Understanding Adversarial Robustness in Deep Learning

by

Bojie Ma

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy

Department of Computer Science
University of Toronto

Understanding Adversarial Robustness in Deep Learning

Bojie Ma
Doctor of Philosophy

Department of Computer Science
University of Toronto
2025

Abstract

This thesis studies the adversarial robustness of deep learning models. Our investigation covers various aspects of this phenomenon, including the development of two new defense algorithms, two new attack algorithms, a novel definition of hierarchical adversarial robustness, and an analysis of how the optimization process affects model robustness.

We begin by introducing the Second-Order Adversarial Regularizer (SOAR) as a defense strategy to improve model robustness. Unlike traditional data augmentation approaches that rely on computationally expensive algorithms to generate adversarially perturbed training samples, we derive a regularizer that mimics the effect of data augmentation, eliminating the need for it during training. We empirically demonstrate the improved adversarial robustness of SOAR-regularized models against white-box and transferred perturbations.

In practice, many machine learning systems still forgo robustification techniques due to the additional computational overhead. This motivates our investigation into the robustness of models obtained through standard training regimes. Focusing on the optimization process, we examine the robustness of models trained with different algorithms. As we will see, models trained using stochastic gradient descent exhibit far greater robustness compared to those trained with adaptive gradient methods, such as Adam and RMSProp. Through a frequency-domain analysis, we discover that specific properties of datasets, seemingly irrelevant to generalization, can actually result in vulnerabilities when models are trained using certain optimizers. These insights underscore the importance of considering both the optimization strategy and dataset characteristics in improving model robustness.

Extending our exploration of dataset properties, we recognize that as datasets grow in size and complexity, the number of classes and their hierarchical relationships become increasingly significant. However, current ways of evaluating adversarial robustness, which treat all misclassifications equally, risk overestimating robustness or underestimating attack effectiveness. To address this, we introduce the concept of hierarchical adversarial robustness. For datasets with hierarchical structures, we define hierarchical adversarial examples as those that lead to misclassifications at the meta-class

level. Building on this, we develop an attack algorithm designed to generate such examples and propose an architectural solution to improve models' robustness against them.

Finally, we take on the role of an adversary. Current robustification strategies predominantly involve replacing the training samples with their adversarially perturbed counterparts, highlighting the importance of understanding and improving adversarial example generation. Therefore, in our last work, we focus on improving the transferability of perturbations. We develop a fine-tuning method called model alignment to transform any model into one from which any attack algorithms can generate more transferable perturbations.

Acknowledgements

I thank my parents for their unconditional love. This love has shaped who I am today, and I hope to continue making you proud.
I thank my advisor, Amir-massoud Farahmand. You taught me to approach research and life with a scientific mind, guiding me through an often noisy world.
I thank my co-advisor, Rich Zemel, and my committee chair, Nicolas Papernot. You showed me what great researchers are like and set a high bar to aspire to.
I thank the senior PhD students I met along the way. Our collaborations taught me lessons no course ever could.
I thank my friends. Your presence was a constant source of joy and laughter throughout this journey.
I thank my dog, AUV. Your companionship meant more than words can say.
Lastly, I thank my wife, Abby Zhu. You are my best friend and greatest supporter. I am excited about our future together.

Contents

1 Introduction: The Brittleness of Neural Networks **1**

1.1 Contributions . . . 3

2 Adversarial Perturbations: Where to Find Them and How to Avoid Them **6**

2.1 Definitions . . . 6

2.2 Conjectures on the Existence of Adversarial Perturbations . . . 8

2.3 Generating Adversarial Perturbations . . . 10

2.4 The Robust Optimization Framework . . . 11

2.5 The Robustness-Accuracy Trade-off . . . 12

2.6 The Transferability of Adversarial Perturbations . . . 13

3 Regularized Training for Improving Adversarial Robustness **15**

3.1 Introduction . . . 15

3.1.1 Contributions . . . 16

3.2 Understanding Adversarial Training Using Linear Regression with an Over-parametrized Model . . . 16

3.3 Adversarial Regularization: An Alternative to Adversarial Training . . . 19

3.3.1 First-Order Adversarial Regularization . . . 19

3.3.2 Second-Order Adversarial Regularization . . . 20

3.3.3 Revisiting the Linear Regression Example . . . 22

3.3.4 Avoiding Gradient Masking . . . 23

3.3.5 Related Work . . . 25

3.4 Experiments . . . 26

3.4.1 Experiment Setup . . . 26

3.4.2 Evaluating Model Robustness . . . 27

3.5 Conclusions . . . 30

3.5.1 Challenges and Limitations . . . 30

3.A Implementation Details . . . 31

3.B Effect of the Number of Randomly Sampled z on SOAR Regularized Loss . . . 32

3.C SOAR and FOAR with Different Initializations . . . 33

3.D Robustness Under ℓ_2-norm Constrained Perturbations . . . 33

3.E Robustness Improvement with Increasing Model Capacities . . . 34

3.F Discussion on Gradient Masking . . . 34

4 Understanding the Robustness Difference between SGD and Adaptive Gradient Methods **36**
4.1 Introduction . . . 36
4.1.1 The Robustness Difference between Models Trained by Different Algorithms . 37
4.1.2 Contributions . . . 38
4.2 Background . . . 38
4.2.1 Optimizations with Adaptive Gradient Algorithms . . . 38
4.2.2 Frequency Representation of Signals . . . 40
4.3 A Claim on How Models Use Irrelevant Frequencies . . . 41
4.3.1 Observation I: Irrelevant Frequencies in Natural Signals . . . 42
4.3.2 Observation II: Model Robustness along Irrelevant Frequencies . . . 45
4.4 Linear Regression Analysis with an Over-parameterized Model . . . 47
4.4.1 Problem Setup . . . 47
4.4.2 Analysis on the Learning Dynamics of GD and signGD . . . 49
4.5 Connecting the Norm of Linear Models to the Lipschitzness of Neural Networks . . . 56
4.6 Conclusions . . . 58
4.6.1 Challenges and Limitations . . . 58
4.A Implementation Details . . . 59
4.B Generalization and Robustness Results . . . 60
4.C Filtering Irrelevant Frequencies . . . 64
4.D Linear Regression Analysis . . . 65
4.D.1 Understanding the Synthetic Dataset . . . 65
4.D.2 Understanding the Dynamics of signGD with $\tilde{\Sigma} = \operatorname{diag}\left\{\tilde{\sigma}_0^2, \tilde{\sigma}_1^2, 0\right\}$ . . . 68
4.D.3 From Irrelevant Frequencies to Spatially Redundant Dimensions . . . 82
4.E Additional Figures . . . 83

5 Understanding and Improving Hierarchical Adversarial Robustness **94**
5.1 Introduction . . . 94
5.1.1 Contributions . . . 95
5.2 Hierarchical Adversarial Robustness . . . 95
5.3 Untargeted PGD's Ineffectiveness in Degrading Hierarchical Adversarial Robustness 97
5.4 Generating Hierarchical Adversarial Perturbations . . . 98
5.5 Hierarchical Adversarial Robust Network . . . 99
5.5.1 Network Design . . . 100
5.5.2 Inference . . . 100
5.5.3 Training . . . 101
5.6 Experiments . . . 101
5.6.1 Experiment Setup . . . 102
5.6.2 Evaluation of Hierarchical Adversarial Robustness Using Hierarchical PGD . 102
5.6.3 Improving Hierarchical Adversarial Robustness with HAR . . . 103
5.7 Conclusions . . . 104
5.7.1 Challenges and Limitations . . . 104
5.A Comparison of Trainable Parameters . . . 105
5.B Results on CIFAR-10 and CIFAR-5x5 with ℓ_∞-norm Constrained Perturbations . . . 105

5.C Results on CIFAR-100 with ℓ_2-norm Constrained Perturbations . . . 106
5.D Comparison of Clean Test Accuracy on CIFAR-10 . . . 106

6 Improving Adversarial Transferability via Model Alignment **107**
6.1 Introduction . . . 107
6.1.1 Contributions . . . 109
6.2 Background . . . 109
6.2.1 Generating Transferable Perturbations . . . 109
6.2.2 Understanding Adversarial Transferability . . . 110
6.3 Formulation of Model Alignment . . . 111
6.4 Understanding Model Alignment . . . 112
6.4.1 Evaluating Similarity between the Source and Witness Model . . . 112
6.4.2 Aligned Model Exploits More Semantic Features . . . 113
6.4.3 Model Alignment Yields Smoother Loss Surface . . . 113
6.5 Experiments . . . 115
6.5.1 Experiment Setup . . . 115
6.5.2 Model Alignment Improves Transferability . . . 116
6.5.3 Ablation Studies . . . 118
6.6 Conclusions . . . 123
6.6.1 Challenges and Limitations . . . 123
6.A Implementation Details . . . 123
6.B Experiments on Additional Datasets . . . 124
6.C Improved Transferability on Defended Models . . . 124

7 Concluding Remarks **126**
7.1 Suggestions for Future Research . . . 126

Bibliography **129**

List of Tables

3.2 Evaluation of classification accuracy for models on the original CIFAR-10 test dataset and under ℓ_∞-norm constrained white-box attacks (%). 27
3.3 Evaluation of classification accuracy for models under ℓ_∞-norm constrained perturbations in the black-box setting (%). 28
3.4 Evaluation of classification accuracy for models under ℓ_∞-norm constrained perturbations generated using AutoAttack (%). 29
3.9 Evaluation of classification accuracy for models under ℓ_2-norm constrained white-box and transferred perturbations (%). 34

4.1 Comparing the upper bound on the Lipschitz constant and the averaged robust accuracy of neural networks trained by SGD, Adam, and RMSProp. 57
4.3 Evaluation of classification accuracy for models trained using SGD, Adam, and RMSProp on both the original and perturbed test dataset (%). 61
4.8 Examples of the synthetic data distribution in the frequency and the spatial domain. 65
4.9 The dynamics of the error term in the frequency domain under signGD. 69

5.1 Hierarchical structure of classes within the CIFAR-10 and CIFAR-100 dataset. . . . 96
5.2 Percentage of the misclassified inputs that are still correctly classified at the meta-class level (%). 98
5.3 Evaluation of meta-class accuracy for flat models and HAR models on the original CIFAR-100 test dataset, under ℓ_∞-norm constrained untargeted and proposed hierarchical PGD attacks (%). 103
5.4 Evaluation of meta-class accuracy for the HAR models under ℓ_∞-norm untargeted PGD perturbations generated based on the meta-classifier (%). 104
5.6 Evaluation of meta-class accuracy for flat models and HAR models on the original CIFAR-100 test dataset, under ℓ_2-norm constrained untargeted and proposed hierarchical PGD attacks (%). 106
5.7 Evaluation of leaf-class accuracy for flat models and HAR models on the CIFAR-10 test dataset. 106

6.1 Evaluation of the change in the similarity between the source and witness model after model alignment. 112
6.2 Evaluation of the ℓ_2-norm of the gradient ($\|\nabla_x \ell(x + \Delta x, y, \theta)\|_2$) and the largest eigenvalue of the Hessian ($\lambda_{\max}(\nabla_x^2 \ell(x + \Delta x, y, \theta))$) on the original ($\theta_s$) and aligned model ($\theta_a$). 115

6.3 Aligning the source model can lead to the generation of more transferable adversarial examples (%). 117
6.4 Analysis on the impact of witness model capacity on the alignment process (%). . . 119
6.5 Aligning the Res50 source model with multiple Res18 as witness models (%). 120
6.6 Improving transferability via embedding space alignment (%). 120
6.7 Compatibility of model alignment with different attack algorithms (%). 121
6.8 Compatibility of model alignment with model modification-based transfer-enhancing attack algorithms (%). 122
6.9 Improved transferability from using aligned models in an ensemble (%). 122
6.10 Improved transferability via model alignment on other datasets (%). 125
6.11 Improved transferability on models equipped with defense methods (%). 125

List of Figures

4.1 Comparison between models trained using SGD, Adam, and RMSProp across seven benchmark datasets. . . . 37
4.2 Illustration of the spectral energy distribution in natural datasets. . . . 42
4.3 Irrelevant frequencies exist in natural datasets. . . . 43
4.4 Examples of filtered images used in Observation I. (Imagenette) 44
4.5 Visualization of the band-limited Gaussian perturbations. . . . 45
4.6 The effect of band-limited Gaussian perturbations on the model. . . . 46
4.7 Empirical validations on (a) the learning dynamics, (b) the standard and adversarial population risk of linear models optimized under GD, Adam, RMSProp, and signGD. 55
4.8 Analyzing the dynamics of A and B by partitioning the set of values of (A, B) in $[-\tilde{\sigma}_0^2\eta, \tilde{\sigma}_0^2\eta] \times \mathbb{R}$. . . . 72

5.1 Overview of the HAR Network. . . . 101

6.1 Attacking the aligned source model for more transferable perturbations. . . . 108
6.2 Frequency-domain visualization of the differences in the perturbation generated using the original and aligned model. . . . 113
6.3 Visualization of the loss surface around adversarial perturbations generated from the original and aligned model. . . . 114

Chapter 1

Introduction: The Brittleness of Neural Networks

Machine learning models are powerful statistical tools. In theory, it was guaranteed that multilayer perceptrons with just a single hidden layer possess significant expressive power as universal function approximators (Hornik et al., 1989). In practice, deep neural networks (DNNs) with billions of parameters can complete a wide range of tasks when provided with sufficient training data — they are capable of classifying objects, translating languages, and even driving cars.

Despite their power, machine learning models are highly brittle. By introducing carefully crafted perturbations to the input, models trained for weeks or months can be rendered useless, producing incorrect results that would otherwise have been accurate (Szegedy et al., 2014). These manipulated inputs are known as **adversarial examples**, and the algorithms used to generate them are called **attacks**. The ability of models to maintain their performance under such perturbations is referred to as **adversarial robustness**, while methods aimed at enhancing this robustness are known as **defenses**. What makes adversarial examples particularly interesting is that the perturbations are often imperceptible to human observers. This underscores the importance of adversarial robustness in deep learning models, especially when deploying them in safety-critical applications. As a result, understanding and improving model robustness has become a significant area of research (Bai et al., 2021).

If neural networks are Superman, then adversarial examples are their Kryptonite, exposing vulnerabilities in these powerful models. However, this is a rather pessimistic view of the situation. What people often overlook is that tracing the origin of Kryptonite leads us to the planet where Superman was born. In other words, studying adversarial examples can provide insights into the inner workings of deep learning models. By understanding the vulnerabilities of these models, we can develop more robust models that are less susceptible to attacks.

In this thesis, adversarial examples are the protagonists of our story. With a focus on the vision domain, we will explore the adversarial robustness of deep learning models, from developing more effective defense and attack algorithms to understanding particular components of the model training pipeline that contribute to vulnerability. In the following, we first provide a few examples to illustrate the consequence of attacks on machine learning models, followed by a summary of the contributions of this thesis.

Beyond the Pig and the Airplane

Perhaps the most common example used to illustrate the effect of an attack is the pig/airplane scenario. In this case, imperceptible perturbations added to an image of a "pig" can lead a highly accurate classifier to misidentify it as an "airplane". While this example adeptly captures the essence of adversarial examples, it does not fully represent the real-world consequences of such attacks on machine learning models.

NeuralHash was an algorithm developed by Apple to detect child sexual abuse material (CSAM) by creating unique digital fingerprints, or "hashes", of images (Apple, 2021). The system was designed to be used to scan and hash iCloud photos and compare them against a database of known CSAM hashes, provided by the National Center for Missing and Exploited Children. By doing so, Apple can identify matches without directly accessing the image content, thereby preserving user privacy.

Traditional hashing algorithms are not suitable for this task because they are sensitive to minor changes in the input (Luo et al., 2023; Chi et al., 2017). For example, a single pixel change could alter the resulting hash significantly, enabling CSAM perpetrators to easily evade detection by making trivial modifications to images.

NeuralHash addressed this by employing a convolutional neural network trained within a contrastive framework. Given pairs of images, it learned to assign similar hashes to positive pairs (images that are simple transformations of each other) and different hashes to negative pairs (completely different images). This way, the hashing algorithm becomes less sensitive to minor changes in the input.

However, vulnerabilities were exposed shortly after NeuralHash's release (Athalye, 2021). Researchers demonstrated that adding imperceptible perturbations to an image could manipulate its hash. This led to two critical vulnerabilities. First, in the "same image, different hash" scenario, an adversary slightly can alter an illegal image (e.g., CSAM), so that it no longer matches its known hash. This evades detection by law enforcement and undermines the system's ability to flag prohibited content. Second, in the "different image, same hash" scenario, an adversary can create a benign image that shares the same hash as known illegal content. This could be weaponized to falsely implicate innocent users or traumatize recipients. Both cases fundamentally compromise the system's reliability and safety.

Autonomous driving systems are among the most safety-critical applications of machine learning models, where the consequences of attacks can be catastrophic (Baidu, 2017; comma.ai, 2018; Tesla, 2018). For example, a 2019 study showed that placing adversarially generated stickers on a stop sign could cause an autonomous vehicle to misclassify it as a speed limit sign (Eykholt et al., 2018). This misclassification could result in the vehicle failing to stop at an intersection, thereby increasing the risk of a collision. While this type of attack is indeed alarming, some argue that it is more practical, low-cost, and reliable to simply remove the stop sign from the intersection (Woitschek et al., 2021). Therefore, such attacks on traffic signs, much like the pig/airplane example, serve more as proofs of concept rather than practical threats.

For systems that rely primarily on camera footage, such as Tesla's Autopilot, lane detection is crucial for maintaining vehicle control. Jing et al. (2021) demonstrated that subtle perturbations added to road markings could mislead a Tesla Model S into veering into the opposite lane. By reverse-engineering Tesla Autopilot's firmware, they discovered the sensitivity of the lane detection

module to such manipulations. The researchers showed that these perturbations could be practically implemented in the real world using materials like stickers, and potentially more covert substances like special paints, which are invisible to the human eye but detectable by cameras. This suggests a more concerning potential for adversaries to disrupt lane detection systems, posing severe safety risks.

Facial recognition systems have seen significant performance improvements due to advancements in deep learning, with applications that range from unlocking smartphones to automated security screenings at border control and military facilities (Kortli et al., 2020). Compromising these systems can have serious consequences. Research has demonstrated that these systems can be deceived by altering a person's physical appearance before capturing their image. Some methods for fooling these systems involve projecting specific patterns onto the face (Zhou et al., 2018), which can be cumbersome. In contrast, others are more practical, utilizing simple accessories like glasses or hats (Sharif et al., 2016; Komkov et al., 2021). This poses substantial security risks, especially where adversaries might bypass checkpoints by using such disguises. For a comprehensive survey on attacks against facial recognition systems, we refer the reader to the work of Vakhshiteh et al. (2021).

1.1 Contributions

The pig/airplane example illustrates a fundamental concept: subtle input perturbations can deceive machine learning models. Meanwhile, the NeuralHash incident exposes the severe real-world implications of such vulnerabilities, demonstrating the potential for misuse. Both autonomous driving and facial recognition systems underscore the critical need for robustness in machine learning applications, especially in those used for security and safety purposes. My goal of this thesis is to investigate various aspects of the adversarial robustness phenomenon, ranging from the development of more effective defense and attack algorithms to understanding specific components of the model training pipeline that contribute to vulnerability. The high-level contributions of this thesis include:

- Introducing two robustification methods: one algorithmic and the other architectural, both designed to improve the adversarial robustness of deep learning models;
- Introducing two attack algorithms: one targeting hierarchical class structures in datasets and the other improving the transferability of perturbations;
- Investigating the robustness difference between models trained by different optimization algorithms.

Throughout this thesis, we focus on the supervised learning setting with applications in computer vision, where the input data is an image and the output is a class label. In addition to this introductory chapter that motivates my research work, the thesis comprises four chapters presenting new contributions (Chapters 3 to 6). It also includes a chapter (Chapter 2) that provides the reader with the necessary background related to adversarial robustness, focusing particularly on topics relevant to the work introduced in this thesis. For background materials that are directly related to the individual chapters, an additional background section will be provided at the beginning of the respective chapter. Chapter 7 summarizes the thesis, highlighting its limitations and suggesting potential areas for future investigation. In the remainder of this section, we summarize the key contributions of each chapter.

SOAR: Second-Order Adversarial Regularization (Chapter 3)

The most common approach to improving the robustness of DNNs against adversarial examples is to train with perturbed inputs. However, this method requires large amount of additional data (Schmidt et al., 2018). In this chapter, we propose a novel regularization approach as an alternative. To derive the regularizer, we formulate the adversarial robustness problem under the robust optimization framework (Chapter 2.4) and approximate the loss function using a second-order Taylor series expansion. The proposed *second-order adversarial regularizer* (SOAR) is an upper bound based on the Taylor approximation of the inner-max in the robust optimization objective. We empirically show that the proposed method significantly improves the robustness of networks against adversarial perturbations. This work is reported as (Ma et al., 2020), a joint work with Fartash Faghri, Nicolas Papernot, and Amir-massoud Farahmand.

Understanding the robustness difference between stochastic gradient descent and adaptive gradient methods (Chapter 4)

Despite attention from the research community, robustness-improving methods are yet to become the norm for machine learning practitioners due to the additional computational overhead. Most people still train their models using the standard training regime, selecting from a pool of optimizers and using the model with the highest validation accuracy. In this chapter, we investigate the robustness differences between models trained with different optimizers. We first empirically show that while the difference between the standard generalization performance of models trained using stochastic gradient descent (SGD) and adaptive gradient methods, such as Adam (Kingma et al., 2015) and RMSProp (Hinton et al., 2012) is small, those trained using SGD exhibit far greater robustness under input perturbations. Notably, we demonstrate the presence of irrelevant frequencies in natural datasets, where noise added to these frequencies do not impact the models' generalization performance. However, models trained with adaptive methods show sensitivity to these changes, suggesting that their use of irrelevant frequencies can lead to solutions sensitive to perturbations. To better understand this difference, we study the learning dynamics of gradient descent (GD) and sign gradient descent (signGD) on a synthetic dataset that mirrors natural signals. With a three-dimensional input space, the models optimized with GD and signGD have standard risks close to zero but vary in their adversarial risks. Our result shows that linear models' robustness to ℓ_2-norm bounded changes is inversely proportional to the model parameters' weight norm: a smaller weight norm implies better robustness. In the context of deep learning, our experiments show that SGD-trained neural networks have smaller Lipschitz constants, explaining the better robustness to input perturbations than those trained with adaptive gradient methods. This work is reported as (Ma et al., 2023a), a joint work with Yangchen Pan and Amir-massoud Farahmand.

Improving hierarchical adversarial robustness of deep neural networks (Chapter 5)

The properties of dataset can play a key role in model robustness, as we later demonstrate in Chapter 4. As datasets grow in size and complexity, the number of classes and the hierarchical structure of the classes also become increasing significant. However, we identify an overlooked aspect in the

current literature: adversarial examples can be dangerous, but not all mistakes have equal consequences. In this chapter, we introduce a novel concept called hierarchical adversarial robustness. For datasets where classes can be grouped into meta-classes, we define hierarchical adversarial examples as those leading to misclassification at the meta-class level. In terms of attacks, we demonstrate that untargeted attacks are ineffective at generating hierarchical adversarial perturbations. Building on this, we propose a hierarchical attack algorithm specifically designed to generate such examples. On the defense side, we propose an architectural solution to improve hierarchical adversarial robustness, utilizing an ensemble approach with a meta-classifier and multiple leaf classifiers, each trained independently using adversarial defense techniques. Our empirical results show that the new model design significantly increases hierarchical adversarial robustness. This work is reported as (Ma et al., 2021), a joint work with Aladin Virmaux, Kevin Scaman, and Juwei Lu.

Improving Adversarial Transferability via Model Alignment (Chapter 6)

Currently, the most effective methods for improving adversarial robustness rely on some form of adversarial training. Developing more effective and potent perturbations can often translate to building more robust models. This motivates us to understand and improve the generation of adversarial examples. In this chapter, we focus on the transferability of adversarial examples and introduce a novel model alignment technique aimed at improving a given source model's ability to generate transferable adversarial perturbations. During the alignment process, the parameters of the source model are fine-tuned to minimize an alignment loss. This loss measures the divergence in the predictions between the source model and another, independently trained model, referred to as the witness model. To understand the effect of model alignment, we conduct a geometric analysis of the resulting changes in the loss landscape. Extensive experiments, using a variety of model architectures, demonstrate that perturbations generated from aligned source models exhibit significantly higher transferability than those from the original source model. This work is reported as (Ma et al., 2024), a joint work with Amir-massoud Farahmand, Yangchen Pan, Philip Torr, and Jindong Gu.

Chapter 2

Adversarial Perturbations: Where to Find Them and How to Avoid Them

The goal of this chapter is to provide the background knowledge in adversarial machine learning upon which this thesis is developed. In Section 2.1, we first introduce the basic setup for a machine learning problem and define key terms used throughout this thesis. Alongside these definitions, we also provide a brief introduction to commonly used jargon in the literature on adversarial robustness. Understanding these concepts clearly is crucial for grasping the broader field, and this thesis will focus on a subset of them. In Sections 2.2 to 2.6, we discuss a selection of topics that are particularly helpful in understanding the technical components of this thesis. Further related work, specific to individual chapters, is developed in their subsequent chapters. For a comprehensive introduction to deep learning, we refer readers to textbooks such as Hastie (2009) and Goodfellow et al. (2016).

2.1 Definitions

The Basic ML Setup

We focus on the supervised learning setting where we are given a distribution $(X, Y) \sim \mathcal{D}$, with X representing the input and Y the corresponding target. The domains of X and Y depend on the specific context of the learning task. The goal of learning is to obtain a function $f : X \times W \to Y$ that maps inputs X to outputs Y, and is parameterized by a member of the parameter space W. Those functions are more often referred to as models, and here we use W to collectively represent the model parameters. The performance of this model is measured by a loss $\ell : Y \times Y \to \mathbb{R}$ that quantifies the difference between the model output and the true output. The pointwise loss on a single input-output pair can be denoted as $\ell(x, y; w) \triangleq \ell(f(x; w), y)$. During learning, we are given samples from the distribution $\mathcal{D}$ in the form of a dataset $\{(x_i, y_i)\}_{i=1}^N$ and the model f is learned by minimizing the average loss over the samples: $w^* = \arg\min_w \frac{1}{N} \sum_{i=1}^N \ell(x_i, y_i; w)$. This process is known as empirical risk minimization (ERM).

The ERM process is a fundamental principle in machine learning and is used to learn a model

that generalizes well to unseen data. The generalization performance of the model is evaluated using a test set, which comprises a separate set of samples drawn from the same distribution $\mathcal{D}$ as the training data but not used during the training phase. The estimation error is typically defined as the difference between the model's performance on the test set and its performance on the training set.

In the context of adversarial machine learning, the robustness of a model refers to its ability to maintain consistent generalization performance, even when small perturbations are introduced to its inputs. Essentially, a model is considered robust if its loss does not increase significantly under slight input modifications. For example, this means that $\ell(x, y; w) \approx \ell(x + \delta, y; w)$, where δ represents a small perturbation. This property is crucial for ensuring that the model performs well in the presence of attacks. We will discuss this in more detail in the subsequent sections.

There are several factors that can influence the generalization and robustness of the learned model. We discuss them below and relate them to relevant parts of this thesis where they are further explored.

Optimization Objective: The choice of the loss function ℓ can depend on factors such as the type of learning task. For regression problems where outputs are continuous, i.e., $Y \in \mathbb{R}$, the mean squared error, mean absolute error, or Huber loss are commonly used. For classification problems where the true outputs are categorical, i.e., $Y \in \mathbb{C} = \{1, 2, \ldots, C\}$ with C classes in total, the cross-entropy loss is widely used. The optimization objective often consists of a loss function with additional terms known as regularizers. Common regularizers, such as ℓ_1 and ℓ_2 regularization, encourage weights to have small magnitudes, often leading to improved generalization and robustness. We will explore optimization, regularizations, and their effects on model robustness in more detail in Section 2.4 and Chapter 3.

Model Architecture: The architecture of the model can vary significantly depending on its intended purpose. For example, linear models are analysis-friendly and easy to interpret, making them suitable for understanding the dynamics under different optimization objectives or parameter update rules. Conversely, convolutional neural networks (CNNs) and more recent transformers are favored in practice due to their ability to capture complex relationships within the data. Throughout this thesis, we primarily focus on linear models for analytical purposes, while CNNs are the main models used to demonstrate our experimental results. Chapter 5 introduces a novel neural network architecture designed to improve robustness when the dataset exhibits specific structures.

Optimization Algorithm: The performance of a model can also be significantly influenced by the choice of optimization algorithm, which depends on the properties of the dataset, the characteristics of the optimization objective, and the availability of computational resources. GD updates the parameters of the model in the direction of the negative gradient of the loss function. Its variant, SGD, updates parameters using a random subset of the training data, known as a mini-batch. This can lead to faster convergence and is particularly useful when the dataset is large. Adaptive optimization algorithms such as Adam (Kingma et al., 2015) and RMSprop (Hinton et al., 2012) are also commonly used in practice. In Chapter 6, we formally present the update rule under those algorithms and investigate how they can influence the robustness of the model.

Adversarial Machine Learning: An Overview

In the beginning of Chapter 1, we described what attacks and defenses are from a very high level. Below, we provide additional definitions commonly used in adversarial machine learning literature.

Adversarial examples are inputs intentionally designed to fool a machine learning model. We focus on adversarial examples created by adding small perturbations to the original input data, referred to as **adversarial perturbations**. To study them, it is important to formalize and constrain the power of the adversary. One key reason for introducing such constraints is that they provide a mathematically precise way to define the adversary's capabilities, making both theoretical analysis and empirical evaluation more tractable. Specifically, we restrict the adversary to modify an input x to $x + \delta$ where $\delta \in \Delta$ and Δ denotes the constraint set. We focus on the commonly-used ϵ-balls constraint with respect to (w.r.t.) the ℓ_p-norms: $\Delta = \left\{ \delta : \|\delta\|_p \leq \epsilon \right\}$, and use $B_p(x, \epsilon)$ to denote the ℓ_p-ball with radius ϵ centered around data point x. Note that other constraint sets have been investigated too (Wong et al., 2019b).

Perturbations are usually **input-specific**, meaning that they are designed to cause misclassification for a specific input. In contrast, **universal adversarial perturbations** are effective against more than one input. Perturbations can also be **non-targeted**, where the model predicts any incorrect label, or **targeted**, where the model predicts a specific incorrect label. In this thesis, we mostly focus on input-specific non-targeted attacks, except in Chapter 5, where we investigate the effects of targeted attacks.

Adversaries can be characterized using the information they can use and the actions they can take to craft perturbations. When evaluating the security consequences of adversarial examples, the term **threat model** is often used to define the capacity of the attackers (Papernot et al., 2016). For instance, a **white-box attack** assumes complete knowledge of the target model, including its parameters, architecture, and its training data. On the contrary, a **black-box** threat model refers to the scenario where the attacker does not know anything about the model under attack. Although the specifics of the model are unknown, adversaries can still interact with it by observing the outputs for any given input. The black-box threat model will be the focus of Chapter 6, where we explore the transferability of adversarial examples.

