features. When using a single adapter, it's hard to control the trade-off. For example, when λ, the natural accuracy of the single adapter drops dramatically to only 72.3%, which is the lowest among all the four methods, but its performance under adversarial attacks is only better than using a new head only. Besides, its performance on the three ImageNet variants is also poor. AdapterFusion is the strongest among the three baselines, but it only has attention at the adapter level, which is agnostic of the robust and predictive features. In contrast, our TORA-ViTs reach the best performance with the trade-off-aware patches-level attention, which can distinguish robust and predictive features. We will further demonstrate the ability of TORA-ViTs to distinguish the two kinds of features via visualization in Section 4.3.5.

4.3.5 Visualization of Attention Maps

We visualize attentions for different adapters in Fig. 4.3. We extract attention scores after the softmax in the gated fusion. They are $\mathbb{R}^{2\times N\times N}$ tensors, where the first dimension is 2 corresponding to 2 adapters, and the remaining dimensions are N corresponding to the number of tokens (including accuracy/robustness tokens and patch tokens). We average scores for each token to get a $\mathbb{R}^{2\times N}$ matrix. Average scores in all blocks are multiplied to get accumulated attention maps of the entire network.

We find the accuracy adapter focuses more on context, and the accuracy adapter focuses more on the object to be classified. When $\lambda = 0.1$ and the model focuses on accuracy, the features yielded by the accuracy adapter have higher attention scores, and the robustness adapter only has a few highlights in the attention maps. However, we can still see those highlights mainly fall in the region of the main object. When $\lambda = 0.9$ and the model focuses on robustness, the features yielded by the robustness adapter have higher attention scores, and those attentions overlap the main object. In this case, the accuracy adapter only has a little attention to the context. If we consider a more balanced trade-off ratio, i.e., $\lambda = 0.5$, we can

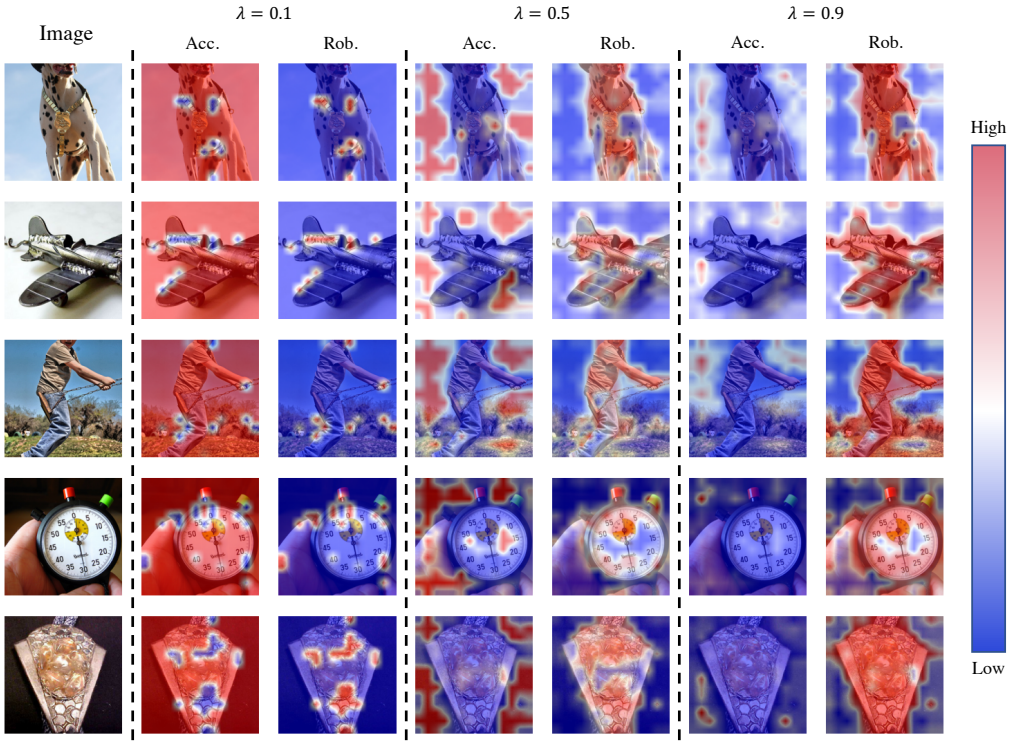


FIGURE 4.3. Visualization of the attentions for different adapters in the gated fusion module with various ratios λ. The blue-white-red color map is used, where **red** represents high attention, and **blue** represents low attention. As can be seen, the features yielded by the accuracy adapter focus more on **context**, and in contrast, the features yielded by the robustness adapter focus more on the main **object** to be classified. This is consistent with the theory of robust non-predictive and predictive non-robust features.

find this phenomenon is clearer. The magnitude of attention of the robustness adapter and the accuracy adapter are similar but are distributed in different regions; the accuracy adapter focuses more on the context, and the robustness adapter focuses more on the object.

If we also take into account Table 4.2, we can find that high attention to the context of the accuracy adapter won't reduce its accuracy. The accuracy on clean data of the accuracy adapter consistently outperforms the robustness adapter, even when $\lambda = 0.9$, and it only has a few attentions on the context. In contrast, we can find such focus on context makes it non-robust under adversarial attacks.

4.4 Chapter Summary

In this chapter, we propose the Trade-off between Robustness and Accuracy of Vision Transformers (**TORA-ViTs**). TORA-ViTs is inspired by the theory of predictive non-robust and robust non-predictive features. By introducing two different adapters, including an accuracy adapter and a robustness adapter, TORA-ViTs is able to extract both predictive and robust features. To combine the two kinds of features in a trade-off-aware manner, an attention-based gated fusion module is further proposed. It takes the outputs of ViT blocks as queries and utilizes the attention mechanism to combine features. Experiments on ImageNet with various robust benchmarks demonstrate that our TORA-ViTs can efficiently improve the robustness of naturally pre-trained ViTs while maintaining competitive natural accuracy. Visualization of the attention map in the gated fusion module empirically proves the theory of robust non-predictive features and predictive non-robust features.

CHAPTER 5

Harnessing Edge Information with the EdgeNet

5.1 Motivation

Previous research (Geirhos et al., 2018; Li and Xu, 2023) suggests that the vulnerability of high-accuracy DNNs to perturbations is rooted in their heavy reliance on *irrelevant and non-robust features* such as textures and backgrounds. In contrast, robust DNNs should instead base their predictions on *relevant and robust features* that pertain to shapes and foreground elements within the images. However, Tsipras et al. (2018) point out that these moderately correlated features, while robust, can adversely affect accurate predictions, making them both robust and non-predictive. Conversely, the key to improving natural accuracy lies in utilizing weakly correlated and non-robust features, which, despite lacking adversarial robustness, exhibit predictive capability. Therefore, improving the adversarial robustness of a DNN without compromising its natural accuracy is challenging.

Based on the aforementioned theorem and the existence of high-accuracy, large-scale pre-trained models, the enhancement of adversarial robustness in naturally pre-trained models has emerged as a prominent subject. Recently, Li and Xu (2023) leverage the capabilities of a fine-tuning technique known as Adapter (Houlsby et al., 2019), effectively enhancing adversarial robustness with an affordable training cost. However, a drawback is also obvious. Through the incorporation of a fusion module to balance predictive and robust features, their model requires tuning a hyper-parameter to manage a trade-off. In certain scenarios, natural accuracy is compromised to enhance robustness and vice versa.

In this chapter, we present an alternative approach wherein, rather than directly augmenting parameters to the backbone network, we introduce a mechanism for integrating specific information extracted from the original images into the intermediate layers of the backbone network. To be specific, our novel approach highlights the potential of edge information extracted from images. This edge information holds the capability to furnish relevant and robust features pertaining to shapes and foreground elements within the images. These features, when integrated, assist pre-trained DNNs in achieving improved adversarial robustness without compromising their natural accuracy in classifying clean images.

To achieve this objective, we propose the incorporation of a side branch named **EdgeNet**. This lightweight, plug-and-play network can seamlessly integrate into existing pre-trained deep models, including state-of-the-art models such as Vision Transformers (ViTs) (Dosovitskiy et al., 2020). Our EdgeNet operates by processing edge information extracted from input images. This process yields a set of robust features that can be strategically injected into the intermediary layers of the frozen backbone DNNs. This augmentation empowers the network to boost its defenses against adversarial perturbations while sustaining its accuracy in recognizing unaltered clean images.

The building blocks feature a "sandwich" architecture, comprising two zero convolutions (Zhang and Agrawala, 2023) at both the input and output, sandwiching a ViT block in the middle. These two zero convolutions selectively transmit relevant inputs to the intermediate block and inject relevant outputs into the pre-trained backbone. Furthermore, the zero convolution at the output position ensures that the information injected into the backbone initiates from a zero point, thereby ensuring the stability and trainability of our method.

Our approach incurs minimal additional computational overhead, comparable to using Adapters for fine-tuning ViT. Firstly, obtaining edge information through conventional edge detection algorithms, such as the well-known Canny edge detector (Canny, 1986), incurs only marginal computational costs compared to DNNs. Furthermore, our EdgeNet-ViT-B/16, which incorporates 4 new blocks, is composed of 119.9M parameters and involves 24.37G floating-point operations (FLOPs). In contrast, the TORA-ViT-B/16 model, relying on Adapters, consists of 111.2M parameters and requires 26.0G FLOPs (Li and Xu, 2023).

Our approach achieves reduced computational overhead while slightly increasing memory consumption. This affordability, combined with its effectiveness, positions EdgeNet as a compelling tool for enhancing DNN robustness in a resource-efficient manner.

Our experiments cover a wide range of robust benchmarks, including white-box and black-box adversarial attacks on ImageNet-1K, employing FGSM (Goodfellow et al., 2014) and PGD (Madry et al., 2017). The robustness of our EdgeNet extends beyond adversarial attacks to encompass scenarios that involve natural adversarial examples in ImageNet-A (Hendrycks et al., 2021b), out-of-distribution data in ImageNet-R (Hendrycks and Dietterich, 2019), and common corruptions in ImageNet-C (Hendrycks et al., 2021a). In particular, our EdgeNet demonstrates slightly superior or comparable performance to the most balanced configuration ($\lambda = 0.5$) of TORA-ViTs across clean ImageNet-1K and ImageNet-A/R/C datasets. Furthermore, it achieves significantly improved accuracy under FGSM and PGD attacks (69.8% compared to 54.7% and 48.8% compared to 38.0%, respectively). The results reveal that our EdgeNet effectively enhances the robustness of pre-trained ViTs.

5.2 Methodology

Firstly, we provide a brief overview of adversarial training as a preliminary. Then, we illustrate the integration of edge information into the backbone and introduce the architecture of building blocks in our EdgeNet. Finally, we provide the necessary details of the edge detection algorithm and establish our joint optimization objective.

5.2.1 Preliminary: Adversarial Training

The common method for achieving robustness of a DNN $\hat{y} = f(\boldsymbol{x})$ against adversarial attacks is adversarial training. This method involves formulating the training objective in a minimax form, wherein the goal is to minimize the loss to discover the optimal model while concurrently maximizing the loss to identify the optimal adversarial examples

$$f^* := \arg\min_f \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}} \left[\max_{\boldsymbol{x}'\in\mathcal{B}_p(\boldsymbol{x},\varepsilon)} \ell\left(f(\boldsymbol{x}'), y\right) \right], \tag{5.1}$$

where f^* is the robust model resulting from adversarial training, $\boldsymbol{x}$ and y represent images and labels sampled from a training distribution $\mathcal{D}$, and $\mathcal{B}_p(\boldsymbol{x}, \varepsilon) = \{\boldsymbol{x}' : \|\boldsymbol{x} - \boldsymbol{x}'\|_p \leq \varepsilon\}$ defines a ball covering all allowed adversarial examples $\boldsymbol{x}'$, with the clean image $\boldsymbol{x}$ as its center, the allowed magnitude of perturbation ε as its radius, and the l_p-norm serving as a measure of distance.

5.2.2 Integration of Edge Information

In Eq. 5.1, the model $f(\cdot)$ solely considers the images $\boldsymbol{x}$ (for clean examples) or $\boldsymbol{x}'$ (when subjected to an attack). We propose to integrate edge information into the model, enhancing the performance of the model

$$\hat{\boldsymbol{y}} = f(\boldsymbol{x}, \boldsymbol{e}), \tag{5.2}$$

where $\boldsymbol{e}$ is the edge obtained by $\boldsymbol{e} = \mathrm{Edge}(\boldsymbol{x})$, and $\mathrm{Edge}(\cdot)$ is an edge detection algorithm, such as the Canny edge detector.

We start with a backbone network composed of L building blocks as expressed in the following equation

$$f_b = \left\{ f_b^{(l)}, l = 1, \ldots, L \right\}, \tag{5.3}$$

where each building block $f_b^{(l)} : \boldsymbol{h}_{l-1} \mapsto \boldsymbol{h}_l$ maps the previous layer's representation $\boldsymbol{h}_{l-1}$ to the subsequent one $\boldsymbol{h}_l$. We enhance the backbone architecture by introducing an additional set of L_e blocks capable of processing edge information, which we refer to as **EdgeNet**

$$f_e = \left\{ f_e^{(l)}, l \in 1, \ldots, L_e \right\}, \tag{5.4}$$

where each building block $f_e^{(l)} : \boldsymbol{e}_{l-1} \mapsto \boldsymbol{e}_l$ are mappings similar to $f_b^{(l)}$ but deals with edge-related features $\boldsymbol{e}_l$.

In order to control the scale of the new blocks, we introduce a hyper-parameter N, which determines the insertion interval for incorporating these additional blocks. This is achieved through the relationship $L_e = L/N$. More specifically, when considering each building block

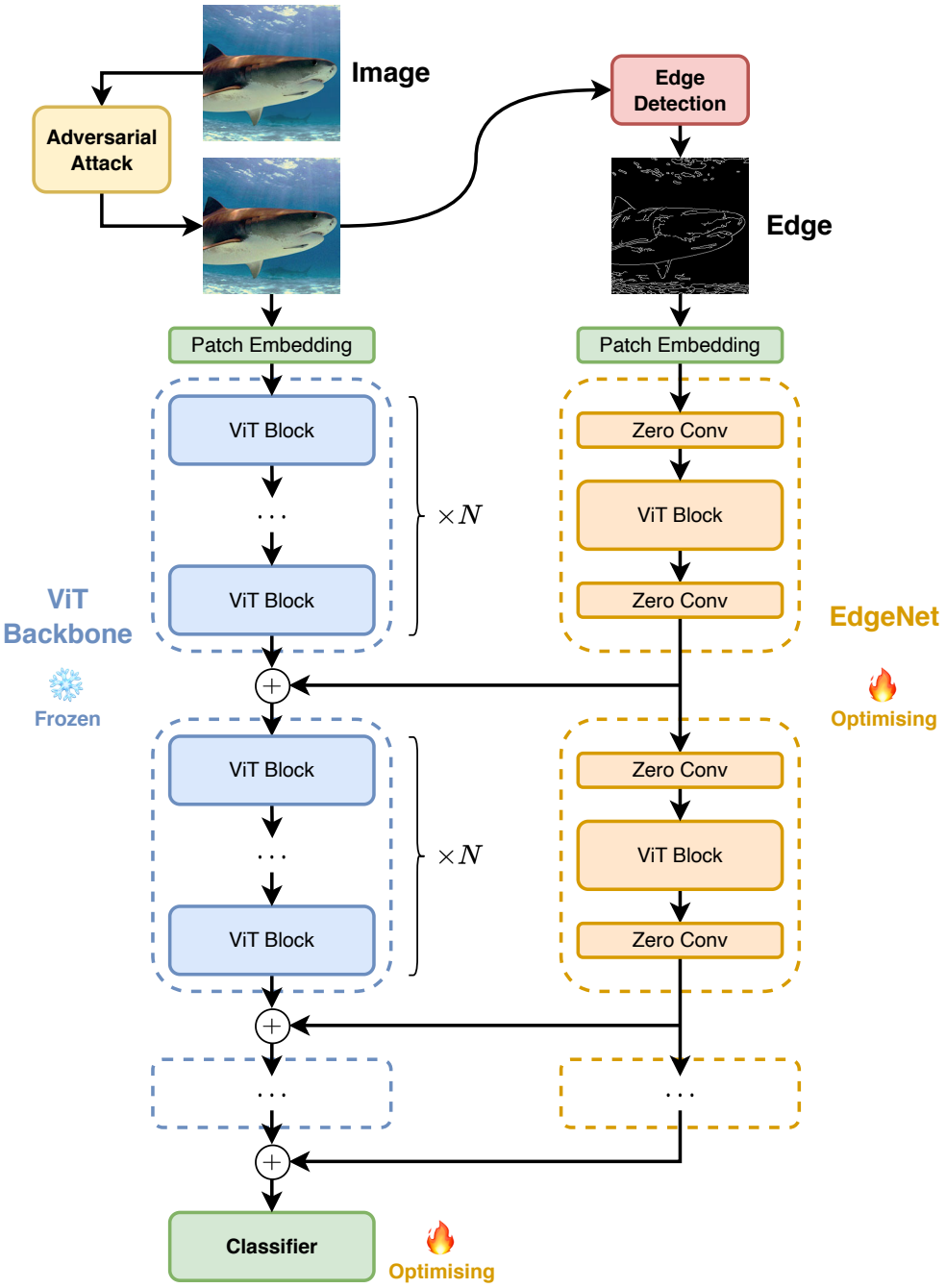


FIGURE 5.1. The architecture of our EdgeNet with ViT as the backbone. We employ an interval of N, signifying the addition of one EdgeNet block for every $N\times$ ViT blocks. Each EdgeNet block features a "sandwich" architecture, commencing with zero convolutions at both the input and output to initialize them with zeros. The output of each EdgeNet block is integrated into the intermediate layer of the ViT backbone through element-wise addition. Throughout the optimization process, the backbone remains frozen while the EdgeNet and classification head undergo training.

indexed by $l = 1, \ldots, L$, we have

$$\begin{cases} \boldsymbol{h}'_l = \boldsymbol{h}_l + \boldsymbol{e}_{l/N} & \text{if } l \bmod N \text{ is } 0, \\ \boldsymbol{h}'_l = \boldsymbol{h}_l & \text{otherwise.} \end{cases} \tag{5.5}$$

$\boldsymbol{h}'_l$ is then used as the input to the $l+1$ block $f_b^{(l+1)}$. Fig. 5.1 demonstrates the overall architecture of this framework.

5.2.3 EdgeNet Building Blocks

We implement a "sandwich" architecture for each building block in our EdgeNet framework, as depicted in Fig. 5.1. To be specific, we add zero convolutions $\mathcal{Z}(\cdot)$ (Zhang and Agrawala, 2023) to both the input and output of each block. Nestled between the two zero convolutions, we place a ViT block $\mathcal{T}(\cdot)$ with randomized initialization, maintaining the same architecture as those found in the backbone

$$\boldsymbol{e}_l = \mathcal{Z}_{\mathbf{out}}^{(l)}\left(\mathcal{T}^{(l)}\left(\mathcal{Z}_{\mathbf{in}}^{(l)}\left(\boldsymbol{e}_{l-1}\right)\right)\right). \tag{5.6}$$

Zero convolutions are defined as 1×1 convolution layer with both weight and bias initialized with zeros. Therefore, the input to the intermediate ViT block and the output of the EdgeNet building block are both start with zero.