In the following sections, we discuss a selection of topics that are particularly helpful in understanding this thesis. While some of the works discussed might appear to be outdated, they are included here because they provide the foundational concepts upon which many subsequent advancements have been built. For more recent studies and works closely related to the specific topics covered in each chapter, we will review and discuss them in detail in the subsequent chapters.

2.2 Conjectures on the Existence of Adversarial Perturbations

Goodfellow et al. (2015) hypothesized that the neural network's vulnerability to adversarial perturbations primarily comes from their linear nature. They demonstrated using a simple linear model that infinitesimal changes made to a high-dimensional input can sum up to one large change to the output, thereby creating an adversarial example. Despite their non-linear transformations, many widely-used activation functions, such as ReLU and Sigmoid, possess linear attributes—ReLU's lin-

ear region and Sigmoid's non-saturating region—which facilitate easier optimization. As such, the authors advanced the conjecture that the existence of adversarial examples in DNNs is largely due to these linear properties.

This hypothesis led to the development of the Fast Gradient Sign Method (FGSM), a technique for generating adversarial perturbations constrained by the ℓ_∞-norm:

$$\begin{aligned}\delta_{\text{FGSM}} &= \arg\max_{\|\delta\|_\infty \leq \epsilon} \tilde{\ell}_{\text{1st}}(x+\delta) \\ &= \arg\max_{\|\delta\|_\infty \leq \epsilon} \ell(x) + \nabla_x \ell(x)^\top \delta \\ &= \epsilon \, \text{sign}(\nabla_x \ell(x)),\end{aligned} \tag{2.1}$$

where the perturbation δ_{FGSM} is the solution to the constrained optimization problem by considering the first-order Taylor approximation of the loss $\tilde{\ell}_{\text{1st}}(x+\delta)$ around the input. Note that we use $\nabla_x \ell(x)$ to denote the gradient of the loss function w.r.t. the input x.

Earlier work by Szegedy et al. (2014) showed that the box-constrained L-BFGS method can reliably generate adversarial examples, though it is a computationally intensive procedure that requires optimization for each input individually. In contrast, FGSM offers a highly efficient alternative, as (2.1) only involves computing a single-step gradient of the loss w.r.t. the input, which can be easily parallelized across an entire batch of data.

Moreover, (2.1) can be easily adapted to generate perturbations under other ℓ_p-norm constraints. For example, the ℓ_2 variant of FGSM, which we refer to as the Fast Gradient Method (FGM), can be formulated as:

$$\begin{aligned}\delta_{\text{FGM}} &= \arg\max_{\|\delta\|_2 \leq \epsilon} \ell(x) + \nabla_x \ell(x)^\top \delta \\ &= \epsilon \frac{\nabla_x \ell(x)}{\|\nabla_x \ell(x)\|_2}.\end{aligned} \tag{2.2}$$

FGSM was simple but very effective. The authors showed that a small maxout network (Goodfellow et al., 2013) misclassifies 87% of the FGSM-perturbed test set data with $\epsilon = 0.1$ on MNIST.

Empirical evidence supporting the linear hypothesis was further discussed in Tramèr et al. (2017). Other hypotheses for the existence of neural network's vulnerability to adversarial examples have also been explored extensively. Song et al. (2018), Lee et al. (2017), and Stutz et al. (2019) suggested that adversarial examples lie off the data manifold and are sampled from a different distribution than the unperturbed dataset. Such a hypothesis has led to the development of adversarial detection methods (Grosse et al., 2017; Gong et al., 2017). Conversely, Gilmer et al. (2018) refuted the hypothesis that adversarial examples are primarily caused by a distribution shift. Using a synthetic dataset with concentric hyperspheres, they demonstrated that adversarial examples can still be found as long as the learned model has a non-zero error rate, even when the adversarial search space is restricted to the data manifold. Additional hypotheses, such as the trade-off between robustness and accuracy (Tsipras et al., 2018; Raghunathan et al., 2020) and relation to decision boundaries (Moosavi-Dezfooli et al., 2017), have also also investigated.

More advanced attack algorithms have since been developed, with FGSM often serving as a building block. In particular, the majority of these algorithms are multi-step variants of FGSM that

iteratively apply FGSM with a small step size to generate adversarial perturbations. In the next section, we discuss other techniques for generating adversarial perturbations.

2.3 Generating Adversarial Perturbations

One general approach to generating adversarial perturbation is to solve an optimization problem that maximizes the loss function subject to a constraint on the perturbation:

$$\delta^* = \underset{\|\delta\|_p \leq \epsilon}{\arg\max}\, \ell(x+\delta), \tag{2.3}$$

where the constraint is typically defined in terms of some ℓ_p-norm. For example, δ_{FGSM} is a special case of this where the maximizer can be found in closed form due to the first-order Taylor approximation of the loss.

To improve the vanilla FGSM, a straightforward extension is to apply multiple iterations of it with a small step size. This iterative approach is known as the Basic Iterative Method (BIM) (Kurakin et al., 2016). Starting with the initial data point $x'_0 = x$, BIM updates as follows:

$$\begin{aligned} g_{t+1} &= \nabla_x \ell(x'_t), \\ x'_{t+1} &= \Pi_{B_\infty(x,\epsilon)} \left\{ x'_t + \alpha \operatorname{sign}\left(\nabla_x \ell(x'_n)\right) \right\}, \end{aligned} \tag{2.4}$$

where g_{t+1} is the input gradient and α denotes the step size at each iteration. The projection operator $\Pi_{B_\infty(x,\epsilon)}$ ensures that the perturbation is within the constrain. Note that one-iteration BIM with $\alpha = \epsilon$ is equivalent to FGSM.

Madry et al. (2018) further improves this approach by introducing a uniformly random initialization of the perturbation within the ℓ_∞-ball before the iterative process. The initialization step is given by:

$$x'_0 = \Pi_{B_\infty(x,\epsilon)}\{x + \epsilon\delta\}, \tag{2.5}$$

where each dimension of δ is independently sampled from a uniform distribution $\mathcal{U}(-1,1)$. This method, known as Projected Gradient Descent (PGD), has been shown by the authors to reliably generate strong adversarial perturbations. Several algorithms have been developed using PGD as a foundation, including those discussed in Chapter 5 and in Croce et al. (2020). In BIM and PGD, the number of iterations is a hyperparameter that requires tuning. Madry et al. (2018) demonstrates on CIFAR-10 that the adversarial loss $\ell(x'_t)$ plateaus after a small number of iterations, opting to use 20 iterations in their evaluations. Consequently, subsequent works, including the evaluations in this thesis, have adopted 20 iterations as a standard setting for PGD.

To find better maximizers of (2.3), more advanced methods borrow techniques initially proposed for optimizing neural networks. Techniques such as momentum, adaptive updates, and variance reduction are successively integrated to create more effective perturbations (Dong et al., 2018; Zou et al., 2020; Wang et al., 2021c; Xiong et al., 2022).

Unlike FGSM and PGD which are formulated as constrained optimization problems, Carlini et al. (2017) formulates the attack using an unconstrained optimization problem. Denote a neural networks as: $f(x) = \text{softmax}(Z(x))$, where $Z(\cdot)$ is the pre-softmax logit, and we use $Z_i(x)$ to denote

the logit of to the i-th class. They proposed the Carlini-Wagner (CW) attack:

$$\min_{\delta} \|\delta\|_p + c \cdot g(x+\delta), \tag{2.6}$$

where $g(x') = \max\{\max_{j \neq t}\{Z_j(x')\} - Z_t(x'), 0\}$, t is the targeted class for misclassification. When $\max_{j \neq t}\{Z_j(x')\} - Z_t(x') \leq 0$, this means the model's most confident output is class t and the optimization is then simplified to $\min \|\delta\|_p$, which has a trivial solution: $\delta = 0$. On the other hand, when it is larger than 0, we penalize the difference of pre-softmax logits between the targeted class and the class with the second highest value.

To ensure that the perturbed input still remains within the valid input space for images, we need to guarantee that $x + \delta \in [0,1]^d$. The authors achieved this by introducing a change of variable, w, and defining $\delta_i = \frac{1}{2}(\tanh(w_i) + 1) - x_i$. Since $-1 \leq \tanh(w_i) \leq 1$, the solution will automatically be within the constraints.

Because the optimization in (2.6) is unconstrained, this requires verifying norm of the perturbation. Additional techniques were discussed in the paper which allow us to generate CW attacks using different norms.

The PGD and CW attacks are two of the most widely used algorithms for generating adversarial perturbations. The CW attack is primarily used to evaluate the robustness of models, while the PGD attack is often utilized to generate adversarial examples for improving model robustness. This topic will be explored further in the next section. Additionally, more recent robustness evaluation toolboxes, such as AutoAttack (Croce et al., 2020), have incorporated these attacks as part of their evaluation suites. In this thesis, we focus on PGD and AutoAttack to evaluate the robustness of our models.

2.4 The Robust Optimization Framework

In addition to developing the PGD attack, Madry et al. (2018) explored the robust optimization framework for improving model robustness of neural networks. The ERM process has excelled in finding solutions with low empirical risk, but it only focuses on reducing the average-case loss on the training set, neglecting the worst-case loss caused by adversarial perturbations. To address this, the authors proposed a shift from the conventional ERM paradigm to one that incorporates robustness directly into the model. This goal was formulated as a robust optimization problem where the objective is to minimize the adversarial population risk given some perturbation constraint Δ:

$$\min_{w} \mathbb{E}_{(X,Y)\sim\mathcal{D}}\big[\max_{\delta \in \Delta} \ell(X+\delta, Y; w)\big], \tag{2.7}$$

where we have an interplay between two goals: the inner-max term looks for the worst-case loss around each input, and the outer-min term aims to optimize model parameters to minimize this loss.

Equation (2.7) represents a saddle point problem involving two variables, δ and w. For each data point (X, Y), the goal is to find a pair of (δ^*, w^*) such that the equation satisfies both the maximization condition w.r.t. δ and the minimization condition w.r.t. w. To solve this, Madry et al. (2018) proposed using alternating GD, which updates the variables in the inner-max and outer-min terms in an alternating manner using gradient ascent and descent. The proposed PGD attack

was then used to approximately solve the inner-max problem for an ℓ_∞-norm bounded perturbation.

For the outer-min problem, the authors proposed training with PGD-perturbed training data. They justified this approach using Danskin's theorem, which states that descent direction at the inner maximizer corresponds to the descent direction for the entire saddle point problem.

However, this theorem assumes that the function is continuously differentiable, a condition not always met in neural networks due to elements like ReLU and max-pooling. Moreover, the PGD perturbation was only an approximation to the inner-max solution. Nevertheless, training with PGD perturbations has become a widely-used method for improving model robustness and remains one of the state-of-the-art (SOTA) defense techniques.

Adversarial training using perturbed training data has been studied extensively by Szegedy et al. (2014), Goodfellow et al. (2015), Kurakin et al. (2016), Madry et al. (2018), and Wong et al. (2019a). These methods primarily differ in how perturbations are generated: L-BFGS (Szegedy et al., 2014), FGSM (Goodfellow et al., 2015), BIM (Kurakin et al., 2016), PGD (Madry et al., 2018) and FGSM with random initialization (Wong et al., 2019a). Under the robust optimization framework, training with adversarial perturbations is seen as an approximation of the min-max problem described in (2.7), through different solutions to the inner-max problem.

Generating PGD perturbations, particularly in the inner loop, is computationally expensive, which notably limits the scalability of PGD adversarial training to large datasets. There are several other attempts to use the min-max objective for robustifying DNNs (Huang et al., 2015; Wong et al., 2018; Shaham et al., 2018). Notably, Wong et al. (2018) proposed adversarial training using randomly initialized FGSM perturbations (i.e., one-step PGD), demonstrating that such models can achieve robustness comparable to PGD-based training but at a significantly reduced computational cost. However, FGSM with random initialization presents a considerably weaker adversary and a less accurate approximation to the inner-max problem, therefore, quantifying the degree to which the inner maximization must be solved in order to perform robust optimization is still an open research area.

Although the robust optimization framework provides an initial step toward a theoretically sound approach for robustifying neural networks, other robustification techniques exist outside this framework. For instance, more recent methods leverage curriculum-based (Cai et al., 2018; Sarkar et al., 2021), ensemble-based (Yang et al., 2020; Wang et al., 2022b; Deng et al., 2024), and contrastive learning approaches (Jiang et al., 2020; Kim et al., 2020; Xu et al., 2024).

Bishop (1995) demonstrated that augmenting the training data with Gaussian noise is equivalent to training with ℓ_2 regularization, a connection we explore in details in Section 3.2. In Chapter 3, we introduce a regularization method which mimics the adversarial training process. Instead of finding the maximizers of the inner loop in (2.7) and substituting the original inputs with these maximizers, we train models directly with a regularizer. The regularized loss upper bounds the worst-case effects of an adversary.

2.5 The Robustness-Accuracy Trade-off

An interesting observation noted across several adversarial training methods is the undesirable trade-off between accuracy on adversarially perturbed inputs and that on unperturbed inputs (Madry et al., 2018; Tsipras et al., 2018; Zhang et al., 2019; Rade et al., 2022; Li et al., 2023). Raghunathan et

al. (2020) analyzed the effect of data augmentation in the linear regression setting where the optimal linear model has a perfect accuracy on both unperturbed and perturbed data. They considered two minimum norm estimators:

$$\begin{aligned} \hat{w}_{\text{std}} &= \arg\min_w \{\|w\|_2 : y_{\text{std}} = X_{\text{std}} w\} \\ \hat{w}_{\text{aug}} &= \arg\min_w \{\|w\|_2 : y_{\text{std}} = X_{\text{std}} w, y_{\text{ext}} = X_{\text{ext}} w\} \end{aligned}, \tag{2.8}$$

where the augmented data X_{ext} are generated by adding noise to the standard data X_{std}.

For linear regression, the authors considered a squared loss: $\ell(w) = (w - w^\star)^\top \Sigma (w - w^\star)$, where $\Sigma = \mathbb{E}\left[XX^\top\right]$ is the data covariance. They showed that the difference in the loss of the standard estimator and the augmented estimator, $\ell(\hat{w}_{std}) - \ell(\hat{w}_{aug})$, hinges on the eigenvalues of Σ. This analysis identified conditions of Σ under which the standard error does not increase when fitting augmented data, thereby leading to methods that minimize the trade-off between standard and robust accuracy.

Using those insights, the authors demonstrated that the robust self-training (RST) procedure proposed by Uesato et al. (2019) and Carmon et al. (2019) satisfies the conditions in linear regression setting. Although RST does not completely resolve the trade-off on neural networks, they showed that it significantly minimizes the trade-off compared to other SOTA defense methods.

Several works have attempted to explain the tension between the goal of adversarial robustness and that of standard generalization. Tsipras et al. (2018) argues that this trade-off is a consequence of robust classifiers learning fundamentally different feature representations than standard classifiers, conjecturing that no classifier is both accurate and robust. Similarly, Zhang et al. (2019) also showed that the trade-off is inevitable and proposed TRADES regularization, which includes a tunable hyperparameter to balance the trade-off between robustness and accuracy. Along with PGD adversarial training, TRADES-regularized models are also included as our baselines in the experiments in Chapters 3 and 5. The objective can be formulated as:

$$\min_w \mathbb{E}_{(X,Y)\sim\mathcal{D}} \left[\ell(X, Y; w) + \frac{1}{\lambda} \cdot \max_{\delta\in\Delta} \text{KL}(f(X+\delta)||f(X))\right],$$

where $f(\cdot)$ is the probability of classes and λ governs the trade-off between clean accuracy and adversarial robustness. Throughout this thesis, we consistently observe a trade-off between standard and robust accuracy across all datasets and models tested.

2.6 The Transferability of Adversarial Perturbations

Attacking the target model using algorithms such as PGD and CW requires knowing the model's architecture and weights, a scenario known as the white-box threat model. This assumes that the adversary has an exact replica of the target model, a strong assumption that is rarely met in practice. However, this does not imply that models are secure from attacks. A unique property of adversarial examples is their transferability, where perturbations generated for one model can often deceive another, even across different architectures (Goodfellow et al., 2015).

Motivated by this phenomenon, researchers have proposed generating adversarial examples using a surrogate model and then transferring them to the target model, leading to the development of the black-box attack framework (Papernot et al., 2017). In this framework, a substitute model is trained

to approximate the decision boundary of the target model by querying its outputs. Adversarial examples are then generated using this surrogate and transferred to the target.

Under this framework, the goal is to limit the number of queries required to learn the substitute model. This led to a heuristic for generating synthetic training set called Jacobian-based Dataset Augmentation. By examining the substitute model's Jacobian matrix at a given input, it reveals directions in which the substitute model's output is varying, and thus requires more input-output pairs to more accurately approximate the decision boundary of the target model. The effectiveness of this technique was demonstrated against online ML classifiers hosted by companies like Amazon and Google, achieving misclassification rates of 96.19% and 88.9%, respectively.

When robustifying models, many defenses only reach an illusion of robustness through methods collectively known as gradient masking (Athalye et al., 2018), which describes the scenario that gradient-based attacks are unable to generate perturbation due to insufficient gradient information. These methods often fail against transferred attacks, such as those proposed by Papernot et al. (2017). Note that with unlimited queries, a substitute model could replicate the target model. This leads to ongoing research aimed at improving the efficiency of query strategies (Cheng et al., 2019; Guo et al., 2019).

What makes adversarial examples transferable? Various hypotheses have been proposed, with one popular theory suggesting that the transferability is due to the similar geometric structures in the loss landscapes of different models (Fawzi et al., 2017; Charles et al., 2019; Zhao et al., 2020; Liu et al., 2016). For instance, Liu et al. (2016) showed that different models often share similar decision boundaries, enabling adversarial examples to transfer between them.

In Chapter 6, we begin by presenting a taxonomy of various transferability-enhancing methods. We then introduce a fine-tuning technique to transform any source model into one from which attacks generate more transferable adversarial perturbations. We also leverage a geometric analysis to understand the effect of the proposed transferability improvement technique.

Chapter 3

Regularized Training for Improving Adversarial Robustness

3.1 Introduction

The intuition behind adversarial training is that by adding sufficiently enough adversarial examples, the network gradually becomes robust to the attack it was trained on (Section 2.4). However, a significant challenge with this approach is the tremendous amount of additional data required for learning a robust model. Schmidt et al. (2018) demonstrated that, under the assumption of a Gaussian data distribution, the sample complexity for achieving robust generalization is $\sqrt{d}$ times greater than that for standard generalization, where d represents the input dimension. This implies that that current datasets may be insufficient for attaining high adversarial accuracy. In this work, as an alternative to data augmentation, we propose to robustify the model by introducing a regularizer that specifically penalizes model parameters vulnerable to attacks. By minimizing the regularized loss function, we get models that are robust to adversarial examples. This method offers a more direct and more efficient way to improve model robustness compared to traditional adversarial training methods.

Both adversarial training and our proposed approach can be formulated under the robust optimization framework (Section 2.4). In this formulation, one is seeking to improve the *worst-case* performance of the model measured by a loss function ℓ. Given each training data, adversarial training first generates a perturbation using a specific attack technique and then updates the model parameters based on the loss evaluated at the perturbed data point. Our proposed method, on the other hand, eliminates the need for finding such perturbations. It is based on approximating the loss at the perturbed data point $x + \delta$ using its second-order Taylor series expansion, i.e.,

$$\ell(x+\delta) \approx \ell(x) + \nabla_x \ell(x)^\top \delta + \frac{1}{2}\delta^\top \nabla_x^2 \ell(x) \delta,$$

and then upper bounding the worst-case loss using these expansion terms. By considering both gradient and Hessian of the loss function w.r.t. the input, we directly identify the model parameters that could be exploited by an adversary and regularize them during the training process. We call

This chapter is based on our work in SOAR: Second-Order Adversarial Regularization.

the method *Second-Order Adversarial Regularizer (SOAR)*.

3.1.1 Contributions

In this chapter, we first introduce an example with linear models to motivate the development of our method in Section 3.2. The derivation of SOAR and a discussion of related regularization approaches are included in Section 3.3. The experiment results are included in Section 3.4. In the course of the development of SOAR, we make the following contributions:

- We show that an over-parameterized linear regression model can be severely affected by an adversary, even though its population loss is zero. We robustify it with a regularizer that **exactly** mimics the adversarial training. This suggests that regularization can be used instead of adversarial training (Section 3.2).
- Inspired by such a possibility, we develop a regularizer which upper bounds the **worst-case** effect of an adversary under an approximation of the loss. In particular, we derive SOAR, which approximates the inner maximization of the robust optimization formulation based on the second-order Taylor series expansion of the loss function (Section 3.3).
- We study SOAR in the logistic regression setting and reveal challenges with regularization using Hessian w.r.t. the input. We develop a simple initialization method to circumvent the issue (Section 3.3.4).
- We empirically show that SOAR significantly improves the adversarial robustness of the network against ℓ_∞ attacks and ℓ_2 attacks generated based on PGD (Madry et al., 2018) on CIFAR-10 (Section 3.4).
- We further investigate SOAR under state-of-the-art method AutoAttack (Croce et al., 2020) and analyze the drop in robustness against it. We also discuss several hypotheses regarding this decrease in robustness (Section 3.4.2).

3.2 Understanding Adversarial Training Using Linear Regression with an Over-parametrized Model

We start with a linear model example to motivate the development of our regularizer. We show that for over-parameterized linear models, gradient descent (GD) finds a solution that has zero population loss, but is prone to attacks. We then show that this problem can be avoided with an appropriate regularizer. Hence, we do not need adversarial training to robustify such a model. This simple illustration motivates the development of our method in next sections.

Consider a linear model $f_w(x) = \langle\, w\,,\, x\,\rangle$ with $x, w \in \mathbb{R}^d$. Suppose that w^* represents the true model that is used to generate the target: $y = x^\top w^*$, where $w^* = (1, 0, \ldots, 0)^\top$. The distribution of $x \sim p$ is confined to a 1-dimensional subspace, $\{\,(x_1, 0, 0, \ldots, 0) \,:\, x_1 \in \mathbb{R}\,\}$. So the density of x is $p\,((x_1, \ldots, x_d)) = p_1(x_1)\delta(x_2)\delta(x_3)\ldots\delta(x_d)$, where $\delta(\cdot)$ is Dirac's delta function. This setup can be thought of as using an over-parameterized model that has many irrelevant dimensions with data that is only covering the relevant dimension of the input space. This is a simplified model of the situation when the data manifold has a dimension lower than the input space. We consider the squared error

pointwise loss $\ell(x; w) = \frac{1}{2} |\langle x, w \rangle - \langle x, w^* \rangle|^2$. Denote the residual by $r(x; w) = \langle x, w - w^* \rangle$, and the population loss by $\mathcal{L}(w) = \mathbb{E}[\ell(X; w)]$.

Suppose that we initialize the weights as $w(0) = W \sim N(0, \sigma^2 \mathbf{I}_{d\times d})$, and use GD on the population loss, i.e., $w(t+1) \leftarrow w(t) - \eta \nabla_w \mathcal{L}(w)$.

The partial derivatives of the population loss are

$$\frac{\partial \mathcal{L}(w)}{\partial w_j} = \int \langle x, w - w^* \rangle p(x) \mathrm{d}x = (w_j - w_j^*) \mathbb{E}[X_j].$$

Notice that the gradient in dimension $j = 1$ is non-zero, unless $(w_1 - w_1^*)\mathbb{E}[X_1] = 0$. Assuming that $\mathbb{E}[X_1] \neq 0$, this implies that the gradient will not be zero unless $w_1 = w_1^*$. On the other hand, the gradients in dimensions $j = 2, \ldots, d$ are all zero, so GD does not change the value of $w_j(t)$ for $j = 2, \ldots, d$. Therefore, under the proper choice of learning rate η, we get that the asymptotic solution of GD solution is $\bar{w} \triangleq \lim_{t\to\infty} w(t) = (w_1^*, w_2(0), w_3(0), \ldots, w_d(0))^\top$.

We make two observations. The first is that $\mathcal{L}(\bar{w}) = 0$, i.e., the population loss is zero. So from the perspective of training under the original loss, we are finding the optimal solution. The second observation is that we can easily attack this model by perturbing x by $\Delta x = (0, \Delta x_2, \Delta x_3, \ldots, \Delta x_d)^\top$. The loss at $x + \Delta x$ is

$$\ell(x + \Delta x; w) = \frac{1}{2} |(w_1 - w_1^*)x_1 + \langle w, \Delta x \rangle|^2 = \frac{1}{2} |r(x; w) + \langle w, \Delta x \rangle|^2 .$$

With the choice of $\Delta x_i = \epsilon \operatorname{sign}(w_i(0))$ (for $i = 2, \ldots, d$) and $\Delta x_1 = 0$, an FGSM-like attack (Goodfellow et al., 2015) at $\bar{w}$ leads to the pointwise loss of

$$\ell(x + \Delta x; \bar{w}) = \frac{1}{2}\epsilon^2 \left[\sum_{j=2}^{d} |w_j(0)|\right]^2 \approx \frac{1}{2}\epsilon^2 \|w(0)\|_1^2 , \tag{3.1}$$

when $d \gg 1$.

Now, to get a better sense of this loss, we compute its expected value w.r.t. the randomness of weight initialization. We have that (including the extra $|w_1(0)|$ term too)

$$\mathbb{E}_{W\sim N(0,\sigma^2\mathbf{I}_{d\times d})}\left[\|W\|_1^2\right] = \mathbb{E}\left[\sum_{i,j=1}^{d} |W_i||W_j|\right] = \sum_{i=1}^{d} \mathbb{E}\left[|W_i|^2\right] + \sum_{i,j=1, i\neq j}^{d} \mathbb{E}[|W_i|]\,\mathbb{E}[|W_j|] ,$$

where we used the independence of the random variable W_i and W_j when $i \neq j$. The expectation $\mathbb{E}\left[|W_i|^2\right]$ is the variance σ^2 of W_i. The random variable $|W_j|$ has a folded normal distribution, and its expectation $\mathbb{E}[|W_j|]$ is $\sqrt{\frac{2}{\pi}}\sigma$. Thus, we get that

$$\mathbb{E}_{W\sim N(0,\sigma^2\mathbf{I}_{d\times 1})}\left[\|W\|_1^2\right] = d\sigma^2 + d(d-1)\frac{2}{\pi}\sigma^2 \approx \frac{2}{\pi}d^2\sigma^2,$$

for $d \gg 1$. The expected population loss of the specified attack Δx at the asymptotic solution $\bar{w}$ is

$$\mathbb{E}_{X,W}\left[\ell(X + \Delta x); \bar{w})\right] \approx O(\epsilon^2 d^2 \sigma^2).$$

The dependence of this loss on dimension d is significant, showing that the learned model is quite

vulnerable to attacks. We note that the conclusions would not change much with initial distributions other than the Normal distribution. The culprit is obviously that GD is not forcing the initial weights to go to zero when there is no data from irrelevant and unused dimensions. This simple problem illustrates how the optimizer and an over-parameterized model might interact and lead to a solution that is prone to attacks.

An effective solution is to regularize the loss to encourage the weights of irrelevant dimensions going to zero. A generic regularizer is to use the ℓ_2-norm of the weights, i.e., formulate the problem as a ridge regression. In that case, the regularized population loss is

$$\mathcal{L}_{\text{ridge}}(w) = \frac{1}{2}\mathbb{E}\left[|\langle X\,,\,w\rangle - \langle X\,,\,w^*\rangle|^2\right] + \frac{\lambda}{2}\|w\|_2^2\,.$$

The solution of $\nabla_w \mathcal{L}_{\text{ridge}}(w) = 0$ is

$$\bar{w}_j(\lambda) = \begin{cases} \frac{\mu_1}{\mu_1+\lambda} w_1^* & j = 1 \\ 0 & j \neq 1. \end{cases}$$

The use of this generic regularizer seems reasonable in this example, as it enforces the weights for dimensions 2 to d to become zero. Its only drawback is that it leads to a biased estimate of w_1^*. We can obtain a similar conclusion for the ℓ_1 regularizer (Lasso). Although the bias can be made small with a small choice for λ, this comes at the cost of extended training time. Rather than relying on generic regularizers, can we define a regularizer specifically designed to improve adversarial robustness?

Bishop (1995) showed the connection between training with random perturbation and Tikhonov Regularization. Inspired by this idea, we develop a regularizer that mimics the adversary itself. Let us assume that a particular adversary attacks the model by adding $\Delta x = (0, \epsilon\,\text{sign}(w_2(0)), \ldots, \epsilon\,\text{sign}(w_d(0))^\top$. The population loss at the perturbed point is

$$\begin{aligned} \mathcal{L}_{\text{robustified}}(w) \triangleq \mathbb{E}\left[\ell(X + \Delta x; w)\right] &= \frac{1}{2}\mathbb{E}\left[\left| r(x; w) + \epsilon \sum_{j=2}^{d} |w_j| \right|^2\right] \\ &= \mathcal{L}(w) + \epsilon\mathbb{E}\left[r(X; w)\right]\|w_{2:d}\|_1 + \frac{\epsilon^2}{2}\|w_{2:d}\|_1^2\,, \end{aligned} \tag{3.2}$$

where $\|w_{2:d}\|_1 = \sum_{j=2}^{d} |w_j|$.[1]

It is important to understand the relation between (3.1) and (3.2). While (3.1) shows the loss under attack for the asymptotic solution, (3.2) shows the loss for any w. By choosing $w = \bar{w}$, the first two terms in (3.2) become zero and we are left with the same term as in (3.1).

Note that minimizing $\mathcal{L}_{\text{robustified}}(w)$ is equivalent to minimizing the model at the point $x' = x + \Delta x$, where $\Delta x = (0, \epsilon\,\text{sign}(w_2(0)), \ldots, \epsilon\,\text{sign}(w_d(0))^\top$. The regularizer $\epsilon\mathbb{E}\left[r(X; w)\right]\|w_{2:d}\|_1 + \frac{\epsilon^2}{2}\|w_{2:d}\|_1^2$ incorporates the effect of adversary in exact form. This motivated the possibility of designing a regularizer tailored to prevent attacks.

Under the robust optimization framework, regularization and adversarial training are two realizations of the inner-max objective in (2.7), but regularization relieved us from finding the perturbation,

[1] A similar, but more complicated result, would hold if the adversary could also attack the first dimension.

as is done in adversarial training. In this section, because of the linear model and the squared error loss, we could design a regularizer that exactly mimics the effect of the adversary. How can we design a regularizer for more complicated models, such as DNNs? We address this in the next section.

3.3 Adversarial Regularization: An Alternative to Adversarial Training

The main idea of the proposed regularization approach is to approximate the loss function using the Taylor series expansion and then solve the inner maximization term of the robust optimization formulation (2.7) using the approximated form. In this section, we first derive a regularizer using the first-order Taylor series expansion, which we refer to as the First-Order Adversarial Regularizer (FOAR). Following that, we derive SOAR using the second-order Taylor series expansion. We then discuss the challenges of using the second-order derivative of the loss w.r.t. the input, known as the input Hessian, in logistic regression and propose a simple initialization method to address the issue.

3.3.1 First-Order Adversarial Regularization

Assuming that the loss is twice-differentiable, we can approximate the loss function around input x by its second-order Taylor expansion

$$\ell(x+\delta, y; w) \approx \tilde{\ell}_{\text{2nd}}(x+\delta, y; w) \triangleq \ell(x, y; w) + \nabla_x \ell(x, y; w)^\top \delta + \frac{1}{2}\delta^\top \nabla_x^2 \ell(x, y; w)\delta, \tag{3.3}$$

where $\nabla_x \ell(x, y; w) \in \mathbb{R}^d$ and $\nabla_x^2 \ell(x, y; w) \in \mathbb{R}^{d\times d}$ are the gradient and Hessian of the loss w.r.t. the input x, respectively.

From a geometric perspective, the gradient term captures the slope of the loss function at x, while the Hessian term represents the curvature at x. Together, they provide a second-order approximation of the loss function around x. Although the gradient is straightforward to compute, computing the input Hessian is computationally expensive for high-dimensional inputs. To address this, we can approximate the Hessian using Hessian-free techniques such as the Hessian-vector product. We will discuss this in more detail in the following section.

For brevity, we drop w, y and use ∇ to denote ∇_x. Let us focus on the ℓ_p attacks, where the constraint set in (2.7) is $\Delta = \{\delta : \|\delta\|_p \le \epsilon\}$ for some $\epsilon > 0$ and $p \ge 1$. We focus on the ℓ_∞ attack because of its popularity, but we also derive the formulation for the ℓ_2 attacks.

As a warm-up, let us solve the inner maximization of (2.7) by considering the first-order Taylor series expansion. We have

$$\ell_{\text{FOAR}}(x) \triangleq \max_{\|\delta\|_p \le \epsilon} \ell(x) + \nabla \ell(x)^\top \delta = \ell(x) + \epsilon \|\nabla \ell(x)\|_q, \tag{3.4}$$

for $1 \le p \le \infty$ and q satisfying $p^{-1} + q^{-1} = 1$. Focusing on the ℓ_∞ attack, we have $p = \infty$ and $q = 1$, so the FOAR is $\epsilon \|\nabla \ell(x)\|_1$. This regularizer is similar to the one proposed by Simon-Gabriel et al. (2019) with the choice of ℓ_∞ perturbation set.

3.3.2 Second-Order Adversarial Regularization

In our work, we take one-step further and solve the inner maximization problem based on the second-order Taylor expansion:

$$\max_{\|\delta\|_p \leq \epsilon} \ell(x) + \nabla \ell(x)^\top \delta + \frac{1}{2} \delta^\top \nabla^2 \ell(x) \delta, \tag{3.5}$$

for $p = 2, \infty$. The second-order expansion in (3.3) can be rewritten as

$$\begin{aligned} \ell(x+\delta) &\approx \ell(x) + \frac{1}{2} \begin{bmatrix} \delta \\ 1 \end{bmatrix}^\top \begin{bmatrix} \nabla^2 \ell(x) & \nabla \ell(x) \\ \nabla \ell(x)^\top & 1 \end{bmatrix} \begin{bmatrix} \delta \\ 1 \end{bmatrix} - \frac{1}{2} \\ &= \ell(x) + \frac{1}{2} \delta'^\top \mathbf{H} \delta' - \frac{1}{2}, \end{aligned} \tag{3.6}$$

where $\delta' = [\delta; 1]$. This allows us to derive an upper bound on the expansion terms using the properties of a single Hessian $\mathbf{H}$. Note that δ' is a $d+1$-dimensional vector and $\mathbf{H}$ is a $(d+1) \times (d+1)$ matrix. To derive SOAR, we need to find an upper bound on $\delta'^\top \mathbf{H} \delta'$ under the attack constraint.

For the ℓ_∞ attack, solving this maximizing problem is not as easy as in (3.4) since the box-constrained quadratic programming problem in formulation (3.5) is NP-hard when the input Hessian is non-convex (Burer et al., 2009), which is typically the case in neural networks. Even though there exist semi-definite programming (SDP) relaxations, such approaches require the exact Hessian w.r.t. the input. And even if we could compute the exact Hessian, SDP itself is a computationally expensive approach, and not suitable to be within the inner loop of a DNN training.

Our goal is to derive a second-order regularizer for perturbations constrained by the ℓ_∞ norm, i.e., $\|\delta\|_\infty \leq \epsilon$. To achieve this, we first consider (3.5) under the constraint $\|\delta\|_2 \leq \sqrt{d}\epsilon$ and then relate it to $\|\delta\|_\infty \leq \epsilon$. As we see later, this approach leads to a computationally efficient solution.

Proposition 3.3.1. *Let $\ell : \mathbb{R}^d \to \mathbb{R}$ be a twice-differentiable function. For any $\epsilon > 0$, we have*

$$\max_{\|\delta\|_2 \leq \sqrt{d}\epsilon} \tilde{\ell}_{2nd}(x+\delta) \leq \ell(x) + \frac{d\epsilon^2 + 1}{2} \mathbb{E}\left[\|\mathbf{H} z\|_2\right] - \frac{1}{2}, \tag{3.7}$$

where $\boldsymbol{H}$ is defined in (3.6) and $z \sim \mathcal{N}(0, \mathbf{I}_{(d+1)\times(d+1)})$.

Proof.

$$\begin{aligned} \max_{\|\delta\|_2 \leq \sqrt{d}\epsilon} \tilde{\ell}_{\text{2nd}}(x+\delta) &= \max_{\|\delta\|_2 \leq \sqrt{d}\epsilon} \ell(x) + \frac{1}{2} \begin{bmatrix} \delta \\ 1 \end{bmatrix}^\top \begin{bmatrix} \nabla^2 \ell(x) & \nabla \ell(x) \\ \nabla \ell(x)^\top & 1 \end{bmatrix} \begin{bmatrix} \delta \\ 1 \end{bmatrix} - \frac{1}{2} \\ &= \ell(x) + \frac{1}{2} \max_{\|\delta\|_2 \leq \sqrt{d}\epsilon} \begin{bmatrix} \delta \\ 1 \end{bmatrix}^\top \mathbf{H} \begin{bmatrix} \delta \\ 1 \end{bmatrix} - \frac{1}{2} \\ &\leq \ell(x) + \frac{1}{2} \max_{\|\delta'\|_2 \leq \sqrt{d\epsilon^2+1}} \delta'^\top \mathbf{H} \delta' - \frac{1}{2}. \end{aligned}$$

To upper bound $\max_{\|\delta'\|_2 \leq \sqrt{d\epsilon^2+1}} \delta'^\top \mathbf{H} \delta'$, we first denote $\epsilon' = \sqrt{d\epsilon^2 + 1}$ and use the Cauchy-

Schwarz inequality to obtain

$$\max_{\|\delta'\|_2 \leq \epsilon'} \delta'^\top \mathbf{H} \delta' \leq \max_{\|\delta'\|_2 \leq \epsilon'} \left| \delta'^\top \mathbf{H} \delta' \right| \leq \max_{\|\delta'\|_2 \leq \epsilon'} \|\delta'\|_2 \|\mathbf{H}\delta'\|_2 = \epsilon' \max_{\|\delta'\|_2 \leq \epsilon'} \|\mathbf{H}\delta'\|_2 = \epsilon'^2 \|\mathbf{H}\|_2 ,$$

where the last equality is obtained using definitions of the ℓ_2-induced matrix norm, commonly referred to as the spectral norm. Since computing $\|\mathbf{H}\|_2$ would again require the exact input Hessian, which we want to avoid, so we further upper bound the spectral norm by the Frobenius norm as

$$\|\mathbf{H}\|_2 = \sigma_{\max}(\mathbf{H}) \leq \|\mathbf{H}\|_{\mathrm{F}} .$$

The Frobenius norm itself satisfies

$$\|\mathbf{H}\|_{\mathrm{F}} = \sqrt{\mathrm{Tr}(\mathbf{H}^\top \mathbf{H})} = \mathbb{E}\left[\|\mathbf{H}z\|_2\right], \tag{3.8}$$

where $z \sim \mathcal{N}(0, \mathbf{I}_{(d+1)\times(d+1)})$. To understand the last equality in (3.8), first note that:

$$\|\mathbf{H}z\|_2^2 = (\mathbf{H}z)^\top (\mathbf{H}z) = z^\top \mathbf{H}^\top \mathbf{H} z.$$

Taking the expectation of both sides, we have:

$$\mathbb{E}\left[\|\mathbf{H}z\|_2^2\right] = \mathbb{E}\left[z^\top \mathbf{H}^\top \mathbf{H} z\right] = \mathrm{Tr}(\mathbf{H}^\top \mathbf{H}),$$

where the last equality is obtained using $\mathbb{E}\left[z^\top A z\right] = \mathrm{Tr}(A)$ for any square matrix A and a standard Gaussian vector z.

Therefore, we can estimate $\|\mathbf{H}\|_{\mathrm{F}}$ by sampling random vectors z and compute the sample average of $\|\mathbf{H}z\|_2$. □

Using Proposition 3.3.1, we obtain an upper bound on the worst-case second-order approximated loss under an ℓ_2 perturbation of size $\sqrt{d}\epsilon$. The next corollary follows directly from this proposition.

Corollary 3.3.2. *Let $\ell : \mathbb{R}^d \to \mathbb{R}$ be a twice-differentiable function. For any $\epsilon > 0$, we have*

$$\max_{\|\delta\|_\infty \leq \epsilon} \tilde{\ell}_{2nd}(x+\delta) \leq \ell(x) + \frac{d\epsilon^2 + 1}{2} \mathbb{E}\left[\|\mathbf{H}z\|_2\right] - \frac{1}{2}, \tag{3.9}$$

where $\boldsymbol{H}$ is defined in (3.6) and $z \sim \mathcal{N}(0, \mathbf{I}_{(d+1)\times(d+1)})$.

Proof. With $\delta \in \mathbb{R}^d$, an ℓ_∞-ball of size ϵ is enclosed by an ℓ_2-ball of size $\sqrt{d}\epsilon$ with the same center. Therefore, we can upper bound the inner maximization by

$$\max_{\|\delta\|_\infty \leq \epsilon} \tilde{\ell}_{2\mathrm{nd}}(x+\delta) \leq \max_{\|\delta\|_2 \leq \sqrt{d}\epsilon} \tilde{\ell}_{2\mathrm{nd}}(x+\delta) \leq \ell(x) + \frac{d\epsilon^2 + 1}{2} \mathbb{E}\left[\|\mathbf{H}z\|_2\right] - \frac{1}{2}, \tag{3.10}$$

where the last inequality follows from Proposition 3.3.1. □

The result in Corollary 3.3.2 upper bounds the maximum of the second-order approximation $\tilde{\ell}_{2\mathrm{nd}}$ over an ℓ_∞ ball with radius ϵ, and relates it to an expectation of a Hessian-vector product. This can be understood as an upper bound on the worst-case damage of an adversary under a second-order approximation of the loss.

Let us take a closer look at $\mathbf{H}z$. By decomposing $z = [z_d, z_1]^\top$, we get

$$\mathbf{H}z = \begin{bmatrix} \nabla^2\ell(x)z_d + z_1\nabla\ell(x) \\ \nabla\ell(x)^\top z_d + z_1 \end{bmatrix}.$$

Now, we have circumvented the computationally intensive process of computing the input Hessian matrix $\nabla^2\ell(x)$, as the term $\nabla^2\ell(x)z_d$ can be computed efficiently using Finite Difference (FD) approximation:

$$\nabla^2\ell(x)z_d \approx \|z_d\|_2 \frac{\nabla\ell(x + h\tilde{z}_d) - \nabla\ell(x)}{h}. \tag{3.11}$$

Note that $\mathbb{E}[\|z_d\|_2] = \sqrt{d}$ for normally distributed z, and to ensure consistence between the step size in the numerator and denominator, we normalize z_d by using $\tilde{z}_d = \frac{z_d}{\|z_d\|_2}$.

To summarize, the SOAR regularizer, formulated to improve robustness against perturbations of size ϵ constrained by the ℓ_∞ norm, evaluated at x, with a direction z, and FD step size $h > 0$ is

$$R(x; z, h, \epsilon) = \frac{d\epsilon^2 + 1}{2} \left\| \begin{bmatrix} \|z_d\|_2 \frac{\nabla\ell(x+h\tilde{z}_d) - \nabla\ell(x)}{h} + z_1\nabla\ell(x) \\ \nabla\ell(x)^\top z_d + z_1 \end{bmatrix} \right\|_2. \tag{3.12}$$

The expectation in (3.9) can then be approximated by taking multiple samples of z drawn from $z \sim \mathcal{N}(0, \mathbf{I}_{(d+1)\times(d+1)})$. These samples would be concentrated around its expectation. Note that $\mathbb{P}\{\|Hz\| - \mathbb{E}[\|Hz\|] > t\} \leq 2\exp(-\frac{ct^2}{\|H\|_2})$, where c is a constant and $\|H\|_2$ is the ℓ_2-induced norm (see Theorem 6.3.2 of Vershynin (2018)). In practice, we observed that taking more than one sample of z do not provide significant improvement for increasing adversarial robustness; we include an empirical study on the the effect of sample sizes in Appendix 3.B.

3.3.3 Revisiting the Linear Regression Example

Before we discuss the remaining details, let us revisit the linear regression example from Section 3.2. Recall that we fully robustify the model with an appropriate regularizer in (3.2). To further motivate a second-order approach such as SOAR, we demonstrate below that we can obtain the first two terms in (3.2) with a first-order regularizer, and we recover the exact form with a second-order formulation. This means that a second-order regularizer captures the exact effect of training with adversarial examples.

To demonstrate this, we first compute the gradient of the loss w.r.t. the input

$$\nabla\ell(x; w) = (\langle\, w\,,\, \Delta x\,\rangle - \langle\, w^*\,,\, \Delta x\,\rangle)(w - w^*) = r(x; w)(w - w^*),$$

and the Hessian w.r.t. the input

$$\nabla^2\ell(x; w) = (w - w^*)(w - w^*)^\top.$$

To approximate (3.2) using its first-order expansion, we have

$$\begin{aligned}\mathbb{E}\left[\ell(X;w)+\nabla\ell(X;w)^\top\Delta x\right] &= \mathcal{L}(w)+\mathbb{E}\left[r(X;w)(w-w^*)^\top\Delta x\right]\\ &= \mathcal{L}(w)+\mathbb{E}\left[r(X;w)w^\top\Delta x\right]\\ &= \mathcal{L}(w)+\epsilon\mathbb{E}\left[r(X;w)\right]\|w_{2:d}\|_1,\end{aligned}$$

where ${w^*}^\top\Delta x=0$ because of our particular choice of Δx and w^*, and we obtain the first two terms in (3.2). Note that the perturbation Δx generally depends on the input x. However, under our definition, $\Delta x=(0,\epsilon\,\mathrm{sign}(w_2(0)),\ldots,\epsilon\,\mathrm{sign}(w_d(0))^\top$ is fixed, as we disregard the contribution from the first dimension.

The second-order approximation is

$$\begin{aligned}&\mathbb{E}\left[l(X;w)+\nabla_x l(X;w)^\top\Delta x+\frac{1}{2}\Delta x^\top\nabla_x^2 l(x;w)\Delta x\right]\\ =&\mathcal{L}(w)+\epsilon\mathbb{E}\left[r(X;w)\right]\|w_{2:d}\|_1+\frac{1}{2}\Delta x^\top(w-w^*)(w-w^*)^\top\Delta x\\ =&\mathcal{L}(w)+\epsilon\mathbb{E}\left[r(X;w)\right]\|w_{2:d}\|_1+\frac{\epsilon^2}{2}\|w_{2:d}\|_1^2,\end{aligned}$$

which recovers the exact form in (3.2).

Next, we study SOAR in the simple logistic regression setting, which shows potential failure of the regularizer. Based on the insight, we provide the remaining details of the method.

3.3.4 Avoiding Gradient Masking

We discussed gradient masking in Section 2.4 and its implications on finding adversarial examples. For SOAR, gradient masking leads to a poor approximation of the loss function, which in turn results in an ineffective regularizer. We provide an example to illustrate this issue and propose a simple initialization method to address it.

Consider a linear classifier $f:\mathbb{R}^d\times\mathbb{R}^d\to[0,1]$ with the form $f(x;w)=\phi(\langle\,w\,,\,x\,\rangle)$, where $x,w\in\mathbb{R}^d$ are the input and the weight, and $\phi(z)=\frac{1}{1+e^{-z}}$ is the sigmoid function. The output of f has the interpretation of being a Bernoulli distribution. For the cross-entropy loss function $\ell(x,y;w)=-[y\log f(x;w)+(1-y)\log(1-f(x;w))]$, the gradient w.r.t. the input x is $\nabla\ell(x)=(f(x;w)-y)w$ and the Hessian w.r.t. the input x is $\nabla^2\ell(x)=f(x;w)(1-f(x;w))ww^\top$.

The second-order expansion in (3.3) with the gradient and Hessian evaluated at x is

$$\ell(x+\delta)\approx\ell(x)+r(x,y;w)w^\top\delta+\frac{1}{2}u(x;w)\delta^\top ww^\top\delta, \tag{3.13}$$

where $r=r(x,y;w)=f(x;w)-y$ and $u=u(x;w)=f(x;w)(1-f(x;w))$. This residual term r describes the difference between the predicted probability and the correct label. The u term can be interpreted as how confident the model is about its predication (correct or incorrect), and is close to 0 whenever the classifier is predicting a value close to 0 or 1. With this setup, the maximization in (3.5) with $p=2$ becomes

$$\ell(x)+\max_{\|\delta\|_2\le\epsilon}\left[rw^\top\delta+\frac{1}{2}u\delta^\top ww^\top\delta\right]=\ell(x)+\epsilon\left|r(x,y;w)\right|\|w\|_2+\frac{\epsilon^2}{2}u(x;w)\|w\|_2^2,$$

Table 3.1: **Comparing the probability of the most likely class across various training methods (%).** Specifically, we compute $\frac{1}{N}\Sigma_{n=1}^{N} \max_{i\in 1,2,\ldots,c} P(x_n)_i$, where $P(x_n)_i$ represent the model's predicted probability of class i given test data x_n. We focus on standard training, adversarial training with 10-step PGD (PGD10), and SOAR with different initializations. We consider three types of input data: clean data (Clean), perturbed with random noise (Random), and perturbed with PGD1 (PGD1). We observe that SOAR with zero and random initializations are extremely confident on all types of data. SOAR with PGD1 initialization is less confident on PGD1 data compared to the other two types, close to the standard training.

Method	Clean	Random	PGD1
Standard	98.11	97.81	96.83
PGD10	70.33	70.04	65.46
SOAR			
- zero init	99.99	99.97	99.99
- random init	99.98	99.98	100.0
- PGD1 init	97.71	97.63	97.94

where the two regularizers encourage the norm of w to be small, weighted according to the residual $r(x, y; w)$ and the uncertainty $u(x; w)$.

Consider a linear interpolation of the cross-entropy loss from x to a perturbed input x'. Specifically, we consider $\ell(\alpha x + (1-\alpha)x')$ for $\alpha \in [0, 1]$. Previous work has empirically shown that as α increases from 0 to 1, the value of the loss follows a logistic function (Madry et al., 2018): exhibiting little to no curvature at x, followed by rapid growth, and eventually plateauing. In such a case, if we use Hessian exactly at x, it leads to an inaccurate approximation of the value at $\ell(x')$. This causes a poor approximation of the inner-max, and the derived regularization will not be effective.

For the approximation in (3.13), this issue corresponds to the scenario in which the classifier is very confident about the clean input at x. Standard training techniques, such as minimizing cross-entropy loss, optimize the model by assigning high probability to the correct class and pushing the probabilities of incorrect classes toward 0. This incentivizes the model to return the correct label with high confidence. Whenever the classifier is correct with a high confidence, both r and u will be close to zero. As a result, the effect of the regularizer diminishes, i.e., the weights are no longer regularized. In such a case, the Taylor series expansion, computed using the gradient and Hessian evaluated at x, becomes an inaccurate approximation to the loss, and hence its maximizer is not a good solution to the inner maximization problem.

Note that this does not mean that Taylor series expansion cannot be used to approximate the loss. In fact, by the mean value theorem, there exists an $h^\star \in (0, 1)$ such that the second-order Taylor expansion is exact: $\ell(x+\delta) = \ell(x) + \nabla\ell(x)^\top\delta + \frac{1}{2}\delta^\top\nabla^2\ell(x + h^\star\delta)\delta$. The issue is that if we compute the Hessian at x (instead of at $x + h^\star\delta$), our approximation might not be very good whenever the curvature profile of the loss function at x is drastically different from the one at $x + h^\star\delta$.

In Table 3.1, we demonstrate the ineffectiveness of SOAR when the gradient and Hessian are evaluated at x. In particular, we compare the average value of the highest probability output for test set data with various initializations, under different training methods. Training with SOAR using zero or random initialization leads to models with nearly 100% confidence on their predictions. Here,

Algorithm 3.1: Improving adversarial robustness via SOAR

Input : Training dataset. Learning rate η, training batch size b, number of iterations T, ℓ_∞ constraint of ϵ, Finite difference step-size h.

Initialize network with pretrained weight w;
for $t \in \{0, 1, \ldots, T\}$ **do**
  Get mini-batch $B = \{(x_1, y_1), \cdots, (x_b, y_b)\}$ from the training set.
  for $j = 1, \ldots, m$ *(in parallel)* **do**
    $x'_j \leftarrow x_j + \delta$, where $\delta \leftarrow (\delta_1, \delta_2, \ldots, \delta_d)^\top$ and $\delta_i \sim \mathcal{U}(-\frac{\epsilon}{2}, \frac{\epsilon}{2})$.
    $x'_j \leftarrow \Pi_{B(x_j, \frac{\epsilon}{2})} \left\{ x'_j + \frac{\epsilon}{2} \operatorname{sign}\left(\nabla_x \ell(x'_j)\right) \right\}$ where $\Pi_{B(x_j, \frac{\epsilon}{2})}$ denotes the projection onto the ℓ_∞-norm ball of radius $\frac{\epsilon}{2}$, centered at x_j.
    Sample $z \sim \mathcal{N}(0, \mathbf{I}_{(d+1)\times(d+1)})$.
    Compute the SOAR regularizer $R(x'_j; z, h, \frac{\epsilon}{2})$ as (3.12).
    Compute the pointwise objective: $\ell_{\text{SOAR}}(x_j, y_j) = \ell(x'_j, y_j) + R(x'_j; z, h, \frac{\epsilon}{2})$.
  end
  $w_{t+1} \leftarrow w_t - \eta \times \frac{1}{b} \sum_{j=1}^{b} \nabla_{w_t} \ell_{\text{SOAR}}$.
end

random initialization denotes randomly initializing x within the ℓ_∞ ball of size ϵ centered at the clean input x: $x' = x + \delta$, where $\delta = (\delta_1, \delta_2, \ldots, \delta_d)^\top$ and $\delta_i \sim \mathcal{U}(-\epsilon, \epsilon)$. The results in Appendix 3.C show that those SOAR-regularized models are still vulnerable under transfer-based black-box attacks.

Highly confident predictions *could be* an indication for gradient masking. Suppose that the model makes predictions with 100% confidence on any given input. This leads to a piece-wise constant loss surface that is either zero (correct predictions) or infinity (incorrect predictions). The gradient of this loss function is either zero or undefined, and thus making gradient ascent ineffective. Therefore, white-box gradient-based attacks such as PGD are unable to find adversarial examples.

This suggests a heuristic to improve SOAR. That is to evaluate the gradient and Hessian, through FD approximation (3.11) at a less confident point within the ℓ_∞ ball of x. We found that evaluating the gradient and Hessian at inputs perturbed by 1-step PGD (PGD1) can alleviate the issue of overconfident predictions, reducing the confidence levels to those comparable with standard training. Also, to ensure the regularization is of the original ℓ_∞ ball of ϵ, we first initialize x with ℓ_∞ PGD1 perturbation of size $\frac{\epsilon}{2}$, and then the SOAR regularizer is applied with the perturbation constrained by $\frac{\epsilon}{2}$. Based on this heuristic, the regularized pointwise objective for a data point (x, y) is

$$\ell_{\text{SOAR}}(x, y) = \ell(x', y) + R(x'; z, h, \frac{\epsilon}{2}), \tag{3.14}$$

where $z \sim \mathcal{N}(0, \mathbf{I}_{(d+1)\times(d+1)})$ and the point x' is initialized at PGD1 adversary. We summarize the full training procedure with SOAR in Algorithm 3.1. Note that it is presented as if the optimizer is SGD, but we may use other optimizers as well. Moreover, we include additional discussions and experiments on gradient masking in Appendix 3.F.

3.3.5 Related Work

Several regularization-based alternatives to adversarial training have been proposed. Simon-Gabriel et al. (2019) considered the ℓ_∞ perturbation set and designed a regularizer under the first-order Taylor approximation, which we refer to as FOAR. Qin et al. (2019) proposed local linearity regu-

larization (LLR), where the local linearity is defined by the maximum error of the first-order Taylor approximation of the loss. The LLR objective consists of the local linearity measure and the magnitude of the projection of gradient along the corresponding direction of the local linearity measure.

CURE (Moosavi-Dezfooli et al., 2019) is the closest to our method. The authors observed that adversarial training leads to a reduction in the magnitude of eigenvalues of the Hessian w.r.t. the input. Thus, they proposed directly minimizing the curvature of the loss function to mimic the effect of adversarial training. An important advantage of our proposed method is that SOAR is derived from a complete second-order Taylor approximation of the loss, while CURE exclusively focuses on the second-order term for the estimation of the curvature. Note the final optimization objective in SOAR, FOAR, LLR and CURE contains derivative w.r.t. the input of the DNN, and such a technique was first introduced to improve generalization by Drucker et al. (1992) as double backpropagation.

Another related line of adversarial regularization methods do not involve approximation to the loss function nor robust optimization. TRADES introduces a regularization term that penalizes the difference between the output of the model on a training data and its corresponding adversarial example (Zhang et al., 2019). Misclassification aware adversarial training (MART) reformulated the training objective by explicitly differentiating between the misclassified and correctly classified examples (Wang et al., 2020b). Ding et al. (2018) present another regularization approach that leverages adaptive margin maximization (MMA) on correctly classified example to robustify the model.

3.4 Experiments

In this section, we verify the effectiveness of the proposed regularization method. Evaluations and discussions focus on the CIFAR-10 dataset, with model robustness evaluated using ℓ_∞-norm constrained perturbations generated by PGD. Additional results on ℓ_2-norm are deferred to Appendix 3.D. Our experiments show that training with SOAR leads to significant improvements in adversarial robustness against white-box and transferred perturbations. We also evaluate SOAR under state-of-the-art AutoAttack (Croce et al., 2020) method and discuss the decreased robustness against it.

3.4.1 Experiment Setup

Model and Dataset: We focus on ResNet-10 (He et al., 2016a) on the CIFAR-10 dataset in this section. Training data is augmented with random crops and horizontal flips. Evaluations on additional model architecture is included in Appendix 3.E.

Baseline: The baseline methods consist of: (1) Standard: training with no adversarially perturbed data; (2) PGD10: training with 10-step PGD adversarial examples; (3) TRADES (Zhang et al., 2019); (4) MART (Wang et al., 2020b) and (5) MMA (Ding et al., 2018). Additionally, we incorporate the first-order regularizer, FOAR (Simon-Gabriel et al., 2019), as part of the baseline to demonstrate the improved robustness of SOAR. For a fair comparison, we also evaluated FOAR with different initializations, as detailed in Appendix 3.C. We found that FOAR achieves the best robustness when initialized with PGD1. Therefore, we only present this variation of FOAR in this section. The optimization procedure for all baseline is described in detail in Appendix 3.A.

Table 3.2: **Evaluation of classification accuracy for models on the original CIFAR-10 test dataset and under ℓ_∞-norm constrained white-box attacks (%).** We evaluate the model robustness on the testing data perturbed using FGSM and various configurations of PGD. We also consider a stronger setting in which multiple runs of attacks are conducted for each given input: PGD20-50 denotes 20-step PGD attacks with 50 restarts. All perturbations are bounded by ℓ_∞-norm with $\epsilon = 8/255$. SOAR outperforms all other methods in terms of robustness against white-box attacks.

Method	Clean	FGSM	PGD20	PGD100	PGD200	PGD1000	PGD20-50
Standard	92.54	21.59	0.14	0.09	0.10	0.08	0.10
PGD10	80.64	50.96	42.86	42.27	42.21	42.17	42.55
TRADES	75.61	50.06	45.38	45.19	45.18	45.16	45.24
MART	75.88	52.55	46.60	46.29	46.25	46.21	46.40
MMA	82.37	47.08	37.26	36.71	36.66	36.64	36.85
FOAR	65.84	36.96	32.28	31.87	31.89	31.89	32.08
SOAR	87.95	67.15	56.06	55.00	54.94	54.69	54.20

Metric: Results reported in this chapter focus on classification accuracy; therefore, higher numbers indicate greater model robustness. Note that all methods in this section are trained to defend against ℓ_∞ norm attacks with $\epsilon = 8/255$,[1] as this ϵ value is commonly used in several early influential works (Kurakin et al., 2016; Madry et al., 2018). The PGD adversaries discussed in this section are generated with $\epsilon = 8/255$ and a step size of $2/255$. PGD20-50 denotes 20-step PGD attacks with 50 restarts. In Section 3.4.2, we compare SOAR with baseline methods on ℓ_∞ AutoAttack (Croce et al., 2020) adversaries using different ϵ values. AutoAttack consists of four individual attack algorithms, and with the exception of two specific targeted methods within AutoAttack, all other perturbations are untargeted (Section 2.1).

3.4.2 Evaluating Model Robustness

Against White-Box Attacks

In Table 3.2, we report the robustness of models trained with SOAR and the baseline methods against PGD and FGSM attacks in the white-box setting. Training with SOAR significantly improves the adversarial robustness against FGSM and all PGD attacks, leading to higher robustness in all k-step PGD attacks on the ResNet model. Also, note that FOAR achieves 32.28% against PGD20 attacks. Despite its uncompetitive performance, this shows that approximating the robust optimization formulation based on Taylor series expansion is a reasonable approach. This justifies our extension to a second-order approximation, as the first-order alone is not sufficient.

Robustness Against Black-box Attacks

Many defenses only reach an illusion of robustness through methods collectively known as gradient masking (Athalye et al., 2018). These methods often fail against transferred perturbations, which

[1]The value 255 in the denominator is due to digital images often being represented in 8-bit format, where any information below 1/255 is discarded.

Table 3.3: **Evaluation of classification accuracy for models under ℓ_∞-norm constrained perturbations in the black-box setting (%).** We evaluate model robustness using test data perturbed by SimBA and various configurations of transferred PGD perturbations. For transferred perturbations, We consider two source models: ResNet10 and WideResNet, denoted by suffixes R and W, respectively. All perturbations are bounded by the ℓ_∞-norm with $\epsilon = 8/255$. SOAR outperforms all other methods in terms of robustness against black-box attacks.