Utilizing zero inputs, $\mathcal{Z}_{\mathbf{in}}^{(l)}(\cdot)$ functions as a filter for extracting information related to the optimization objective. Employing zero outputs, $\mathcal{Z}_{\mathbf{out}}^{(l)}(\cdot)$ functions as a filter for determining information to be integrated into the backbone. Furthermore, the addition of zeros to the backbone at the beginning ensures that the information flow within the backbone remains unaffected. Consequently, the subsequent fine-tuning of EdgeNet is significantly streamlined.

5.2.4 Edge Detection

We utilize the Canny edge detector (Canny, 1986) for edge detection. Firstly, the image is processed with a Gaussian filter to reduce noise and smooth the intensity variations. Subsequently, the gradient magnitude and direction are computed using convolution with

Sobel filters. The gradient direction helps determine the orientation of the edges. Non-maximum suppression is then applied to thin out the edges by retaining only the local maxima in the gradient magnitude along the gradient direction. Finally, a double thresholding step categorizes the edge pixels as strong, weak, or non-edges. Strong edges are retained, while weak edges are subjected to connectivity analysis to determine if they should be preserved.

Within the double thresholding phase, we employ the following equations to automatically determine the lower and upper thresholds

$$\text{lower} = \max(0, 0.7 \times \text{median\_value}), \tag{5.7}$$

$$\text{upper} = \min(255, 1.3 \times \text{median\_value}), \tag{5.8}$$

where median_value is the median value of pixels obtained from the previous step.

5.2.5 Joint Optimization

During the training process, the pre-existing ViT blocks and the patch embedding layer within the backbone remain fixed and undergo no updates. The optimization objective solely focuses on the new ViT blocks and patch embedding layer introduced for edge features, in addition to the classification head within the backbone.

Considering that our primary focus is not directed toward balancing the trade-off between accuracy and robustness, we adopt a simplified joint optimization objective

$$\min_{f} \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}\Big[\alpha \cdot \ell\left(f(\boldsymbol{x}, \text{Edge}(\boldsymbol{x})), y\right) + \beta \cdot \max_{\boldsymbol{x}'\in\mathcal{B}_p(\boldsymbol{x},\varepsilon)} \ell\left(f(\boldsymbol{x}', \text{Edge}(\boldsymbol{x}')), y\right)\Big], \tag{5.9}$$

where α is the weight for accuracy and β is the weight for robustness. The cross-entropy loss is used for $\ell(\cdot, \cdot)$. Through the adjusting of α and β, we can fine-tune our EdgeNet in a manner that enhances its robustness, meanwhile ensuring that the accuracy won't drop significantly.

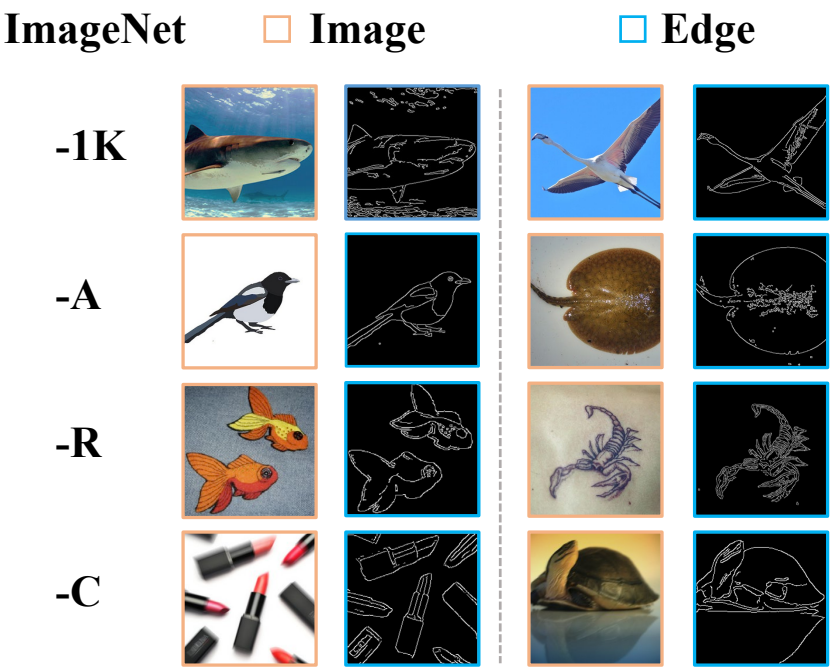


FIGURE 5.2. Instances selected from ImageNet-1K, -A, -R, and -C, accompanied by their respective edges extracted by the Canny edge detector.

5.3 Experiments

5.3.1 Settings

Pre-trained ViTs. In our experiments, we adopt the vanilla ViT architecture introduced by Dosovitskiy et al. (2020). In our specific approach, we employ the ViT-B/16 variant, which is characterized by several key parameters. This variant encompasses an input size of 224×224 pixels, with each image divided into patches of dimensions 16×16. The embedding dimension is set at 768, and the architecture is comprised of a total of 12 blocks. To initialize the network, we employ pre-trained parameters made available by Steiner et al. (2021).

Training. For the joint training objective in Eq. 5.9, we set the hyper-parameters $\alpha = 1.2$ and $\beta = 0.8$. We use FGSM with l_∞-norm for adversarial training and adopt a perturbation magnitude of $\varepsilon = 1/255$. We use the SGD optimizer, with a fixed learning rate of 1×10^{-4}, a momentum of 0.9, and a weight decay of 2×10^{-5}.

Evaluation. Our evaluations cover 5 distinct settings.

1. We initiate our analysis by addressing ***white-box attacks***. To investigate the robustness of our model, we employ both single-step FGSM (Goodfellow et al., 2014) and multi-step PGD (Madry et al., 2017) on the ImageNet-1K dataset. Consistent with Mao et al. (2022), we adopt a l_∞-norm and a perturbation magnitude of $\varepsilon = 1/255$ for both FGSM and PGD. For PGD, we execute it for 5 steps, using a step size of $0.5/255$.
2. Moving on, we delve into the realm of ***black-box attacks***. Initially, the ViT backbone is used to generate adversarial perturbations to attack our EdgeNet-ViT. Subsequently, we employ a ResNet-50 model to generate adversarial perturbations to attack both the ViT backbone and our EdgeNet-ViT.

Expanding the scope beyond adversarial attacks, we extend our evaluations to assess the robustness of our EdgeNet-ViT in broader scenarios.

3. In the domain of ***natural adversarial examples***, we use the ImageNet-A dataset (Hendrycks et al., 2021b). This dataset places the ImageNet objects in unusual contexts or orientations, challenging the model's adaptability to unconventional scenarios.
4. In the domain of ***out-of-distribution data***, we use the ImageNet-R dataset (Hendrycks et al., 2021a). This dataset contains abstract or rendered versions of objects, probing the model's capacity to generalize beyond its trained data distribution.
5. In the domain of ***common corruptions***, we use the ImageNet-C dataset (Hendrycks and Dietterich, 2019), which applies 19 common corruptions categorized into 5 groups (e.g., motion blur, Gaussian noise, fog, JPEG compression, etc.), mimicking real-world distortions that a model might encounter.

Illustrations of samples sourced from ImageNet-1K, -A, -R, and -C, along with their corresponding edges extracted by the Canny edge detector, are presented in Fig. 5.2.

# Intervals	# New Blocks	FLOPs (G)	Params (M)	Throughput (Images/Sec)	Clean	Attacks		ImageNet Variants		
						FGSM	PGD	A	R	C (↓)
1	12	37.88	186.14	375.16	83.4	69.0	48.0	39.5	56.8	**34.3**
3	4	24.37	119.99	543.40	**83.7**	**69.8**	**48.8**	**39.6**	56.9	34.4
6	2	21.00	103.45	601.64	83.3	66.8	46.3	37.6	**57.2**	35.0
-	0	17.60	88.1	635.81	80.2	41.1	15.5	22.1	42.0	56.9

TABLE 5.1. The performance of EdgeNet across varying scales. The "# Intervals" determines the frequency of adding a new block in relation to existing ones, while "# New Blocks" denotes the total number of added blocks. We also include results achieved by fine-tuning the classification head of the backbone for comparison (the last row).

5.3.2 Different Scales of EdgeNet

As we introduce an interval hyper-parameter, we manipulate its value to adjust the scale of EdgeNet. We present the performance of EdgeNet across different scales on the aforementioned benchmarks alongside reporting metrics such as the count of floating-point operations (FLOPs), the number of parameters, and the inference throughput (measured in images per second). We assess the throughput using a single NVIDIA RTX4090 GPU with 24GB of memory. As we maintain the backbone blocks in a frozen state and solely optimize our newly introduced blocks while fine-tuning the classification head, we establish a baseline by including the ViT-B/16 backbone. In this baseline, no new blocks are added, but the classification head is fine-tuned. The results are reported in Table 5.1.

When incorporating a total of 12 new blocks into the model, a substantial increase in computational overhead is observed. Additionally, the convergence of the model becomes challenging under these circumstances. While the inclusion of a larger number of new blocks results in improved performance compared to inserting only 2 new blocks, it falls short in performance when compared to the outcome of inserting 4 new blocks. In contrast, introducing 4 new blocks emerges as the most optimal configuration for EdgeNet, yielding its peak performance. This configuration does exhibit a slightly elevated computational overhead, yet it retains a commendable throughput, albeit slightly lower than the setup with only 2 new blocks (approximately 58.24 images/second lower). When incorporating a mere 2 new blocks, the achieved enhancement is not as pronounced as what is observed when inserting 12 or 4 new blocks.

However, this configuration still outperforms the scenario of fine-tuning the classification head in isolation.

Taking into account both classification performance and computational considerations, we identify the configuration with # Intervals = 3 as the optimal setting. In this configuration, EdgeNet achieves significantly improved clean accuracy and robustness compared to the baseline, albeit at the expense of approximately 14.5% reduction in throughput. It strikes a balanced compromise between classification performance, computational requirements, and robustness. This configuration demonstrates substantial gains in clean accuracy and robustness over the baseline while maintaining a reasonable trade-off in terms of computational efficiency.

5.3.3 Comparison to Baseline Methods

Table 5.2 presents a comprehensive comparison between our proposed EdgeNet and 5 distinct categories of baseline methods. These categories encompass naturally trained and robust CNNs, naturally trained and robust ViTs, along with robust fine-tuned ViTs evaluated across various benchmarks. The reported metrics include accuracy under adversarial attacks (FGSM and PGD), on ImageNet-A, and on ImageNet-R. Additionally, the mean Corruption Error (mCE) is reported for ImageNet-C, with lower values indicating better performance. As can be seen, our method showcases superior performance when subjected to both FGSM and PGD attacks. Meanwhile, our approach attains similar levels of performance on the clean ImageNet-1K dataset and its variants when compared to baseline methods from previous research.

We commence by comparing our EdgeNet with the robust fine-tuning method. When compared to the most balanced setting of TORA-ViT-B/16, indicated by $\lambda = 0.5$, we observe remarkable enhancements in accuracy under FGSM and PGD attacks, registering improvements of 15.1% and 10.8%, respectively. This performance augmentation is achieved while maintaining the same level of clean accuracy (83.7%). Furthermore, when considering ImageNet variants, our EdgeNet exhibits accuracy gains of 0.4% for ImageNet-A and 0.6% for ImageNet-R while consistently preserving the identical mCE for ImageNet-C. When

Categories	Models	Clean	Attacks		ImageNet Variants		
			FGSM	PGD	A	R	C (↓)
CNNs	ResNet-50 (He et al., 2016)	76.1	12.2	0.9	0.0	36.1	76.7
	ResNeXt50-32x4d (Xie et al., 2017)	79.8	34.7	13.5	10.7	41.5	64.7
	EfficientNet-B4 (Tan and Le, 2019)	83.0	44.6	18.5	26.3	47.1	71.1
	ConvNeXt-B (Liu et al., 2022b)	83.8	-	-	36.7	51.3	46.8
Robust CNNs	ANT (Rusak et al., 2020)	76.1	17.8	3.1	1.1	39.0	63.0
	AugMix (Hendrycks et al., 2019)	77.5	20.2	3.8	3.8	41.0	65.3
	Debiased CNN (Li et al., 2020c)	76.9	20.4	5.5	3.5	40.8	67.5
	DeepAugment (Hendrycks et al., 2021a)	75.8	27.1	9.5	3.9	46.7	53.6
	Anti-Aliased CNN (Zhang, 2019)	79.3	32.9	13.5	8.2	41.1	68.1
ViTs	ViT-B/16 (Dosovitskiy et al., 2020)	72.8	-	-	8.0	27.1	74.8
	ViT-B/16 + CutMix (Dosovitskiy et al., 2020)	75.5	-	-	14.8	28.5	64.1
	ViT-B/16 + MixUp (Dosovitskiy et al., 2020)	77.8	-	-	12.2	34.9	61.8
	ViT-B/16 + AugReg (Steiner et al., 2021)	79.9	-	-	17.5	38.2	52.5
	PVT-Large (Wang et al., 2021)	81.7	33.1	7.3	26.6	42.7	59.8
	ConViT-B (d'Ascoli et al., 2021)	82.4	45.4	20.8	29.0	48.4	46.9
	DeiT-B/16 (Touvron et al., 2021)	82.0	46.4	21.3	27.4	44.9	48.5
	T2T-ViT_t-24 (Yuan et al., 2021)	82.6	46.7	17.5	28.9	47.9	48.0
	Swin-B (Liu et al., 2021)	83.4	49.2	21.3	35.8	46.6	54.4
	PiT-B (Heo et al., 2021)	82.4	49.3	23.7	33.9	43.7	48.2
Robust ViTs	PyramidAT (Herrmann et al., 2022)	81.7	-	-	23.0	47.7	45.0
	PyramidAT-384 (Herrmann et al., 2022)	83.3	-	-	36.4	46.7	47.8
	RVT-B (Mao et al., 2022)	82.5	52.3	27.4	27.7	48.2	47.3
	RVT-B* (Mao et al., 2022)	82.7	53.0	29.9	28.5	48.7	46.8
	MAE-ViT-B (He et al., 2022)	83.6	-	-	35.9	48.3	51.7
	FAN-L-ViT (Zhou et al., 2022)	83.9	-	-	34.2	53.1	43.3
Robust Fine-tuning	TORA-ViT-B/16 ($\lambda = 0.1$) (Li and Xu, 2023)	84.1	48.4	23.3	46.5	57.6	31.7
	TORA-ViT-B/16 ($\lambda = 0.5$) (Li and Xu, 2023)	83.7	54.7	38.0	39.2	56.3	34.4
	TORA-ViT-B/16 ($\lambda = 0.9$) (Li and Xu, 2023)	80.3	74.2	57.5	22.2	53.7	41.6
	EdgeNet-ViT-B/16 **(Ours)**	83.7	69.8	48.8	39.6	56.9	34.4

TABLE 5.2. Evaluation of baseline methods on ImageNet-1K and its variants (A, R, and C). The top-1 accuracy is used to assess performance on clean ImageNet-1K, under adversarial attacks (FGSM and PGD), on ImageNet-A, and -R. In the case of ImageNet-C, the focus is on the mean Corruption Error (mCE), where lower values indicate better performance (marked by ↓). "ViT-B/16-384 + AugReg" and "PyramidAT-384" employ input dimensions of 384×384 inputs, while the remaining models utilize input dimensions of 224×224.

compared to TORA-ViT-B/16 with $\lambda = 0.1$, we have slightly lower clean accuracy (0.4%). This is because this model is fine-tuned for better performance on natural images. Therefore, our improvements in terms of adversarial robustness are even larger. We improve accuracy under FGSM and PGD attacks by 21.4% and 25.5%. We also have slightly lower performance

on ImageNet variants because they find their performance on ImageNet variants is correlated to clean accuracy instead of adversarial robustness.

In comparison to TORA-ViT-B/16 employing $\lambda = 0.1$, our clean accuracy exhibits a minor decrease of 0.4%. This diminishment can be attributed to the fact that this version of TORA has been fine-tuned for optimized performance on natural images. Consequently, our pronounced advancements in terms of adversarial robustness are even more notable. Under FGSM and PGD attacks, our approach displays substantial improvements, improving accuracy by 21.4% and 25.5%, respectively. Additionally, our performance is slightly lower than theirs when assessed on ImageNet variants. This can be attributed to the observation that their performance on ImageNet variants is closely associated with clean accuracy rather than adversarial robustness.

In the final setting of TORA-ViT-B/16, denoted by $\lambda = 0.9$, which is their most robust setting. Although their accuracy against FGSM and PGD attacks sees an increase of 4.4% and 8.7%, respectively, this progress comes at the expense of a 3.4% reduction in clean accuracy. Additionally, in comparison to our approach, their performance on ImageNet variants experiences relative drops of 17.4%, 3.2%, and 7.2%. Finally, we would like to emphasize once again that TORA controls a trade-off by introducing a specialized module into the backbone network to control the balance between robust features and predictive features. In contrast, our method aims to enhance robustness by introducing edge information without altering the backbone network itself. Therefore, in a fair comparison against their most balanced setting ($\lambda = 0.5$), our improvements are even more significant. However, even when compared to their favorably biased models, it is evident that our performance gap in their advantageous metrics is minimal, while our enhancements are more pronounced in their weaker aspects. In summary, our approach represents a more comprehensive, unbiased, and balanced model.

In addition to the robust fine-tuning, our EdgeNet outperforms all the other previous approaches under adversarial attacks and on the ImageNet variants. In terms of clean performance, our performance is only slightly lower than ConvNext-B4 and FAN-L-ViT for 0.1% and

Source Model	Defense Model	Valid Acc. (%)	
		FGSM	PGD
ViT-B/16	ViT-B/16	35.03	14.26
ViT-B/16	EdgeNet-ViT-B/16	74.41	70.32
ViT-S/16	ViT-B/16	74.09	75.59
ViT-S/16	EdgeNet-ViT-B/16	79.34	80.09
ViT-L/16	ViT-B/16	78.31	77.29
ViT-L/16	EdgeNet-ViT-B/16	80.62	80.18
Swin-B	ViT-B/16	82.94	82.40
Swin-B	EdgeNet-ViT-B/16	83.24	82.96

TABLE 5.3. The validation accuracy under black-box attacks on ImageNet-1K. Using ViT-B/16 as both the source model and defense model is equivalent to a white-box attack, which is included here solely for the purpose of comparison.

0.2%, respectively. These differences are very marginal. Furthermore, our clean performance surpasses that of other previous methods.

5.3.4 Black-box Attacks

In the previous experiments, white-box attacks are investigated, involving scenarios where the attacker possesses access to the parameters of target models. In Table 5.3, we extend our analysis to a more realistic black-box attack scenario, where the assumption is made that the attacker lacks access to the parameters of the target models. We consider various models as the source model for generating adversarial perturbations. These models encompass the backbone ViT-B/16, as well as two of its size variants, namely ViT-S/16 (a smaller version) and ViT-L/16 (a larger version). Furthermore, we include another Vision Transformer architecture known as Swin-B in our considerations.

Initially, we consider attacks using ViT-B/16, the backbone itself, as the source model. The results show that when EdgeNet is incorporated as an additional component, attacks originating from the backbone no longer successfully compromise our model, increasing the classification accuracy from 35.03% to 74.41% under FGSM and from 14.26% to 70.32% under PGD respectively.

Input	Clean	Attacks		ImageNet Variants		
		FGSM	**PGD**	**A**	**R**	**C** (↓)
Image	82.7	64.4	47.0	32.2	56.1	37.2
Edge	83.7	69.8	48.8	39.6	56.9	34.4

TABLE 5.4. The performance of integrating image or edge information into the backbone.

When utilizing other models as the source model, it becomes evident that our EdgeNet demonstrates effective defense against these attacks, showcasing stronger robustness compared to the ViT-B/16 backbone itself. Furthermore, it is noteworthy that even when employing the Swin-B with a different architecture as the source model, both the ViT-B/16 backbone and our method exhibit substantial robustness. However, even in this scenario, our approach manages to further enhance the backbone's robustness.

5.3.5 Integrating Image or Edge Information

In order to illustrate the effectiveness of incorporating edge information, we conduct an experiment by replacing the inputs to EdgeNet with images. For this configuration, we maintain the exact same architecture and hyper-parameters for the new blocks, opting for the optimal # Intervals = 3 setting. As shown in Table 5.4, both the integration of images and edge information yield performance improvements compared to the classification head fine-tuning method presented in Table 5.1. Furthermore, it is noteworthy that the integration of edge information consistently outperforms the integration of image information. This is because integrating image information again may have redundancy in relation to the image features already present within the backbone.

5.4 Chapter Summary

In this chapter, we have uncovered a significant pathway to enhance the robustness of Deep Neural Networks, specifically Vision Transformers, against adversarial attacks. By leveraging edge information extracted from images, we developed EdgeNet, a lightweight

and seamlessly integrable module that brings about improved adversarial robustness. The efficiency of EdgeNet, demonstrated through minimal additional computational overhead and wide applicability across various robust benchmarks, makes it a compelling advancement in the field. The experiment results, including superior performance against different types of adversarial attacks and maintained accuracy on clean images, underline the potential of edge information as a robust and relevant feature in vision classification tasks. Notably, the robustness of EdgeNet extends beyond adversarial attacks to scenarios involving natural adversarial examples (ImageNet-A), out-of-distribution data (ImageNet-R), and common corruptions (ImageNet-C). This broader application underlines EdgeNet's versatility and its potential as a comprehensive solution for diverse challenges in vision classification tasks.

CHAPTER 6

Adapting Neural Architectures for OOD Generalization

6.1 Motivation

Neural architecture search (NAS) aims to automate the design of neural network architectures. Recently, convolutional neural networks (CNNs) designed using NAS methods have surpassed the performance of manually designed architectures. However, early NAS methods are computationally intensive due to their need for training and evaluating a large number of architectures (Shah et al., 2018; Zoph et al., 2018). As a result, conducting architecture searches directly on large-scale benchmarks like ImageNet becomes intractable. Therefore, many NAS methods perform searches on proxy tasks, such as CIFAR-10, before retraining the obtained architectures on ImageNet. However, even on CIFAR-10, most methods still require thousands of GPU days.

In recent years, significant efforts have been made to reduce the computational cost of NAS. Notably, research on differentiable approaches for architecture search (Liu et al., 2018) has reduced the search cost on the small-scale CIFAR-10 dataset to just several GPU days or even a few GPU hours. DARTS (Liu et al., 2018) relaxes the discrete NAS search space into continuous architecture parameters and constructs a supernetwork by weighting all candidate operations. In this supernetwork, both network weights and architecture parameters can be jointly optimized using gradient descent. The search cost of DARTS on CIFAR-10 is only 1 GPU day, but the parallel optimization of all candidate operations requires a large amount of GPU memory. GDAS (Dong and Yang, 2019) addresses this issue by introducing a differentiable sampler that selects a single operation per connection in each

epoch. This dramatically reduces GPU memory usage, allowing the search to be completed in approximately 4 to 5 GPU hours, depending on the specific settings.

Despite the success in efficiently searching for architectures with strong in-distribution (ID) performance on small-scale proxy datasets, a fundamental problem remains: the architectures found on these small tasks do not come with out-of-distribution (OOD) performance guarantees and do not always generalize well to large-scale target datasets. This issue arises due to the distribution shift between proxy and target datasets. NAS optimizes architectures for performance on the proxy dataset, but the same architectures may not perform optimally when applied to the final target dataset. This domain gap can lead to two major risks: (1) the omission of architectures that would perform well on the target dataset but are overlooked due to suboptimal proxy performance, and (2) the selection of architectures that perform well on the proxy task but degrade significantly on the target domain. This inconsistency hinders the reliability of NAS in real-world applications.

Recent efforts have attempted to search directly on large datasets like ImageNet. For instance, MnasNet (Tan et al., 2019) leveraged reinforcement learning but required 288 TPU days for a single search, making it impractical. ProxylessNAS (Cai et al., 2018a) introduced a more efficient differentiable NAS approach, reducing search cost on ImageNet to 8.33 GPU days. However, this is still significantly more expensive than proxy-based methods, which can complete a search in just 4 to 5 GPU hours (Dong and Yang, 2019; Chen et al., 2019).

To bridge this gap, we introduce Adaptable Neural Architecture Search (AdaptNAS), which explicitly models and minimizes the generalization gap between proxy tasks and target domains. Inspired by domain adaptation principles, we analyze the relationship between the empirical validation error on the proxy task and the expected error on the target dataset. We derive generalization bounds to quantify this gap and propose a lightweight adaptation strategy that incorporates a subset of target data during search, ensuring better transferability of architectures. Our experiments on both synthetic datasets and large-scale benchmarks demonstrate that AdaptNAS consistently outperforms traditional NAS methods in OOD generalization. It also achieves competitive accuracy with significantly lower search costs compared to methods that search directly in the target domain.

6.2 Generalization Analysis for AdaptNAS

Let $\mathcal{X}$ be the input data space, $\mathcal{Z}$ be a latent representation space and $\mathcal{Y}$ be the label space. A convolutional neural network (CNN) $f : \mathcal{X} \to \mathcal{Y}$ can be disassembled into a representation mapping $\mathcal{R} : \mathcal{X} \to \mathcal{Z}$ and a classification hypothesis $h : \mathcal{Z} \to \mathcal{Y}$. In general, h is usually a naive single-layer feed-forward network with weights $\boldsymbol{w}_h$, and $\mathcal{R}$ can have complex topology described by network weights $\boldsymbol{w}_\mathcal{R}$ and the neural architecture $\boldsymbol{A}$. The target of NAS is to find an optimum architecture $\boldsymbol{A}^* \in \mathbb{A}$ that minimize the classification loss $\mathcal{L}(\boldsymbol{w}^*(\boldsymbol{A}), \boldsymbol{A}) = \mathbb{E}_{\boldsymbol{x}_i \sim \mathcal{D}}[\ell(f(\boldsymbol{x}_i; \boldsymbol{w}^*(\boldsymbol{A}), \boldsymbol{A}), y_i)]$

$$\boldsymbol{A}^* = \arg\min_{\boldsymbol{A}} \mathcal{L}(\boldsymbol{w}^*(\boldsymbol{A}), \boldsymbol{A}), \tag{6.1}$$

where $\mathbb{A}$ is a predefined search space, $\mathcal{D}$ is a distribution over the input space $\mathcal{X}$, ℓ is a loss function and $\boldsymbol{w}^*(\boldsymbol{A})$ is the optimal value of network weights $\boldsymbol{w} = \{\boldsymbol{w}_\mathcal{R}, \boldsymbol{w}_h\}$ depending on the current architecture $\boldsymbol{A}$. We consider the bi-level optimization form of NAS

$$\min_{\boldsymbol{A}} \quad \mathcal{L}_{valid}(\boldsymbol{w}^*(\boldsymbol{A}), \boldsymbol{A}) \tag{6.2}$$

$$\textbf{s.t.} \quad \boldsymbol{w}^*(\boldsymbol{A}) = \arg\min_{\boldsymbol{w}} \mathcal{L}_{train}(\boldsymbol{w}, \boldsymbol{A}), \tag{6.3}$$

where $\mathcal{L}_{train}$ and $\mathcal{L}_{valid}$ are losses on the training distribution $\mathcal{D}_{train}$ and the held-out validation distribution $\mathcal{D}_{valid}$, respectively. In such a bi-level form, $\boldsymbol{w}$ and $\boldsymbol{A}$ are optimized alternately with Eqs. (6.3) and (6.2) until convergence or reach a maximum iteration number.

We define *ProxyNAS* as those existing NAS methods that conduct optimization on a relatively small proxy task (e.g., CIFAR10) and evaluate the searched architectures on the large-scale task (e.g., ImageNet). In this chapter, we tend to revisit such a tradition of NAS training and evaluation from the perspective of domain adaptation and propose the AdaptNAS. The smaller training data of the architecture is taken as *source domain*, and we can also leverage a few data from the *target domain* (e.g., ImageNet) to improve the generalization of the architecture.

Formally, a domain can be considered as a pair of a distribution $\mathcal{D}$ on input space $\mathcal{X}$ and a labeling function $f : \mathcal{X} \to \mathcal{Y}$. We can thus define the source and target domains as $\langle \mathcal{D}_S, f_S \rangle$ and $\langle \mathcal{D}_T, f_T \rangle$, respectively. In this section, we first introduce a generalization bound in NAS

constrained by the source domain validation error and a domain distance. Then, we introduce the target domain validation error into the boundary to utilize any accessible target domain information. Detailed proofs are provided in the supplementary material.

6.2.1 Generalization Bounds via Validation of Source Domain

A domain distance measurement is necessary to quantify the generalization gap between domains. We use the $\mathcal{A}$-distance (Kifer et al., 2004) as the measurement. The $\mathcal{A}$-distance is defined as follow

DEFINITION 1 ($\mathcal{A}$-distance). Let $\mathcal{D}$ and $\mathcal{D}'$ be distributions on $\mathcal{X}$, and $\mathcal{A}$ be a collection of subsets of $\mathcal{X}$ such that every $A \in \mathcal{A}$ is measurable w.r.t $\mathcal{D}$ and $\mathcal{D}'$. The $\mathcal{A}$-distance between $\mathcal{D}$ and $\mathcal{D}'$ is

$$d_{\mathcal{A}}(\mathcal{D}, \mathcal{D}') := 2 \sup_{A \in \mathcal{A}} |\Pr_{\mathcal{D}}[A] - \Pr_{\mathcal{D}'}[A]|, \tag{6.4}$$

where $\Pr_{\mathcal{D}}[A]$ is the probability of A under $\mathcal{D}$.

The complexity of the $\mathcal{A}$-distance can be limited by the *symmetric difference hypothesis space* $\mathcal{H}\Delta\mathcal{H}$ (Blitzer et al., 2008). For simplification, we discuss the binary classification scenario, where $\mathcal{Y} = \{0, 1\}$. The theory results can be easily generalized to the multi-class case. Under the binary setting, we have $\mathcal{H}\Delta\mathcal{H} = \{h(\boldsymbol{z}) \oplus h'(\boldsymbol{z}) | h, h' \in \mathcal{H}\}$, where $\oplus$ is the XOR operation, and $\mathcal{H}$ is a hypothesis space. Based on this, $\mathcal{A}_{\mathcal{H}\Delta\mathcal{H}}$ can be defined as a collection of all subsets A such that $A = \{\boldsymbol{x} | \boldsymbol{x} \in \mathcal{X}, h(x) \neq h'(x)\}$ for some $h, h' \in \mathcal{H}$. Letting $\mathcal{A} = \mathcal{A}_{\mathcal{H}\Delta\mathcal{H}}$ in Eq. 6.4, we can have the symmetric difference $\mathcal{A}$-distance, notated as $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}, \mathcal{D}')$. The advantage of using $d_{\mathcal{H}\Delta\mathcal{H}}(\cdot, \cdot)$ is that it satisfies

$$\forall h, h' \in \mathcal{H}, \quad |\varepsilon_S(h, h') - \varepsilon_T(h, h')| \leq \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T), \tag{6.5}$$

where $\varepsilon(\cdot, \cdot)$ measures the disagreement of two hypothesis. The measure in the source domain is defined as $\varepsilon_S(h, h') = \mathbb{E}_{\boldsymbol{x} \sim \mathcal{D}_S}[|h(\boldsymbol{x}) - h'(\boldsymbol{x})|]$, and we use a similar definition for the target domain. Similar to $\mathcal{D}_S$, $\mathcal{D}_T$, we notate the source and target latent distribution on $\mathcal{Z}$ as $\widetilde{\mathcal{D}}_S$ and $\widetilde{\mathcal{D}}_T$. The labelling functions from $\mathcal{Z}$ to $\mathcal{X}$ are represented by $\tilde{f}_S$ and $\tilde{f}_T$, respectively. We define the expected error of h in a domain S as the disagreement between h and $\tilde{f}_S$, notated

as $\varepsilon_S(h) := \varepsilon_S(h, \tilde{f}_S)$. The similar notation is also used for the target domain. Then, Eq. 6.5 can lead to Lemma 2.

LEMMA 2. (Blitzer et al., 2008) *Let $\mathcal{R}$ be a representation function $\mathcal{R} : \mathcal{X} \to \mathcal{Z}$, and $\widetilde{\mathcal{D}}_S$ and $\widetilde{\mathcal{D}}_T$ be the source and target distribution over $\mathcal{Z}$, respectively. For $h \in \mathcal{H}$*

$$\varepsilon_T(h) \leq \varepsilon_S(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\widetilde{\mathcal{D}}_S, \widetilde{\mathcal{D}}_T) + \lambda, \tag{6.6}$$

where λ is combined error of the optimum hypothesis $h^* = \arg\min_{h \in \mathcal{H}} \varepsilon_S(h) + \varepsilon_T(h)$ on both domains: $\lambda = \varepsilon_S(h^*) + \varepsilon_T(h^*)$.

Lemma 2 reveals that the cross-domain generalization gap is bounded by the expected source error and the $\mathcal{A}$-distance of latent distributions. This distance can be minimized by optimizing the representation function $\mathcal{R}$. In NAS, $\boldsymbol{w} = \{\boldsymbol{w}_{\mathcal{R}}, \boldsymbol{w}_h\}$ are optimized over the training data, while in the validation phase, given the fixed $\boldsymbol{w}$, the architecture $\boldsymbol{A}$ is further optimized to minimize the validation error (see Eq. 6.2). We, therefore, proceed to extend the above analysis to the validation set. Let $\widetilde{\mathcal{U}}_{S,train}$ and $\widetilde{\mathcal{U}}_{S,valid}$ be a training set and a held-out validation set of i.i.d. sample drawn from $\widetilde{\mathcal{D}}_S$, respectively, such that $\widetilde{\mathcal{U}}_{S,train} \cap \widetilde{\mathcal{U}}_{S,valid} = \varnothing$. The validation is of a subset $\mathcal{H}'$ of $\mathcal{H}$ depending on $\widetilde{\mathcal{U}}_{S,train}$ but is independent of $\widetilde{\mathcal{U}}_{S,valid}$. The following theorem provides an analysis on the expected target error in terms of the empirical source validation error on $\widetilde{\mathcal{U}}_{S,valid}$ and an empirical $\mathcal{A}$-distance.

THEOREM 6.2.1. *Let m be the size of $\widetilde{\mathcal{U}}_{S,valid}$, d' be the VC-dimension of $\mathcal{H}'$, and $\widetilde{\mathcal{U}}_S$ and $\widetilde{\mathcal{U}}_T$ be sets of unlabelled i.i.d. samples drawn from $\widetilde{\mathcal{D}}_S$ and $\widetilde{\mathcal{D}}_T$, each with size m'. With probability at least $1 - \delta$, for $h \in \mathcal{H}'$*

$$\begin{aligned} \varepsilon_T(h) \leq \hat{\varepsilon}_{S,valid}(h) + \frac{d' \log m - \log \delta}{3m} + \sqrt{\frac{2(d' \log m - \log \delta)}{m}} \\ + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\widetilde{\mathcal{U}}_S, \widetilde{\mathcal{U}}_T) + 4\sqrt{\frac{d' \log(2m') + \log(4/\delta)}{m'}} + \lambda. \end{aligned} \tag{6.7}$$

Theorem 6.2.1 provides an empirical estimate of the cross-domain generalizability of architectures by validation. The target expected error of an architecture $\boldsymbol{A}$ depends on two terms: the validation error of the entire network (including both $\mathcal{R}$ and h, but h is fixed during

the validation) in source domain and the $\mathcal{A}$-distance of $\widetilde{\mathcal{U}}_S$ and $\widetilde{\mathcal{U}}_T$ generated by the neural architecture.

6.2.2 Generalization Bounds via a Hybrid Validation

In Theorem 6.2.1, $\widetilde{\mathcal{U}}_{S,valid}$ is requested to compute $\hat{\varepsilon}_{S,valid}(h)$. Besides the validation set on the source domain, we could further have labeled samples from the target domain for validation use in practice. A hybrid validation set of m examples is therefore defined as the composition of βm source examples and $(1-\beta)m$ target examples, where $\beta \in [0,1]$. Validation errors on the source and target domain are combined by weighted sum with $\alpha \in [0,1]$

$$\hat{\varepsilon}_{\alpha,valid}(h) = \alpha\hat{\varepsilon}_{S,valid}(h) + (1-\alpha)\hat{\varepsilon}_{T,valid}(h). \tag{6.8}$$

The following lemma bounds the expected target error with the expected hybrid error. This bound can be extended to the validation set as well.

LEMMA 3. *Let $\varepsilon_\alpha(h)$ be an expected hybrid error weighted by $\alpha \in [0,1]$. For $h \in \mathcal{H}$*

$$\varepsilon_T(h) \leq \varepsilon_\alpha(h) + \alpha\left(\frac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(\widetilde{D}_S, \widetilde{D}_T) + \lambda\right). \tag{6.9}$$

By applying Lemma 3 to Theorem 6.2.1, we can have the following corollary.