Method	SimBA	PGD20-R	PGD20-W	PGD1000-R	PGD1000-W
PGD10	47.27	77.19	79.48	77.22	79.55
TRADES	47.67	72.28	74.39	72.24	74.37
MART	48.57	72.99	74.91	72.99	75.04
MMA	43.53	78.70	80.39	78.72	81.35
FOAR	35.97	63.56	65.20	63.60	65.27
SOAR	68.57	79.25	86.35	79.49	86.47

are generated from an independently trained, undefended model.

In our evaluation, the transferred attacks consist of PGD20 and PGD1000 perturbations, generated using an independently initialized and trained ResNet and WideResNet source model, denoted by suffixes R and W, respectively. Beyond these transferred perturbations, Tramer et al. (2020) recommends considering score-based attacks such as Simple Black-box Attack (SimBA) (Guo et al., 2019), which are particularly relevant in real-world applications where gradient information is unavailable. SimBA is a query-based algorithm that iteratively samples, evaluates model responses, and refines the next sample to improve query efficiency. Empirically, attacks like SimBA have proven more effective than transferred PGD perturbations. For these reasons, we have included SimBA in our evaluations as well. All perturbations discussed in this section are ℓ_∞ constrained at $\epsilon = 8/255$.

SOAR achieves the highest level of robustness against all baseline methods trained on ResNet, as shown in Table 3.3. It is important to note that all defense methods are substantially more vulnerable to the score-based SimBA method, yet the SOAR regularized model remains the most robust method against SimBA.

Robustness Against AutoAttack

We also evaluate SOAR against a state-of-the-art method called AutoAttack (Croce et al., 2020). We observe that all models tested become more vulnerable to AutoAttack. This attack algorithm consists of an ensemble of four attacks: two parameter-free versions of PGD (APGD-CE and APGD-DLR), Fast Adaptive Boundary (FAB) attack (Croce et al., 2019), and a score-based black-box method called Square Attack (Andriushchenko et al., 2020). The major difference between the two PGD attacks is the loss they are based on: APGD-CE is based on the cross-entropy loss similar to the vanilla PGD (Madry et al., 2018), and APGD-DLR is based on the logit difference similar to the CW attack (Carlini et al., 2017).

For robustness evaluation, testing each sample across all four attack algorithms is not only computationally expensive but also unnecessary, as AutoAttack is successful if misclassification happens under any one of the algorithms. For a complete understanding of why models are vulnerable,

Table 3.4: **Evaluation of classification accuracy for models under ℓ_∞-norm constrained perturbations generated using AutoAttack (%).** AutoAttack is an ensemble of four ℓ_∞-bounded attacks: APGD-CE, APGD-DLR, FAB, and Square Attack. Here, we tested it against the four individual attacks included in AutoAttack.

Method	Untargeted APGD-CE		Targeted APGD-DLR		Targeted FAB		Square Attack	
ϵ	8/255	4/255	8/255	4/255	8/255	4/255	8/255	4/255
PGD10	41.57	61.95	38.99	60.11	39.68	60.26	47.84	63.02
TRADES	44.69	60.67	40.27	58.25	40.64	58.45	46.16	61.22
MART	45.01	62.05	39.22	58.38	39.90	58.65	46.90	62.03
MMA	35.59	58.15	34.77	57.82	35.50	58.24	45.24	63.99
FOAR	31.15	49.87	27.56	46.91	27.92	47.04	35.92	51.05
SOAR	53.40	63.87	18.25	52.64	20.22	53.29	35.94	63.90

however, it is necessary to evaluate them against each of the four individual attacks included in AutoAttack.

The results are summarized in Table 3.4. We observe that the robustness against untargeted APGD-CE is similar to that against PGD (Table 3.2), which is expected, as both attacks are based on cross-entropy PGD. The key difference is that APGD-CE incorporates a step-size selection strategy, whereas vanilla PGD uses a fixed step size. However, SOAR's robustness decreases against targeted APGD-DLR and targeted FAB attacks. Notably, at $\epsilon = 8/255$, SOAR is particularly vulnerable to targeted APGD-DLR, with a robust accuracy of only 18.25%. To explore whether SOAR could improve robustness at smaller perturbation scales, we evaluate under different ϵ values. At $\epsilon = 4/255$, SOAR demonstrated more consistent improvements in robustness.

We propose two hypotheses to explain the observed decrease in robustness. First, SOAR may overfit to perturbations generated using the cross-entropy loss. This is because APGD-DLR, which is based on logit differences, and FAB, which aims to find minimal perturbation distances, are fundamentally different from PGD which maximizes the cross-entropy loss. In theory, SOAR could be derived using losses other than cross-entropy; however, two issues remain: 1. previous work have discussed the challenges of formulating the logit difference objective for ℓ_∞-norm constrained perturbations (Carlini et al., 2017), and 2. could SOAR overfit to these alternative objectives, resulting in decreased robustness against PGD? These challenges need to be addressed.

The second hypothesis is that, although SOAR shows more consistent robustness improvements at smaller values of ϵ, this suggests that the techniques discussed in Section 3.3.4 may not fully resolve the issues arising from the second-order approximation. The PGD1-based initialization might still be insufficient for an accurate second-order approximation, limiting SOAR's effectiveness at improving robustness against perturbations with the original ϵ for which it was designed.

Finally, it is important to recognize SOAR's promising potential as a defense, as demonstrated by our results with transfer-based perturbations and score-based, query-based attacks like SimBA. This highlights that SOAR offers meaningful improvements in robustness beyond gradient-based attack algorithms.

3.5 Conclusions

This chapter proposed SOAR, a regularizer that improves the robustness of DNN to adversarial perturbations. SOAR was obtained using the second-order Taylor series approximation of the loss function w.r.t. the input, and approximately solving the inner maximization of the robust optimization formulation. We showed that training with SOAR leads to significant improvement in adversarial robustness under the ℓ_∞ and the ℓ_2 attacks.

In the following chapters, we build upon the analytical tools used in SOAR to explore different dimensions of adversarial robustness. Most notably, the robust optimization framework, along with the analysis of linear models, helps us understand the robustness of models trained with different optimization techniques in Chapter 4. SOAR, as a regularizer, represents an algorithmic approach to improving adversarial robustness. In the next chapter, we shift our focus to model architecture and investigate how robustness can be improved when the dataset exhibits a hierarchical structure between classes.

3.5.1 Challenges and Limitations

Batch Normalization: We observe that networks with BatchNorm layers do not benefit from SOAR in terms of adversarial robustness. Despite an extensive hyperparameter search, we were unable to achieve meaningful improvements in such networks. A related study by Galloway et al. (2019) explores the connection between BatchNorm and adversarial robustness, highlighting a significant gap in robustness between networks with and without BatchNorm layers in VGG-based architectures (Simonyan et al., 2014) under standard training. Further investigation into the interaction between SOAR and BatchNorm is necessary, and we consider this an important future research direction.

Starting from Pretrained Models: We found it challenging to train with SOAR on newly-initialized models. Fine-tuning on a pretrained model for specific tasks is a common approach, and similar to curriculum learning (Bengio et al., 2009), SOAR benefits from this approach. Initially, the model is trained on an easier task (standard training), followed by regularization for a related but more challenging task (improving adversarial robustness).

Catastrophic Overfitting: An important observation is that when models achieve high adversarial accuracy and continue training for extended periods, both standard and adversarial accuracy drop significantly. This phenomenon is similar to what has been described as *catastrophic forgetting* (Cai et al., 2018) and *catastrophic overfitting* (Wong et al., 2019a). Wong et al. (2019a) address this with early stopping. In our experiments, we found that using a large learning rate accelerates adversarial accuracy but leads to earlier catastrophic overfitting. As a solution, we fixed the number of epochs to 200 and carefully tuned learning rates to prevent overfitting.

Computational Complexity: Our primary goal is to propose regularization as an alternative approach to improving adversarial robustness. Although we have discussed techniques to improve implementation efficiency, there remains potential for further speedup. While SOAR is faster than MART and TRADES, it is still slower than PGD10 adversarial training. The computational complexity is characterized by the number of forward and backward passes for a single mini-batch, as shown in Table 3.5.

Table 3.5: **Comparison of Forward and Backward Passes for Different Methods**

Method	**Forward Passes**	**Backward Passes**
Standard Training	1	1
PGD-based Adversarial Training (k-step)	k+1	k+1
FOAR	1	2
SOAR	3	4

Appendices

In the following appendices, we provide some auxiliary results that are omitted from the main body of the chapter. In Appendix 3.A, we first describe the exact optimization schedule to train the baseline models. In Appendix 3.B, we compare the effect in the number of randomly sampled z on the SOAR regularized loss. In Appendix 3.C, we compare trained using SOAR and FOAR with different initializations. In Appendix 3.D, we evaluate the robustness of SOAR against ℓ_2-norm constrained perturbations on CIFAR-10. In Appendix 3.E, we evaluate the robustness of SOAR against ℓ_∞ bounded white-box PGD perturbations on models with different capacities. Finally, in Appendix 3.F, we include additional experiments to verify that SOAR improves robustness of the model without gradient masking.

3.A Implementation Details

Standard training: Models are trained for a total of 200 epochs, with an initial learning rate of 0.1. The learning rate decays by an order of magnitude at epoch 100 and 150. We used a mini-batch size of 128 for testing and training. We used SGD optimizer with momentum of 0.9 and a weight decay of 2e-4.

Adversarial training with PGD10: The optimization setting is the same as the one used for standard training. Additionally, to ensure that the final model has the highest adversarial robustness, we save the model at the end of every epoch, and the final evaluation is based on the one with the highest PGD20 accuracy.

SOAR: SOAR refers to continuing the training of the standard model on ResNet. It is trained for a total of 200 epochs with an initial learning rate of 0.004 and decay by an order of magnitude at epoch 100. We used SGD optimizer with momentum of 0.9 and a weight decay of 2e-4. We use a FD step-size $h = 0.01$ for the regularizer. Additionally, we apply a clipping of 10 on the regularizer.

MART and TRADES: We used the same optimization setup as the ones in their respective public repository.[2] We briefly summarize it here. The model is trained for a total of 120 epochs, with an initial learning rate of 0.1. The learning rate decays by an order of magnitude at epoch 75, 90, 100. We used SGD optimizer with momentum of 0.9 and a weight decay of 2e-4. We performed a hyperparameter sweep on the strength of the regularization term β and selected one that resulted in the best performance against PGD20 attacks.

[2] https://github.com/YisenWang/MART

Table 3.6: **Comparing the values of the regularized loss computed using different numbers of z at the beginning and the end of SOAR regularization.**

Checkpoints	$n = 1$	$n = 10$	$n = 100$
Beginning of SOAR	10.58	10.58	10.58
End of SOAR	1.57	1.56	1.56

Table 3.7: **Comparing SOAR with different initializations (%).** We follow the evaluation setting in Table 3.2 and evaluate FOAR on the test data perturbed using FGSM and various configurations of PGD.

Initialization	Standard accuracy	White-box PGD20	Transferred PGD20
zero	91.73	89.24	2.86
rand	91.70	90.82	9.16
PGD1	87.95	56.06	79.25

MMA: We used the same optimization setup as the one in its public repository.[3] We briefly summarize it here. The model is trained for a total of 50000 iterations, with an initial learning rate of 0.3. The learning rate changes to 0.09 at the 20000 iteration, 0.03 at the 30000 iteration and lastly 0.009 at the 40000 iteration. We used SGD optimizer with momentum of 0.9 and a weight decay of 2e-4. We performed a hyperparameter sweep on the margin term and selected the one that resulted in the best performance against PGD20 attacks.

3.B Effect of the Number of Randomly Sampled z on SOAR Regularized Loss

Suppose we slightly modify (3.14) by introducing

$$\ell_{\text{SOAR}}(x, y, n) = \ell(x', y) + \frac{1}{n}\sum_{i=0}^{n} R(x'; z_{(i)}, h, \epsilon)$$

to account for the effect of using multiple randomly sampled $z_{(i)}$ in computing the SOAR regularized loss. In Table 3.6, we observed the model at two checkpoints–initially and at the completion of SOAR regularization–and found that the value of the regularized loss remains almost unchanged as we increased n from 1 to 100, so the results reported in Section 3.4 uses $n = 1$.

Table 3.8: **Comparing FOAR with different initializations (%).** We evaluate the performance of SOAR with different initializations on CIFAR-10 against white-box and transfer-based black-box ℓ_∞ bounded adversarial perturbations ($\epsilon = 8/255$).

Initialization	Standard	FGSM	PGD20	PGD100	PGD200	PGD1000	PGD20-50
zero	74.43	33.39	23.65	22.83	22.79	23.23	68.58
rand	73.96	33.62	24.71	23.96	23.93	24.37	69.04
PGD1	65.84	36.96	32.28	31.87	31.89	32.08	63.56

3.C SOAR and FOAR with Different Initializations

SOAR with Different Initializations

We report the adversarial robustness of the model trained using SOAR with different initialization techniques in Table 3.7. Despite the high accuracy against white-box PGD perturbations, models initialized with zero and random settings exhibit poor performance against transferred attacks. This suggests the presence of gradient masking with zero and random initializations. In contrast, SOAR with PGD1 initialization appears to mitigates this issue.

FOAR with Different Initializations

For a more accurate second-order approximation, we approximate the loss function using inputs perturbed with PGD1 initialization. For a fair comparison, we also evaluate FOAR with different initializations. We observe that FOAR also benefits from PGD1 initialization, as shown in Table 3.8. Therefore, we only present this variation of FOAR in Section 3.4.

3.D Robustness Under ℓ_2-norm Constrained Perturbations

We evaluate SOAR and two of the baseline methods, PGD10 and TRADES, against ℓ_2 white-box and transferred attacks on CIFAR-10 in Table 3.9. No ℓ_2 results were reported by MART and we are not able to reproduce the ℓ_2 results using the implementation by MMA, thus those two methods are not included in our evaluation.

In Section 3.3.2, we show that the ℓ_∞ formulation of SOAR with $\|\delta\|_\infty = \epsilon$ is upper-bounded by the ℓ_2 formulation of SOAR with $\|\delta\|_2 = \epsilon\sqrt{d}$. In other words, models trained with SOAR to be robust against ℓ_∞ attacks with $\epsilon = \frac{8}{255}$ should also exhibit improved robustness against ℓ_2 attacks with $\epsilon = \frac{8}{255}\sqrt{32 * 32 * 3} = 1.74$. In our evaluation, all ℓ_2 adversaries used during PGD10 and TRADES are generated with 10-step PGD ($\epsilon = 1.74$) and a step size of 0.44. Note that the goal here is to show the improved robustness of SOAR against ℓ_2 attacks rather than being SOTA. The optimization procedures are the same as the ones used in the ℓ_∞ evaluation.

We observe that training with SOAR improves the robustness of the model against ℓ_2 attacks. We demonstrate the improved robustness using an increasing range of ϵ, with perturbations generated using 100 iterations of PGD and a step size of $\frac{2.5\epsilon}{100}$. In Table 3.9, we find that training with SOAR

[3] https://github.com/BorealisAI/mma_training

Table 3.9: **Evaluation of classification accuracy for models under ℓ_2-norm constrained white-box and transferred perturbations (%).** We evaluate the model robustness on the testing data perturbed using PGD100 under various ϵ. The last column shows the robustness against transferred ℓ_2 PGD perturbations with $\epsilon = 1.74$.

Method	$\epsilon = \frac{60}{255}$	$\epsilon = \frac{120}{255}$	$\epsilon = \frac{255}{255}$	$\epsilon = 1.74$	Transfer
PGD10	68.13	62.03	47.53	28.09	70.86
TRADES	68.59	62.39	45.42	25.67	69.02
SOAR	75.39	66.81	60.90	56.89	69.52

significantly increases robustness against white-box and transferred ℓ_2 adversaries. As ϵ increases, SOAR remains robust against white-box ℓ_2 attacks ($\epsilon = 1$), while the effectiveness of other methods diminishes.

The last column of Table 3.9 shows the robustness against transferred ℓ_2 attacks ($\epsilon = 1.74$), where the source model is a separately trained ResNet10 using the unperturbed training set. SOAR achieves the second-highest robustness compared to baseline methods against these transferred ℓ_2 attacks. This result supports findings by Simon-Gabriel et al. (2019), who empirically showed that adversarial robustness achieved through regularization can defend against multiple ℓ_p-norm perturbations simultaneously, a benefit not observed with adversarial training.

3.E Robustness Improvement with Increasing Model Capacities

Empirical studies reveal that increasing model capacity can significantly enhance adversarial robustness (Madry et al., 2018; Wang et al., 2020b) gained through adversarial training with PGD adversaries. One hypothesis is that models with higher capacity have a greater ability to generalize across different examples, including adversarial ones. Our result shows that such an improvement with increasing model capacity is not limited to adversarial training but also extends to regularization-based methods like SOAR.

Table 3.10 compares the robustness of SOAR against ℓ_∞-norm bounded white-box PGD perturbations on networks with varying capacities. CNN6 and CNN8 denote convolutional networks with 6 and 8 convolution layers, respectively, while ResNet10 is the network used in Section 3.4. The results show that as network capacity increases, there are improvements in both standard accuracy and adversarial accuracy. We expect further robustness improvement with larger capacity networks.

3.F Discussion on Gradient Masking

To confirm that SOAR improves the robustness of the model without inducing gradient masking, we conduct several experiments detailed below.

In the ℓ_∞ attack setting, PGD generates perturbations using the sign of the gradient, $\text{sign}(\nabla_x \ell(x))$. When elements in $\nabla_x \ell(x)$ is too small or zero, the gradient direction may not reflect the true ascent direction. To assess the strength of the gradient, we measure the number of non-zero elements in

Table 3.10: **Evaluation of classification accuracy for SOAR-regularized models with different capacities (%).** In addition to ResNet10, we consider two smaller convolutional networks, denoted as CNN6 and CNN8. Evaluations are conducted on CIFAR-10 against white-box ℓ_∞ bounded PGD perturbations ($\epsilon = 8/255$). We find that SOAR-regularized models with larger capacities exhibit better robustness against adversarial perturbations.

Model	Standard	PGD20	PGD100	PGD200
CNN6	81.73	32.83	31.20	31.15
CNN8	83.65	47.30	46.07	45.83
ResNet10	87.95	56.06	55.00	54.94

the gradient, averaged over all test inputs. This value is defined as:

$$\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{d}\mathbb{I}(|\nabla_{x_i}\ell(x_i)_j| > 10^{-6}),$$

where N denotes the number of test inputs, d is the dimension of the input, and $\mathbb{I}(\cdot)$ is the indicator function. For CIFAR-10, $N = 10000$ and $d = 32 \times 32 \times 3 = 3072$.

A model with gradient masking exhibits far fewer non-zero elements. In our experiments, the average number of non-zero gradient elements was 3072 for PGD10-trained, indicating no gradient masking, 3069 for SOAR with PGD1 initialization, and only 1043 for SOAR with zero initialization, which indicates gradient masking. These results suggest that SOAR with PGD1 initialization preserves a similar number of meaningful gradient elements as the PGD10 model, allowing the PGD adversary to generate effective perturbations.

Furthermore, in Section 3.4, 20-iteration ℓ_∞ PGD adversaries are generated with a step size of $\frac{2}{255}$ and $\epsilon = \frac{8}{255}$. Suppose we use $\epsilon = 1$ instead of $\epsilon = \frac{8}{255}$, while keeping all other parameters the same. This means allowing the maximum ℓ_∞ perturbation to span the entire input range $[0, 1]$ and generating PGD20 attacks. We observe that these attacks result in near black-and-white images for SOAR with PGD1 initialization, and the model has 0% accuracy against such PGD20 attacks, indicating that the model is not robust against such strong perturbations. This suggests that SOAR does not induce gradient masking and that the model is indeed robust against white-box perturbations.

Chapter 4

Understanding the Robustness Differences between SGD and Adaptive Gradient Methods

4.1 Introduction

Adaptive gradient methods, such as Adam (Kingma et al., 2015) and RMSProp (Hinton et al., 2012), are a family of popular techniques to optimize machine learning models. They are an extension of the traditional GD method, which uses the gradient of a differentiable objective function to update the model's parameters in the direction that improves the objective. To speed up the optimization procedure, the adaptive gradient methods introduce a coordinate-wise learning rate to adjust the update for each parameter based on its individual gradient. Previous empirical work investigates the difference in the standard generalization between models trained using SGD and adaptive gradient methods (Wilson et al., 2017; Agarwal et al., 2020), while recent efforts have focused on understanding the implicit bias of SGD (Gunasekar et al., 2017; Soudry et al., 2018; Lyu et al., 2020) and adaptive gradient algorithms (Qian et al., 2019; Wang et al., 2021a).

Nevertheless, our result shows that in practice such a gap in the standard generalization is relatively small, in contrast to the difference between the robustness of models trained using those algorithms. While more ML-based systems are deployed in the real world, the models' robustness, their ability to maintain their performance when faced with noisy or corrupted inputs, has become an important criterion. There is a large volume of literature on developing specialized methods to improve the robustness of neural networks (e.g., Section 2.4 and Chapter 3), yet practitioners still simply use standard methods to train their models (Silva et al., 2020). In fact, a recent survey shows that only 3 of the 28 organizations have developed their ML-based systems with the improvement in robustness in mind (Kumar et al., 2020).

Therefore, this motivates us to understand the effect of optimizers on the robustness of models obtained in the standard training regime. In particular, we focus on models trained using SGD and

This chapter is based on our work in Understanding the robustness difference between stochastic gradient descent and adaptive gradient methods.

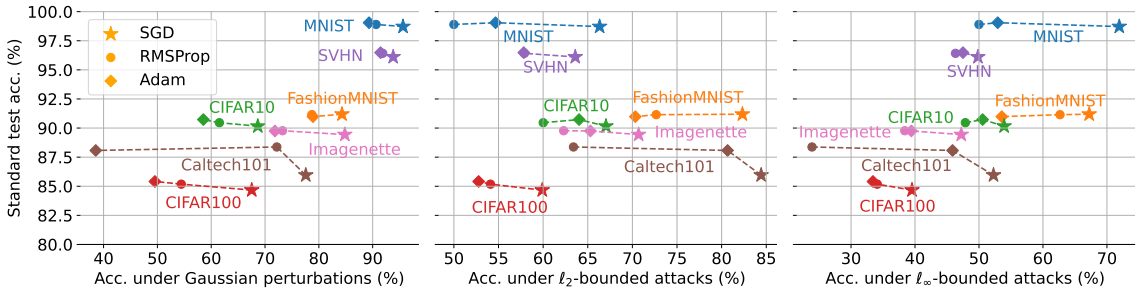


Figure 4.1: **Comparison between models trained using SGD, Adam, and RMSProp across seven benchmark datasets.** Each colored triplet denotes models on the same dataset. Models trained by different algorithms have similar standard generalization performance, but there is a distinct robustness difference as measured by the test data accuracy under Gaussian noise, ℓ_2 and ℓ_∞ bounded adversarial perturbations (Croce et al., 2020). Results are averaged over three independent model initializations and trainings.

adaptive gradient methods.

4.1.1 The Robustness Difference between Models Trained by Different Algorithms

As a first step, we compare how models, trained with SGD, Adam, and RMSProp, differ in their standard generalization and robustness on seven benchmark datasets (LeCun, 1998; Xiao et al., 2017; Krizhevsky et al., 2009; Netzer et al., 2011; Howard, 2019; Li et al., 2004). In our experiments, we evaluate standard generalization using the accuracy of the trained classifier on the original test dataset. To measure robustness, we consider the classification accuracy on the test dataset perturbed by Gaussian noise, as well as ℓ_2- and ℓ_∞-bounded adversarial perturbations (2.3), generated using the methods described in (Croce et al., 2020). We follow the default Pytorch configuration to train all the models and sweep through a wide range of learning rates. The final model is selected with the highest validation accuracy. Implementation details are discussed in Appendix 4.A.

The complete result of the experiment can be found in Appendix 4.B. In this section, we visualize the difference between models trained with SGD and the adaptive gradient methods in Figure 4.1, pointing to two important observations. First, the relatively small vertical differences among the three models, on a given dataset, show that the models have similar standard generalization performance despite being trained by different algorithms. On the other hand, we observe, under all three types of perturbations, a large horizontal span with SGD always positioned on the far right side among the three. This indicates that models trained by SGD significantly outperform models trained by the other two in terms of their robustness against perturbations.

While our primary experiments are centered around models based on convolutional neural networks, within the computer vision domain, we also extend our analysis to include results from experiments on Vision Transformers (Dosovitskiy et al., 2021) and an audio dataset (Warden, 2018). The results of these additional experiments are consistent with the findings presented in Figure 4.1 and are detailed in Appendix 4.B. Visualizations of the perturbations in the evaluation can be found in Appendix 4.E.

4.1.2 Contributions

Previous optimization work often studies how the structure of the dataset affects the dynamics of learning. For example, some focus on a dataset with different feature strengths (Amari et al., 2021; Pezeshki et al., 2021), while others assume a linearly separable dataset (Wilson et al., 2017; Gunasekar et al., 2017; Soudry et al., 2018). In our work, we investigate how the frequency characteristics of the dataset impact the robustness of models trained by SGD and adaptive gradient methods. We make the following contributions:

- We demonstrate that natural datasets contain irrelevant frequencies, which, when removed, have negligible effects on standard generalization performance (Section 4.3.1).
- We also observe that neural networks trained by different algorithms can have very different robustness against perturbations in the direction of the irrelevant frequencies (Section 4.3.2).
- Those observations lead to our claim that models only need to learn how to correctly use relevant information in the data to optimize the training objective, and because their use of the irrelevant information is under-constrained, it can lead to solutions sensitive to perturbations (Section 4.3).
- Our analysis of linear models on least square regression shows that linear models' robustness to ℓ_2-norm bounded changes is inversely proportional to the model parameters' weight norm: a smaller weight norm implies better robustness (Section 4.4.1).
- We study the learning dynamics of GD and signGD, a memory-free version of Adam and RMSProp, with linear models. With a three-dimensional input space, the analysis shows that models optimized with GD exhibit a smaller weight norm compared to their signGD counterparts (Section 4.4.2).
- To generalize this result in the deep learning setting, we demonstrate that neural networks trained by Adam and RMSProp often have a larger Lispchitz constant and, consequently, are more prone to perturbations (Section 4.5).

Specifically, in the analysis of linear models, we design a least square regression task using a synthetic dataset whose frequency representation mimics the natural datasets. This setting allows us to i) mathematically define the standard and adversarial population risks, ii) design a learning task that has multiple optima for the standard population risk, each with a different adversarial risk, and iii) theoretically analyze the learning dynamics of various algorithms.

4.2 Background

In this section, we briefly review the essential background to help understand our work, including formulations of adaptive gradient methods and methods of representing signals in the frequency domain.

4.2.1 Optimizations with Adaptive Gradient Algorithms

Consider the ERM problem introduced in Section 2.1. A common approach in solving ERM is SGD, which minimizes the loss by iteratively updating the model parameters, w, using mini-batches of

data points sampled uniformly and independently from the training set $(X_n, Y_n)_{n=1}^N$:

$$g(w) = \frac{1}{|\mathcal{B}|} \sum_{n \in \mathcal{B}} \nabla_w \ell(X_n, Y_n; w), \tag{4.1}$$

where $\mathcal{B} \subset \{1, ..., N\}$ denotes the mini-batch and has a size of $|\mathcal{B}| \ll N$. The update rule of SGD is $w(t+1) = w(t) - \eta(t) g(w(t))$, where $\eta(t) \in \mathbb{R}^+$ denotes the learning rate.

A family of adaptive gradient methods has been used to accelerate training by updating the model parameters based on a coordinate-wise scaling of the original gradients. Methods such as Adam and RMSprop have demonstrated significant acceleration in training DNNs (Kingma et al., 2015). Many adaptive gradient methods can be written as

$$\begin{aligned} m(t+1) &= \beta_1 g(w(t)) + (1-\beta_1) m(t) \\ v(t+1) &= \beta_2 g(w(t))^2 + (1-\beta_2) v(t) \\ w(t+1) &= w(t) - \eta(t) \frac{m(t+1)}{\sqrt{v(t+1)} + \epsilon}, \end{aligned} \tag{4.2}$$

where $g(w(t))$ is the stochastic estimate of gradient used by SGD (4.1), m and v are the first and second-order memory terms with their strength specified by β_1 and β_2, and ϵ is a small constant used to avoid division-by-zero. Such a general form has been widely used to study the dynamics of adaptive gradient algorithm (Wilson et al., 2017; Silva et al., 2020; Ma et al., 2022b). For example, Adam corresponds to $\beta_1, \beta_2 \in (0, 1)$, and RMSProp is recovered when $\beta_1 = 1$ and $\beta_2 \in (0, 1)$. Notice that such updates rely on the history of past gradients, and this makes the precise understanding and analysis of adaptive gradient methods more challenging (Duchi et al., 2011).

Recent work analyzes the learning dynamics of adaptive gradient methods by separately considering the direction and the magnitude of the update (Kingma et al., 2015; Balles et al., 2018; Ma et al., 2022b). As a simple example, to demonstrate how adaptive gradient methods can potentially accelerate learning compared to the vanilla SGD, consider a memory-free version of (4.2) with $\beta_1 = \beta_2 = 1$ and $\epsilon = 0$. It is easy to see that the update rule in (4.2) becomes signGD:

$$\begin{aligned} w(t+1) &= w(t) - \eta(t) \operatorname{sign}(g(w(t))) \\ &= w(t) - \vec{\eta}(t) \odot g(w(t)), \end{aligned} \tag{4.3}$$

where $\odot$ denotes Hadamard product, $\vec{\eta}(t) \in \mathbb{R}^d$ is a coordinate-wise learning rate based on the absolute value of the weight, i.e., $\vec{\eta}(t) = \frac{\eta(t)}{|g(w(t))|}$. Therefore, $\vec{\eta}(t)$ accounts for the magnitude of the weight and a larger learning rate is used for parameters with smaller gradients.

In general, gradient-sign-based optimization methods are not successful in training deep learning models (Riedmiller et al., 1993; Ma et al., 2022b), nevertheless, methods such as signGD can shed light on the learning dynamics of adaptive gradient methods (Karimi et al., 2016; Balles et al., 2018; Moulay et al., 2019). For example, recent work by Ma et al. (2022b) studies the behavior of adaptive gradient algorithms in the continuous-time limit. They demonstrate that under a fixed β_1 and β_2, the memory effect for both Adam and RMSprop diminishes and the continuous-time limit of the two algorithms follows the dynamics of signGD flow. In this work, the deep learning models on which we observe the robustness difference are trained using Adam and RMSProp, with the exception of

Section 4.4, where we focus on signGD, a memory-free version of Adam and RMSProp, and GD to help us understand the robustness gap between models trained using SGD and adaptive gradient methods in a simple setting.

4.2.2 Frequency Representation of Signals

Natural signals are highly structured (Schwartz et al., 2001). They often consist of statistically significant (or insignificant) patterns with a large amount of predictability (or redundancy). Such a phenomenon has been observed in both natural images (Ruderman, 1994; Simoncelli, 1997; Huang et al., 1999) and natural audio signals (McAulay et al., 1986; Attias et al., 1996; Turner, 2010). To understand the structure of signals and identify patterns from them, one technique is to decompose the signal into multiples of "harmonics" or "overtones": a superposition of periodic waves with varying amplitudes and in varying phases. For example, Fourier (1822) first proposed to analyze complicated heat equations using well-understood trigonometric functions, a method now called the Fourier transformation. This new representation allows us to precisely study the structure and the magnitude of any repeating patterns presented in the original waveform. For the understanding of digital signals, such a process is called discrete-time signal processing (Oppenheim et al., 2001).

Many discrete harmonic transformations exist, such as the discrete Fourier transform, the discrete cosine transform (DCT) (Ahmed et al., 1974) and the wavelet transform (Mallat, 1999). The analysis in this work utilizes the type-II DCT, but other techniques can be applied as well and we expect similar results.

Concretely, consider a d-dimensional signal $x \in \mathbb{R}^d$ in the spatial domain. The same signal can be alternatively represented as a discrete sum of amplitudes multiplied by its cosine harmonics:

$$\tilde{x}_k = \sum_{j=0}^{d-1} x_j \cos\left[\frac{\pi}{d}\left(j+\frac{1}{2}\right)k\right],$$

for $k = 0, ..., d-1$, where the transformed signal $\tilde{x}$ has a frequency-domain representation.[1] Because DCT is linear, it can be carried out using a matrix operation, i.e., $\tilde{x} = Cx$, where C is a $d \times d$ DCT transformation matrix with values specified by

$$C_{kj}^{(d)} = \sqrt{\frac{\alpha_k}{d}} \cos\left[\frac{\pi}{d}\left(j+\frac{1}{2}\right)k\right], \tag{4.4}$$

where $\alpha_0 = 1$ and $\alpha_k = 2$ for $k > 0$. In particular, $\tilde{x}$ can be written as a matrix-vector product between the transformation matrix C and the column vector x:

$$\begin{bmatrix} \tilde{x}_0 \\ \tilde{x}_1 \\ \vdots \\ \tilde{x}_{d-1} \end{bmatrix} = \begin{bmatrix} \sqrt{\frac{1}{d}} & \sqrt{\frac{1}{d}} & \cdots & \sqrt{\frac{1}{d}} \\ \sqrt{\frac{2}{d}}\cos\frac{\pi(2(0)+1)(1)}{2d} & \sqrt{\frac{2}{d}}\cos\frac{\pi(2(1)+1)(1)}{2d} & \cdots & \sqrt{\frac{2}{d}}\cos\frac{\pi(2(d-1)+1)(1)}{2d} \\ \vdots & \vdots & \vdots & \vdots \\ \sqrt{\frac{2}{d}}\cos\frac{\pi(2(0)+1)(d-1)}{2d} & \sqrt{\frac{2}{d}}\cos\frac{\pi(2(1)+1)(d-1)}{2d} & \cdots & \sqrt{\frac{2}{d}}\cos\frac{\pi(2(d-1)+1)(d-1)}{2d} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{d-1} \end{bmatrix}.$$

Notice that C is a real orthogonal matrix whose rows consists of periodic cosine bases with increasing frequencies. Therefore, the absolute value of $\tilde{x}$ at a particular dimension indicates the

[1] Indices range from 0 to $d-1$, as zero-frequency is commonly used to refer to a signal with a constant everywhere.

magnitudes of the corresponding basis function, and a higher dimension in $\tilde{x}$ means the basis function is of higher frequency. Another important property of DCT is its invertibility. That is, signals in the frequency domain can be converted back to the spatial-temporal domain via the inverse DCT (iDCT):

$$x = C^{-1}\tilde{x} = C^{\top}\tilde{x}.$$

In the example above, we discussed one-dimensional DCT which is applied to vectors and is used in the linear analysis in Section 4.4. Transformations on images require two-dimensional DCT and can be done using $\tilde{x} = CxC^{\top}$, where $x, \tilde{x} \in \mathbb{R}^{d\times d}$, and C is defined in (4.4); and the inverse two-dimensional DCT is $x = C^{\top}\tilde{x}C$. For more details on two-dimensional DCT, we refer the reader to Pennebaker et al. (1992).

Previous work analyzes the sensitivity of neural network classifiers by examining the frequency characteristics of various types of perturbations, with an emphasis on understanding how data augmentation affects the robustness of the model (Dong et al., 2019a). In our work, the frequency interpretation of signals is an integral part of understanding the robustness difference between models trained by SGD and adaptive gradient methods. This perspective allows us to study the structure of complex signals using well-understood periodic basis functions such as cosines and understand the energy distribution of signals by examining the amplitude of the basis function. In particular, the energy of a discrete signal x is defined as $E(x) = \sum_{i=0}^{d-1} |x_i|^2$, and by Parseval's theorem, is equivalent to the sum of squared amplitudes across all the bases, i.e., $E(x) = E(\tilde{x}) = \sum_{i=0}^{d-1} |\tilde{x}_i|^2$.

Natural images are primarily made of low-frequency signals[2]: a high concentration of energy in the low-frequency harmonics renders the amplitude of the higher-frequency harmonics almost negligible (Tolhurst et al., 1992; Schaaf et al., 1996).

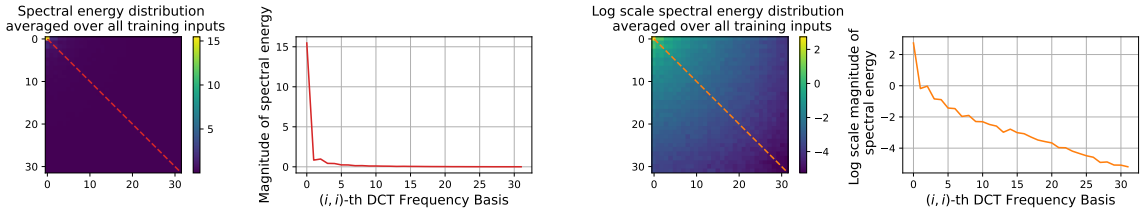
In Figure 4.2, we visualize the energy distribution for CIFAR-100 and Imagenette, with each dataset illustrated through four plots. The (i, j) coordinate in the first plot indicates the average amplitude, $\frac{1}{N}\sum_{n=1}^{N} |\tilde{x}_{n;(i,j)}|$, for the (i, j)-th basis across all N training images, where $\tilde{x}_n$ represents the DCT transformation of the n-th image, x_n, and $\tilde{x}_{n;(i,j)}$ represents the amplitude of the (i, j)-th basis in the n-th sample. The second plot focuses on the diagonal elements of the first, $\left\{\frac{1}{N}\sum_{n=1}^{N} |\tilde{x}_{n;(i,i)}|\right\}_{i=0,\ldots,d-1}$. The two plots are then drawn on a logarithmic scale to highlight the pronounced concentration of energy around the low-frequency harmonics, while the amplitudes for higher-frequency harmonics diminish significantly. Additional figures for other datasets can be found in Appendix 4.E.

Moreover, we show in Section 4.3.1 that there exist frequencies in natural datasets, which if removed from the training data, do not affect the standard generalization performance of the model. Based on this observation, in Section 4.4, we construct a synthetic dataset that mimics the characteristics of natural signals, and it allows us to study the learning dynamics of various optimization algorithms in a controlled setting.

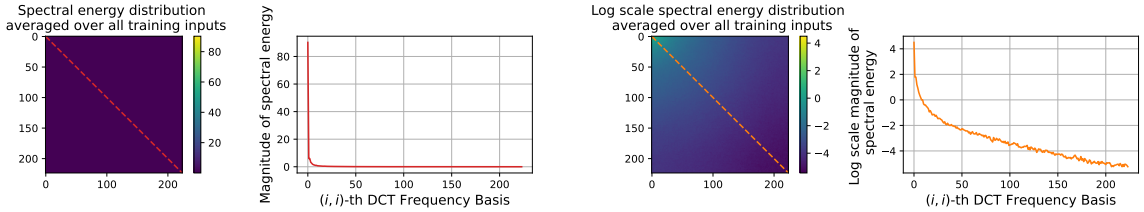
4.3 A Claim on How Models Use Irrelevant Frequencies

Why do models trained by different optimization algorithms behave similarly in the standard setting where the training and the test inputs are i.i.d., while they perform drastically differently when

[2]We will always use the term "high" or "low" frequency on a relative scale.



a. CIFAR-100



b. Imagenette

Figure 4.2: **Illustration of the spectral energy distribution in natural datasets.** Distribution of the spectral energy heavily concentrates at low frequencies and decays exponentially towards higher frequencies.

faced with noisy or corrupted data? To answer this question, we first observe that there is irrelevant information in the natural dataset (Observation I), and attenuating them from the training input has negligible effects on the standard generalization of the model. This leads to our claim:

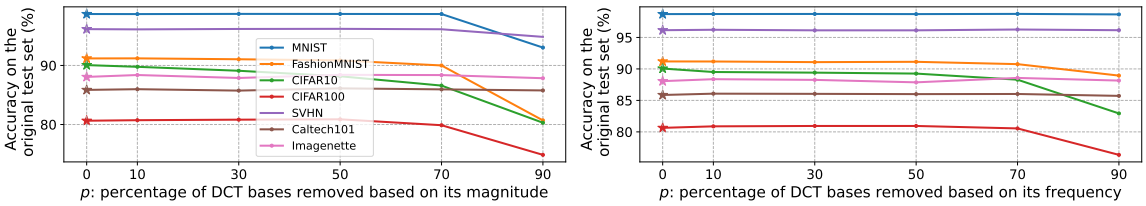
Claim 4.3.1. *To optimize the standard training objective, models only need to learn how to correctly use relevant information in the data. Their use of irrelevant information in the data, however, is under-constrained and can lead to solutions sensitive to perturbations.*

Because of this, by targeting the perturbations toward the subset of the signal that contains irrelevant information, we notice that models trained by different algorithms exhibit very different performance changes (Observation II).

4.3.1 Observation I: Irrelevant Frequencies in Natural Signals

Previous work demonstrated that the magnitude of the frequency components in natural images decreases as the frequency increases (Ruderman, 1994; Wainwright et al., 1999), which we have observed in Figure 4.2. The spectral sensitivity of the human eyes is limited (Gross, 2005), so patterns with low magnitudes and high frequencies are not important from the perspective of human observers, as they appear to us as nearly invisible and unintuitive information in the scene (Schwartz et al., 2001; Schwartz, 2004). For machines, image-processing methods have long exploited the fact that most of the content-defining information in natural images is represented in low frequencies, and the high-frequency signal is redundant, irrelevant, and is often associated with noise (Wallace, 1991; Guo et al., 2020a; Sharma et al., 2019).

Similarly, the notion of irrelevant frequencies also exists when training a neural network classifier. One way to illustrate this is by taking a supervised learning task, removing the irrelevant information from the training input, and then assessing the model's performance using the original test data. We observe that when modifying the training dataset by removing subsets of the signal with low



a. Parts of the signal with low spectral energy is irrelevant. b. Parts of the signal with high-frequency basis is irrelevant.

Figure 4.3: **Irrelevant frequencies exist in natural datasets.** Accuracy on the original test set remains high when the training inputs are modified by removing parts of the signal with a) low spectrum energy and b) high frequencies. Stars represent test accuracy on models trained using the original training input. In setting a), training images are filtered based on the magnitude of the DCT basis. Specifically, parts of the image with DCT bases that have a magnitude in the bottom $\frac{p}{100}$-th percentile are removed, so a large p means more information is discarded. In setting b), training images are low-pass filtered, and p denotes the percentage of the high-frequency components that are discarded in the training data. We explain the formulation of the two settings in Appendix 4.C. Examples of the modified inputs are included in Appendix 4.E.

spectral energy (Figure 4.3a) or high frequencies (Figure 4.3b), there is a negligible effect on models' classification accuracy on the original test data. In Figure 4.4, we illustrate the filtering process in both scenarios. The image remains nearly identical to the original, even with 90% of the DCT bases removed. Note that our filtering is different from JPEG compression (Pennebaker et al., 1992) in how it handles irrelevant frequencies. JPEG removes high-frequency components locally, introducing block-like artifacts in the form of class-irrelevant, high-frequency DCT bases. In contrast, our filtering is a global operation that does not introduce such compression artifacts. For a comprehensive explanation of the modifications applied to the images, refer to Appendix 4.C. Further, Appendix 4.E contains additional visualizations of the modified images across various datasets.

Results in Figure 4.3 shows that in both settings after reducing more than half of the DCT basis vectors to zeroes in the training data, the model's generalization ability remains strong. This observation suggests there is a considerable amount of irrelevant information in naturally occurring data from the perspective of a neural network classifier, and such information is often featured with low spectrum energy or lives at the high end of the frequency spectrum.

This observation leads to the first part of Claim 4.3.1. That is, models only need to learn how to correctly use the crucial class-defining information from the training data to optimize the training objective. On the other hand, the extent to which they utilize irrelevant information in the data is not well-regulated. This can be problematic and lead to solutions sensitive to perturbations. In Section 4.4, we validate Claim 4.3.1 using a linear regression analysis with a synthetic dataset that contains irrelevant information. We demonstrate there exist multiple optima of the training objective and those solutions can all correctly use the relevant information in the data, but the way they exclude irrelevant information from computing the output is different. Specifically, a robust model disregards irrelevant information by assigning a weight of zero to it, but a non-robust model has certain non-zero weights which, when combined with the irrelevant information in the input, yield a net-zero effect in the output. In this case, although the two models are indistinguishable under the original training objective, the non-robust model will experience a reduction in model performance should this irrelevant information become corrupted at test time.

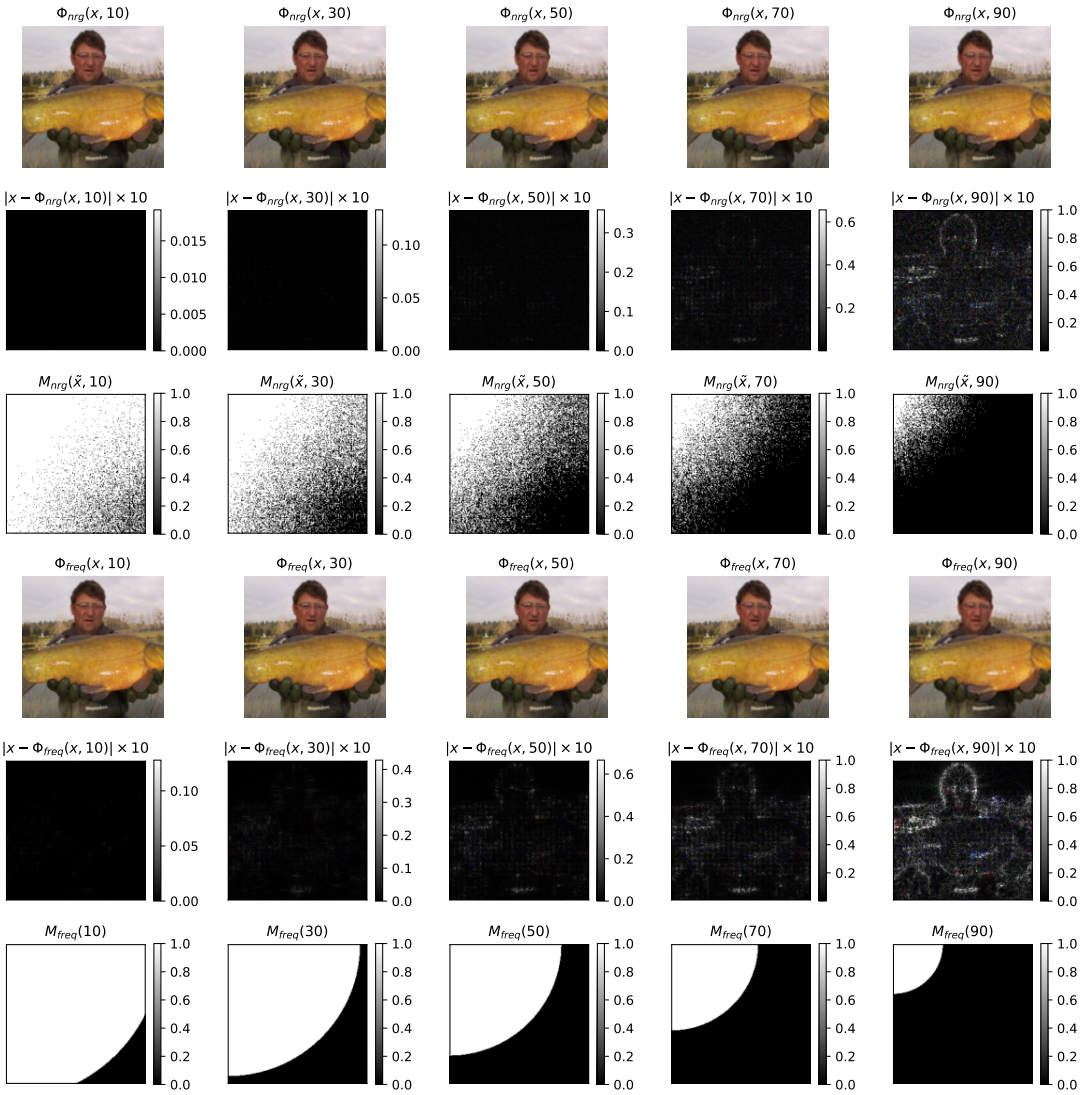


Figure 4.4: **Examples of filtered images used in Observation I. (Imagenette)** We use a threshold value of $\texttt{threshold} = \{10, 30, 50, 70, 90\}$ to modify images based on its magnitude of DCT basis and their frequency basis. The threshold value is the percentage of the DCT basis vectors that are removed from the image. We show images modified by removing DCT basis vectors whose magnitudes are in the bottom `threshold` percentage (row 1), the differences between the modified images and the original image (row 2), the binary mask used to remove the DCT basis: black means removed (row 3), images modified by removing high-frequency DCT basis vectors (row 4), the differences between the modified images and the original image (row 5) and the binary mask used to remove the DCT basis: black means removed (row 6). Notice that the masks in row 6 only depends on the dimension of the images, whereas the masks in row 3 differs from images to images.

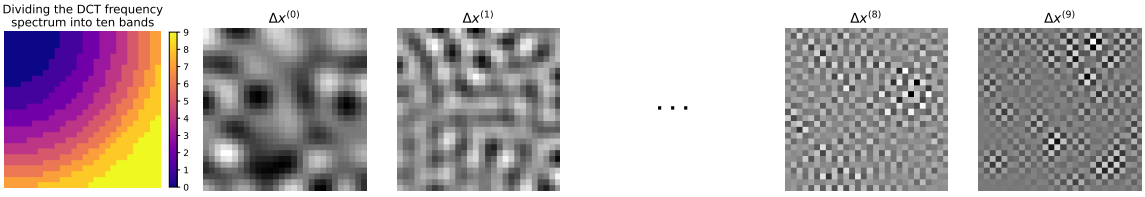


Figure 4.5: **Visualization of the band-limited Gaussian perturbations.** The DCT spectrum is divided into ten equally sized bands to generate band-limited Gaussian perturbations. Denote them by using $\Delta x^{(i)}$, where $i \in \{0, 1, ..., 9\}$. The frequency represented in the spectrum plot increases from the top-left (lowest frequency) to the bottom-right corner (highest frequency). Therefore, as the band moves towards higher frequencies, perturbations exhibit more high-frequency checkerboard patterns.

4.3.2 Observation II: Model Robustness along Irrelevant Frequencies

Let us now focus on the second part of the claim. If models' responses to perturbations along the irrelevant frequencies explain their robustness difference, then we should expect a similar accuracy drop between models when perturbations are along relevant frequencies, but a much larger accuracy drop on less robust models when test inputs are perturbed along irrelevant frequencies. Consider the robustness of the models when the test data are corrupted with Gaussian noise: the perturbation along each spatial dimension is i.i.d and drawn from a zero-mean Gaussian distribution with finite variance. This type of noise is commonly referred to as the additive white Gaussian noise, where white refers to the property that the noise has uniform power across the frequency spectrum (Diebold, 1998). Nevertheless, the previous discussion suggests that noise along different frequencies does not have an equal impact on the models' output. To verify this, we assess the impact on model accuracy by perturbing only specific frequency ranges of the test inputs with band-limited Gaussian noise.

To construct the band-limited Gaussian noise, we first follow the previous work (Wang et al., 2020a) to group DCT basis vectors based on their distance to the 0-frequency DC term and divide the entire DCT spectrum into ten bands where each band occupies the same number of DCT bases. This is to ensure an identical ℓ_2 norm among all the perturbations. Denote the binary mask of the i-th band by using $M^{(i)} \in \{0, 1\}^{d \times d}$, its corresponding band-limited Gaussian noise is $\Delta x^{(i)} = C^\top (M^{(i)} \odot \delta) C$, where $\delta \sim \mathcal{N}(0, \sigma^2 I_{d \times d})$ and C is the DCT transformation matrix defined in (4.4). Figure 4.5 illustrates how the frequency bases are grouped into ten equally sized bands and examples of the band-limited Gaussian noise. Denote the perturbations by using $\Delta x^{(i)}$, with $\Delta x^{(0)}$ and $\Delta x^{(9)}$ representing the lowest and the highest band, respectively. To investigate the effect of the perturbation $\Delta x^{(i)}$ on the models, we measure the change in classification accuracy when the test inputs are perturbed by $\Delta x^{(i)}$:

$$\frac{1}{N} \sum_{n=1}^{N} \mathbb{I}\{F(X_n) = Y_n\} - \frac{1}{NK} \sum_{n=1}^{N} \sum_{k=1}^{K} \mathbb{I}\left\{F(X_n + \Delta x_k^{(i)}) = Y_n\right\}, \tag{4.5}$$

where F is a neural network classifier, $\{(X_n, Y_n)\}_{n=1}^{N}$ denotes the test dataset, and each input X_n is perturbed by $K = 10$ i.i.d. noise samples $\Delta x_k^{(i)}$. The same K perturbations are applied across all inputs. It is important to realize in (4.5) that the additive noise Δx is applied to the spatial signal X, although we are limiting the frequency band of the noise.

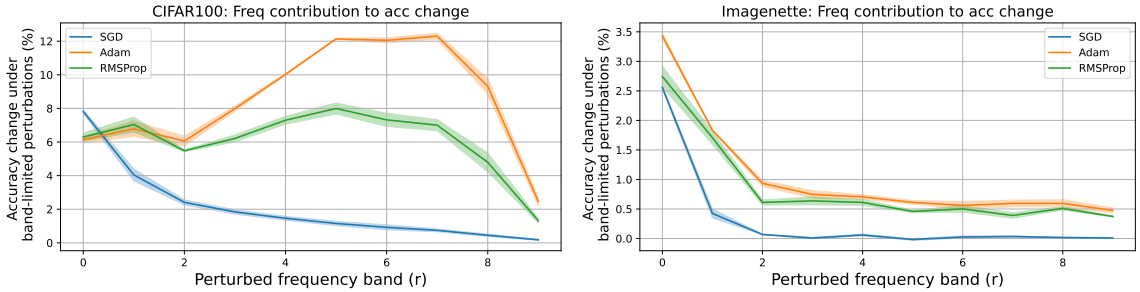


Figure 4.6: **The effect of band-limited Gaussian perturbations on the model.** Perturbations from the lowest band, i.e., $\Delta x^{(0)}$, have a similar effect on all the models, despite being trained by different algorithms and exhibiting different robustness properties. On the other hand, models' responses vary significantly when the perturbation focuses on higher frequency bands. The results are averaged over three independently initialized and trained models, and the shaded area indicates the standard error among the three models.

Figure 4.6 demonstrates how the classification accuracy degrades under different band-limited Gaussian noises on CIFAR-100 and Imagenette; and results on the other datasets are included in Appendix 4.E. First, notice that the perturbation from the lowest band $\Delta x^{(0)}$ has a similar impact on all the models regardless of the algorithm they are trained by. There is however a noticeable difference in how models trained by SGD and adaptive gradient methods respond to perturbations from higher frequency bands. On models trained by SGD, the flattened curve implies that the effect of high-frequency perturbations on the generalization performance quickly diminishes to zero, suggesting that models are not sensitive to changes along the dimensions of irrelevant frequencies. Contrarily on models trained by the two adaptive gradient methods, we observe a difference in the way models respond to perturbations of higher frequency bands. On CIFAR-100, for example, the two models are highly vulnerable to Gaussian perturbations from bands 5 to 7. This observation shows that when models, during their training phase, do not have mechanisms in place to limit their use of irrelevant frequencies, their performance can be compromised if data along irrelevant frequencies become corrupted at test time.

We also observe that models' responses to high-frequency Gaussian perturbations varies among datasets. This can be attributed to the fact that (ir-)relevant frequencies are most likely going to be a unique characteristic for a particular dataset. We do not expect a dataset that solely consists of hand-written digits to share the same (ir-)relevant frequencies as one that consists of real-world objects. Moreover, the dimension (image resolution) of inputs for a given dataset matters, as a higher dimension potentially can allow more irrelevant frequencies. Therefore, we emphasize that the goal of our work is not to identify the exact (ir-)relevant frequencies among datasets. Rather, the analysis is built on the **presence** of irrelevant frequencies in the dataset, especially towards the higher end of the frequency spectrum, and how models differ in their robustness when trained by different algorithms. In the next section, we investigate the reason for such a robustness difference by studying how the irrelevant frequencies affect the learning dynamics of GD and signGD under a synthetic linear regression task.

4.4 Linear Regression Analysis with an Over-parameterized Model

In this section, we study the learning dynamics of GD and signGD on least squares regression with linear models to understand why models trained using the two algorithms have the same standard generalization performance but exhibit different robustness against perturbations. On a synthetic dataset that emulates the energy distribution of natural datasets in the frequency domain, we design a learning task that has multiple optima for the standard population risk, each with a different adversarial risk. We analyze the weight adaptation under GD and signGD in both spatial and frequency domains and show that training with signGD can result in larger weights associated with irrelevant frequencies, resulting in models with a higher adversarial risk. Our result verifies Claim 4.3.1. We report the main results here and defer the full derivations to Appendix 4.D.

4.4.1 Problem Setup

Consider a linear model $f(x, w) = \langle\, w\,,\, x\,\rangle$ with $x, w \in \mathbb{R}^d$, where w and x are the weight and the signal represented in the spatial domain, respectively. Since the DCT transformation matrix C is an orthogonal matrix, with rows and columns forming unit vectors (and thus orthonormal), an alternative way to represent this model is:

$$f(x, w) = \langle\, w\,,\, x\,\rangle = w^\top x = w^\top C^\top C x = \langle\, \tilde{w}\,,\, \tilde{x}\,\rangle = f(\tilde{x}, \tilde{w}),$$

where $\tilde{w}$ and $\tilde{x}$ are the exact same weight and signal but are now represented in the frequency domain. This means that for linear models, computing the output of the model can be carried out in either domain as long as we use the matching representation of the signal and the weight. The goal of the linear analysis is to study the learning dynamics of different algorithms in a synthetic and controlled environment where we can clearly define the frequency-domain signal-target (ir)relevance to help understand the behavior of models in more complex settings. For this reason, let $\tilde{w}^*$ denote the frequency-domain representation of the true model that is used to interact with the input $\tilde{x}$ and generate the target: $y = \tilde{x}^\top \tilde{w}^*$, where $\tilde{w}^* = (\tilde{w}_0^*, \tilde{w}_2^*, \ldots, \tilde{w}_{d-1}^*)^\top$. We consider the squared error pointwise loss, which can be equally formulated in both domains:

$$\ell(x, y; w) = \frac{1}{2}\,|f(x, w) - y|^2 = \frac{1}{2}\,|\langle\, x\,,\, w\,\rangle - \langle\, x\,,\, w^*\,\rangle|^2 \quad \text{and} \quad \ell(\tilde{x}, y; \tilde{w}) = \frac{1}{2}\,|f(\tilde{x}, \tilde{w}) - y|^2 = \frac{1}{2}\,|\langle\, \tilde{x}\,,\, \tilde{w}\,\rangle - \langle\, \tilde{x}\,,\, \tilde{w}^*\,\rangle|^2 .$$

Denote the error between the learned weight and the true weight at iteration t by $e(t) = w(t) - w^*$, and the standard risk by $\mathcal{R}_{\mathrm{s}}(w) = \mathbb{E}\left[\ell(X, Y; w)\right]$. In a similar way, those terms can be represented in the frequency domain as $\tilde{e}(t) = \tilde{w}(t) - \tilde{w}^*$ and $\mathcal{R}_{\mathrm{s}}(\tilde{w}) = \mathbb{E}\left[\ell(\tilde{X}, Y; \tilde{w})\right]$. Now we are ready to explain the design philosophy behind the synthetic dataset, the structure of the true model $\tilde{w}^*$, and particularly, with regard to robustness, the ideal model that minimizes the effect of perturbations.

Suppose that $\tilde{X}$ follows a Gaussian distribution $\mathcal{N}(\tilde{\mu}, \tilde{\Sigma})$. For analytical tractability, we consider $\tilde{\mu} = 0$ and a diagonal structure of $\tilde{\Sigma}$, i.e., $\tilde{\Sigma} = \mathrm{diag}(\tilde{\sigma}_0^2, ..., \tilde{\sigma}_{d-1}^2)$. This implies that in the spatial domain, X follows a Gaussian distribution $\mathcal{N}(0, \Sigma)$ where $\Sigma = C^\top \tilde{\Sigma} C$. In Appendix 4.D.1, we provide examples of the spatial-domain structure of the data, where we define the distribution

directly in the frequency domain. In Section 4.3, we demonstrate that natural datasets exhibit a particular energy profile where signals contain irrelevant information represented by high-frequency and low-amplitude waves. To emulate this setting with a synthetic dataset, we define frequencies that are (ir)relevant in generating the target. Let $\mathbb{I}_{\text{irrel}} \subseteq \{1, 2, ..., d-1\}$ and $\mathbb{I}_{\text{rel}} = \{0, 1, 2, ..., d-1\} - \mathbb{I}_{\text{irrel}}$ denote the set of irrelevant and relevant frequencies, respectively. Recall that the goal is to make high-frequency components of the dataset irrelevant, so we exclude the DC term ($0 \notin \mathbb{I}_{\text{irrel}}$) when considering irrelevant frequencies, as it is the lowest frequency possible. Next, we specify the energy distribution of the synthetic dataset. The expected energy of a random signal following such a distribution is

$$\mathbb{E}\left[E(\tilde{X})\right] = \mathbb{E}\left[\sum_{i=0}^{d-1} |\tilde{X}_i|^2\right] = \sum_{i=0}^{d-1} \mathbb{E}\left[\tilde{X}_i^2\right] = \sum_{i=0}^{d-1} \tilde{\sigma}_i^2,$$

where $\tilde{X}$ is defined earlier in Section 4.2.2. We assume that $\tilde{\sigma}_i^2 = 0$ if $i \in \mathbb{I}_{\text{irrel}}$, meaning the irrelevant frequencies of the data from the synthetic dataset have zero energy contributions. The purpose of this is to imitate the behavior of real-world datasets, where the high-frequency components have a negligible impact on the overall energy of the signal.