COROLLARY 6.2.1.1. *Let $\alpha \in [0,1]$ be the weight of the hybrid error, and $\beta \in [0,1]$ be the ratio of i.i.d. samples drawn from $\widetilde{\mathcal{D}}_S$ and $\widetilde{\mathcal{D}}_T$ in a held-out validation set. With probability at least $1-\delta$, for $h \in \mathcal{H}'$*

$$\begin{aligned}\varepsilon_T(h) \leq \hat{\varepsilon}_{\alpha,valid}(h) + \left(\frac{\alpha}{\beta} + \frac{1-\alpha}{1-\beta}\right)\left(\frac{d'\log m - \log\delta}{3m} + \sqrt{\frac{2(d'\log m - \log\delta)}{m}}\right) \\ + \alpha\left(\frac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(\widetilde{\mathcal{U}}_S, \widetilde{\mathcal{U}}_T) + 4\sqrt{\frac{d'\log(2m') + \log(4/\delta)}{m'}} + \lambda\right).\end{aligned} \tag{6.10}$$

To utilize Corollary 6.2.1.1, α and β need to be determined. When $\alpha = 1$ and the target validation error is not considered, Corollary 6.2.1.1 will be reduced to Theorem 6.2.1. With

$\alpha \in (0, 1)$, we will introduce both source and target samples for validation, and the generalizability of architectures could be improved (see the α before $\mathcal{A}$-distance). The selection of β is a trade-off. With a fixed source validation set, a smaller β means more target samples and higher computation costs. Besides, with a β approaches 0 or 1, the source and target sample number become highly unbalanced, and the factor $\alpha/\beta + (1-\alpha)/(1-\beta)$ approaches infinite, which makes the architecture optimization unpredictable. This, therefore, reminds us of carefully balancing the sample size in source and target domains. More empirical discussions can be found in experiments.

6.3 AdaptNAS Algorithm

Motivated by theorems in Section 6.2, we propose two versions of AdaptNAS. The former, **AdaptNAS-S**ource, following Theorem 6.2.1, optimizes network weights with source training samples and estimates the $\mathcal{A}$-distance in the training phase. In the searching phase, the architecture $\boldsymbol{A}$ is optimized to reduce both the source validation loss and the $\mathcal{A}$-distance. The latter, **AdaptNAS-C**ombined, following Corollary 6.2.1.1, uses a similar schema as AdaptNAS-S, but further considers a subset of target samples to optimize both network weights and architectures by utilizing Eq. 6.8.

It is intractable to directly compute the $\mathcal{A}$-distance, but we can approximate it with a domain discriminator (Ben-David et al., 2007). With a domain discriminator $h_d \in \mathcal{H}$, we have

$$d_{\mathcal{H}\Delta\mathcal{H}}(\widetilde{\mathcal{U}}_S, \widetilde{\mathcal{U}}_T) = 2\left(1 - 2\min_{h_d \in \mathcal{H}} \hat{\varepsilon}_d(h_d)\right), \tag{6.11}$$

where $\hat{\varepsilon}_d(h_d) = \frac{1}{2m'}\sum_{i=1}^{2m'} |h_d(\boldsymbol{z}_i) - y_{d,i}|$ is the empirical discrimination error on $\boldsymbol{z}_i \in \widetilde{\mathcal{U}}_S \cup \widetilde{\mathcal{U}}_T$, and $y_{d,i}$ is the domain label. Although the optimal h_d is normally unsolvable, the $\mathcal{A}$-distance can still be approximated arbitrarily well by optimizing it. An useful property of Eq. 6.11 is that $d_{\mathcal{H}\Delta\mathcal{H}}(\widetilde{\mathcal{U}}_S, \widetilde{\mathcal{U}}_T) \propto \frac{1}{\min_{h_d \in \mathcal{H}} \hat{\varepsilon}_d(h_d)}$. With such an observation, it is possible to learn a domain discriminator during NAS and use adversarial learning to minimize the $\mathcal{A}$-distance by maximizing discrimination error. In AdaptNAS, we first learn an h_d to distinguish the latent representation produced by $\mathcal{R}$ in the training phase to minimize a discrimination loss:

$\mathcal{L}_d(h_d; \boldsymbol{w}_\mathcal{R}, \boldsymbol{A}) = \mathbb{E}_{\boldsymbol{x}_i \sim \mathcal{D}_S \cup \mathcal{D}_T} \ [\ell(h_d(\mathcal{R}(\boldsymbol{x}_i; \boldsymbol{w}_\mathcal{R}, \boldsymbol{A})), d_i)]$. Then, $\boldsymbol{A}$ of $\mathcal{R}$ is optimized in the searching phase with an adversarial loss to maximize the discrimination loss.

In AdaptNAS-S, the lower-level optimization in Eq. 6.3 can be reformed as Eq. 6.13, where $\mathcal{L}_{S,train}(h; \boldsymbol{w}_\mathcal{R}, \boldsymbol{A}) = \mathbb{E}_{\boldsymbol{x}_i \sim \mathcal{D}_{S,train}} [\ell(h(\mathcal{R}(\boldsymbol{x}_i; \boldsymbol{w}_\mathcal{R}, \boldsymbol{A})), y_i)]$ is the source training loss. Similar notations are also used for the source validation loss and the target training and validation losses. Similarly, the upper-level optimization in Eq. 6.2 can be reformed as Eq. 6.12.

$$\min_{\boldsymbol{A}} \quad \mathcal{L}_{S,valid}(h; \boldsymbol{w}_\mathcal{R}, \boldsymbol{A}) - \mathcal{L}_d(h_d; \boldsymbol{w}_\mathcal{R}, \boldsymbol{A}), \tag{6.12}$$

$$\textbf{s.t.} \quad \max_{h_d} \ \min_{\boldsymbol{w}, h} \ \mathcal{L}_{S,train}(h; \boldsymbol{w}_\mathcal{R}, \boldsymbol{A}) - \mathcal{L}_d(h_d; \boldsymbol{w}_\mathcal{R}, \boldsymbol{A}). \tag{6.13}$$

However, the discriminator gradients at the early stage could be noisy and will corrupt the entire network. To control them in back-propagation, we apply a gradient reversal technique (Ganin and Lempitsky, 2014). In the training phase, with gradient reversal, h and h_d are still updated with their own gradients, but $\boldsymbol{w}_\mathcal{R}$ is updated with additional reversed discriminator gradients weighted by γ as in Eq. 6.14. The weight term $\gamma \in [0, 1]$ can be dynamically adjusted during optimization. In the searching phase, by utilizing differentiable NAS (Dong and Yang, 2019; Liu et al., 2018), architectures can be relaxed as continuous parameters and updated with adversarial learning as in Eq. 6.15.

$$\boldsymbol{w}_\mathcal{R} \leftarrow \boldsymbol{w}_\mathcal{R} - \eta \left(\frac{\partial \mathcal{L}_{S,train}(h; \boldsymbol{w}_\mathcal{R}, \boldsymbol{A})}{\partial \boldsymbol{w}_\mathcal{R}} - \gamma \frac{\partial \mathcal{L}_d(h_d; \boldsymbol{w}_\mathcal{R}, \boldsymbol{A})}{\partial \boldsymbol{w}_\mathcal{R}} \right), \tag{6.14}$$

$$\boldsymbol{A} \leftarrow \boldsymbol{A} \quad - \eta \left(\frac{\partial \mathcal{L}_{S,valid}(h; \boldsymbol{w}_\mathcal{R}, \boldsymbol{A})}{\partial \boldsymbol{A}} - \gamma \frac{\partial \mathcal{L}_d(h_d; \boldsymbol{w}_\mathcal{R}, \boldsymbol{A})}{\partial \boldsymbol{A}} \right). \tag{6.15}$$

In AdaptNAS-C, we cannot simply replace $\mathcal{L}_S(h; \boldsymbol{w}, \boldsymbol{A})$ by $\mathcal{L}_\alpha(h; \boldsymbol{w}, \boldsymbol{A})$, because Corollary 6.2.1.1 has already revealed that the $\mathcal{A}$-distance term should also be weighted by α. We, therefore, rewrite the optimization problem into the following form, where the source loss

and discrimination loss are weighted together

$$\min_{\boldsymbol{A}} \quad \alpha\left(\mathcal{L}_{S,valid}(h; \boldsymbol{w}_{\mathcal{R}}, \boldsymbol{A}) - \mathcal{L}_d(h_d; \boldsymbol{w}_{\mathcal{R}}, \boldsymbol{A})\right) + (1-\alpha)\mathcal{L}_{T,valid}(h; \boldsymbol{w}_{\mathcal{R}}, \boldsymbol{A}), \tag{6.16}$$

$$\textbf{s.t.} \quad \max_{h_d} \min_{h, \boldsymbol{w}_{\mathcal{R}}} \alpha\left(\mathcal{L}_{S,train}(h; \boldsymbol{w}_{\mathcal{R}}, \boldsymbol{A}) - \mathcal{L}_d(h_d; \boldsymbol{w}_{\mathcal{R}}, \boldsymbol{A})\right) + (1-\alpha)\mathcal{L}_{T,train}(h; \boldsymbol{w}_{\mathcal{R}}, \boldsymbol{A}). \tag{6.17}$$

Compared to the origin bi-level optimization in Eqs. 6.2 and 6.3, AdaptNAS introduces an adversarial loss to both levels and the AdaptNAS-C version also introduces the target loss. This ensures the generalizability of both levels. A general difficulty in the bi-level optimization setting is the upper-level optimization highly depends on the lower-level one and is impacted by the quality of the lower-level solution. Similarly, if the solution to the lower-level problem has a large generalization gap, it will be hard for the upper-level one to generalize well. The symmetrical constraint on both levels can alleviate this issue.

A remaining problem is that for the hybrid loss calculation on CIFAR-10 and ImageNet, we cannot directly use their labels. The reason is that to let the bound in Corollary 6.2.1.1 work, the hypothesis h should be identical for both domains. In practice, the classifier depends on the dimension of output, and there is a large gap between categories in CIFAR-10 and ImageNet (10 versus 1,000 different classes). To bridge this gap, we apply self-supervised learning. In self-supervised learning, samples are transformed and labeled based on some predefined rules. The labels are no longer correlated to objects in samples but are correlated to the rule we defined to transform the samples. Besides, self-supervised learning has been demonstrated to learn feature mapping on one dataset and then well apply the learned mapping to another dataset (Gidaris et al., 2018; Jing and Tian, 2020). To be specific, we utilize a rotation task (Gidaris et al., 2018) for its impressive performance. In the rotation task, each sample in the dataset is rotated to different degrees and labeled with them. We use four different degrees: $0°$, $90°$, $180°$ and $270°$. We can, therefore, learn a 4-class classification task with identical categories on both CIFAR-10 and ImageNet.

6.4 Experiments

We perform extensive experiments on various domains to demonstrate the practical generalizability of AdaptNAS. Firstly, we use three relatively small digits datasets to compare our results with the results of searching on the source domain only and on the target domain directly. Then, we search with both versions of our method under the standard NAS setting (i.e., with CIFAR-10 as the source domain and ImageNet as the target domain) multiple times with various hyperparameters to justify our claims following Theorem 6.2.1 and 6.2.1.1. Finally, the obtained architectures with our optimal settings are compared with baseline architectures.

6.4.1 Search Setting

Following many previous works (Chen et al., 2019; Dong and Yang, 2019; Liu et al., 2018; Zheng et al., 2019; Zoph et al., 2018), we use the NASNet search space (Zoph et al., 2018). There are 2 kinds of cells, including normal cells and reduction cells, and each cell has 7 nodes, including 2 input nodes, 1 output node, and 4 computation nodes. We use a set of 8 different candidate operations. The source dataset, CIFAR-10, contains 50,000 samples in the training set. For target domain samples, we construct a subset of 50,000 samples from ImageNet, containing 50 samples from each category, as target samples that we have access to during searching. This is about 3.90% of the entire ImageNet. More details of our experiment settings are available in the supplementary material.

6.4.2 Cross-Domain Generalization with AdaptNAS

We use three pairs of source and target domains of digits to demonstrate the generalizability of AdaptNAS. The first pair is MNIST (LeCun et al., 1998) and MNIST-M (Ganin et al., 2016). MNIST is a dataset of greyscale handwritten digits (Fig. 6.1a). MNIST-M modifies MNIST by blending greyscale images over random patches of color photos in BSDS500 (Arbelaez et al., 2010) (Fig. 6.1b). The blending introduces extra color and texture. In the second pair, we still use MNIST as the source domain, and the target domain is SVHN (Netzer et al.,

(A) MNIST; (B) MNIST-M; (C) SVHN;

FIGURE 6.1. Sample images from different domains.

Search Method	Source Dataset		
	MNIST	MNIST	MNIST-M
	Target Dataset		
	MNIST-M	SVHN	SVHN
Search on Source	98.56	94.70	95.28
AdaptNAS (ours)	98.75	95.63	95.48
Search on Target	98.61	95.60	95.60

TABLE 6.1. The generalizability of AdaptNAS: Test accuracy of obtained architectures on the target domain. The first row corresponds to the ProxyNAS method without generalization constraints. The last row is our aiming performance. The middle row is our method.

2011), which includes natural images of street house numbers (Fig. 6.1c). The last pair still includes SVHN as the target domain, but the source domain is the more divergent MNIST-M. Intuitively, the first setting is the simplest, the second one is the hardest, and the last one is moderate.

Table 6.1 shows the test accuracy of all obtained architectures on the target domain. Our method can consistently outperform the source-only search method. It is also competitive to directly search on the target domain and can even occasionally outperform it. This is because we only remain architectures after searching and retraining the network weights from scratch. It is possible for an architecture whose generalizability is explicitly optimized to outperform another architecture searched on a single domain.

6.4.3 Better Generalization with The Hybrid Loss

Firstly, we test different parameters for the AdaptNAS-C, including α and β, as shown in Table 6.2. We use 5 different values for α from 0 to 1 with an interval of 0.25. When $\alpha = 1$

α	β	Source Err. (%) (CIFAR-10)		Target Err. (%) (ImageNet)	
		Valid	**Test**	**Valid**	**Test**
0.00	0.50	49.26	3.00	42.52	24.5
0.25	0.50	30.00	2.97	40.13	**24.2**
0.50	0.50	25.16	**2.50**	40.13	24.5
0.75	0.50	22.78	2.62	42.41	25.1
1.00	0.50	23.15	2.53	55.37	25.4
0.00	0.83	52.19	3.21	53.65	25.5
0.25	0.83	38.06	3.17	51.56	25.0
0.50	0.83	33.82	**2.95**	49.86	**24.7**
0.75	0.83	28.68	3.00	54.17	25.5
1.00	0.83	23.89	2.98	56.39	25.8
0.00	0.98	74.80	3.91	69.65	29.5
0.25	0.98	67.31	3.66	70.90	26.5
0.50	0.98	51.93	3.56	64.25	25.8
0.75	0.98	40.68	3.02	62.75	**25.1**
1.00	0.98	30.15	**2.93**	61.85	25.7

TABLE 6.2. Performance of various AdaptNAS-C settings.

and only the source loss is considered, AdaptNAS-C becomes AdaptNAS-S, which is identical to the first row of Table 6.3. A relatively new case is when $\alpha = 0$, and only the target loss is considered. We also use 3 different values for β, including 0.50, 0.83 and 0.98. Table 6.2 shows the validation error during search and the test error of retraining on CIFAR-10 and the full ImageNet. In the first group where $\beta = 0.50$, the source and target domain have the same number of samples (50,000 samples from each domain). With the extreme setting that $\alpha = 0$, the performance on CIFAR-10 is the worst. The target error is the same as the one with $\alpha = 0.50$ but is lower than the one with $\alpha = 0.25$. Although a decent performance might be achieved by solely using target loss if there are sufficient target samples, if there are increasingly few target samples (e.g., the second group where $\beta = 0.83$ and the third group where $\beta = 0.98$), the effect of using domain discriminator loss can be even more remarkable. In the second group, the number of target samples decreased to 10,000, and in the third group, the number further decreased to 1,000. With fewer target samples, using the hybrid loss can improve the target domain performance by up to 4.4%.

Hybrid Loss		Source Err. (%) (CIFAR-10)	Target Err. (%) (ImageNet)
Train	Search		
N	N	2.77	25.3
N	Y	2.84	24.8
Y	Y	2.50	24.5

TABLE 6.3. Compare different versions of AdaptNAS.

We further compare AdaptNAS-S and C. When we introduce the target loss to AdaptNAS-C, we symmetrically introduce it to the training and searching phase because the upper-level optimization highly depends on the lower-level one. Despite that, we explore one more setting, where the target loss is solely introduced to the searching phase. Table 6.3 shows the test error of retraining. By using hybrid loss, the target error of architectures decreases. Even if the hybrid loss is only used in searching, the improvement in the target domain is remarkable. By using hybrid loss in both the training and searching phases, the lowest target error is reached.

6.4.4 Gradient Reversal Scheduler in Adversarial Learning

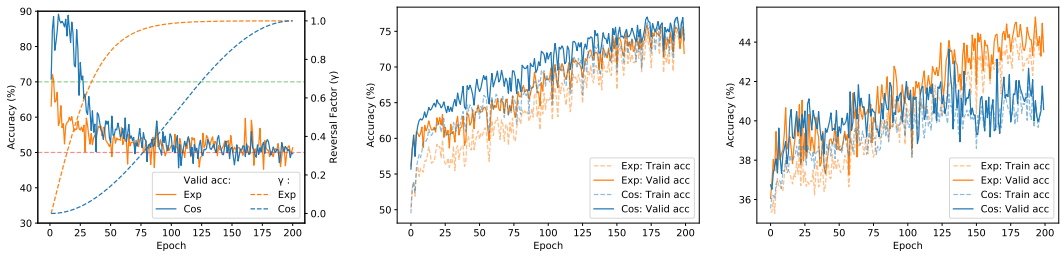
We compare two different schedulers for γ in Eqs. (6.14) and (6.15). An exponential scheduler is proposed by Ganin and Lempitsky (2014), which updates γ by

$$\gamma_p = \frac{2}{1 + \exp(-10 \cdot p)} - 1, \tag{6.18}$$

where $p \in [0, 1]$ is the training procedure calculated by dividing the current epoch by the total number of epochs. However, as shown by the blue dashed line in Fig. 6.2a, the exponential scheduler rises too fast. We also test a cosine-based scheduler, which rises slower

$$\gamma_p = \frac{1 - \cos(p \cdot \pi)}{2} \tag{6.19}$$

where the definition of p is the same as above. Both schedulers are experimented with AdaptNAS-S, which does not consider the target domain loss, to emphasize the impact of the discriminator.



(A) Discrimination accuracy; (B) Source accuracy; (C) Target accuracy;

FIGURE 6.2. The search curves.

Scheduler	**Source Err. (%) (CIFAR-10)**	**Target Err. (%) (ImageNet)**
Exponential	2.93	25.1
Cosine	2.77	25.3

TABLE 6.4. Test error of searching with different γ schedulers.

Fig. 6.2 shows accuracy curves during the search. We also retrain obtained architectures on CIFAR-10 and the entire ImageNet after search (Table 6.4). As shown in Fig. 6.2a, the initial accuracy of both discriminators is similar, and the one with an exponential scheduler immediately drops, while the one with a cosine scheduler can achieve relatively high accuracy, and then declines as γ increases. Common sense is that a strong discriminator usually leads to small loss and vanishing gradients (Arjovsky and Bottou, 2017), which makes the network hard to learn. This is verified by Fig. 6.2b and 6.2c. Although the cosine scheduler corresponds to a better performance in the source domain, its target domain performance is overtaken by the exponential scheduler, which means the network trained with exponential generalized better. In Table 6.4, the test performance by retraining also shows the same conclusion. When the cosine scheduler is used, the error is low on CIFAR-10 but is high on ImageNet which indicates the architecture is not adapted successfully.