To see how having irrelevant frequencies affect the structure of the true model, notice that the definition of the synthetic dataset implies $\tilde{X}_i = 0$ for all $i \in \mathbb{I}_{\text{irrel}}$. This means that the true target value does not depend on those irrelevant frequencies. Clearly, this linear model is over-parameterized because one only needs to specify $\tilde{w}_i^*$ for all $i \in \mathbb{I}_{\text{rel}}$ to establish the signal-target relationship.

The objective of the standard risk with such a synthetic dataset is not strictly convex, i.e., there are multiple minimizers with zero standard risk, as the value of $\tilde{w}_i^*$ for all $i \in \mathbb{I}_{\text{irrel}}$ has no impact on the model output. For clarity, let us define $\tilde{\mathcal{W}}^* = \{\, \tilde{w}^* : \mathcal{R}_s(\tilde{w}^*) = 0 \,\}$ as the set that includes all standard risk minimizers. Having multiple standard risk minimizers is the result of over-parametrization; however, there is a unique solution that achieves zero standard risk and makes the model immune to any perturbations parallel to the directions of the irrelevant frequencies, and it corresponds to having zero weight at irrelevant frequencies: $\tilde{w}_i^* = 0$ for all $i \in \mathbb{I}_{\text{irrel}}$. Define such a robust standard risk minimizer as $\tilde{w}^{\text{R}} \in \tilde{\mathcal{W}}^*$, we have

$$\tilde{w}_i^{\text{R}} \triangleq \begin{cases} \tilde{w}_i^* & \text{for all} \quad i \in \mathbb{I}_{\text{rel}} \\ 0 & \text{otherwise.} \end{cases} \tag{4.6}$$

Note that we use $\tilde{w}^*$ to denote any arbitrary standard minimizers in $\tilde{\mathcal{W}}^*$. To see why $\tilde{w}^{\text{R}}$ is the most robust standard minimizer, we introduce the adversarial risk to capture the worst-case performance of the model under an ℓ_2-constrained perturbation. Similar to the squared error loss, the adversarial risk can also be equally formulated in both domains:

$$\mathcal{R}_{\text{a}}(w) \triangleq \mathbb{E}_{(X,Y)}\left[\max_{||\Delta x||_2 \leq \epsilon} \ell(X + \Delta x, Y; w)\right] \quad \text{and} \quad \mathcal{R}_{\text{a}}(\tilde{w}) \triangleq \mathbb{E}_{(\tilde{X},Y)}\left[\max_{||\Delta \tilde{x}||_2 \leq \epsilon} \ell(\tilde{X} + \Delta \tilde{x}, Y; \tilde{w})\right],$$

where the ℓ_2-constraint with a size of ϵ has an equivalent effect in both domains, as guaranteed by Parseval's theorem. To understand the adversarial risk from a frequency-domain perspective, let us focus on $\mathcal{R}_{\text{a}}(\tilde{w})$:

$$\mathcal{R}_{\text{a}}(\tilde{w}) = \mathbb{E}_{\tilde{X}}\left[\max_{||\Delta \tilde{x}||_2 \leq \epsilon} \frac{1}{2} \left| \left\langle\, \tilde{X}\,,\, \tilde{w} - \tilde{w}^* \,\right\rangle + \langle\, \Delta \tilde{x}\,,\, \tilde{w} \,\rangle \right|^2\right], \tag{4.7}$$

where we focus on the expectation over $\tilde{X}$, as Y is replaced with $\left\langle \tilde{X}\,,\,\tilde{w}^*\right\rangle$. Notice that the maximization is inside the expectation. This means that we are finding a separate perturbation for each input. Therefore, the maximizer, $\Delta\tilde{x}^*$, of a given $\tilde{X}$ within the expectation in (4.7) is

$$\Delta\tilde{x}^* \triangleq \underset{||\Delta\tilde{x}||_2\leq\epsilon}{\arg\max}\frac{1}{2}\left|\left\langle \tilde{X}\,,\,\tilde{w}-\tilde{w}^*\right\rangle + \langle\,\Delta\tilde{x}\,,\,\tilde{w}\,\rangle\right|^2 = \epsilon\,\mathrm{sign}\left[\left\langle \tilde{X}\,,\,\tilde{w}-\tilde{w}^*\right\rangle\right]\frac{\tilde{w}}{||\tilde{w}||_2}. \tag{4.8}$$

Now knowing the worst-case perturbation to any $\tilde{X}$, we can continue the derivation in (4.7) with

$$\begin{aligned}\mathcal{R}_{\mathrm{a}}(\tilde{w}) &= \frac{1}{2}\mathbb{E}_{\tilde{X}}\left[\left|\left\langle \tilde{X}\,,\,\tilde{w}-\tilde{w}^*\right\rangle + \epsilon\,\mathrm{sign}[\left\langle \tilde{X}\,,\,\tilde{w}-\tilde{w}^*\right\rangle]\,\|\tilde{w}\|_2\right|^2\right]\\ &= \frac{1}{2}\sum_{i\in\mathbb{I}_{\mathrm{rel}}}\tilde{\sigma}_i^2(\tilde{w}_i-\tilde{w}_i^*)^2 + \epsilon\sqrt{\frac{2}{\pi}\sum_{i\in\mathbb{I}_{\mathrm{rel}}}\tilde{\sigma}_i^2(\tilde{w}_i-\tilde{w}_i^*)^2}\,\|\tilde{w}\|_2 + \frac{\epsilon^2}{2}\,\|\tilde{w}\|_2^2\,.\end{aligned} \tag{4.9}$$

Finding the exact minimizer to (4.9) is not only more involved but also beyond the scope of this chapter. The primary objective here is to understand the robustness properties of solutions obtained by minimizing the standard risk. For an arbitrary standard risk minimizer $\tilde{w}^*$ from $\tilde{\mathcal{W}}^*$, we can evaluate (4.9) at $\tilde{w}^*$ and obtain its the adversarial risk as

$$\mathcal{R}_{\mathrm{a}}(\tilde{w}^*) = \frac{\epsilon^2}{2}||\tilde{w}^*||_2^2, \tag{4.10}$$

where the first two terms in (4.9) become zero at any fixed standard risk minimizer. This shows that the robustness of the standard risk minimizers against ℓ_2-bounded perturbations is inversely proportional to the norm of the linear model. That is, a smaller norm implies better robustness. Recall that when evaluating the standard risk of the model, the weights associated with irrelevant frequencies do not matter, since they are never used in computing the output of the model. On the contrary, the $||\tilde{w}^*||_2^2$ term in (4.10) implies that those weights matter when considering the robustness of the model under perturbations. It is not difficult to see that the minimum adversarial risk can be achieved on a unique standard risk minimizer $\tilde{w}^{\mathrm{R}}$ (4.6).

Therefore, in the over-parameterized linear regression setting, a standard risk minimizer with a minimum norm is preferred when considering the robustness of the model, and a model with zero weight at irrelevant frequencies is the most robust solution among the standard risk minimizers. With this example, we verify Claim 4.3.1. While standard risk minimizers can correctly use the relevant information of the data, their use of irrelevant information is under-constrained. This can result in significant weight assigned to irrelevant frequencies, making models more susceptible to perturbations.

Next, we study the learning dynamics of GD and signGD and demonstrate that the solutions found by GD and signGD differ in the weight of the irrelevant frequencies. This causes the solutions found by the two algorithms to have a similar standard population risk, but behave very differently under perturbations.

4.4.2 Analysis on the Learning Dynamics of GD and signGD

We now analyze the weight adaptation of a linear model under GD and signGD, and experimentally verify our results. Our analysis shows that for the over-parameterized linear model, GD finds

solutions with a standard risk of exactly zero, and signGD finds solutions with a standard risk close to zero. However, they have different robustness properties. With irrelevant frequencies present in the data distribution, GD is more likely to converge to a solution that is less sensitive to perturbations along the direction of irrelevant frequencies, whereas signGD is more likely to converge to solutions that are more prone to such perturbations.

GD Dynamics

Let us start with GD in the spatial domain. Suppose that we initialize the weights in the spatial domain as $w(0) = W \sim N(0, \Sigma_W)$, where $\Sigma_W \in \mathbb{R}^{d\times d}$. Similar to how both $\tilde{X}$ and X follow a Gaussian distribution, the frequency representation of the initialized weight also follows a Gaussian distribution: $\tilde{w}(0) = \tilde{W} \sim \mathcal{N}(0, \tilde{\Sigma}_W)$ where $\tilde{\Sigma}_W = C\Sigma_W C^\top$. To train the model, we use GD on the population risk:

$$w(t+1) \leftarrow w(t) - \eta \nabla_w \mathcal{R}_{\mathrm{s}}(w(t)).$$

The gradient computed using the population risk is $\nabla_w \mathcal{R}_{\mathrm{s}}(w(t)) = \mathbb{E}\left[XX^\top\right] e(t) = \Sigma e(t)$, where $e(t)$ denotes the error between the current weight and the optimal weight. Using this, the learning dynamics of GD in the spatial domain is given by:

$$\begin{aligned} e(t+1) &= w(t+1) - w^* \\ &= w(t) - \eta \nabla_w \mathcal{R}_{\mathrm{s}}(w(t)) - w^* \\ &= w(t) - w^* - \eta \Sigma e(t) \\ &= e(t) - \eta \Sigma e(t) \\ &= (I - \eta\Sigma) e(t) \\ &= (I - \eta\Sigma)^{t+1} e(0). \end{aligned} \tag{4.11}$$

This shows that the learned weight converges to the optimal weight w^* at a rate depending on Σ. To see the GD dynamics in the frequency domain, we can simply perform DCT on both sides of (4.11):

$$\begin{aligned} \tilde{e}(t+1) &= C(I - \eta\Sigma)^{t+1} e(0) \\ &= C(I - \eta\Sigma)^{t+1} C^\top \tilde{e}(0) \\ &= C(I - \eta\Sigma)^{t} C^\top C (I - \eta\Sigma) C^\top \tilde{e}(0) \\ &= C(I - \eta\Sigma)^{t-1} C^\top C (I - \eta\Sigma) C^\top C (I - \eta\Sigma) C^\top \tilde{e}(0) \\ &= \left[C(I - \eta\Sigma)C^\top\right]^{t+1} \tilde{e}(0) \\ &= (I - \eta C\Sigma C^\top)^{t+1} \tilde{e}(0) \\ &= (I - \eta\tilde{\Sigma})^{t+1} \tilde{e}(0), \end{aligned} \tag{4.12}$$

where $\tilde{\Sigma}$ is the covariance of $\tilde{x}$. It is easy to see that no weight adaptation happens for the irrelevant frequencies because $\tilde{\sigma}_i^2 = 0$ for all $i \in \mathbb{I}_{\mathrm{irrel}}$. As $\tilde{\Sigma}$ is diagonal, choosing the learning rate η such that

$\left|1-\eta \max_{i\in\{0,\ldots,d-1\}} \tilde{\Sigma}_{ii}\right| < 1$, we get that the asymptotic solution is

$$\tilde{\boldsymbol{w}}_i^{\mathrm{GD}} \triangleq \lim_{t\to\infty} \tilde{w}_i(t) = \begin{cases} \tilde{w}_i^* & \text{for all} \quad i \in \mathbb{I}_{\mathrm{rel}} \\ \tilde{w}_i(0) & \text{otherwise.} \end{cases}$$

That is, the initial random weights at the irrelevant frequencies do not change. Using (4.10), we have

$$\mathcal{R}_{\mathrm{a}}(\tilde{\boldsymbol{w}}^{\mathrm{GD}}) = \frac{\epsilon^2}{2}||\tilde{\boldsymbol{w}}^{\mathrm{GD}}||_2^2 = \frac{\epsilon^2}{2}\left\{\sum_{i\in\mathbb{I}_{\mathrm{rel}}} \tilde{w}_i^{*2} + \sum_{j\in\mathbb{I}_{\mathrm{irrel}}} \tilde{w}_j(0)^2\right\}. \tag{4.13}$$

Comparing the standard risk minimizer found by GD with the robust standard risk minimizer in (4.6), we notice that the GD solution is not the most robust among all standard risk minimizers, as it is sensitive to perturbations along irrelevant frequencies. Suppose that the initialized weight in the frequency domain is randomly sampled from $\mathcal{N}(0, \sigma^2 I_{d\times d})$, and the signal-target relationship is determined by a handful of relevant frequencies. Taking the expectation of (4.13) over the randomly initialized weight, we have $\mathbb{E}_{\tilde{w}(0)}\left[\mathcal{R}_{\mathrm{a}}(\tilde{\boldsymbol{w}}^{\mathrm{GD}})\right] \approx O(\epsilon^2 d\sigma^2)$, so the adversarial risk can be quite significant when the difference between relevant and irrelevant frequencies is large, i.e., $|\mathbb{I}_{\mathrm{rel}}| \ll |\mathbb{I}_{\mathrm{irrel}}| \approx d$. A similar phenomenon, where adversarial risk increases with large d, was observed in the motivating example of SOAR introduced in Section 3.2.

This example shows that the GD solution is sensitive to initialization. Because there is no mechanism in place to actively ensure that the weights associated with these irrelevant frequencies become zero, GD is not forcing the initial weights to go to zero at those frequencies. One solution is to include the weight norm as a penalty term along with the original optimization objective, but this can result in learning a biased solution. Another simple fix is to initialize the weight at exactly 0. This robustifies the GD solution by initializing those irrelevant weights at the most robust state.

SignGD Dynamics

Adaptive gradient algorithms like Adam and RMSProp utilize historical gradient information as a momentum mechanism for updating model parameters, thereby expediting the learning process. However, it is important to note that their acceleration is not solely attributable to this feature, nor is their adaptiveness limited to it. In (4.3), we have demonstrated how signGD, a memory-free version of Adam and RMSProp, can adaptively adjust the update using a coordinate-wise learning rate. Although signGD is not a suitable choice for training DNNs (Riedmiller et al., 1993; Ma et al., 2022b), examining its behavior can provide insights into the learning dynamics of other adaptive gradient methods (Karimi et al., 2016; Balles et al., 2018; Moulay et al., 2019). Furthermore, later in this section, we empirically demonstrate that signGD serves as a suitable alternative for understanding the learning dynamics of Adam and RMSProp.

Again, let us start with signGD in the spatial domain. The update rule using the population risk takes the sign of the gradient computed using the population risk

$$w(t+1) \leftarrow w(t) - \eta \operatorname{sign}[\nabla_w \mathcal{R}_{\mathrm{s}}(w)],$$

and its learning dynamics in the spatial domain is given by

$$\begin{aligned} e(t+1) &= w(t+1) - w^* \\ &= w(t) - \eta \operatorname{sign}[\nabla_w \mathcal{R}_{\mathrm{s}}(w)] - w^* \\ &= e(t) - \eta \operatorname{sign}[\Sigma e(t)]. \end{aligned} \tag{4.14}$$

Unlike the GD dynamics in (4.11), (4.14) shows that the behavior of signGD depends on the sign of $\Sigma e(t)$, and this means that when $|[\Sigma e(t)]_i| \ll 1$, training using signGD can accelerate the learning along the i-th dimension. Although we can obtain Σ from $\Sigma = C^\top \tilde{\Sigma} C$, the structure of Σ is subject to variation based on $\tilde{\Sigma}$, so it is difficult to find an analytical solution for the dynamics of the model trained under signGD, such as (4.11) where we have a closed form for $e(t)$ as a function of $e(0)$ for models trained under GD. This means that analyzing the signGD dynamics is limited to studying its step-by-step update based on the sign of the entries in $\Sigma e(t)$.

The signGD learning dynamics in the frequency domain can be obtained by taking the DCT transformation on both sides of (4.14):

$$\begin{aligned} \tilde{e}(t+1) &= \tilde{e}(t) - \eta C \operatorname{sign}[\Sigma e(t)] \\ &= \tilde{e}(t) - \eta C \operatorname{sign}[C^\top \tilde{\Sigma} C C^\top \tilde{e}(t)] \\ &= \tilde{e}(t) - \eta C \operatorname{sign}[C^\top \tilde{\Sigma} \tilde{e}(t)]. \end{aligned} \tag{4.15}$$

where the error and the covariance terms inside of the sign are also transformed into their frequency-domain representations. Equation 4.15 shows that analyzing the behavior of signGD in the frequency domain requires knowing the sign of the entries in $C^\top \tilde{\Sigma} \tilde{e}(t)$. This term can be understood as an inverse DCT transformation of $\tilde{\Sigma}\tilde{e}(t)$, and with a diagonal structure of $\tilde{\Sigma}$, we know that $\tilde{\Sigma}\tilde{e}(t) = \left[\tilde{\sigma}_0^2, ..., \tilde{\sigma}_{d-1}^2\right]^\top \odot \tilde{e}(t)$. However, similar to the situation in (4.14), the sign of the entries in $C^\top \tilde{\Sigma} \tilde{e}(t)$ is dependent on $\tilde{e}(t)$ at different steps, so obtaining an analytical solution for the frequency-domain dynamics is also challenging.

In both (4.14) and (4.15), we see that understanding the signGD dynamics for any general $\tilde{\Sigma}$ can be complicated. Thus, we focus on a structure of $\tilde{\Sigma}$ that simplifies the analysis but still allows us to understand why training with signGD results in vulnerable models. In particular, we consider a data distribution where $\tilde{X} \sim \mathcal{N}(\tilde{\mu} = 0, \tilde{\Sigma} = \operatorname{diag}\left\{\tilde{\sigma}_0^2, \tilde{\sigma}_1^2, 0\right\})$. This definition implies that the data distribution contains irrelevant information at the highest frequency basis and we have $\tilde{X}_2 = 0$ for all data points.

Now, we continue with signGD learning dynamics in the frequency domain from (4.15). Let us denote $A(t) = \frac{\sqrt{3}}{3}\tilde{\sigma}_0^2 \tilde{e}_0(t)$ and $B(t) = \frac{\sqrt{2}}{2}\tilde{\sigma}_1^2 \tilde{e}_1(t)$, and $C = C^{(3)}$ follows the DCT transformation matrix defined in (4.4). With some algebraic manipulation, we have

$$\tilde{e}(t+1) = \tilde{e}(t) - \eta \begin{bmatrix} \frac{\sqrt{3}}{3}(\operatorname{sign}[A(t) + B(t)] + \operatorname{sign}[A(t)] + \operatorname{sign}[A(t) - B(t)]) \\ \frac{\sqrt{2}}{2}(\operatorname{sign}[A(t) + B(t)] - \operatorname{sign}[A(t) - B(t)]) \\ \frac{\sqrt{6}}{6}\operatorname{sign}[A(t) + B(t)] - \frac{\sqrt{6}}{3}\operatorname{sign}[A(t)] + \frac{\sqrt{6}}{6}\operatorname{sign}[A(t) - B(t)] \end{bmatrix}, \tag{4.16}$$

and we include its complete derivation in Appendix 4.D. With this particular choice of $\tilde{\Sigma}$, (4.16) shows that weight adaptation depends on the sign of three terms: $A(t)$, $A(t) + B(t)$ and $A(t) - B(t)$.

In Table 4.9 of Appendix 4.D.2, we study the learning dynamics of signGD by analyzing all 27 sign combinations and their corresponding updates. We report the main results here and defer the detailed analysis to Appendix 4.D.2.

With a constant learning rate of η, the asymptotic signGD solution converges to an $O(\eta)$ neighborhood of the standard risk minimizer:

$$\limsup_{t\to\infty} |\tilde{w}_i(t) - \tilde{w}_i^*| = O(\eta),$$

where $i \in \{0, 1\}$. In particular, we demonstrate that $\tilde{w}_0$ oscillates in an $O(\eta)$ neighborhood of $\tilde{w}_0^*$. Consider T as the first iteration after which $\tilde{w}_0$ starts oscillating, and define $\Delta\tilde{w}_2$ as the sum of all the updates in $\tilde{w}_2$ up to the T-th iteration. The limiting behavior of $\tilde{w}_2$ under signGD update is

$$\limsup_{t\to\infty} |\tilde{w}_2(t)| = |\tilde{w}_2(T) + O(\eta)| = |\tilde{w}_2(0) + \Delta\tilde{w}_2 + O(\eta)|\,,$$

where $\tilde{w}_2(0)$ is the weight at initialization. This means that after T iterations, for all $t' > T$, $\tilde{w}_2(t')$ stays in an $O(\eta)$ neighborhood of $\tilde{w}_2(T)$. As such, we have the asymptotic solution found by signGD:

$$\tilde{\boldsymbol{w}}^{\text{signGD}} = [\tilde{w}_0^*,\ \tilde{w}_1^*,\ \tilde{w}_2(0) + \Delta\tilde{w}_2]^\top + O(\eta).$$

From the perspective of training under the standard risk, the signGD solution is close to the optimum. Specifically, its standard risk is

$$\mathcal{R}_{\text{s}}(\tilde{\boldsymbol{w}}^{\text{signGD}}) = \mathbb{E}\left[\ell(\tilde{X}, Y; \tilde{\boldsymbol{w}}^{\text{signGD}})\right] = \frac{1}{2}\mathbb{E}\left[\left\langle \tilde{X}\,,\, \tilde{\boldsymbol{w}}^{\text{signGD}} - \tilde{w}^* \right\rangle^2\right] = O((\tilde{\sigma}_0^2 + \tilde{\sigma}_1^2)\eta^2).$$

Note that the standard risk of the GD solution is exactly zero; and by choosing a sufficiently small learning rate η, the standard risk of the signGD solution can also be close to zero as well. However, their adversarial risks are very different. Specifically, the adversarial risk of the asymptotic signGD solution is

$$\mathcal{R}_{\text{a}}(\tilde{\boldsymbol{w}}^{\text{signGD}}) = \frac{\epsilon^2}{2}||\tilde{\boldsymbol{w}}^{\text{signGD}}||_2^2 = \frac{\epsilon^2}{2}\left\{\tilde{w}_0^{*2} + \tilde{w}_1^{*2} + (\tilde{w}_2(0) + \Delta\tilde{w}_2)^2\right\}. \tag{4.17}$$

We can compare it with the adversarial risk of the asymptotic solution found by GD under the same setup:

$$\mathcal{R}_{\text{a}}(\tilde{\boldsymbol{w}}^{\text{GD}}) = \frac{\epsilon^2}{2}\left\{\tilde{w}_0^{*2} + \tilde{w}_1^{*2} + \tilde{w}_2(0)^2\right\}. \tag{4.18}$$

It can be observed that the main difference between the two adversarial risks in (4.17) and (4.18) arises from the difference in weights learned at the irrelevant frequency. Since their use of irrelevant frequency in the data is under-constrained, neither algorithm can reduce $\tilde{w}_2$ to zero, thereby neither solution is the most robust standard risk minimizer. As discussed in Section 4.4.2, the GD solution is sensitive to weight initialization. Before understanding the $\Delta\tilde{w}_2$ term in the signGD solution, we first introduce two assumptions on the synthetic dataset that serve to better emulate the distribution found in the natural dataset. Consider a dataset with a strong task-relevant correlation between the relevant frequency component of the data and the target, a realistic scenario as we discussed in Section 4.3.2. In this case, $|\tilde{w}_0^*|$ and $|\tilde{w}_1^*|$ can be large. Additionally, with a weight initialization

around zero, such as in methods by He et al., 2015 and Glorot et al., 2010, the initial error $|\tilde{e}_0(0)|$ and $|\tilde{e}_1(0)|$ can be large and close to $|\tilde{w}_0^*|$ and $|\tilde{w}_1^*|$ when $|\tilde{w}_0^*| \gg |\tilde{w}_0(0)|$ and $|\tilde{w}_1^*| \gg |\tilde{w}_1(0)|$. Moreover, we have observed in Figure 4.2 that the distribution of spectral energy heavily concentrates at the low end of the frequency spectrum and decays quickly towards higher frequencies. Since $\tilde{\sigma}_i^2$ is interpreted as the expected energy of a random variable at the i-th frequency, it is reasonable to assume that $\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2} < \frac{1}{3}$.

With the two assumptions, we demonstrate that $\Delta\tilde{w}_2$ is proportional to $|\tilde{w}_0^*|$ or $|\tilde{w}_1^*|$ depending on the initialization of $|A(0)|$ and $|B(0)|$. In particular, we have

$$|\Delta\tilde{w}_2| \approx \begin{cases} \sqrt{3}C\,|\tilde{w}_0^*| & \text{if} \quad |A(0)| < |B(0)| \\ \frac{3\sqrt{2}\tilde{\sigma}_1^2}{2\tilde{\sigma}_0^2} C\,|\tilde{w}_1^*| & \text{if} \quad |A(0)| > |B(0)|\,, \end{cases}$$

where $C \in [\frac{\sqrt{6}}{6}, \frac{\sqrt{6}}{3}]$. To quantitatively understand the robustness difference between solutions found by the two algorithms, we consider the ratio between the adversarial risk of the standard risk minimizers found by GD (4.18) and signGD (4.17) with a three-dimensional input space. We observe that the solution found by signGD is more sensitive to perturbations compared to the GD solution:

$$\frac{\mathcal{R}_{\mathrm{a}}(\tilde{\boldsymbol{w}}^{\mathrm{signGD}})}{\mathcal{R}_{\mathrm{a}}(\tilde{\boldsymbol{w}}^{\mathrm{GD}})} \approx \begin{cases} 1 + C_3 \frac{\tilde{w}_0^{*2}}{\tilde{w}_0^{*2} + \tilde{w}_1^{*2}} & \text{if} \quad |A(0)| < |B(0)| \\ 1 + C_4 \frac{\tilde{w}_1^{*2}}{\tilde{w}_0^{*2} + \tilde{w}_1^{*2}} & \text{if} \quad |A(0)| > |B(0)|\,, \end{cases} \tag{4.19}$$

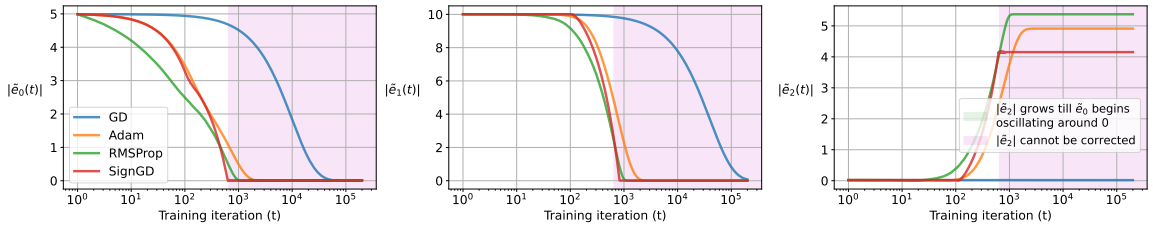
where $\frac{1}{2} \le C_3 \le 2$ and $\frac{3}{4}\frac{\tilde{\sigma}_1^4}{\tilde{\sigma}_0^4} \le C_4 \le 3\frac{\tilde{\sigma}_1^4}{\tilde{\sigma}_0^4}$. Given that this ratio is always greater than 1, the linear model obtained through GD is always more robust against ℓ_2-bounded perturbations in comparison to the model obtained from signGD.

Empirical Validation

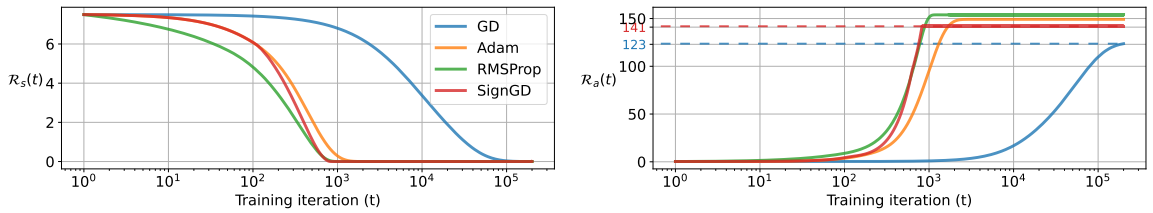
To validate our analysis, in Figure 4.7 we create a three-dimensional dataset using $(\tilde{\sigma}_0^2, \tilde{\sigma}_1^2, \tilde{\sigma}_2^2) = (0.01, 0.0025, 0)$, and $(\tilde{w}_0^*, \tilde{w}_1^*, \tilde{w}_0^*) = (5, 10, 0)$, and compare the dynamics of the frequency-domain weight error on models trained by GD, Adam, RMSProp, and signGD. All models are initialized with the same weight and are trained using a fixed learning rate of 0.01. At each training iteration, we sample 1000 data points and compute the gradient based on the sampled data. We want to clarify that even though we analyze the weight update dynamics in both frequency and spatial domains, the actual training still takes place in the spatial domain.

In (4.12), we show that the GD solution $\tilde{\boldsymbol{w}}_i^{\mathrm{GD}}(t)$ converges to $\tilde{w}_i^*$ with a rate of $1-\eta\tilde{\sigma}_i^2$. Therefore, when $\tilde{\sigma}_i^2$ is small, learning can be particularly slow for weights associated with the i-th frequency, as shown in Figure 4.7a. On the other hand, notice in Table 4.9 in Appendix 4.D.2 that $|\tilde{e}_0|$ gets reduced by at least $\frac{\sqrt{3}}{3}$ regardless of the magnitude of $\tilde{\sigma}_0^2$ for signGD. This means that the magnitude of $\tilde{\sigma}_i^2$ does not directly affect the convergence speed. Instead, the relative magnitude between $A(t)$ and $B(t)$ determines the frequency which receives priority during the learning process. As a result, we observe an acceleration for models trained by signGD.

Next, we observe that the error trajectory for the model trained by signGD closely resembles the one from the model trained by Adam for $\tilde{e}_0$ and $\tilde{e}_1$. In the analysis of signGD, we show that $|\tilde{e}_2|$ increases till $|\tilde{e}_0|$ starts oscillating in $O(\eta)$. Figure 4.7a shows that this pattern can be observed in



a. Dynamics of the error term



b. The standard population risk $\mathcal{R}_s$ and the adversarial population risk $\mathcal{R}_a$

Figure 4.7: **Empirical validations on (a) the learning dynamics, (b) the standard and adversarial population risk of linear models optimized under GD, Adam, RMSProp, and signGD.** We create a three-dimensional dataset created using $(\tilde{\sigma}_0^2, \tilde{\sigma}_1^2, \tilde{\sigma}_2^2) = (0.01, 0.0025, 0)$, and $(\tilde{w}_0^*, \tilde{w}_1^*, \tilde{w}_2^*) = (5, 10, 0)$. All models are initialized with the same weight $(\tilde{w}_0(0), \tilde{w}_1(0), \tilde{w}_2(0)) = (0.01, -0.01, 0.02)$ and trained using a fixed learning rate of 0.01. Note that the iterations are plotted on a logarithmic scale, which makes the initial changes appear smoother than they actually are. **(a) Dynamics of the error term.** During the signGD training process, the error along the irrelevant frequency grows until $\tilde{e}_0$ starts to oscillate around 0. In our example, the green highlighted areas in the figure correspond to the iterations before $\tilde{e}_0$ starts to oscillate, and the red areas show that the error along the irrelevant frequency cannot be corrected. **(b) The standard population risk and the adversarial population risk ($\epsilon = \sqrt{2}$).** We notice that despite all models reaching zero standard population risk, their adversarial population risks are different. The adversarial population risk of models trained by adaptive gradient methods is higher than the one from the model trained by GD, indicating lower robustness.

models trained by Adam as well. This shows that signGD is a suitable alternative to understanding the learning dynamics of models under the proposed linear regression task. For models trained by GD, since there is no update on the weight associated with the irrelevant frequency, $\tilde{e}_2$ remains at the initialized value throughout training. To demonstrate the weight adaptation under signGD, we divide the training into two phases, as highlighted by two background colors. The green area indicates that $|\tilde{e}_0|$ decreases and $|\tilde{e}_2|$ increases in the meanwhile. Once oscillation begins for $|\tilde{e}_0|$, $|\tilde{e}_2|$ can no longer be corrected. This behavior corresponds to the purple area in Figure 4.7a.

In Figure 4.7b, we compare the standard population risk and the adversarial population risk of different models. We notice that despite all models reaching near zero standard population risk, their adversarial population risk is different. In particular, the adversarial population risk of models trained by adaptive gradient methods is higher than the one from the model trained by GD, indicating lower robustness. Choosing $\epsilon = \sqrt{2}$ in (4.10), the adversarial risk of those standard risk minimizers is exactly the squared ℓ_2 norm of the weight. With our choice of initialization, the resulting $|A(0)|$ and $|B(0)|$ are 0.0289 and 0.0177 respectively. This means that the ratio between the two adversarial risks is $\frac{\mathcal{R}_{\mathrm{a}}(\tilde{\boldsymbol{w}}^{\mathrm{signGD}})}{\mathcal{R}_{\mathrm{a}}(\tilde{\boldsymbol{w}}^{\mathrm{GD}})} \in [1.04, 1.15]$ according to (4.19), and this aligns with the ratio of 1.146 obtained empirically from the experiments.

This simple problem illustrates how the optimization algorithms and an over-parameterized model might interact, and learning with signGD can lead to a solution that is more prone to perturbations. In this section, we focus on analyzing the robustness of the solution from a frequency domain perspective, that is, the behavior of $\tilde{w}$ with an input perturbation of $\Delta\tilde{x}$. In Appendix 4.D.3, we present a spatial interpretation of the result and demonstrate how signals with irrelevant frequencies contain spatially redundant dimensions.

4.5 Connecting the Norm of Linear Models to the Lipschitzness of Neural Networks

The takeaway from the over-parameterized linear regression analysis is that among all standard risk minimizers, the minimum norm solution is the most robust one. That is, a smaller weight norm implies better robustness. This suggests a connection between the weight norm and model robustness. Nonetheless, the major limitation of the analysis is that it is designed for a linear model. In this section, we generalize such a connection to the deep learning setting and verify it using the robustness of neural networks trained by different algorithms.

A significant challenge is that the concept of weight norm, as defined for linear models, does not directly apply to neural networks due to their hierarchical structure and non-linear activation functions. Nonetheless, we can relate a network's weights to its sensitivity to input perturbations, forming a basis for analyzing robustness in neural networks.

Consider the ℓ_p vector norm, for all $x_1, x_2 \in \mathbb{R}$, a function f is said to be Lipschitz continuous if $||f(x_1) - f(x_2)||_p \leq L||x_1 - x_2||_p$, for some real-valued Lipschitz constant $L \geq 0$.[3] The Lipschitz constant effectively quantifies the sensitivity of the function to changes in the input space.

Now, consider a single-layer ReLU-activated feedforward network with $x \in \mathbb{R}^d$ and $W \in \mathbb{R}^{D\times d}$. For an input perturbation of $\Delta x \in \mathbb{R}^d$ constrained by the vector ℓ_p-norm, the maximum change in the model output as measured by the same norm can be bounded using

$$\|\mathrm{ReLU}\left(W(x+\Delta x)\right) - \mathrm{ReLU}\left(Wx\right)\|_p \leq \|W\Delta x\|_p \leq \|W\|_p \|\Delta x\|_p ,$$

where $\|W\|_p$ denotes the vector ℓ_p-norm induced matrix norm of the weight W, which serves as the Lipschitz constant for this single-layer model. This bound demonstrates how weight norms influence sensitivity to input perturbations, linking them to robustness.

In the single-layer model example, its Lipschitz constant is exactly the matrix norm of the weight. More generally, consider the feedforward neural network as a series of function compositions:

$$f(x) = (\phi_l \circ \phi_{l-1} \circ \ldots \circ \phi_1)(x),$$

where each ϕ_i is a linear operation, an activation function, or a pooling operation. A particularly useful property of the Lipschitz function is that the composition of Lipschitz functions with Lipschitz constant L_1, L_2, ..., L_N w.r.t. the same norm is also Lipschitz with an upper-bound on the Lipschitz constant $L \leq L_1 L_2 ... L_N$. Denoting the Lipschitz constant of function f as $L(f)$, we can establish

[3] Any value of L satisfying the Lipschitz condition is considered a valid Lipschitz constant. For the sake of clarity, we will refer to the smallest (optimal) Lipschitz constant as L.

Table 4.1: **Comparing the upper bound on the Lipschitz constant and the averaged robust accuracy of neural networks trained by SGD, Adam, and RMSProp.** We follow (Gouk et al., 2021) to compute the Lipschitz constants of each layer in isolation and multiply them together to establish an upper bound on the constant of the entire network. Notice that across all selected datasets, models trained by SGD have a considerably smaller upper bound compared to models trained by Adam and RMSProp. In Figure 4.1, we demonstrate the robustness of the neural networks under Gaussian noise, ℓ_2 and ℓ_∞ bounded adversarial perturbations. Here, we average the accuracy across the perturbations and get a single score quantifying the model's robustness. All results are averaged over three independently initialized and trained models.

	Dataset	MNIST	Fashion	CIFAR-10	CIFAR-100	SVHN	Caltech-101	Imagenette
$\prod_{i=1}^{l} L(\phi_i)$	SGD	**3.80**	**3.83**	**26.81**	**40.41**	**22.65**	**18.53**	**23.99**
	Adam	5.75	8.12	28.70	41.87	30.45	26.20	28.55
	RMSProp	6.21	5.11	37.75	41.71	28.31	45.84	27.11
Average Robust Acc.	SGD	**77.97%**	**77.95%**	**63.21%**	**55.65%**	**69.08%**	**71.42%**	**67.59%**
	Adam	65.64%	67.60%	57.71%	45.25%	65.60%	55.03%	58.86%
	RMSProp	63.54%	71.34%	56.47%	47.55%	65.37%	53.16%	57.98%

an upper bound on the Lipschitz constant for the entire feedforward neural network using

$$L(f) \leq \prod_{i=1}^{l} L(\phi_i). \tag{4.20}$$

For a multi-layer neural network that comprises repeated layers of linear operation followed by non-linear activation, this result allows us to upper bound the change in model output w.r.t. changes in the input space by multiplying the operator norms of the weights. However, it is important to note that (4.20) provides a loose upper bound, and computing the exact Lipschitz constant for a neural network is NP-hard (Virmaux et al., 2018). Nonetheless, this approximation enables us to draw meaningful connections between the weights and the robustness of neural networks.

Results in Section 4.4 indicate that linear models trained by signGD have larger weight norms, indicating reduced robustness. Extending this to the deep learning setting, we expect neural networks trained with Adam to have larger Lipschitz upper bounds, leading to less robust models, as illustrated in Figure 4.1. To verify this, we follow the techniques in Gouk et al. (2021) and compute an upper bound on the Lipschitz constant of the same neural networks trained by SGD, Adam, and RMSProp in Figure 4.1. The results, presented in Table 4.1, confirm that models trained with SGD consistently exhibit smaller Lipschitz upper bounds across all datasets and architectures compared to those trained with the two adaptive gradient methods.

In Figure 4.1, we demonstrate the robustness of the neural networks under Gaussian noise, ℓ_2 and ℓ_∞ bounded adversarial perturbations (Croce et al., 2020). In Table 4.1, we average the accuracy across the perturbations and get a single score quantifying the model's robustness. The results clearly indicate that models trained with SGD are more robust than those trained with Adam and RMSProp, consistent with their smaller Lipschitz constants and the observations from the linear model analysis.

4.6 Conclusions

In this chapter, we highlighted the robustness difference between models trained by SGD and adaptive gradient methods, particularly Adam and RMSProp. To understand this phenomenon, we leveraged a frequency-domain analysis, and demonstrated that natural datasets contain frequencies that are irrelevant to minimizing the standard training loss. Empirically, through a band-limited perturbation analysis on neural networks trained on common vision datasets, we showed that models trained by the adaptive gradient method utilize the statistics in the irrelevant frequency, and thus they experience a huge drop in performance when the same statistics become corrupted. Analytically, on a synthetic linear regression task where the dataset was designed to contain target-irrelevant frequencies, we showed that while both GD and signGD can find the solution with standard risks close to zero, the adversarial risk of the asymptotic solution found by signGD can be larger than that of GD. Such results from the linear analysis explained the observation in Figure 4.1 and suggested that a smaller model parameters' weight norms may indicate a larger model robustness. Finally, in the deep learning setting, we showed that models trained by SGD have a noticeably smaller Lipschitz constant than those trained by Adam and RMSProp.

4.6.1 Challenges and Limitations

Our work has some limitations. First, when conducting a theoretical analysis of various optimizers, we opted for signGD as a simpler alternative to Adam and RMSProp. Second, our focus was primarily on linear models. However, it is crucial to acknowledge that DNNs inherently possess non-linear characteristics, which limit the depth of insights derived from linear models. Therefore, one promising future direction is to incorporate tools such as neural tangent kernels (Jacot et al., 2018), which provide a deeper understanding of network dynamics. Third, our analysis focuses on optimization algorithms along with the standard objective function. We can also study the effect of optimizer with alternative objectives that are designed to improve the robustness of the model (Simon-Gabriel et al., 2019; Wen et al., 2020; Ma et al., 2020; Foret et al., 2021). For instance, the effect of adversarial training using perturbations similar to the Fast Gradient Sign Method (FGSM) under the linear regression setup has been studied in Chapter 3. In linear classification, Wei et al. (2023) showed that minimizing the sharpness-aware loss (SAM) (Foret et al., 2021) can lead to robust models. Another promising direction for future research is to analyze model robustness by coupling various optimization algorithms with different optimization objectives.

Appendices

In the following appendices, we provide some auxiliary results that are omitted from the main body of the chapter. The source code for this project can be found at `https://github.com/averyma/opt-robust`. In Appendix 4.A, we first describe the data augmentation, the exact optimization schedule, and the model architectures used to train the models. In Appendix 4.B, we describe the complete generalization and robustness results in Table 4.3 and how they are used to generate Figure 4.1. In Appendix 4.C, we discuss how the training inputs are modified when making the observations in Section 4.3. In Appendix 4.D, we provide additional detail on the linear analysis in Section 4.4. Finally, in Appendix 4.E, we provide additional figures including visualization of

the perturbed images, the modified images used in Section 4.3.1, and the frequency sensitivity comparison in Section 4.3.2.

4.A Implementation Details

Data augmentation: In our chapter, we study how the presence of irrelevant information in the dataset affects the robustness of the model when trained by different algorithms. We approach this problem from a frequency-domain perspective. Specifically, we leverage the structure and energy profile of the dataset in the frequency domain. While data augmentation methods are widely used in training machine learning models to improve generalization and reduce overfitting, understanding how those methods affect the datasets in the frequency domain requires additional analysis tailored for each augmentation method. Therefore, on Fashion-MNIST, CIFAR-10, CIFAR-100, Caltech-101, and Imagenette, training inputs are augmented with random horizontal flipping, a method that does not change the frequency profile of the image.

Optimization schedule: For all models, we use the following default PyTorch (v1.12.1) optimization settings. For SGD, we disable all of the following mechanism: dampening, weight decay, and Nesterov. For Adam, we use the default values of $\beta_1 = 0.9$ $\beta_2 = 0.999$, $\epsilon = 10^{-8}$ and disable weight decay and disable AMSgrad (Reddi et al., 2018) to eliminate other factors that may affect the robustness of the model. For RMSProp, we use default values of $\alpha = 0.99$, $\epsilon = 10^{-8}$, and disable momentum and disable centered RMSProp which normalizes the gradient by an estimation of its variance. All models are trained for 200 epochs. In Table 4.2, we list the initial learning rate. The learning rate decreases by a factor of 0.1 at epoch 100 and 150.

Model architecture: For MNIST and Fashion-MNIST, we use a ReLU-activated, two-layer convolutional neural network ending with two fully-connected layers. For CIFAR-10, CIFAR-100, SVHN, Caltech-101, and Imagenette, we use PreActResNet18 (He et al., 2016b) and Vision Transformers (ViT-B/16) (Dosovitskiy et al., 2021). For the Speech Commands dataset (Warden, 2018), we use the M5 network architecture defined by Dai et al. (2017). See Table 4.2 for details of all architectures used in this chapter. We denote Conv(i, o, k) as a convolution layer having i input channels, o output channels with k by k filters, FN(i, o) as a fully-connect layer with i input channels and o output channels, and SM(o) as the soft-max layer with o output. The stride for all convolution layers is 1. The main experiments in our work are centered around models based on convolutional neural networks, within the computer vision domain. Additional results from using ViT-B/16 and on the Speech Commands dataset can be found in Table 4.5 and Table 4.6, respectively.

Batch normalization: We concentrate on a particular aspect of the training process: the selection of optimizers. Our aim is to shed light on how this critical component influences the robustness of trained models. It has been recently shown that the use of batch normalization (BN) can also affect the robustness of the model (Benz et al., 2021b; Benz et al., 2021a; Wang et al., 2022a). Consequently, to maintain focus on the impact of optimizers, we have omitted BN in the training phase for the experiments leading to the results in Figure 4.1 and the analysis in Section 4.3. However, to show that our conclusions remain valid for models with BN, we have included additional results that incorporate BN in Table 4.4.

Table 4.2: **Experiment setup:** the initial learning rate and the definition of neural networks in this work.

Dataset	Optimization	Initial Learning Rate
MNIST	SGD	0.1
	Adam	0.0005
	RMSProp	0.0005
Fashion-MNIST	SGD	0.1
	Adam	0.0005
	RMSProp	0.0005
CIFAR-10	SGD	0.2
	Adam	0.0002
	RMSProp	0.0005
CIFAR-100	SGD	0.3
	Adam	0.0005
	RMSProp	0.0005
SVHN	SGD	0.2
	Adam	0.0002
	RMSProp	0.0002
Caltech-101	SGD	0.05
	Adam	0.0002
	RMSProp	0.001
Imagenette	SGD	0.1
	Adam	0.0002
	RMSProp	0.0002
Speech Commmands	SGD	0.1
	Adam	0.1
	RMSProp	0.1

Dataset	Structure
MNIST Fashion-MNIST	Conv(1, 16, 4) - ReLU - Conv(16, 32, 4) - ReLU - FN(21632, 100) - FN(100, 10) - SM(10)
CIFAR-10 CIFAR-100 SVHN Caltech-101 Imagenette	PreActResNet18 ViT-B/16
Speech Commands	M5 (Dai et al., 2017)

4.B Generalization and Robustness Results

Main results: Table 4.3 summarizes the result on the standard generalization ability and robustness properties of the models trained by SGD, Adam, and RMSProp on seven vision datasets. All results are averaged over three independently initialized and trained models. To evaluate standard generalization, we measure the classification accuracy of the models on the testing data. To capture model robustness, we measure the classification accuracy of the models on the testing data perturbed using Gaussian perturbations, ℓ_2 and ℓ_∞-bounded perturbations (Croce et al., 2020). Perturbations with varying degrees of severity are included in the evaluation to ensure the observation of the robustness difference is not limited to perturbations with any particular parameters. The degree of severity is determined by the variance of the Gaussian perturbation and an ℓ_2 and ℓ_∞ norm for the attacks. We select those parameters so the range of the accuracy differences between models is similar across different datasets. Particularly, the highlighted results in Table 4.3 are in a similar range, so we use them to plot Figure 4.1.We also ensure that the original image semantics remains in the perturbed images, and we provide a visualization of the perturbed images in Figure 4.18 to 4.20.

Finally, for CIFAR-100 and Caltech-101, because of the large number of classes in the dataset, we use the top-5 classification accuracy to plot Figure 4.1 as the results are within a range similar to other datasets with 10 classes. The observation of the similar standard generalization and different robustness holds on both top-1 and top-5 accuracy.

Table 4.3: Evaluation of classification accuracy for models trained using SGD, Adam, and RMSProp on both the original and perturbed test dataset (%). We evaluate the model robustness on the testing data perturbed using Gaussian perturbations, ℓ_2 and ℓ_∞-bounded perturbations. We include various severity of perturbations to better capture the model robustness. Models trained by SGD are the most robust against the three types of perturbations across all datasets. The highlighted results are used in Figure 4.1, as they are in relatively similar ranges. Results are averaged over three independently initialized and trained models.

Dataset	Optimization	Test	Gaussian perturbations			ℓ_2-bounded attack			ℓ_∞-bounded attack		
			$\sigma^2 = 0.01$	$\sigma^2 = 0.05$	$\sigma^2 = 0.1$	$\epsilon = 0.5$	$\epsilon = 0.7$	$\epsilon = 1.0$	$\epsilon = 0.05$	$\epsilon = 0.07$	$\epsilon = 0.1$
MNIST	SGD	98.72	98.59	97.94	95.64	93.00	87.33	66.33	87.53	71.93	31.50
	Adam	99.05	98.86	96.76	89.33	92.33	86.00	54.67	85.40	52.93	8.77
	RMSProp	98.90	98.70	97.02	90.63	91.67	83.00	50.00	82.73	50.00	7.33
			$\sigma^2 = 0.001$	$\sigma^2 = 0.005$	$\sigma^2 = 0.01$	$\epsilon = 0.1$	$\epsilon = 0.5$	$\epsilon = 0.7$	$\epsilon = 0.01$	$\epsilon = 0.03$	$\epsilon = 0.05$
Fashion-MNIST	SGD	91.20	90.49	87.78	84.30	82.33	24.33	12.33	67.23	22.30	4.17
	Adam	90.98	89.91	85.03	78.91	70.33	6.00	0.33	53.57	5.33	0.00
	RMSProp	91.15	90.09	85.23	78.69	72.67	15.93	4.67	62.67	16.33	1.67
			$\sigma^2 = 0.001$	$\sigma^2 = 0.005$	$\sigma^2 = 0.007$	$\epsilon = 0.1$	$\epsilon = 0.2$	$\epsilon = 0.3$	$\epsilon = \frac{1}{255}$	$\epsilon = \frac{2}{255}$	$\epsilon = \frac{4}{255}$
CIFAR-10	SGD	90.16	87.35	74.13	68.66	67.40	37.57	16.30	53.93	21.27	0.93
	Adam	90.73	86.93	67.50	58.54	64.03	29.60	11.10	50.57	13.93	0.20
	RMSProp	90.46	86.25	70.03	61.52	60.10	25.47	8.87	47.87	14.03	0.33
			$\sigma^2 = 0.001$	$\sigma^2 = 0.005$	$\sigma^2 = 0.007$	$\epsilon = 0.1$	$\epsilon = 0.2$	$\epsilon = 0.3$	$\epsilon = \frac{1}{255}$	$\epsilon = \frac{2}{255}$	$\epsilon = \frac{4}{255}$
CIFAR-100(top1)	SGD	59.76	56.88	46.26	41.28	28.47	12.93	5.90	18.80	5.57	1.17
	Adam	61.10	55.30	31.54	24.92	24.30	7.13	1.87	14.43	2.47	0.33
	RMSProp	60.36	56.46	36.42	29.50	28.90	10.47	2.83	17.90	3.47	0.20
CIFAR-100(top5)	SGD	84.67	81.77	72.84	67.53	80.30	70.20	59.90	75.30	61.70	39.53
	Adam	85.41	81.37	58.11	49.54	80.53	66.73	52.77	74.70	53.70	33.43
	RMSProp	85.18	81.66	62.71	54.44	80.93	68.57	54.10	74.60	59.10	34.10
			$\sigma^2 = 0.001$	$\sigma^2 = 0.003$	$\sigma^2 = 0.005$	$\epsilon = 0.1$	$\epsilon = 0.2$	$\epsilon = 0.3$	$\epsilon = \frac{1}{255}$	$\epsilon = \frac{2}{255}$	$\epsilon = \frac{4}{255}$
SVHN	SGD	96.11	95.68	94.47	93.81	85.53	63.60	39.63	80.67	49.83	17.03
	Adam	96.48	96.03	94.04	91.46	80.77	57.83	35.93	78.93	47.50	12.43
	RMSProp	96.42	95.91	94.07	91.87	81.13	57.90	34.10	76.93	46.33	11.30
			$\sigma^2 = 0.01$	$\sigma^2 = 0.05$	$\sigma^2 = 0.1$	$\epsilon = 0.5$	$\epsilon = 1.0$	$\epsilon = 1.5$	$\epsilon = \frac{1}{255}$	$\epsilon = \frac{2}{255}$	$\epsilon = \frac{4}{255}$
Caltech-101(top1)	SGD	70.80	68.13	57.67	43.46	58.77	47.03	35.17	48.27	27.87	4.70
	Adam	72.32	58.34	19.88	8.55	57.70	44.47	29.60	45.47	22.03	2.63
	RMSProp	73.82	69.38	51.34	33.17	37.80	11.30	2.80	14.77	1.93	0.03
Caltech-101(top5)	SGD	85.96	84.84	77.57	67.16	85.30	84.43	79.00	83.97	75.47	52.27
	Adam	88.08	79.48	38.55	19.67	82.03	80.67	77.63	83.97	72.07	45.87
	RMSProp	88.37	85.90	72.19	52.74	80.20	63.40	45.93	63.97	41.33	23.90
			$\sigma^2 = 0.01$	$\sigma^2 = 0.05$	$\sigma^2 = 0.1$	$\epsilon = 0.5$	$\epsilon = 1.0$	$\epsilon = 1.5$	$\epsilon = \frac{1}{255}$	$\epsilon = \frac{2}{255}$	$\epsilon = \frac{4}{255}$
Imagenette	SGD	89.44	84.83	67.23	50.21	70.70	44.13	21.33	47.23	14.33	0.37
	Adam	89.75	71.84	29.05	17.72	65.30	31.53	11.83	39.43	6.23	0.07
	RMSProp	89.77	73.25	28.28	17.16	62.30	30.27	11.27	38.40	6.20	0.03

Results on Models with Batch Normalization Enabled

When BN layers are activated in PreActResNet18, we observe that the models exhibit similar standard generalization performance, yet the robustness difference between SGD and adaptive gradient methods remains evident. This observation is in line with the results presented in Table 4.4, where BN layers are disabled. Notably, the accuracy of models with BN enabled is significantly lower compared to their BN-disabled counterparts under almost all types of perturbations, particularly under stronger perturbations. This finding aligns with the results from the previous work (Benz et al., 2021b; Benz et al., 2021a; Wang et al., 2022a).

Results on Vision Transformers

In addition to the network designs considered in Table 4.2, we extend our work to ViT in order to verify whether similar observations can be drawn on other neural network architectures.

It is important to note that the dataset utilized in this chapter is significantly smaller in size compared to larger datasets such as ImageNet and JFT-300M. Recent research, such as the work by Zhu

Table 4.4: **Results on standard generalization and robustness of models trained with BN enabled (%).** We follow the exact optimization configuration as the ones used in generating Table 4.3. The only modification is that **BN is enabled**.

Dataset	Optimization	Test	Gaussian perturbations			ℓ_2-bounded attack			ℓ_∞-bounded attack		
			$\sigma^2 = 0.001$	$\sigma^2 = 0.005$	$\sigma^2 = 0.007$	$\epsilon = 0.1$	$\epsilon = 0.2$	$\epsilon = 0.3$	$\epsilon = \frac{1}{255}$	$\epsilon = \frac{2}{255}$	$\epsilon = \frac{4}{255}$
CIFAR-10	SGD	92.24	86.91	55.23	43.29	64.84	27.86	6.96	48.64	9.14	0.04
	Adam	93.38	85.67	50.41	39.11	56.5	15.96	2.63	37.7	3.8	0
	RMSProp	93.57	86.97	52.09	39.86	55.93	15.16	2.23	37.7	3.76	0
			$\sigma^2 = 0.001$	$\sigma^2 = 0.005$	$\sigma^2 = 0.007$	$\epsilon = 0.1$	$\epsilon = 0.2$	$\epsilon = 0.3$	$\epsilon = \frac{1}{255}$	$\epsilon = \frac{2}{255}$	$\epsilon = \frac{4}{255}$
CIFAR-100	SGD	72.24	57.99	26.08	19.23	34.36	10.2	3.63	21.3	4.8	0.2
	Adam	71.36	55.85	24.55	18.23	27.56	6.66	1.73	15.26	2.9	0.23
	RMSProp	70.99	56.13	24.56	18.03	23.8	5.1	1.1	13.7	2	0.1
			$\sigma^2 = 0.001$	$\sigma^2 = 0.003$	$\sigma^2 = 0.005$	$\epsilon = 0.1$	$\epsilon = 0.2$	$\epsilon = 0.3$	$\epsilon = \frac{1}{255}$	$\epsilon = \frac{2}{255}$	$\epsilon = \frac{4}{255}$
SVHN	SGD	94.16	95.81	94.96	92.90	81.73	60.56	41.96	76.13	49.26	15.3
	Adam	96.62	96.09	93.96	91.19	80.13	47.76	21.86	72.36	32.76	4.36
	RMSProp	96.44	94.86	93.75	91.02	79.86	48.43	22.16	72.36	33.7	3.96
			$\sigma^2 = 0.01$	$\sigma^2 = 0.05$	$\sigma^2 = 0.1$	$\epsilon = 0.5$	$\epsilon = 1.0$	$\epsilon = 1.5$	$\epsilon = \frac{1}{255}$	$\epsilon = \frac{2}{255}$	$\epsilon = \frac{4}{255}$
Caltech-101	SGD	78.61	72.69	45.38	25.87	61.23	44.03	23.03	46.80	13.13	0.83
	Adam	79.58	62.21	21.08	10.35	56.76	34.46	13.06	37.3	5.66	0.23
	RMSProp	75.56	69.89	45.38	23.13	58.6	44.6	20.5	42.6	11.3	0.6
			$\sigma^2 = 0.01$	$\sigma^2 = 0.05$	$\sigma^2 = 0.1$	$\epsilon = 0.5$	$\epsilon = 1.0$	$\epsilon = 1.5$	$\epsilon = \frac{1}{255}$	$\epsilon = \frac{2}{255}$	$\epsilon = \frac{4}{255}$
Imagenette	SGD	89.35	76.67	46.18	28.27	73.43	42.96	17.56	48.93	14.33	0.03
	Adam	91.88	67.29	24.30	14.92	67.66	24.06	3.06	31.4	6.23	0
	RMSProp	91.93	67.02	23.79	15.43	68.76	26.1	4.1	33.66	6.20	0

et al. (2023), has shown that ViT tends to generalize poorly on small datasets when trained from scratch. In particular, Zhu et al. (2023) empirically demonstrated that ViT and ResNet learn distinct representations on small datasets while converging to similar representations on larger datasets.

Therefore, we perform fine-tuning on a pretrained ViT-B/16. Among the datasets we considered, Imagenette is a 10-class subset of the ImageNet-1k dataset, making it especially suitable for the fine-tuning task, since the publicly available ViT checkpoint was pretrained on ImageNet-1k. Also, it is important to note that the pretrained models were originally trained using Adam. In our fine-tuning process, we treat ViT as a feature extractor (i.e., no weight update on the transformer encoder), with a focus on fine-tuning the Multi-Layer Perceptrons (MLP) head. Our approach follows prior work (Steiner et al., 2022) and incorporates the three different optimizers, each fine-tuned for 10 epochs. We initiated the fine-tuning process with an initial learning rate of 0.01, followed by a cosine decay learning rate schedule and a linear warmup. Throughout this process, we maintained a fixed batch size of 512.

To evaluate the robustness of the fine-tuned models, we maintained the exact same perturbation strengths, including the variance of Gaussian noise and ϵ for adversarial perturbations, as used in Table 4.3. The results can be found in Table 4.5. We draw three observations.

First, all models fine-tuned with the three different optimizers achieve near 100% test accuracy, a substantial improvement from the 89% accuracy when training from scratch using PreActResNet18. This significant boost in standard generalization highlights the effectiveness of fine-tuning with ViT. Second, we observe that the fine-tuned models exhibit a notable increase in robustness to Gaussian noise. However, they are highly vulnerable to adversarial perturbations. This observation is consistent with the results from existing literature (Zhang et al., 2019), where a trade-off is

Table 4.5: **Results on standard generalization and robustness of ViT-B/16 fined-tuned on the Imagenette dataset (%).**

Model	Optimization	Test	Gaussian perturbations			ℓ_2-bounded attack			ℓ_∞-bounded attack		
			$\sigma^2 = 0.01$	$\sigma^2 = 0.05$	$\sigma^2 = 0.1$	$\epsilon = 0.5$	$\epsilon = 1.0$	$\epsilon = 1.5$	$\epsilon = \frac{1}{255}$	$\epsilon = \frac{2}{255}$	$\epsilon = \frac{4}{255}$
ViT-b/16	SGD	99.18	99.05	96.16	91.25	6.2	0	0	1.3	0	0
	Adam	99.93	99.51	94.43	88.59	5.1	0	0	0.7	0	0
	RMSProp	99.92	99.54	95.38	89.91	5	0	0	0.8	0	0

Table 4.6: **Results on standard generalization and robustness of models on an audio classification task on the Speech Commands dataset (%).**

Dataset	Optimization	Test	Gaussian perturbations			ℓ_2-bounded attack			ℓ_∞-bounded attack		
			$\sigma^2 = 0.001$	$\sigma^2 = 0.003$	$\sigma^2 = 0.005$	$\epsilon = 0.01$	$\epsilon = 0.05$	$\epsilon = 0.1$	$\epsilon = 0.0001$	$\epsilon = 0.0005$	$\epsilon = 0.001$
Speech Commands	SGD	85.14	55.76	39.88	33.42	70.01	20.31	9.81	75.6	36.71	13.47
	Adam	85.47	54.73	38.87	31.95	60.74	17.87	8.49	71.97	29.2	10.15
	RMSProp	84.67	52.37	36.59	27.94	59.57	19.04	8.88	70.50	30.95	11.01

often present between standard accuracy and adversarial robustness. Finally, we make a similar observation on the robustness difference between models fine-tuned with the three optimizers, where models fine-tuned with SGD exhibited greater robustness to both Gaussian noise and adversarial perturbations when compared to models fine-tuned using Adam and RMSProp.