Domain	Method	GPU Days	CIFAR-10		ImageNet			
			Params (M)	Err. (%)	Params (M)	+× (M)	Err. (%) Top-1	Err. (%) Top-5
CIFAR-10	NASNet-A (Zoph et al., 2018)	2K	3.3	2.65	5.3	564	26.0	8.4
	ENAS (micro) (Pham et al., 2018)	0.45	4.6	2.89	-	-	-	-
	DARTS (2nd order) (Liu et al., 2018)	1	3.3	$2.76_{\pm 0.09}$	4.7	574	26.7	8.7
	GDAS (Dong and Yang, 2019)	0.21	3.4	2.93	5.3	581	26.0	8.5
	P-DARTS (Chen et al., 2019)	0.3	3.4	2.50	4.9	557	24.4	7.4
	Proxyless-G (Cai et al., 2018a)	4.0	5.7	2.08	-	-	-	-
	HM-NAS (2nd order) (Yan et al., 2019)	1.4	1.8	$2.41_{\pm 0.05}$	-	-	-	-
	MdeNAS (Zheng et al., 2019)	0.16	3.6	2.55	6.1	≤600	25.5	7.9
CIFAR-100	P-DARTS (Chen et al., 2019)	0.3	3.6	2.62	5.1	577	24.7	7.5
ImageNet[1]	MnasNet-A3 (Tan et al., 2019)	3.8K[2]	-	-	5.2	403	23.3	6.7
	FBNet-C[3] (Wu et al., 2019)	9	-	-	5.5	375	25.1	-
	Proxyless-R (Mobile) (Cai et al., 2018a)	8.3	-	-	-	-	25.4	7.8
	Proxyless (GPU) (Cai et al., 2018a)	8.3	-	-	7.1	465	24.9	7.5
	HM-NAS[3] (Yan et al., 2019)	∼5	-	-	3.6	482	26.2	-
	MdeNAS (CPU)[4] (Zheng et al., 2019)	2	-	-	-	≤600	24.8	-
	MdeNAS (GPU)[4] (Zheng et al., 2019)	2	-	-	-	≤600	25.9	-
Cross-Domain	AdaptNAS-S (Rot-1)	0.5	3.6	2.59	5.2	575	24.7	7.6
	AdaptNAS-S (Rot-4)	1.8	3.5	2.77	5.0	552	25.3	7.8
	AdaptNAS-C (Rot-1)	0.7	3.9	2.72	5.4	603	24.3	7.4
	AdaptNAS-C (Rot-4)	2.0	3.7	2.50	5.3	583	24.2	7.4

TABLE 6.5. Comparison with baseline NAS methods searching on different domains. For error rates on CIFAR-10, if a paper provides results with the cutout, we use that version because the cutout always yields its best performance, and we use it, too. On ImageNet, the cutout is normally not used.

[1] Include methods using subset of ImageNet.
[2] MnasNet takes 4.5 days on 64 TPUv2 for one search. The GPU days is estimated by Wu et al. (2019).
[3] FBNet and HM-NAS searches on a subset of ImageNet with 100 classes.
[4] MdeNas searches with the MobileNetV2 (Howard et al., 2017) as backbone and accelerated by the structure in Cai et al. (2018a).

6.4.5 Comparison with Baseline

Table 6.5 compares AdaptNAS with baseline NAS methods, including methods both searching with proxy tasks or directly searching on ImageNet. We notice one drawback of using self-supervised learning is it dramatically increases the searching time. In the rotation task, where we rotate an image to 4 different degrees, the sample number increases 4 times. To balance the performance and efficiency trade-off, we add a simplified Rot-1 setting, where each sample is randomly rotated to only 1 of 4 degrees in each epoch. The origin task is then notated as Rot-4.

Compared to methods searching on CIFAR-10, our method can reach a lower ImageNet top-1 error and competitive CIFAR-10 error. The simplified AdaptNAS-S Rot-1 is our

fastest setting and is even faster than several differentiable NAS methods on CIFAR-10, including DARTS, HM-NAS, and Proxyless-G. Our best top-1 error on ImageNet is reached by AdaptNAS-C with Rot-4, which costs 2 GPU days for searching. Even though it is slower than many ProxyNAS methods, it is faster than most NAS methods that directly search on ImageNet, including the ones using a subset of ImageNet. Only MdeNAS, which searches with acceleration, can reach a similar search cost.

6.5 Chapter Summary

In this chapter, we study the fundamental generalization issue in proxy-based Neural Architecture Search (NAS) and analyze the distribution shift problem that limits the reliability of architectures when transferred to larger-scale target datasets. To address this challenge, we derive generalization bounds that quantify the transferability of architectures between proxy and target domains. Motivated by this theoretical analysis, we propose AdaptNAS, a novel NAS method that explicitly incorporates domain adaptation principles into the search process. Instead of relying solely on proxy dataset performance, AdaptNAS introduces a domain distance constraint to optimize architectures for both ID and OOD generalization. This allows us to mitigate the risks of architecture misselection and improve reliability when deploying models on real-world tasks. Extensive experiments on CIFAR-10 and ImageNet demonstrate that AdaptNAS achieves superior generalization performance compared to both proxy-based and proxyless NAS methods, while maintaining a significantly lower computational cost. These results highlight the importance of explicitly addressing the generalization gap in NAS and provide a new perspective on optimizing architectures for improved generalizability.

CHAPTER 7

Conclusion and Future Work

Deep Neural Networks (DNNs) have revolutionized computer vision, achieving near-human or even superhuman accuracy in various tasks. However, their deployment in real-world scenarios is severely hindered by their vulnerability to adversarial perturbations and out-of-distribution (OOD) shifts, both of which can drastically degrade model performance. Despite current progress, previous methods often struggle with the robustness-accuracy dilemma, as improving adversarial or OOD robustness can come at the cost of degrading standard or in-distribution (ID) accuracy. In this thesis, we introduce a novel paradigm that mitigates the robustness-accuracy dilemma from an architectural perspective. By introducing architectural innovations, we have demonstrated that architectural enhancements can significantly improve adversarial robustness while maintaining or even improving natural accuracy.

This thesis has presented various architectural innovations that address the robustness-accuracy dilemma.

- NADAR enhances adversarial robustness while preserving accuracy by leveraging and dilating existing backbone architecture. This approach optimizes dilation architectures using theoretical error bounds and ensures computational efficiency with performance constraints. Experimental results confirm that NADAR surpasses existing adversarial training and NAS-based methods.
- TORA is inspired by the existence of predictive non-robust and robust non-predictive features. It introduces accuracy and robustness adapters into ViT architectures to disentangle these features. A gated fusion module dynamically balances their contributions using the attention mechanism, which allows the model to adaptively

fuse robust and predictive features. Empirical results on robust benchmarks validate the effectiveness of this approach in enhancing the robustness and accuracy of ViTs.

- EdgeNet explicitly incorporates edge information to mitigate the over-reliance on texture-based features that are vulnerable to perturbations. The novel "sandwich" architecture ensures a seamless integration of edge-based features while preserving the integrity of the backbone network. Experimental evaluations demonstrate that EdgeNet not only improves adversarial robustness but also enhances performance against natural adversarial examples, distribution shifts, and common corruptions.
- AdaptNAS extends the impact of architectural innovations by optimizing architectures for both ID accuracy and OOD robustness. Unlike conventional NAS methods that prioritize ID performance and often fail to generalize under distribution shifts, AdaptNAS introduces domain adaptation constraints to explicitly optimize for robustness across diverse environments. Empirical evaluations confirm that AdaptNAS consistently outperforms traditional NAS approaches by achieving architectures that maintain strong performance under both ID and OOD conditions.

While our architectural innovations have significantly advanced the field of adversarial robustness, there remain broader challenges and opportunities for further refinement.

- For NADAR, future research should explore its role in fostering not only adversarial robustness but also broader trustworthiness aspects such as interpretability and fairness. Understanding how dilation impacts feature extraction in varied adversarial scenarios may provide deeper insights into ensuring robustness across diverse applications.
- For TORA, investigating the interplay between predictive and robust features in the context of distribution shifts and unforeseen adversarial attacks could strengthen its generalizability. Extending this feature disentanglement framework beyond Vision Transformers to multimodal and sequential data may enhance the robustness of models in real-world, dynamic environments. Furthermore, ensuring that TORA does not inadvertently introduce biases in predictive and robust feature selection remains an important direction for future research.

- EdgeNet presents a novel approach to leveraging structural information for robustness. Future work should explore its potential to enhance the explainability of deep models, particularly in settings where transparency and accountability are crucial. Additionally, examining how EdgeNet interacts with adversarial perturbations across different image modalities, including medical and satellite imagery, may extend its applicability and impact.
- Future research on AdaptNAS could extend beyond domain shifts to tackle broader robustness challenges, such as adversarial robustness. Joint optimization of OOD generalization and adversarial robustness may lead to more reliable architectures, while refining domain adaptation constraints with techniques like optimal transport or contrastive alignment could further enhance performance. Interpretability remains crucial. Exploring explainable AI to analyze learned architectures and their learning trajectory could offer deeper insights into how design choices influence robustness and generalization.

Bibliography

Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 33(5):898–916, 2010.

Martin Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. arxiv e-prints, art. *arXiv preprint arXiv:1701.04862*, 2017.

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

Bowen Baker, Otkrist Gupta, Nikhil Naik, and Ramesh Raskar. Designing neural network architectures using reinforcement learning. *arXiv preprint arXiv:1611.02167*, 2016.

Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In *Advances in neural information processing systems*, pages 137–144, 2007.

Srinadh Bhojanapalli, Ayan Chakrabarti, Daniel Glasner, Daliang Li, Thomas Unterthiner, and Andreas Veit. Understanding robustness of transformers for image classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10231–10241, 2021.

John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman. Learning bounds for domain adaptation. In *Advances in neural information processing systems*, pages 129–136, 2008.

Han Cai, Ligeng Zhu, and Song Han. Proxylessnas: Direct neural architecture search on target task and hardware. *arXiv preprint arXiv:1812.00332*, 2018a.

Qi-Zhi Cai, Min Du, Chang Liu, and Dawn Song. Curriculum adversarial training. *arXiv preprint arXiv:1805.04807*, 2018b.

John Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, pages 679–698, 1986.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. Ieee, 2017.

Huanran Chen, Yinpeng Dong, Zhengyi Wang, Xiao Yang, Chengqi Duan, Hang Su, and Jun Zhu. Robust classification via a single diffusion model. *arXiv preprint arXiv:2305.15241*, 2023.

Xin Chen, Lingxi Xie, Jun Wu, and Qi Tian. Progressive differentiable architecture search: Bridging the depth gap between search and evaluation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1294–1303, 2019.

Zhi Cheng, Yanxi Li, Minjing Dong, Xiu Su, Shan You, and Chang Xu. Neural architecture search for wide spectrum adversarial robustness. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(1):442–451, 2023.

Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *International Conference on Machine Learning*, pages 854–863. PMLR, 2017.

Koby Crammer, Michael Kearns, and Jennifer Wortman. Learning from multiple sources. *Journal of Machine Learning Research*, 9(Aug):1757–1774, 2008.

Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International Conference on Machine Learning*, pages 2206–2216. PMLR, 2020.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

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

Minjing Dong, Yanxi Li, Yunhe Wang, and Chang Xu. Adversarially robust neural architectures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

Xuanyi Dong and Yi Yang. Searching for a robust neural architecture in four gpu hours. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1761–1770, 2019.

Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Xiaolin Hu, and Jun Zhu. Discovering adversarial examples with momentum. *arXiv preprint arXiv:1710.06081*, 2017.

Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9185–9193, 2018.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Chengbin Du, Yanxi Li, Zhongwei Qiu, and Chang Xu. Stable diffusion is unstable. *Advances in Neural Information Processing Systems*, 36, 2024.

John C Duchi and Hongseok Namkoong. Learning models with uniform performance via distributionally robust optimization. *The Annals of Statistics*, 49(3):1378–1406, 2021.

Stéphane d'Ascoli, Hugo Touvron, Matthew L Leavitt, Ari S Morcos, Giulio Biroli, and Levent Sagun. Convit: Improving vision transformers with soft convolutional inductive biases. In *International Conference on Machine Learning*, pages 2286–2296. PMLR, 2021.

Simon Föll, Alina Dubatovka, Eugen Ernst, Siu Lun Chau, Martin Maritsch, Patrik Okanovic, Gudrun Thäter, Joachim M Buhmann, Felix Wortmann, and Krikamol Muandet. Gated domain units for multi-source domain generalization. *arXiv preprint arXiv:2206.12444*, 2022.

Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. *arXiv preprint arXiv:1409.7495*, 2014.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape

bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018.

Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Jindong Gu, Volker Tresp, and Yao Qin. Are vision transformers robust to patch perturbations? In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XII*, pages 404–421. Springer, 2022.

Minghao Guo, Yuzhe Yang, Rui Xu, Ziwei Liu, and Dahua Lin. When nas meets robustness: In search of robust architectures against adversarial attacks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 631–640, 2020.

Kai Han, An Xiao, Enhua Wu, Jianyuan Guo, Chunjing Xu, and Yunhe Wang. Transformer in transformer. *Advances in Neural Information Processing Systems*, 34:15908–15919, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *Proceedings of the International Conference on Learning Representations*, 2019.

Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International Conference on Learning Representations*, 2017.

Dan Hendrycks, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. *arXiv preprint arXiv:1912.02781*, 2019.

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness:

A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8340–8349, 2021a.

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. *CVPR*, 2021b.

Byeongho Heo, Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Junsuk Choe, and Seong Joon Oh. Rethinking spatial dimensions of vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11936–11945, 2021.

Charles Herrmann, Kyle Sargent, Lu Jiang, Ramin Zabih, Huiwen Chang, Ce Liu, Dilip Krishnan, and Deqing Sun. Pyramid adversarial training improves vit performance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13419–13429, 2022.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International conference on learning representations*, 2017.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019.

Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

Han Hu, Zheng Zhang, Zhenda Xie, and Stephen Lin. Local relation networks for image recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3464–3473, 2019.

Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

Daniel Kifer, Shai Ben-David, and Johannes Gehrke. Detecting change in data streams. In *VLDB*, volume 4, pages 180–191. Toronto, Canada, 2004.

Junho Kim, Byung-Kwan Lee, and Yong Man Ro. Distilling robust and non-robust features in adversarial examples by information bottleneck. *Advances in Neural Information Processing Systems*, 34:17148–17159, 2021.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25: 1097–1105, 2012.

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

Yann LeCun, LD Jackel, Léon Bottou, Corinna Cortes, John S Denker, Harris Drucker, Isabelle Guyon, Urs A Muller, Eduard Sackinger, Patrice Simard, et al. Learning algorithms for classification: A comparison on handwritten digit recognition. *Neural networks: the statistical mechanics perspective*, 261:276, 1995.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Minjong Lee and Dongwoo Kim. Robust evaluation of diffusion-based adversarial purification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 134–144, 2023.

Yanxi Li and Chengbin Du. Optimizing quantized diffusion models via distillation with cross-timestep error correction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 18530–18538, 2025.

Yanxi Li and Chang Xu. Trade-off between robustness and accuracy of vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7558–7568, 2023.

Yanxi Li, Minjing Dong, Yunhe Wang, and Chang Xu. Neural architecture search in a proxy validation loss landscape. In *International Conference on Machine Learning*, pages 5853–5862. PMLR, 2020a.

Yanxi Li, Zhaohui Yang, Yunhe Wang, and Chang Xu. Adapting neural architectures between domains. *Advances in neural information processing systems*, 33:789–798, 2020b.

Yanxi Li, Zean Wen, Yunhe Wang, and Chang Xu. One-shot graph neural architecture search with dynamic search space. *Proceedings of the AAAI conference on artificial intelligence*, 35(10):8510–8517, 2021a.

Yanxi Li, Zhaohui Yang, Yunhe Wang, and Chang Xu. Neural architecture dilation for adversarial robustness. *Advances in Neural Information Processing Systems*, 34:29578–29589, 2021b.

Yanxi Li, Xinghao Chen, Minjing Dong, Yehui Tang, Yunhe Wang, and Chang Xu. Spatial-channel token distillation for vision mlps. In *International Conference on Machine Learning*, pages 12685–12695. PMLR, 2022a.

Yanxi Li, Minjing Dong, Yunhe Wang, and Chang Xu. Neural architecture search via proxy validation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(6): 7595–7610, 2022b.

Yanxi Li, Minjing Dong, Yixing Xu, Yunhe Wang, and Chang Xu. Neural architecture tuning with policy adaptation. *Neurocomputing*, 485:196–204, 2022c.

Yanxi Li, Chengbin Du, and Chang Xu. Harnessing edge information for improved robustness in vision transformers. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38 (4):3252–3260, 2024.

Yingwei Li, Qihang Yu, Mingxing Tan, Jieru Mei, Peng Tang, Wei Shen, Alan Yuille, and Cihang Xie. Shape-texture debiased neural network training. *arXiv preprint arXiv:2010.05981*, 2020c.

Fangzhou Liao, Ming Liang, Yinpeng Dong, Tianyu Pang, Xiaolin Hu, and Jun Zhu. Defense against adversarial attacks using high-level representation guided denoiser. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1778–1787, 2018.

Hanxiao Liu, Karen Simonyan, Oriol Vinyals, Chrisantha Fernando, and Koray Kavukcuoglu. Hierarchical representations for efficient architecture search. *arXiv preprint arXiv:1711.00436*, 2017.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. In *International Conference on Learning Representations*, 2018.

Jiashuo Liu, Zheyan Shen, Peng Cui, Linjun Zhou, Kun Kuang, and Bo Li. Distributionally robust learning with stable adversarial training. *IEEE Transactions on Knowledge and Data Engineering*, 35(11):11288–11300, 2022a.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11976–11986, 2022b.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations*, 2019.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Kaleel Mahmood, Rigel Mahmood, and Marten Van Dijk. On the robustness of vision transformers to adversarial examples. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7838–7847, 2021.

Xiaofeng Mao, Gege Qi, Yuefeng Chen, Xiaodan Li, Ranjie Duan, Shaokai Ye, Yuan He, and Hui Xue. Towards robust vision transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12042–12051, 2022.

Dongyu Meng and Hao Chen. Magnet: a two-pronged defense against adversarial examples. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pages 135–147, 2017.

Muhammad Muzammal Naseer, Kanchana Ranasinghe, Salman H Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Intriguing properties of vision transformers. *Advances in Neural Information Processing Systems*, 34:23296–23308, 2021.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Anima Anandkumar. Diffusion models for adversarial purification. *arXiv preprint arXiv:2205.07460*, 2022.

Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *International conference on machine learning*, pages 4055–4064. PMLR, 2018.

Sayak Paul and Pin-Yu Chen. Vision transformers are robust learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 2071–2081, 2022.

Xiaohuan Pei, Yanxi Li, Minjing Dong, and Chang Xu. Neural architecture retrieval. In *The Twelfth International Conference on Learning Representations*, 2024.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. Adapterfusion: Non-destructive task composition for transfer learning. *arXiv preprint arXiv:2005.00247*, 2020.

Hieu Pham, Melody Y Guan, Barret Zoph, Quoc V Le, and Jeff Dean. Efficient neural architecture search via parameter sharing. *arXiv preprint arXiv:1802.03268*, 2018.

Rahul Rade and Seyed-Mohsen Moosavi-Dezfooli. Helper-based adversarial training: Reducing excessive margin to achieve a better accuracy vs. robustness trade-off. In *ICML 2021 Workshop on Adversarial Machine Learning*, 2021a.

Rahul Rade and Seyed-Mohsen Moosavi-Dezfooli. Reducing excessive margin to achieve a better accuracy vs. robustness trade-off. In *International Conference on Learning Representations*, 2021b.

Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc V Le, and Alexey Kurakin. Large-scale evolution of image classifiers. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2902–2911. JMLR. org, 2017.

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*, volume 33, pages 4780–4789, 2019.

Evgenia Rusak, Lukas Schott, Roland S Zimmermann, Julian Bitterwolf, Oliver Bringmann, Matthias Bethge, and Wieland Brendel. A simple way to make neural networks robust against diverse image corruptions. In *European Conference on Computer Vision*, pages 53–69. Springer, 2020.

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.

Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. *arXiv preprint arXiv:1805.06605*, 2018.

Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! In *Advances in Neural Information Processing Systems*, pages 3358–3369, 2019.

Syed Asif Raza Shah, Wenji Wu, Qiming Lu, Liang Zhang, Sajith Sasidharan, Phil DeMar, Chin Guok, John Macauley, Eric Pouyoul, Jin Kim, et al. Amoebanet: An sdn-enabled network service for big data science. *Journal of Network and Computer Applications*, 119: 70–82, 2018.

Aman Sinha, Hongseok Namkoong, Riccardo Volpi, and John Duchi. Certifying some distributional robustness with principled adversarial training. *arXiv preprint arXiv:1710.10571*, 2017.

Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. *arXiv preprint arXiv:1710.10766*, 2017.

Andreas Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your vit? data, augmentation, and regularization in vision transformers. *arXiv preprint arXiv:2106.10270*, 2021.

Xiu Su, Tao Huang, Yanxi Li, Shan You, Fei Wang, Chen Qian, Changshui Zhang, and Chang Xu. Prioritized architecture sampling with monto-carlo tree search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10968–10977,

2021.

Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pages 843–852, 2017.

Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2820–2828, 2019.

Yehui Tang, Kai Han, Jianyuan Guo, Chang Xu, Yanxi Li, Chao Xu, and Yunhe Wang. An image patch is a wave: Phase-aware vision mlp. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10935–10944, 2022.

Damien Teney, Yong Lin, Seong Joon Oh, and Ehsan Abbasnejad. Id and ood performance are sometimes inversely correlated on real-world datasets. *Advances in Neural Information Processing Systems*, 36:71703–71722, 2023.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021.

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

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. *arXiv preprint arXiv:1805.12152*, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Haotao Wang, Tianlong Chen, Shupeng Gui, TingKuei Hu, Ji Liu, and Zhangyang Wang. Once-for-all adversarial training: In-situ tradeoff between robustness and accuracy for free.

Advances in Neural Information Processing Systems, 33:7449–7461, 2020.

Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 568–578, 2021.

Yunke Wang, Yanxi Li, and Chang Xu. Position: Ai scaling: From up to down and out. In *International Conference on Machine Learning Position Track*, 2025.

Tsui-Wei Weng, Huan Zhang, Pin-Yu Chen, Jinfeng Yi, Dong Su, Yupeng Gao, Cho-Jui Hsieh, and Luca Daniel. Evaluating the robustness of neural networks: An extreme value theory approach. *arXiv preprint arXiv:1801.10578*, 2018.

Florian Wenzel, Andrea Dittadi, Peter Gehler, Carl-Johann Simon-Gabriel, Max Horn, Dominik Zietlow, David Kernert, Chris Russell, Thomas Brox, Bernt Schiele, et al. Assaying out-of-distribution generalization in transfer learning. *Advances in Neural Information Processing Systems*, 35:7181–7198, 2022.

Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pages 5286–5295. PMLR, 2018.

Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. *arXiv preprint arXiv:2001.03994*, 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10734–10742, 2019.

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

Yuhui Xu, Lingxi Xie, Xiaopeng Zhang, Xin Chen, Guo-Jun Qi, Qi Tian, and Hongkai Xiong. Pc-darts: Partial channel connections for memory-efficient differentiable architecture search. *arXiv preprint arXiv:1907.05737*, 2019.

Shen Yan, Biyi Fang, Faen Zhang, Yu Zheng, Xiao Zeng, Mi Zhang, and Hui Xu. Hm-nas: Efficient neural architecture search via hierarchical masking. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019.

Mengyue Yang, Furui Liu, Zhitang Chen, Xinwei Shen, Jianye Hao, and Jun Wang. Causalvae: Disentangled representation learning via neural structural causal models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9593–9602, 2021a.

Shuo Yang, Tianyu Guo, Yunhe Wang, and Chang Xu. Adversarial robustness through disentangled representations. In *AAAI*, pages 3145–3153, 2021b.

Zhaohui Yang, Yunhe Wang, Xinghao Chen, Boxin Shi, Chao Xu, Chunjing Xu, Qi Tian, and Chang Xu. Cars: Continuous evolution for efficient neural architecture search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1826–1835, 2020.

Jongmin Yoon, Sung Ju Hwang, and Juho Lee. Adversarial purification with score-based generative models. In *International Conference on Machine Learning*, pages 12062–12072. PMLR, 2021.

Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zi-Hang Jiang, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 558–567, 2021.

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

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P Xing, Laurent El Ghaoui, and Michael I Jordan. Theoretically principled trade-off between robustness and accuracy. *arXiv preprint arXiv:1901.08573*, 2019.

Jiacheng Zhang, Benjamin IP Rubinstein, Jingfeng Zhang, and Feng Liu. Ddad: A two-pronged adversarial defense based on distributional discrepancy. *arXiv preprint arXiv:2503.02169*, 2025.

Jingfeng Zhang, Xilie Xu, Bo Han, Gang Niu, Lizhen Cui, Masashi Sugiyama, and Mohan Kankanhalli. Attacks which do not kill training make adversarial learning stronger. In

International Conference on Machine Learning, pages 11278–11287. PMLR, 2020.

Jingfeng Zhang, Jianing Zhu, Gang Niu, Bo Han, Masashi Sugiyama, and Mohan S. Kankanhalli. Geometry-aware instance-reweighted adversarial training. In *Proceeding of International Conference on Learning Representations*, 2021.

Lvmin Zhang and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. *arXiv preprint arXiv:2302.05543*, 2023.

Richard Zhang. Making convolutional networks shift-invariant again. In *International conference on machine learning*, pages 7324–7334. PMLR, 2019.

Hengshuang Zhao, Jiaya Jia, and Vladlen Koltun. Exploring self-attention for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10076–10085, 2020.

Xiawu Zheng, Rongrong Ji, Lang Tang, Baochang Zhang, Jianzhuang Liu, and Qi Tian. Multinomial distribution learning for effective neural architecture search. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1304–1313, 2019.

Daquan Zhou, Zhiding Yu, Enze Xie, Chaowei Xiao, Animashree Anandkumar, Jiashi Feng, and Jose M Alvarez. Understanding the robustness in vision transformers. In *International Conference on Machine Learning*, pages 27378–27394. PMLR, 2022.

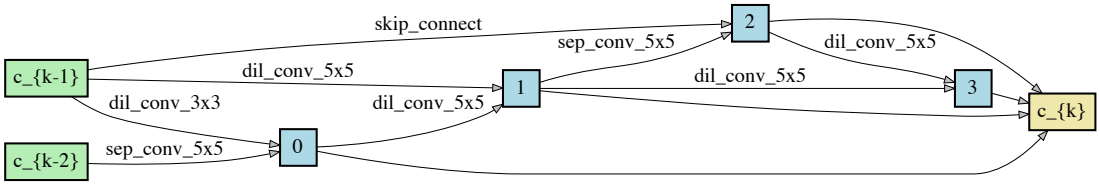
Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8697–8710, 2018.

APPENDIX A

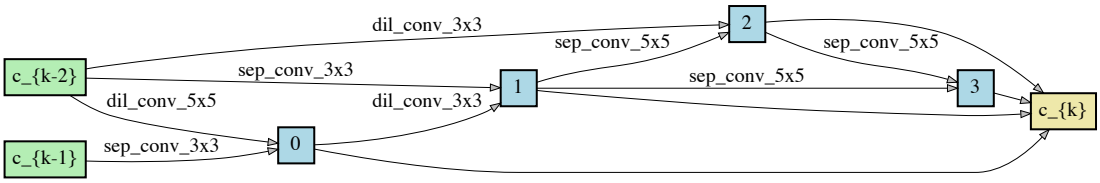
Appendix for Chapter 3

A1 Search Space and Dilated Architectures

For the dilation architecture, we use a DAG with 4 nodes as the supernetwork. There are 8 operation candidates for each edge, including 4 convolutional operations: 3×3 separable convolutions, 5×5 separable convolutions, 3×3 dilated separable convolutions and 5×5 dilated separable convolutions, 2 pooling operations: 3×3 average pooling and 3×3 max pooling, and two special operations: an $\mathrm{identity}$ operation representing skip-connection and a zero operation representing two nodes are not connected. During dilating, we stack 3 cells for each of the 3 blocks in the WRN34-10. During retraining, the number is increased to 6.



(A) NADAR-A (with FLOPs constraint);



(B) NADAR-B (without FLOPS constraint).

FIGURE A.1. Visualization of the dilated cells.

Dataset	FLOPs Const.	+× (M)	Valid Acc. Against (%)		
			FGSM	MI-FGSM	PGD-40
MNIST	T	104.02	98.19	98.11	98.90
	F	131.26	98.27	98.25	98.97
MNIST-M	T	89.23	92.50	92.31	91.79
	F	138.81	93.47	93.04	92.62

TABLE A.1. The adversarial validation accuracy of NADAR under the FGSM, MI-FGSM, and PGD-40 attack on MNIST and MNIST-M.

The dilated architectures designed by NADAR are shown in Fig. A.1. We find that NADAR prefers deep architecture, which can bring more non-linearity improvement with a limited number of parameters. Non-linearity is closely related to network capacity. Such deep architectures can bring more capacity and adversarial robustness to the hybrid network.

A2 Additional Results

A2.1 MNIST

Despite adaptation, we report the adversarial validation accuracy of architectures dilated by NADAR under various attack methods on MNIST and a colorful variant of MNIST, namely MNIST-M Ganin et al. (2016). MNIST-M blends greyscale images in MNIST over random patches of colour photos in BSDS500 (Arbelaez et al., 2010). The blending introduces extra colour and texture. In this experiment, we use the ResNet-18 as our backbone.

Table A.1 shows the adversarial validation accuracy. We report the results of NADAR with and without FLOPs constraint. As shown in the table, the FLOPs constraint can reduce the FLOPs by $20.75 \sim 35.72\%$ while the performance is still competitive. On the MNIST, we observe that NADAR can reach better accuracy under PGD-40 than under FGSM and MI-FGSM. We argue that this is because the MNIST dataset is relatively simple, and the 40-step PGD causes overfitting. We therefore perform experiments on the MNIST-M, and the results show that FGSM $>$ MI-FGSM $>$ PGD-40. We also compare the results to those of

Network	Valid Acc. Against (%)	
	Natural	PGD-20
WRN34-10 w/o dilation	87.25	45.84
ResNet-18 + NADAR	81.35	50.92
ResNet-34 + NADAR	83.57	52.64
ResNet-50 + NADAR	83.23	52.89
ResNet-101 + NADAR	84.39	**53.89**
WRN34-10 + NADAR	**86.23**	53.43

TABLE A.2. NADAR with various backbones.

the SOTA methods. On the MNIST dataset, PGD-7 only reaches 96.01% validation accuracy under the PGD-40 attack, and TRADES-6 only reaches 96.07%.

A2.2 Dilation with Various Backbones

To demonstrate the generalizability of NADAR, we test it with different scales of ResNet backbones from ResNet-18 to ResNet-101. The standard accuracy and adversarial accuracy under the PGD-20 attack of various backbones are reported in Table A.2. All the hybrid networks are retrained with PGD-7 in the same setting.

The results demonstrate that NADAR can effectively improve the robustness of different backbones compared to the WRN34-10 baseline without any dilation. The largest ResNet-101 backbone can reach competitive adversarial accuracy to the dilated WRN34-10. Regarding standard accuracy on natural images, all the ResNet backbones suffer higher performance drops compared to the WRN34-10 backbone due to the inherent limitation of the small capacity of the backbone itself. This illustrates that although NADAR can improve the robustness regardless of the capacity of the backbone, it is still crucial to select a proper backbone for better standard accuracy.

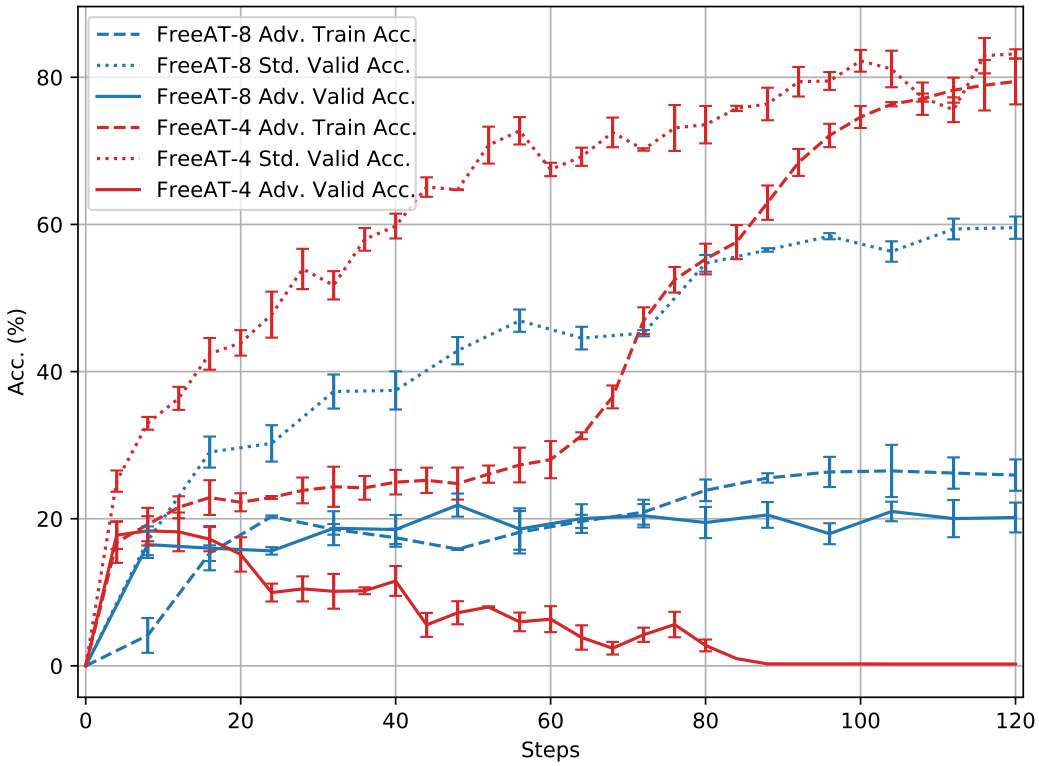


FIGURE A.2. The accuracy curves of dilating architectures with different adversarial training settings of FreeAT.

A3 Additional Ablation Studies

A3.1 Adversarial Training for Dilation

As mentioned, we use FressAT as the adversarial training method to optimize the dilation architecture for efficiency. FreeAT requires a repeat number K on each mini-batch for better perturbation generation. According to their paper, $K = 8$ reaches the best robustness. We also perform experiments regarding the selection of K. Fig. A.2 illustrates the accuracy curves of the hybrid network during dilating. We report the adversarial training accuracy of FreeAT, the standard validation accuracy, and the adversarial validation accuracy under PGD-20 attacks. There is no standard training accuracy because the hybrid network is not directly optimized under the standard classification task (recall the standard performance constraint). After each complete epoch, all the values are obtained when the K-repeat of all

# Dilation Cells	Params (M)		$+\times$ (G)		Valid Acc. Against (%)	
	Back.	Arch.	Back.	Arch.	Natural	PGD-20
3×3	46.2	2.0	6.7	0.3	86.45±0.22	47.78±0.41
3×6	46.2	4.4	6.7	0.7	86.28±0.26	49.63±0.18
3×9	46.2	6.8	6.7	1.1	85.63±0.12	45.25±0.57

TABLE A.3. Different number of stacked cells in the dilation network.

the mini-batches is finished. The horizontal axis represents the total number of optimization steps, which equals the epoch number multiplied by K.

When $K = 4$, the hybrid network reaches outstanding adversarial training accuracy, but the validation only increases slightly at the very beginning of training and then keeps decreasing until it reaches 0. In contrast, the standard validation accuracy increases continuously and reaches a competitive level. This implies that the perturbation generated with $K = 4$ is not powerful enough to dilate the network for the defense against PGD-20, and the framework might be dominated by the standard training of the backbone or the standard constraint on the dilation architecture. When $K = 8$, although the standard validation accuracy is much lower than the previous result, the adversarial training and validation accuracy are competitive and close to each other.

A3.2 Different Scales of Dilation

Besides the FLOPs constraint, there is another factor that impacts the model capacity and the computation cost of a dilation network, which is the number of stacked cells. In Section 3.4.2, we stack 6 cells for each of the 3 blocks in the WRN34-10 for retraining. Intuitively, a large network capacity corresponds to better performance. However, we demonstrate that the network cannot be dilated unlimitedly. There is a sweet spot of neural architecture dilation. In this experiment, we test two more scales of stacked cells. Table A.3 compares the validation results of different scales of dilation.

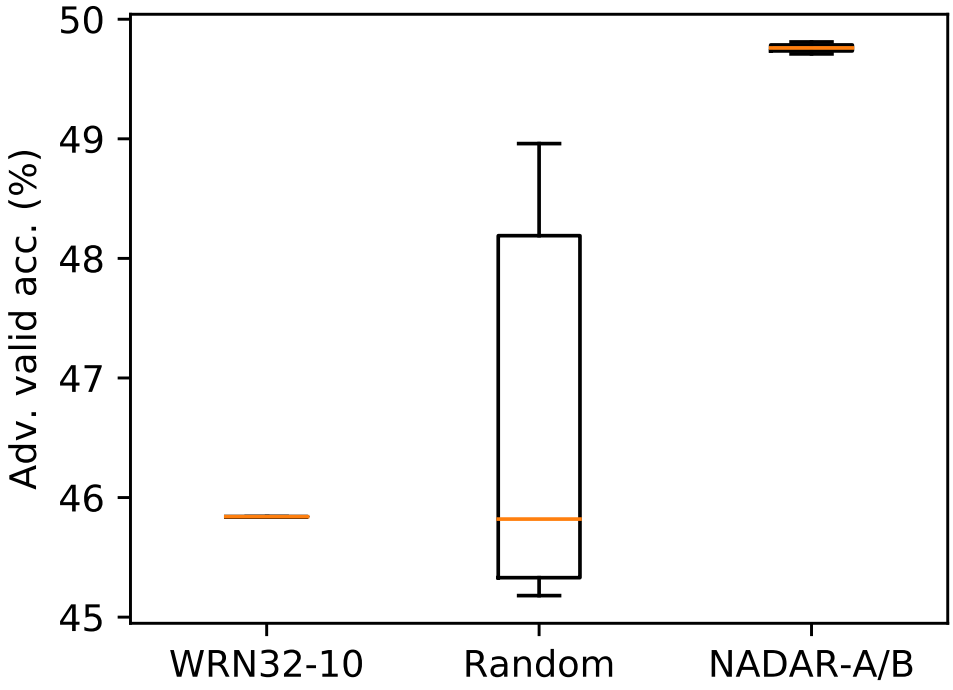


FIGURE A.3. Comparison of NADAR to WRN34-10 backbone and randomly dilated hybrid networks.

We can observe that as the scale of the dilation network increases, the standard accuracy consistently declines. In terms of the adversarial accuracy, it first increases with the dilation scale and then drops significantly. This might be because the network becomes difficult to converge as the network capacity increases. Therefore, we stack 3×6 cells in the dilation network, which reaches the best adversarial accuracy and has a lower standard accuracy drop.

A3.3 Comparison to Random Dilation

To demonstrate the effectiveness of neural architecture dilation, we compare five randomly dilated architectures to our NADAR architectures and the WRN34-10 backbone. We train all the networks with PGD-7 and test their robustness under PGD-20. The adversarial validation accuracy is as shown in Fig. A.3. The median accuracy of random architectures is similar to the WRN34-10 backbone but with a great possibility of reaching better performance.

However, there is still a certain possibility that the dilation architectures can slightly harm the performance of the hybrid network. This shows that neural architecture dilation has the potential to improve the robustness of a backbone, but it still needs to be optimized. The accuracy of NADAR-A and -B is significantly better than the best results of random dilation, which shows that our approach can indeed improve the robustness of backbones effectively and stably.

A4 Proof of Theorems

This section proves the lemmas and theorems in Chapter 3.

A4.1 Standard Error Bound

THEOREM A4.1. *Let $h_{\texttt{bck}}(\boldsymbol{x}) = h_b(\boldsymbol{x})$ be a standard hypothesis, $h_{\texttt{hyb}}(\boldsymbol{x}) = h_b(\boldsymbol{x}) + h_d(\boldsymbol{x})$ be a hybrid hypothesis, and $\mathcal{R}_{\texttt{std}}(h_{\texttt{bck}})$ and $\mathcal{R}_{\texttt{std}}(h_{\texttt{hyb}})$ be the standard error of $h_{\texttt{bck}}$ and $h_{\texttt{hyb}}$, respectively. For any mapping $h_b, h_d : \mathcal{X} \mapsto \mathbb{R}$, we have*

$$\mathcal{R}_{\texttt{std}}(h_{\texttt{hyb}}) \leq \mathcal{R}_{\texttt{std}}(h_{\texttt{bck}}) + \mathbb{E}\left[e^{-h_b(\boldsymbol{x})h_d(\boldsymbol{x})}\right], \tag{A.1}$$

where $\boldsymbol{x} \in \mathcal{X}$ is the input.

PROOF. In Theorem A4.1, we compare the standard error $\mathcal{R}_{\texttt{std}}$ of $h_{\texttt{bck}}$and $h_{\texttt{hyb}}$. The error bound can be defined as the disagreement between the two hypotheses under the condition that $h_{\texttt{bck}}$ is correct. Formally, it can be written as

$$\begin{aligned}
&\mathcal{R}_{\texttt{std}}(h_{\texttt{adv}}) - \mathcal{R}_{\texttt{std}}(h_{\texttt{bck}}) && \text{(A.2)}\\
=&\mathbb{E}\left[\mathbf{1}\left(yh_{\texttt{bck}}(\boldsymbol{x}) > 0,\ h_{\texttt{hyb}}(\boldsymbol{x})h_{\texttt{bck}}(\boldsymbol{x}) \leq 0\right)\right] && \text{(A.3)}\\
\leq&\mathbb{E}\left[\mathbf{1}\left(h_{\texttt{hyb}}(\boldsymbol{x})h_{\texttt{bck}}(\boldsymbol{x}) \leq 0\right)\right]. && \text{(A.4)}
\end{aligned}$$

By applying a simple inequality

$$\mathbf{1}\{yh(\boldsymbol{x}) \leq 0\} \leq e^{-yh(\boldsymbol{x})}, \tag{A.5}$$

we have

$$\mathbb{E}\left[\mathbf{1}\left(h_{\mathtt{hyb}}(\boldsymbol{x})h_{\mathtt{bck}}(\boldsymbol{x}) \leq 0\right)\right] \tag{A.6}$$

$$\leq \mathbb{E}\left[e^{-h_{\mathtt{hyb}}(\boldsymbol{x})h_{\mathtt{bck}}(\boldsymbol{x})}\right] \tag{A.7}$$

$$= \mathbb{E}\left[e^{-(h_{\mathtt{b}}(\boldsymbol{x})+h_{\mathtt{w}}(\boldsymbol{x}))h_{\mathtt{b}}(\boldsymbol{x})}\right] \tag{A.8}$$

$$= \mathbb{E}\left[e^{-h_{\mathtt{b}}(\boldsymbol{x})h_{\mathtt{b}}(\boldsymbol{x})}e^{-h_{\mathtt{b}}(\boldsymbol{x})h_{\mathtt{w}}(\boldsymbol{x})}\right]. \tag{A.9}$$

As $h_{\mathtt{b}}(\boldsymbol{x})h_{\mathtt{b}}(\boldsymbol{x}) \in [0, +\infty)$, we have $e^{-h_{\mathtt{b}}(\boldsymbol{x})h_{\mathtt{b}}(\boldsymbol{x})} \in (0, 1]$. Therefore, we have

$$\mathcal{R}_{\mathtt{std}}(h_{\mathtt{hyb}}) - \mathcal{R}_{\mathtt{std}}(h_{\mathtt{bck}}) \leq \mathbb{E}\left[e^{-h_{\mathtt{b}}(\boldsymbol{x})h_{\mathtt{w}}(\boldsymbol{x})}\right]. \tag{A.10}$$

Theorem A4.1 is proved. □

A4.2 Adversarial Error Bound

We first prove Lemma 4, which is used to prove Theorem A4.2.

LEMMA 4. *For any mapping $h : \mathcal{X} \mapsto \mathbb{R}$, we have*

$$\mathbb{E}\left[\max_{\boldsymbol{x}' \in \mathcal{B}_p(\boldsymbol{x},\varepsilon)} e^{-yh(\boldsymbol{x}')}\right] \leq \mathbb{E}\left[\max_{\boldsymbol{x}' \in \mathcal{B}_p(\boldsymbol{x},\varepsilon)} e^{-yh(\boldsymbol{x})}e^{-h(\boldsymbol{x})h(\boldsymbol{x}')}\right], \tag{A.11}$$

where $\boldsymbol{x} \in \mathcal{X}$ is the input, $y \in \{-1, +1\}$ is the corresponding label, and ε is the bound of allowed adversarial perturbation.

PROOF. Lemma 4 aims to describe the inherent feature of a hypothesis on adversarial tasks. It bounds the adversarial error of a hypothesis with its standard error and its disagreement between standard and adversarial examples. Formally, it can be written as

$$\mathbb{E}\left[\max_{\boldsymbol{x}' \in \mathcal{B}_p(\boldsymbol{x},\varepsilon)} \mathbf{1}\left(yh(\boldsymbol{x}') > 0\right)\right] = \mathbb{E}\left[\mathbf{1}\left(yh(\boldsymbol{x}) > 0\right)\right] + \mathbb{E}\left[\max_{\boldsymbol{x}' \in \mathcal{B}_p(\boldsymbol{x},\varepsilon)} \mathbf{1}\left(yh(\boldsymbol{x}) > 0,\ h(\boldsymbol{x})h(\boldsymbol{x}') \leq 0\right)\right]. \tag{A.12}$$

By applying Eq. A.5 again, we have

$$\mathbb{E}\left[\max_{\boldsymbol{x}'\in\mathcal{B}_p(\boldsymbol{x},\varepsilon)} e^{-yh(\boldsymbol{x}')}\right] \tag{A.13}$$

$$\leq \mathbb{E}\left[e^{-yh(\boldsymbol{x})}\right] + \mathbb{E}\left[\max_{\boldsymbol{x}'\in\mathcal{B}_p(\boldsymbol{x},\varepsilon)} e^{-h(\boldsymbol{x})h(\boldsymbol{x}')}\right] \tag{A.14}$$

$$= \mathbb{E}\left[\max_{\boldsymbol{x}'\in\mathcal{B}_p(\boldsymbol{x},\varepsilon)} e^{-yh(\boldsymbol{x})}e^{-h(\boldsymbol{x})h(\boldsymbol{x}')}\right]. \tag{A.15}$$

Lemma 4 is proved. □

THEOREM A4.2. *Let $h_{\texttt{bck}}(\boldsymbol{x}) = h_b(\boldsymbol{x})$ be a standard hypothesis, $h_{\texttt{hyb}}(\boldsymbol{x}) = h_b(\boldsymbol{x}) + h_d(\boldsymbol{x})$ be a dilated hypothesis, $\mathcal{R}_{\texttt{std}}(h_{\texttt{bck}})$ be the standard error of $h_{\texttt{bck}}$, and $\mathcal{R}_{\texttt{adv}}(h_{\texttt{hyb}})$ be the adversarial error of $h_{\texttt{hyb}}$. For any mapping $h_b, h_d : \mathcal{X} \mapsto \mathbb{R}$, we have*

$$\mathcal{R}_{\texttt{adv}}(h_{\texttt{hyb}}) \leq \mathcal{R}_{\texttt{std}}(h_{\texttt{bck}}) + \mathbb{E}\left[\max_{\boldsymbol{x}'\mathcal{B}_p(\boldsymbol{x},\varepsilon)} e^{-yh_b(\boldsymbol{x})}\left(e^{-h_b(\boldsymbol{x})h_b(\boldsymbol{x}')}e^{-yh_d(\boldsymbol{x}')} - 1\right)\right]. \tag{A.16}$$

where $\boldsymbol{x} \in \mathcal{X}$ is the input, $y \in \{-1, +1\}$ is the corresponding label, and ε is the bound of allowed adversarial perturbation.

PROOF. In Theorem A4.2, we directly compare the adversarial error $\mathcal{R}_{\texttt{adv}}$ of $h_{\texttt{hyb}}$ and the standard error $\mathcal{R}_{\texttt{std}}$ of $h_{\texttt{bck}}$. Formally, it can be written as

$$\mathcal{R}_{\texttt{adv}}(h_{\texttt{hyb}}) - \mathcal{R}_{\texttt{std}}(h_{\texttt{bck}}) \tag{A.17}$$

$$= \mathbb{E}\left[\mathbf{1}(\exists \boldsymbol{x}' \in \mathcal{B}_p(\boldsymbol{x},\varepsilon),\ \textbf{s.t.}\ \ yh_{\texttt{hyb}}(\boldsymbol{x}') \leq 0)\right] - \mathbb{E}\left[\mathbf{1}(yh_{\texttt{bck}}(\boldsymbol{x}) \leq 0)\right] \tag{A.18}$$

$$\leq \mathbb{E}\left[\max_{\boldsymbol{x}'\mathcal{B}_p(\boldsymbol{x},\varepsilon)} e^{-yh_{\texttt{hyb}}(\boldsymbol{x}')}\right] - \mathbb{E}\left[e^{-yh_{\texttt{bck}}(\boldsymbol{x})}\right] \tag{A.19}$$

$$= \mathbb{E}\left[\max_{\boldsymbol{x}'\mathcal{B}_p(\boldsymbol{x},\varepsilon)} e^{-yh_{\texttt{hyb}}(\boldsymbol{x}')} - e^{-yh_{\texttt{bck}}(\boldsymbol{x})}\right] \tag{A.20}$$

By applying Lemma 4, we have

$$\mathcal{R}_{\texttt{adv}}(h_{\texttt{hyb}}) - \mathcal{R}_{\texttt{std}}(h_{\texttt{bck}}) \leq \mathbb{E}\left[\max_{\boldsymbol{x}'\mathcal{B}_p(\boldsymbol{x},\varepsilon)} e^{-yh_{\texttt{b}}(\boldsymbol{x})}\left(e^{-h_{\texttt{b}}(\boldsymbol{x})h_{\texttt{b}}(\boldsymbol{x}')}e^{-yh_{\texttt{w}}(\boldsymbol{x}')} - 1\right)\right]. \tag{A.21}$$

Theorem A4.2 is proved. □

APPENDIX B

Appendix for Chapter 4

B1 Different Backbones Sizes

Models	Clean	FGSM	PGD	A	R	C(↓)
RVT-Ti	79.2	42.7	18.9	14.4	43.9	57.0
Ours Ti/16 (0.1)	**82.1**	38.5	13.8	**33.2**	**51.4**	**42.1**
Ours Ti/16 (0.5)	80.9	44.4	29.5	26.8	49.0	43.5
Ours Ti/16 (0.9)	77.5	**63.3**	**47.9**	10.2	48.1	51.4
RVT-S	81.9	51.8	28.2	25.7	47.7	49.4
Ours S/16 (0.1)	**83.6**	48.1	22.5	**45.1**	**54.0**	**35.1**
Ours S/16 (0.5)	82.1	54.2	37.8	37.4	52.7	38.8
Ours S/16 (0.9)	78.6	**73.3**	**56.6**	20.5	50.9	43.3

TABLE B.1. The results of using backbones of various sizes.

In Table B.1, we present the results of including ViT-Ti/16 and -S/16 Dosovitskiy et al. (2020) as backbones in our evaluation. Among previous SOTA methods, RVT Mao et al. (2022) also utilizes these backbones and adjusts their architectures to improve robustness. To evaluate the effectiveness of our approach, we compared our method with RVT-Ti and -S. Despite introducing different sizes of backbones, our enhancement still holds, indicating that our proposed approach is scalable and adaptable to various backbone sizes.

B2 Evaluation of Various Softmax Functions

Table B.2 presents a comparison of our method for applying the softmax function, which is situated on the right side of Fig. 4.2b, with the other two existing methods located in the

	Clean	FGSM	PGD	A	R	C(↓)
Left	77.9	45.8	23.9	20.5	33.8	62.1
Mid	81.6	53.3	37.1	28.3	52.0	41.1
Right	**83.7**	**54.7**	**38.0**	**39.2**	**56.3**	**34.4**

TABLE B.2. The results of using different methods for applying the softmax function.

# Blocks	Clean	A	R	C(↓)
12	83.7	39.2	56.3	34.4
6	82.8	32.2	51.5	37.4
1	80.5	24.8	45.2	45.8

TABLE B.3. The results of inserting our modules into various numbers of blocks.

middle and on the left side of the figure. The results consistently demonstrate the superiority of our method over the alternative methods.

B3 The Number of Blocks with Adapters

In this thesis, we use ViT-B/16 as our backbone, which consists of a total of 12 blocks. We inserted our proposed adapters and gated fusion modules into all 12 blocks. To explore the impact of the number of blocks on the performance of our approach, we conducted an ablation study, where we evaluated the effectiveness of inserting adapters into only the last 6 blocks or the last 1 block of the backbone. In Table B.3, the results indicate that using adapters in fewer blocks leads to a decline in performance. We use $\lambda = 0.5$ in this experiment.

APPENDIX C

Appendix for Chapter 6

C1 Proofs

C1.1 Proof of Lemma 5

LEMMA 5. (Blitzer et al., 2008) *Let $\mathcal{R}$ be a representation function $\mathcal{R} : \mathcal{X} \to \mathcal{Z}$, and $\widetilde{\mathcal{D}}_S$ and $\widetilde{\mathcal{D}}_T$ be the source and target distribution over $\mathcal{Z}$, respectively. For $h \in \mathcal{H}$*

$$\varepsilon_T(h) \leq \varepsilon_S(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\widetilde{\mathcal{D}}_S, \widetilde{\mathcal{D}}_T) + \lambda, \tag{C.1}$$

where λ is combined error of the optimum hypothesis $h^* = \arg\min_{h \in \mathcal{H}} \varepsilon_S(h) + \varepsilon_T(h)$ on both domains: $\lambda = \varepsilon_S(h^*) + \varepsilon_T(h^*)$.

PROOF. The proof bases on the triangle inequality for classification error (Ben-David et al., 2007; Crammer et al., 2008): for any labeling function f_1, f_2 and f_3, $\varepsilon(f_1, f_2) \leq \varepsilon(f_1, f_3) + \varepsilon(f_2, f_3)$. With the definition that $\varepsilon_S(h) := \varepsilon_S(h, \tilde{f}_S)$, for the source domain, we have

$$\varepsilon_S(h, h^*) \leq \varepsilon_S(h, \tilde{f}_S) + \varepsilon_S(h^*, \tilde{f}_S) = \varepsilon_S(h) + \varepsilon_S(h^*). \tag{C.2}$$

For the target domain, we use a slightly different version

$$\varepsilon_T(h) = \varepsilon_T(h, \tilde{f}_T) \leq \varepsilon_T(h^*, \tilde{f}_T) + \varepsilon_T(h, h^*) = \varepsilon_T(h^*) + \varepsilon_T(h, h^*). \tag{C.3}$$

The symmetric difference $\mathcal{A}$-distance has the following property

$$\forall h, h' \in \mathcal{H}, \quad |\varepsilon_S(h, h') - \varepsilon_T(h, h')| \leq \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T). \tag{C.4}$$

We can have

$$\begin{aligned}
\varepsilon_T(h) &\leq \varepsilon_T(h^*) + \varepsilon_T(h, h^*) && \text{(Applying Eq. C.3)} \\
&\leq \varepsilon_T(h^*) + \varepsilon_S(h, h^*) + |\varepsilon_S(h, h') - \varepsilon_T(h, h')| \\
&\leq \varepsilon_T(h^*) + \varepsilon_S(h, h^*) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T) && \text{(Applying Eq. C.4)} \\
&\leq \varepsilon_T(h^*) + \varepsilon_S(h) + \varepsilon_S(h^*) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T) && \text{(Applying Eq. C.2)} \\
&= \varepsilon_S(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T) + \lambda,
\end{aligned}$$

where $\lambda = \varepsilon_S(h^*) + \varepsilon_T(h^*)$. □

C1.2 Proof of Theorem C1.1

THEOREM C1.1. *Let m be the size of $\widetilde{\mathcal{U}}_{S,valid}$, d' be the VC-dimension of $\mathcal{H}'$, and $\widetilde{\mathcal{U}}_S$ and $\widetilde{\mathcal{U}}_T$ be sets of unlabelled i.i.d. samples drawn from $\widetilde{\mathcal{D}}_S$ and $\widetilde{\mathcal{D}}_T$, each with size m'. With probability at least $1-\delta$, for $h \in \mathcal{H}'$*

$$\begin{aligned}
\varepsilon_T(h) \leq \hat{\varepsilon}_{S,valid}(h) + \frac{d' \log m - \log \delta}{3m} + \sqrt{\frac{2(d' \log m - \log \delta)}{m}} \\
+ \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\widetilde{\mathcal{U}}_S, \widetilde{\mathcal{U}}_T) + 4\sqrt{\frac{d' \log(2m') + \log(4/\delta)}{m'}} + \lambda.
\end{aligned} \tag{C.5}$$

PROOF. Firstly, we derive the bound between the expected source error $\varepsilon_S(h)$ in Eq. C.1 and the empirical source validation error $\hat{\varepsilon}_{S,valid}(h)$. Let $\kappa_i(h) = \varepsilon_S(h) - \ell(h(z_i), y_i)$ for $h \in \mathcal{H}'$ and $z_i \in \widetilde{\mathcal{U}}_{S,valid}$. Therefore,

$$\varepsilon_S(h) - \hat{\varepsilon}_{S,valid}(h) = \frac{1}{m} \sum_{i=1}^{m} \kappa_i(h). \tag{C.6}$$

Because $\varepsilon_S(h) \in [0,1]$ and $\ell(h(z_i), y_i) \in [0,1]$, we have $\varepsilon_S(h) - \ell(h(z_i), y_i) \in [-1,1]$ and $\mathbb{E}[\kappa_i(h)^2] \leq 1$, $|\kappa_i(h)| \leq 1$. By applying Bernstein inequality,

$$\mathbb{P}\left(\frac{1}{m} \sum_{i=1}^{m} \kappa_i(h) > \xi \right) \leq \exp\left(-\frac{\xi^2 m/2}{1 + \xi/3} \right). \tag{C.7}$$

By taking union bound of Eq. C.7 over all $h \in \mathcal{H}'$ with VC-dimension d',

$$\mathbb{P}\left(\cup_{h\in\mathcal{H}'}\frac{1}{m}\sum_{i=1}^{m}\kappa_i(h) > \xi\right) \leq m^{d'}\exp\left(-\frac{\xi^2 m/2}{1+\xi/3}\right). \tag{C.8}$$

Let $\delta = m^{d'}\exp\left(-\frac{\xi^2 m/2}{1+\xi/3}\right)$ and solve the equation for ξ

$$\xi = \frac{d'\log m - \log\delta}{3m} \pm \sqrt{\left(\frac{d'\log m - \log\delta}{3m}\right)^2 + \frac{2(d'\log m - \log\delta)}{m}}. \tag{C.9}$$

Because $\xi \geq 0$ and $\sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$, Eq. C.9 can be simplified as

$$\xi \leq \frac{d'\log m - \log\delta}{3m} + \sqrt{\frac{2(d'\log m - \log\delta)}{m}}. \tag{C.10}$$

Thus, for any $\delta > 0$, with probability at least $1-\delta$, for $h \in \mathcal{H}'$,

$$\varepsilon_S(h) - \hat{\varepsilon}_{S,valid}(h) \leq \frac{d'\log m - \log\delta}{3m} + \sqrt{\frac{2(d'\log m - \log\delta)}{m}}. \tag{C.11}$$

Finally, by applying the bound between the expected domain distance with the empirical domain distance according to (Kifer et al., 2004), we can have Eq. C.5. □

C1.3 Proof of Lemma 6

LEMMA 6. *Let $\varepsilon_\alpha(h)$ be an expected hybrid error weighted by $\alpha \in [0,1]$. For $h \in \mathcal{H}$*

$$\varepsilon_T(h) \leq \varepsilon_\alpha(h) + \alpha\left(\frac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(\widetilde{D}_S, \widetilde{D}_T) + \lambda\right). \tag{C.12}$$

PROOF. According to the triangle inequality for classification error,

$$\varepsilon_T(h) \leq \varepsilon_T(h,h^*) + \varepsilon_T(h^*) \Rightarrow \varepsilon_T(h) - \varepsilon_T(h,h^*) \leq \varepsilon_T(h^*). \tag{C.13}$$

Similarly, for the source domain, we have

$$\varepsilon_S(h) - \varepsilon_S(h,h^*) \leq \varepsilon_S(h^*). \tag{C.14}$$

Therefore, the bound between the expected target error $\varepsilon_T(h)$ and the expected hybrid error $\varepsilon_\alpha(h)$ can be derived by

$$
\begin{aligned}
|\varepsilon_T(h) - \varepsilon_\alpha(h)| &= |\varepsilon_T(h) - \alpha\varepsilon_S(h) - (1-\alpha)\varepsilon_T(h)| \\
&= \alpha|\varepsilon_T(h) - \varepsilon_S(h)| \\
&= \alpha|(\varepsilon_T(h) + \varepsilon_T(h,h^*) - \varepsilon_T(h,h^*)) - (\varepsilon_S(h) + \varepsilon_S(h,h^*) - \varepsilon_S(h,h^*))| \\
&\leq \alpha\,|(\varepsilon_T(h) - \varepsilon_T(h,h^*)) + (\varepsilon_T(h,h^*) - \varepsilon_S(h,h^*)) + (\varepsilon_S(h,h^*) - \varepsilon_S(h))| \\
&\leq \alpha\,(\varepsilon_T(h^*) + |\varepsilon_T(h,h^*) - \varepsilon_S(h,h^*)| + \varepsilon_S(h^*)) \quad \text{(Applying Eqs. C.13 and C.14)} \\
&\leq \alpha\left(\frac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(\widetilde{D}_S, \widetilde{D}_T) + \lambda\right), \quad \text{(Applying Eq. C.4)}
\end{aligned}
$$

where $\lambda = \varepsilon_S(h^*) + \varepsilon_T(h^*)$. □

C1.4 Proof of Corollary C1.1.1

COROLLARY C1.1.1. *Let $\alpha \in [0,1]$ be the weight of the hybrid error, and $\beta \in [0,1]$ be the ratio of i.i.d. samples drawn from $\widetilde{\mathcal{D}}_S$ and $\widetilde{\mathcal{D}}_T$ in a held-out validation set. With probability at least $1-\delta$, for $h \in \mathcal{H}'$*

$$
\begin{aligned}
\varepsilon_T(h) \leq \hat{\varepsilon}_{\alpha,valid}(h) + \left(\frac{\alpha}{\beta} + \frac{1-\alpha}{1-\beta}\right)\left(\frac{d'\log m - \log\delta}{3m} + \sqrt{\frac{2(d'\log m - \log\delta)}{m}}\right) \\
+ \alpha\left(\frac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(\widetilde{\mathcal{U}}_S, \widetilde{\mathcal{U}}_T) + 4\sqrt{\frac{d'\log(2m') + \log(4/\delta)}{m'}} + \lambda\right).
\end{aligned} \tag{C.15}
$$

PROOF. By combining Theorem C1.1 and Lemma 6, we can derive the proof of Corollary C1.1.1. Let $\widetilde{\mathcal{U}}_{\beta,valid}$ be a hybrid validation set with $\boldsymbol{z}_i \in \widetilde{\mathcal{U}}_{\beta,valid}$ for $i \in [1, \beta m]$ from source domain and $\boldsymbol{z}_i \in \widetilde{\mathcal{U}}_{\beta,valid}$ for $i \in [\beta m + 1, m]$ from target domain. Let $\kappa_i(h) = (\alpha/\beta)(\varepsilon_S(h) - \ell(h(z_i), y_i))$ for $i \in [1, \beta m]$, and $\kappa_i(h) = (1-\alpha)/(1-\beta)(\varepsilon_S(h) - \ell(h(z_i), y_i))$

for $i \in [\beta m + 1, m]$. Therefore,

$$\begin{aligned}
&\varepsilon_{\alpha,valid}(h) - \hat{\varepsilon}_{\alpha,valid}(h) \\
=&\alpha(\varepsilon_{S,valid}(h) - \hat{\varepsilon}_{S,valid}(h)) + (1-\alpha)(\varepsilon_{T,valid}(h) - \hat{\varepsilon}_{T,valid}(h)) \\
=&\frac{\alpha}{\beta m}\sum_{i=1}^{\beta m}(\varepsilon_S(h) - \ell(h(z_i), y_i)) + \frac{1-\alpha}{(1-\beta)m}\sum_{i=\beta m+1}^{m}(\varepsilon_S(h) - \ell(h(z_i), y_i)) \\
=&\frac{1}{m}\sum_{i=1}^{m}\kappa_i(h).
\end{aligned} \tag{C.16}$$

The rest of the proof is similar to the proof of Theorem C1.1, but with $\mathbb{E}[\kappa_i(h)^2] \leq (\alpha/\beta + (1-\alpha)/(1-\beta))^2$ and $|\kappa_i(h)| \leq \alpha/\beta + (1-\alpha)/(1-\beta)$. We can have: for any $\delta > 0$, with probability at least $1-\delta$, for $h \in \mathcal{H}'$,

$$\varepsilon_{\alpha,valid}(h) - \hat{\varepsilon}_{\alpha,valid}(h) \leq \left(\frac{\alpha}{\beta} + \frac{1-\alpha}{1-\beta}\right)\left(\frac{d'\log m - \log\delta}{3m} + \sqrt{\frac{2(d'\log m - \log\delta)}{m}}\right) \tag{C.17}$$

Finally, by applying the bound between the expected domain distance with the empirical domain distance according to (Kifer et al., 2004), we can have Eq. C.15. □

C2 Experiment Details

C2.1 NAS Search Space

Following many previous works (Chen et al., 2019; Dong and Yang, 2019; Liu et al., 2018; Zheng et al., 2019; Zoph et al., 2018), we use the NASNet search space (Zoph et al., 2018). There are 2 kinds of cells in the search space, including normal cells and reduction cells. Normal cells use stride 1 and maintain the size of feature maps. Reduction cells use stride 2 and reduce the height and width of feature maps to a half. After a reduction cell, the channel number is doubled. Each cell has 7 nodes, including 2 input nodes, 1 output node and 4 computation nodes. The connection pattern of cells in the NASNet search space is illustrated in Figure C.1, where $\boldsymbol{h}_{i-2}$ and $\boldsymbol{h}_{i-1}$ are input nodes connected to the previous two cells, $\boldsymbol{h}_i$ is an output node concatenating all computation nodes of the current cell, and $\boldsymbol{x}^{(0)}$ to $\boldsymbol{x}^{(3)}$ are computation nodes taking outputs of previous nodes as their inputs and applying some

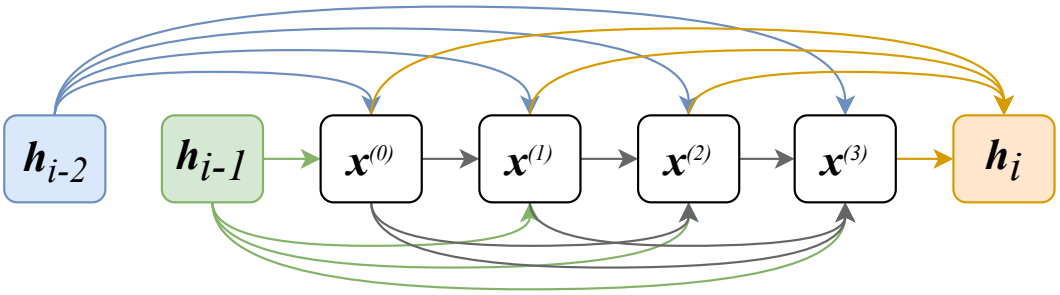


FIGURE C.1. The connection pattern of cells in the NASNet search space.

operations on them. Cells are stacked sequentially to build a network. In the network, cells located at the $1/3$ and $2/3$ are reduction cells, while others are normal cells.

We use a set of 8 different candidate operations, including:

- 3×3 separable convolution;
- 5×5 separable convolution;
- 3×3 dilated separable convolution;
- 5×5 dilated separable convolution;
- 3×3 max pooling;
- 3×3 average pooling;
- identity (i.e., skip-connection);
- zero (i.e., not connected).

All the operations follow the ReLU-Conv/Pooling-BN pattern except identity and zero.

C2.2 Search and Evaluation on Digits

For searching on digits datasets, we use a network with 5 cells, where the 2nd and 3rd cells are reduction cells. The first cell has 16 initial channels. We search for 100 epochs. After searching, we use the same network size for evaluation. The network is trained for 100 epochs.

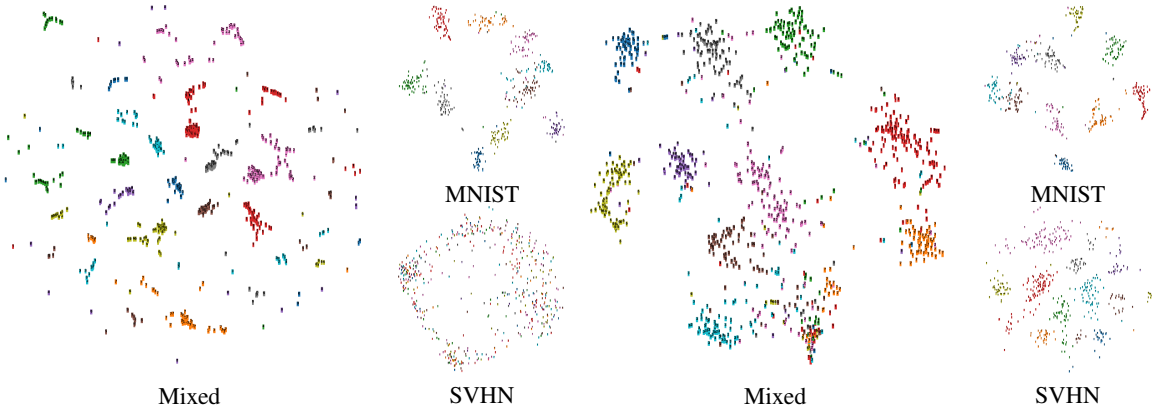
C2.3 Search and Evaluation on CIFAR-10 and ImageNet

For searching on CIFAR-10 and ImageNet, we use a network with 8 cells, where the 3rd and 6th cells are reduction cell. The first cell has 16 initial channels. We search for 200 epochs. For evaluation on CIFAR-10, we use a network with 20 cells and 36 initial channels. The network is trained for 600 epochs with cutout. The network for ImageNet generalization is relatively shallow but wide, which has 14 cells and 48 initial channels. The network is trained for 250 epochs. Auxiliary heads are used for evaluation on both datasets, which is inserted after the 2nd reduction cell. We follow DARTS Liu et al. (2018) and PC-DARTS Xu et al. (2019) and train networks for 600 and 250 epochs on CIFAR-10 and ImageNet, respectively.

C3 More Results

C3.1 Latent Space Visualization (on Digits)

We visualize the latent space learned during search. The setting with MNIST as the source domain and SVHN as the target domain, which has the maximum generalization gap (5.3% test error by searching in the source domain versus 4.4% test error by searching in the target domain), is selected for demonstration. 500 random samples from each domain are chosen. Figure C.2 shows the latent space learned by different searching methods, including searching in the source domain only and searching with AdaptNAS-S. The origin latent representation is 256-dimension and is reduced to 2-dimension with t-distributed Stochastic Neighbor Embedding (t-SNE). Figure C.2a shows the latent space learned by searching in the source domain only. As can be seen, samples from MNIST clusters by their labels, while samples from SVHN distributes almost randomly. When they are mixed together, there are more than one cluster for each label. Figure C.2b shows the latent space learned by AdaptNAS-S. Samples from both domains clusters well. When they are mixed together, there are only one major cluster for each label with several outliers.

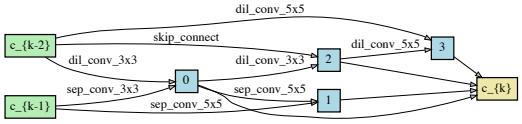


(A) Search in the source domain; (B) Search with AdaptNAS-S.

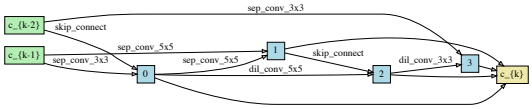
FIGURE C.2. Latent spaces learned by different searching methods with MNIST as the source domain and SVHN as the target domain. The dimension is reduced with t-SNE. Different colors stand for different categories. There are 10 categories for different digits from 0 to 9.

C3.2 Architectures of Reported Results (on CIFAR-10 and ImageNet)

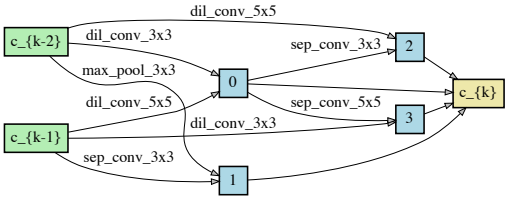
Figure C.3 shows architectures of the reported results compared with SOTAs in the paper. Both normal and reduction cells found by different AdaptNAS settings are provided.



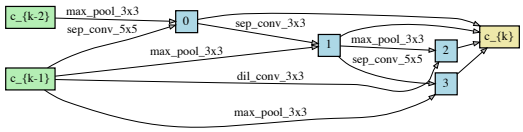
(A) Normal: AdaptNAS-S (Rot-1);



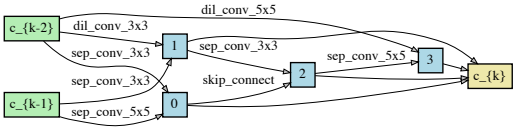
(B) Reduction: AdaptNAS-S (Rot-1);



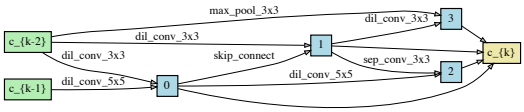
(C) Normal: AdaptNAS-S (Rot-4);



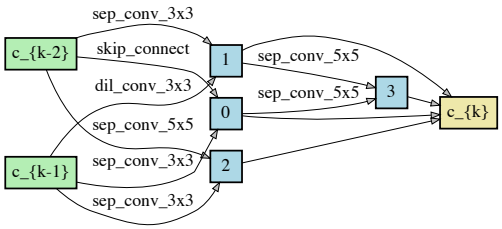
(D) Reduction: AdaptNAS-S (Rot-4);



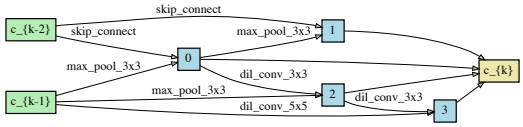
(E) Normal: AdaptNAS-C (Rot-1);



(F) Reduction: AdaptNAS-C (Rot-1);



(G) Normal: AdaptNAS-C (Rot-4);



(H) Reduction: AdaptNAS-C (Rot-4);

FIGURE C.3. Architectures found by different settings of AdaptNAS.