Results with an Audio Dataset

Besides the vision domain, we extend our work to the audio domain since audio signals offer a frequency-based interpretation as well. We include additional results in Table 4.6, which compare the standard generalization and robustness properties of an audio classifier trained on the Speech Commands dataset (Warden, 2018). We focus on the PreActResNet18 architectures and all models are trained for 200 epochs, with an initial learning rate of 0.1 and learning rate decay by a factor of 0.1 at epoch 100 and 150. We consider the accuracy of models under Gaussian- and adversarially-perturbed test sets. Manual verification was conducted to ensure that the noisy audio phrase could still be recognizable.

Results demonstrate that despite similar test accuracy, the models trained using SGD exhibit greater robustness when compared to the other two optimization methods. These insights provide valuable context to the generalizability of our initial observations, offering a more comprehensive understanding of how optimizers perform in the context of different data modalities.

Results with Momentum-enabled SGD

Additional results with momentum-enabled SGD (SGD-m) are included in Table 4.7. We maintain the exact same optimization configuration as that is used for generating the SGD results presented in Table 4.3, and the only variation is an additional momentum term with a coefficient of $\beta = 0.9$. The result shows that models optimized by both vanilla SGD and SGD-m exhibit similar trends in terms of generalization and robustness.

Table 4.7: **Results on standard generalization and robustness of models trained by SGD without and with momentum (%).**

Dataset	Optimization	Test	Gaussian perturbations			ℓ_2-bounded attack			ℓ_∞-bounded attack		
			$\sigma^2 = 0.001$	$\sigma^2 = 0.005$	$\sigma^2 = 0.007$	$\epsilon = 0.1$	$\epsilon = 0.2$	$\epsilon = 0.3$	$\epsilon = \frac{1}{255}$	$\epsilon = \frac{2}{255}$	$\epsilon = \frac{4}{255}$
CIFAR-10	SGD	90.16	87.35	74.13	68.66	67.40	37.57	16.30	53.93	21.27	0.93
	SGD-m	89.79	87.28	73.207	66.783	67.1	38.067	15.9	55.667	20.233	0.6333
			$\sigma^2 = 0.001$	$\sigma^2 = 0.005$	$\sigma^2 = 0.007$	$\epsilon = 0.1$	$\epsilon = 0.2$	$\epsilon = 0.3$	$\epsilon = \frac{1}{255}$	$\epsilon = \frac{2}{255}$	$\epsilon = \frac{4}{255}$
CIFAR-100	SGD	59.76	56.88	46.26	41.28	28.47	12.93	5.90	18.80	5.57	1.17
	SGD-m	56.08	55.03	44.97	40.03	29.2	12.4	5.7	19.7	6.5	0.8
			$\sigma^2 = 0.001$	$\sigma^2 = 0.003$	$\sigma^2 = 0.005$	$\epsilon = 0.1$	$\epsilon = 0.2$	$\epsilon = 0.3$	$\epsilon = \frac{1}{255}$	$\epsilon = \frac{2}{255}$	$\epsilon = \frac{4}{255}$
SVHN	SGD	96.11	95.68	94.47	93.81	85.53	63.60	39.63	80.67	49.83	17.03
	SGD-m	96.14	95.71	94.44	92.80	82.7	60.53	39.03	77.33	49.26	15.96
			$\sigma^2 = 0.01$	$\sigma^2 = 0.05$	$\sigma^2 = 0.1$	$\epsilon = 0.5$	$\epsilon = 1.0$	$\epsilon = 1.5$	$\epsilon = \frac{1}{255}$	$\epsilon = \frac{2}{255}$	$\epsilon = \frac{4}{255}$
Caltech-101	SGD	70.80	68.13	57.67	43.46	58.77	47.03	35.17	48.27	27.87	4.70
	SGD-m	69.89	67.77	55.77	41.34	54.86	43.36	31.06	44.16	23.9	3.133
			$\sigma^2 = 0.01$	$\sigma^2 = 0.05$	$\sigma^2 = 0.1$	$\epsilon = 0.5$	$\epsilon = 1.0$	$\epsilon = 1.5$	$\epsilon = \frac{1}{255}$	$\epsilon = \frac{2}{255}$	$\epsilon = \frac{4}{255}$
Imagenette	SGD	89.44	84.83	67.23	50.21	70.70	44.13	21.33	47.23	14.33	0.37
	SGD-m	88.69	85.05	68.58	51.15	75.53	49.2	24.53	56.43	17.76	0.2

4.C Filtering Irrelevant Frequencies

We demonstrate irrelevant frequencies in two settings: i) DCT basis vectors with a low magnitude are irrelevant and ii) high-frequency DCT bases are irrelevant. In Appendix 4.E, we visualize the original images and the modified images used in Section 4.3.1.

To understand how the modified training images are generated, we use $\Phi_{\text{nrg}}(x, p)$ with $0 < p < 100$ to denote the operation that modifies the input image x by removing the DCT basis vectors whose magnitudes are in the bottom $\frac{p}{100}$-th percentile. We use $M_{\text{nrg}}(x, p)$ to denote the binary mask used in the process. Consider an image $x \in \mathbb{R}^{d \times d}$, the entire process can be formulated as

$$\Phi_{\text{nrg}}(x, p) = C(\tilde{x} \odot M_{\text{nrg}}(x, p))C^\top,$$

where $\odot$ is the element-wise product and C is the DCT transformation matrix. The binary mask $M_{\text{nrg}} \in \{0, 1\}^{d \times d}$ is defined as

$$M_{\text{nrg}}(x, p) = \begin{cases} 1 & \text{if} \quad |\tilde{x}_{i,j}| > \phi(\tilde{x}, p) \\ 0 & \text{otherwise,} \end{cases}$$

where $\phi(\tilde{x}, p) \in \mathbb{R}$ computes the $\frac{p}{100}$-th percentile in $|\tilde{x}|$. Therefore, DCT basis vectors with a magnitude smaller than the threshold are first discarded in $\tilde{x}$, and then this filtered $\tilde{x}$ is converted back to the spatial domain.

Similarly, we use $\Phi_{\text{freq}}(x, p)$ to denote the operation that modifies the input image x by removing the DCT basis vectors whose frequency are in the highest $\frac{p}{100}$-th percentile. We use $M_{\text{freq}}(p)$ to denote the binary mask used in the process. This operation can be formulated as

$$\Phi_{\text{freq}}(x, p) = C(\tilde{x} \odot M_{\text{freq}}(p))C^\top,$$

Table 4.8: **Examples of the synthetic data distribution in the frequency and the spatial domain.**

$\tilde{\Sigma}$	Frequency-domain Representation	Spatial-domain Representation
$\text{diag}\{\tilde{\sigma}_0^2, 0, 0\}$	$(\tilde{X}_0, 0, 0)$	$(\sqrt{\frac{1}{3}}\tilde{X}_0, \sqrt{\frac{1}{3}}\tilde{X}_0, \sqrt{\frac{1}{3}}\tilde{X}_0)$
$\text{diag}\{0, \tilde{\sigma}_1^2, 0\}$	$(0, \tilde{X}_1, 0)$	$(\sqrt{\frac{1}{2}}\tilde{X}_1, 0, -\sqrt{\frac{1}{2}}\tilde{X}_1)$
$\text{diag}\{0, 0, \tilde{\sigma}_2^2\}$	$(0, 0, \tilde{X}_2)$	$(\sqrt{\frac{1}{6}}\tilde{X}_2, -\sqrt{\frac{2}{3}}\tilde{X}_2, +\sqrt{\frac{1}{6}}\tilde{X}_2)$
$\text{diag}\{0, \tilde{\sigma}_1^2, \tilde{\sigma}_2^2\}$	$(0, \tilde{X}_1, \tilde{X}_2)$	$(\sqrt{\frac{1}{2}}\tilde{X}_1 + \sqrt{\frac{1}{6}}\tilde{X}_2, -\sqrt{\frac{2}{3}}\tilde{X}_2, -\sqrt{\frac{1}{2}}\tilde{X}_1 + \sqrt{\frac{1}{6}}\tilde{X}_2)$
$\text{diag}\{\tilde{\sigma}_0^2, 0, \tilde{\sigma}_2^2\}$	$(\tilde{X}_0, 0, \tilde{X}_2)$	$(\sqrt{\frac{1}{3}}\tilde{X}_0 + \sqrt{\frac{1}{6}}\tilde{X}_2, \sqrt{\frac{1}{3}}\tilde{X}_0 - \sqrt{\frac{2}{3}}\tilde{X}_2, \sqrt{\frac{1}{3}}\tilde{X}_0 + \sqrt{\frac{1}{6}}\tilde{X}_2)$
$\text{diag}\{\tilde{\sigma}_0^2, \tilde{\sigma}_0^2, 0\}$	$(\tilde{X}_0, \tilde{X}_1, 0)$	$(\sqrt{\frac{1}{3}}\tilde{X}_0 + \sqrt{\frac{1}{2}}\tilde{X}_1, \sqrt{\frac{1}{3}}\tilde{X}_0, \sqrt{\frac{1}{3}}\tilde{X}_0 - \sqrt{\frac{1}{2}}\tilde{X}_1)$
$\text{diag}\{\tilde{\sigma}_0^2, \tilde{\sigma}_1^2, \tilde{\sigma}_2^2\}$	$(\tilde{X}_0, \tilde{X}_1, \tilde{X}_2)$	$(\sqrt{\frac{1}{3}}\tilde{X}_0 + \sqrt{\frac{1}{2}}\tilde{X}_1 + \sqrt{\frac{1}{6}}\tilde{X}_2, \sqrt{\frac{1}{3}}\tilde{X}_0 - \sqrt{\frac{2}{3}}\tilde{X}_2, \sqrt{\frac{1}{3}}\tilde{X}_0 - \sqrt{\frac{1}{2}}\tilde{X}_1 + \sqrt{\frac{1}{6}}\tilde{X}_2)$

where $M_{\text{freq}} \in \{0,1\}^{d\times d}$ is defined as:

$$M_{\text{freq}}(p) = \begin{cases} 1 & \text{if} \quad i^2 + j^2 > \frac{p}{100}\sqrt{2}d \\ 0 & \text{otherwise,} \end{cases}$$

where i, j are frequency bases. Notice that M_{freq} only depends on the size of the image, whereas M_{nrg} depends on the input x since we identify the threshold value in $|\tilde{x}|$. Examples of the modified images and the modification process are shown in Appendix 4.E.

4.D Linear Regression Analysis

4.D.1 Understanding the Synthetic Dataset

The goal of the linear analysis is to study the learning dynamics of different algorithms on a synthetic dataset where we can clearly define the frequency-domain signal-target (ir)relevance. This motivates us to directly define the distribution of the input signal in the frequency domain. In Section 4.4, we consider $\tilde{X}$ follows a Gaussian distribution $\mathcal{N}(\tilde{\mu}, \tilde{\Sigma})$, and for analytical tractability, we consider $\tilde{\mu} = 0$ and a diagonal structure of $\tilde{\Sigma}$, i.e., $\tilde{\Sigma} = \text{diag}(\tilde{\sigma}_0^2, ..., \tilde{\sigma}_{d-1}^2)$. Admittedly, it is quite unconventional to define the data distribution directly in the frequency domain, so we provide a few examples in Table 4.8 to illustrate the structure of the input data in both representations.

Similar to the analysis in Section 4.4.2, we focus on a low dimensional setting with $d = 3$. The first six rows in Table 4.8 represent the scenario when there are zero variances in $\tilde{\Sigma}$. Notice the notion of irrelevant information in the data is different in the two representations. In the frequency domain, an irrelevant frequency indicates that the data has a value of zero at the particular frequency. In the spatial domain, having irrelevant frequency means that there are redundant dimensions in the spatial representation of the data because the value of data at some dimensions can be fully predictable by knowing the values of data at some other dimensions.

Derivation of Equation 4.7

The adversarial risk under an ℓ_2-norm bounded perturbation with a size of ϵ is

$$\begin{aligned}\mathcal{R}_{\mathrm{a}}(\tilde{w}) &\triangleq \mathbb{E}_{(\tilde{X},Y)}\left[\max_{||\Delta\tilde{x}||_2\leq\epsilon} \ell(\tilde{X}+\Delta\tilde{x},Y;\tilde{w})\right] \\ &= \mathbb{E}_{(\tilde{X},Y)}\left[\max_{||\Delta\tilde{x}||_2\leq\epsilon} \frac{1}{2}\left|f(\tilde{X}+\Delta\tilde{x},\tilde{w})-Y\right|^2\right] \\ &= \mathbb{E}_{\tilde{X}}\left[\max_{||\Delta\tilde{x}||_2\leq\epsilon} \frac{1}{2}\left|\left\langle \tilde{X}+\Delta\tilde{x}\,,\,\tilde{w}\right\rangle - \left\langle \tilde{X}\,,\,\tilde{w}^*\right\rangle\right|^2\right] \\ &= \mathbb{E}_{\tilde{X}}\left[\max_{||\Delta\tilde{x}||_2\leq\epsilon} \frac{1}{2}\left|\left\langle \tilde{X}\,,\,\tilde{w}-\tilde{w}^*\right\rangle + \left\langle\,\Delta\tilde{x}\,,\,\tilde{w}\,\right\rangle\right|^2\right],\end{aligned}$$

where we focus on the expectation over $\tilde{X}$, as Y is replaced with $\left\langle \tilde{X}\,,\,\tilde{w}^*\right\rangle$.

Derivation of Equation 4.8

Given a random variable $\tilde{X}$, we define $\Delta\tilde{x}^*$ to be the maximizer of the term inside the expectation of (4.7):

$$\Delta\tilde{x}^* \triangleq \arg\max_{||\Delta\tilde{x}||_2\leq\epsilon} \frac{1}{2}\left|\left\langle \tilde{X}\,,\,\tilde{w}-\tilde{w}^*\right\rangle + \left\langle\,\Delta\tilde{x}\,,\,\tilde{w}\,\right\rangle\right|^2.$$

To maximize this term, we need the two inner product terms to have the same sign. This means

$$\Delta\tilde{x}^* = \operatorname{sign}\left[\left\langle \tilde{X}\,,\,\tilde{w}-\tilde{w}^*\right\rangle\right] \arg\max_{||\Delta\tilde{x}||_2\leq\epsilon} \left|\left\langle\,\Delta\tilde{x}\,,\,\tilde{w}\,\right\rangle\right|^2.$$

For the remaining argmax term, we can first use the Cauchy-Schwarz inequality to obtain

$$\max_{||\Delta\tilde{x}||_2\leq\epsilon} \left|\left\langle\,\Delta\tilde{x}\,,\,\tilde{w}\,\right\rangle\right|^2 \leq \max_{||\Delta\tilde{x}||_2\leq\epsilon} \|\Delta\tilde{x}\|_2^2\,\|\tilde{w}\|_2^2 = \epsilon^2\,\|\tilde{w}\|_2^2,$$

which leads to

$$\arg\max_{||\Delta\tilde{x}||_2\leq\epsilon} \left|\left\langle\,\Delta\tilde{x}\,,\,\tilde{w}\,\right\rangle\right|^2 = \epsilon\frac{\tilde{w}}{\|\tilde{w}\|_2}.$$

Finally, we have

$$\Delta\tilde{x}^* = \epsilon\operatorname{sign}\left[\left\langle \tilde{X}\,,\,\tilde{w}-\tilde{w}^*\right\rangle\right]\frac{\tilde{w}}{||\tilde{w}||_2}.$$

Derivation of Equation 4.9

The adversarial risk is

$$\begin{aligned}\mathcal{R}_{\mathrm{a}}(\tilde{w}) &= \frac{1}{2}\mathbb{E}_{\tilde{X}}\left[\left|\left\langle \tilde{X}\,,\,\tilde{w}-\tilde{w}^*\right\rangle + \epsilon\operatorname{sign}[\left\langle \tilde{X}\,,\,\tilde{w}-\tilde{w}^*\right\rangle]||\tilde{w}||_2\right|^2\right] \\ &= \frac{1}{2}\mathbb{E}_{\tilde{X}}\left[\left\langle \tilde{X}\,,\,\tilde{w}-\tilde{w}^*\right\rangle^2 + 2\epsilon\left|\left\langle \tilde{X}\,,\,\tilde{w}-\tilde{w}^*\right\rangle\right|||\tilde{w}||_2 + \epsilon^2||\tilde{w}||_2^2\right] \\ &= \frac{1}{2}\sum_{i\in\mathbb{I}_{\mathrm{rel}}}\tilde{\sigma}_i^2(\tilde{w}_i-\tilde{w}_i^*)^2 + \epsilon\mathbb{E}_{\tilde{X}}\left[\left|\left\langle \tilde{X}\,,\,\tilde{w}-\tilde{w}^*\right\rangle\right|\right]||\tilde{w}||_2 + \frac{\epsilon^2}{2}||\tilde{w}||_2^2.\end{aligned}$$

To compute the expectation, we first denote $Z = \sum_{i\in\mathbb{I}_{\mathrm{rel}}}\tilde{X}_i(\tilde{w}_i-\tilde{w}_i^*)$. Because $\tilde{\sigma}_i^2 = 0$ for all

$i \in \mathbb{I}_{\text{irrel}}$, this allows us to ignore those irrelevant frequencies in the summation in Z. This leads us to

$$\mathbb{E}_{\tilde{X}}\left[\left|\left\langle \tilde{X}\,,\, \tilde{w}-\tilde{w}^*\right\rangle\right|\right] = \mathbb{E}_Z\Big[|Z|\Big].$$

Since Z is a linear combination of zero-mean Gaussian random variable's, this makes it also a zero-mean Gaussian r.v, i.e., $\mathbb{E}[Z]=0$. The variance of Z is

$$\begin{aligned}
\sigma_Z^2 &= \mathbb{E}[Z^2] - \mathbb{E}[Z]^2 \\
&= \mathbb{E}\Bigg[\sum_{i\in\mathbb{I}_{\text{rel}}}\sum_{j\in\mathbb{I}_{\text{rel}}, i\neq j}\Big[\tilde{X}_i\tilde{X}_j(\tilde{w}_i-\tilde{w}_i^*)(\tilde{w}_j-\tilde{w}_j^*)\Big] + \sum_{i\in\mathbb{I}_{\text{rel}}}\Big[\tilde{X}_i^2(\tilde{w}_i-\tilde{w}_i^*)^2\Big]\Bigg] \\
&= \mathbb{E}\Bigg[\sum_{i\in\mathbb{I}_{\text{rel}}}\Big[\tilde{X}_i^2(\tilde{w}_i-\tilde{w}_i^*)^2\Big]\Bigg] \\
&= \sum_{i\in\mathbb{I}_{\text{rel}}}\tilde{\sigma}_i^2(\tilde{w}_i-\tilde{w}_i^*)^2,
\end{aligned}$$

where the expectation on the cross-multiplication term is zero because $\tilde{X}_i$ and $\tilde{X}_j$ are independent random variable's. This means $Z\sim\mathcal{N}(0,\sigma_Z^2)$ with $\sigma_Z^2 = \sum_{i\in\mathbb{I}_{\text{rel}}}\tilde{\sigma}_i^2(\tilde{w}_i-\tilde{w}_i^*)^2$. Therefore, $\mathbb{E}_Z\Big[|Z|\Big]$ is the expectation of a folded normal distribution:

$$\mathbb{E}_Z\Big[|Z|\Big] = \sigma_Z\sqrt{\frac{2}{\pi}} = \sqrt{\frac{2}{\pi}\sum_{i\in\mathbb{I}_{\text{rel}}}\tilde{\sigma}_i^2(\tilde{w}_i-\tilde{w}_i^*)^2}.$$

Derivation of the signGD Dynamics for $\tilde{\Sigma} = \text{diag}\left\{\tilde{\sigma}_0^2, \tilde{\sigma}_1^2, 0\right\}$

To understand $\tilde{e}(t+1)$ with our specific choice of $\Sigma = C^\top\tilde{\Sigma}C$ and $\tilde{\Sigma} = \text{diag}\left\{\tilde{\sigma}_0^2,\tilde{\sigma}_1^2,0\right\}$, first notice that the DCT transformation matrix $C = C^{(3)}$ follows the definition in (4.4):

$$C = \begin{bmatrix} \sqrt{\frac{1}{3}} & \sqrt{\frac{1}{3}} & \sqrt{\frac{1}{3}} \\ \sqrt{\frac{2}{3}}\cos\frac{\pi}{6} & \sqrt{\frac{2}{3}}\cos\frac{\pi}{2} & \sqrt{\frac{2}{3}}\cos\frac{5\pi}{6} \\ \sqrt{\frac{2}{3}}\cos\frac{\pi}{3} & \sqrt{\frac{2}{3}}\cos\pi & \sqrt{\frac{2}{3}}\cos\frac{5\pi}{3} \end{bmatrix} = \begin{bmatrix} \sqrt{\frac{1}{3}} & \sqrt{\frac{1}{3}} & \sqrt{\frac{1}{3}} \\ \sqrt{\frac{1}{2}} & 0 & -\sqrt{\frac{1}{2}} \\ \sqrt{\frac{1}{6}} & -\sqrt{\frac{2}{3}} & \sqrt{\frac{1}{6}} \end{bmatrix}.$$

Denote $\sqrt{\frac{1}{3}}\tilde{\sigma}_0^2\tilde{e}_0(t)$ and $\sqrt{\frac{1}{2}}\tilde{\sigma}_1^2\tilde{e}_1(t)$ by using $A(t)$ and $B(t)$, respectively. Putting it all together, we have:

$$
\begin{aligned}
\tilde{e}(t+1) &= \tilde{e}(t) - \eta C\,\mathrm{sign}[C^\top\tilde{\Sigma}Ce(t)] \\
&= \tilde{e}(t) - \eta C\,\mathrm{sign}[C^\top\tilde{\Sigma}\tilde{e}(t)] \\
&= \tilde{e}(t) - \eta \begin{bmatrix} \sqrt{\frac{1}{3}} & \sqrt{\frac{1}{3}} & \sqrt{\frac{1}{3}} \\ \sqrt{\frac{1}{2}} & 0 & -\sqrt{\frac{1}{2}} \\ \sqrt{\frac{1}{6}} & -\sqrt{\frac{2}{3}} & \sqrt{\frac{1}{6}} \end{bmatrix} \mathrm{sign}\left[\begin{bmatrix} \sqrt{\frac{1}{3}} & \sqrt{\frac{1}{3}} & \sqrt{\frac{1}{3}} \\ \sqrt{\frac{1}{2}} & 0 & -\sqrt{\frac{1}{2}} \\ \sqrt{\frac{1}{6}} & -\sqrt{\frac{2}{3}} & \sqrt{\frac{1}{6}} \end{bmatrix}^\top \begin{bmatrix} \tilde{\sigma}_0^2\tilde{e}_0(t) \\ \tilde{\sigma}_1^2\tilde{e}_1(t) \\ 0 \end{bmatrix} \right] \\
&= \tilde{e}(t) - \eta \begin{bmatrix} \sqrt{\frac{1}{3}} & \sqrt{\frac{1}{3}} & \sqrt{\frac{1}{3}} \\ \sqrt{\frac{1}{2}} & 0 & -\sqrt{\frac{1}{2}} \\ \sqrt{\frac{1}{6}} & -\sqrt{\frac{2}{3}} & \sqrt{\frac{1}{6}} \end{bmatrix} \begin{bmatrix} \mathrm{sign}\left[\sqrt{\frac{1}{3}}\tilde{\sigma}_0^2\tilde{e}_0(t) + \sqrt{\frac{1}{2}}\tilde{\sigma}_1^2\tilde{e}_1(t)\right] \\ \mathrm{sign}\left[\sqrt{\frac{1}{3}}\tilde{\sigma}_0^2\tilde{e}_0(t)\right] \\ \mathrm{sign}\left[\sqrt{\frac{1}{3}}\tilde{\sigma}_0^2\tilde{e}_0(t) - \sqrt{\frac{1}{2}}\tilde{\sigma}_1^2\tilde{e}_1(t)\right] \end{bmatrix} \\
&= \tilde{e}(t) - \eta \begin{bmatrix} \sqrt{\frac{1}{3}} & \sqrt{\frac{1}{3}} & \sqrt{\frac{1}{3}} \\ \sqrt{\frac{1}{2}} & 0 & -\sqrt{\frac{1}{2}} \\ \sqrt{\frac{1}{6}} & -\sqrt{\frac{2}{3}} & \sqrt{\frac{1}{6}} \end{bmatrix} \begin{bmatrix} \mathrm{sign}[A(t) + B(t)] \\ \mathrm{sign}[A(t)] \\ \mathrm{sign}[A(t) - B(t)] \end{bmatrix} \\
&= \tilde{e}(t) - \eta \begin{bmatrix} \frac{\sqrt{3}}{3}(\mathrm{sign}[A(t) + B(t)] + \mathrm{sign}[A(t)] + \mathrm{sign}[A(t) - B(t)]) \\ \frac{\sqrt{2}}{2}(\mathrm{sign}[A(t) + B(t)] - \mathrm{sign}[A(t) - B(t)]) \\ \frac{\sqrt{6}}{6}\mathrm{sign}[A(t) + B(t)] - \frac{\sqrt{6}}{3}\mathrm{sign}[A(t)] + \frac{\sqrt{6}}{6}\mathrm{sign}[A(t) - B(t)] \end{bmatrix}.
\end{aligned}
$$

4.D.2 Understanding the Dynamics of signGD with $\tilde{\Sigma} = \mathbf{diag}\{\tilde{\sigma}_0^2, \tilde{\sigma}_1^2, 0\}$

Previous work has shown that for strictly convex problems with a unique minimum, the signGD solution converges to the minimum under a sequence of decaying learning rate: $\lim_{t\to\infty}\eta(t) = 0$ (Moulay et al., 2019). In this section, we follow Section 4.4.2 where the GD dynamics is studied under a constant learning rate and investigate the behavior of signGD under a fixed η. Compared to the asymptotic GD solution that converges exactly to the standard risk minimizer, we demonstrate that the asymptotic signGD solution converges to an $O(\eta)$ neighborhood of the standard risk minimizer.

For the rest of this section, we first perform a partition-based analysis to study the learning dynamics of $\tilde{e}_0$ and $\tilde{e}_1$ in Appendix 4.D.2. Proposition 4.D.4 summarizes how the value of weight changes in different partitions, and the weight adaptation of $\tilde{e}_0$ and $\tilde{e}_1$ is summarized in Corollary 4.D.6. Based on the corollary, we analyze the dynamics of $\tilde{e}_2$ in Appx. 4.D.2. Lastly, we focus on the differences between the adversarial risk of solutions found by GD and signGD in Appx. 4.D.2.

Dynamics of $\tilde{e}_0$ and $\tilde{e}_1$ under signGD

With our choice of $\tilde{\Sigma}$, (4.16) shows that weight adaptation depends on the sign of three terms: $A(t)$, $A(t)+B(t)$ and $A(t)-B(t)$. This allows us to study the learning dynamics of signGD by analyzing the three terms in Table 4.9. There are 27 sign combinations in total; however, not all of them are valid. For example, consider the combination with $\mathrm{sign}[A(t)] = 1$, $\mathrm{sign}[A(t) + B(t)] = -1$. Those

Table 4.9: **The dynamics of the error term in the frequency domain under signGD.** Dynamics of the error term in the frequency domain can be written in a tabular format and the exact update depends on the initialized weight $\tilde{w}(0)$ and the true model $\tilde{w}^*$. We use $A(t)$ and $B(t)$ to denote $\frac{\sqrt{3}}{3}\tilde{\sigma}_0^2\tilde{e}_0(t)$ and $\frac{\sqrt{2}}{2}\tilde{\sigma}_1^2\tilde{e}_1(t)$, respectively. Invalid sign combinations are denoted by using n/a.

No.	sign$[A(t)]$	sign$[A(t)+B(t)]$	sign$[A(t)-B(t)]$	$\tilde{e}(t+1)$	$\lvert A(t)\rvert$ vs. $\lvert B(t)\rvert$
1	1	1	1	$\tilde{e}(t)-\eta\left[\sqrt{3},0,0\right]^\top$	$\lvert A(t)\rvert > \lvert B(t)\rvert$
2	1	1	-1	$\tilde{e}(t)-\eta\left[\frac{\sqrt{3}}{3},\sqrt{2},-\frac{\sqrt{6}}{3}\right]^\top$	$\lvert A(t)\rvert < \lvert B(t)\rvert$
3	1	1	0	$\tilde{e}(t)-\eta\left[\frac{2\sqrt{3}}{3},\frac{\sqrt{2}}{2},-\frac{\sqrt{6}}{6}\right]^\top$	$\lvert A(t)\rvert = \lvert B(t)\rvert$
4	1	-1	1	$\tilde{e}(t)-\eta\left[\frac{\sqrt{3}}{3},-\sqrt{2},-\frac{\sqrt{6}}{3}\right]^\top$	$\lvert A(t)\rvert < \lvert B(t)\rvert$
n/a	1	-1	-1	n/a	n/a
n/a	1	-1	0	n/a	n/a
5	1	0	1	$\tilde{e}(t)-\eta\left[\frac{2\sqrt{3}}{3},-\frac{\sqrt{2}}{2},-\frac{\sqrt{6}}{6}\right]^\top$	$\lvert A(t)\rvert = \lvert B(t)\rvert$
n/a	1	0	-1	n/a	n/a
n/a	1	0	0	n/a	n/a
n/a	-1	1	1	n/a	n/a
6	-1	1	-1	$\tilde{e}(t)-\eta\left[-\frac{\sqrt{3}}{3},\sqrt{2},\frac{\sqrt{6}}{3}\right]^\top$	$\lvert A(t)\rvert < \lvert B(t)\rvert$
n/a	-1	1	0	n/a	n/a
7	-1	-1	1	$\tilde{e}(t)-\eta\left[-\frac{\sqrt{3}}{3},-\sqrt{2},\frac{\sqrt{6}}{3}\right]^\top$	$\lvert A(t)\rvert < \lvert B(t)\rvert$
8	-1	-1	-1	$\tilde{e}(t)-\eta\left[-\sqrt{3},0,0\right]^\top$	$\lvert A(t)\rvert > \lvert B(t)\rvert$
9	-1	-1	0	$\tilde{e}(t)-\eta\left[-\frac{2\sqrt{3}}{3},-\frac{\sqrt{2}}{2},\frac{\sqrt{6}}{6}\right]^\top$	$\lvert A(t)\rvert = \lvert B(t)\rvert$
n/a	-1	0	1	n/a	n/a
10	-1	0	-1	$\tilde{e}(t)-\eta\left[-\frac{2\sqrt{3}}{3},\frac{\sqrt{2}}{2},\frac{\sqrt{6}}{6}\right]^\top$	$\lvert A(t)\rvert = \lvert B(t)\rvert$
n/a	-1	0	0	n/a	n/a
n/a	0	1	1	n/a	n/a
11	0	1	-1	$\tilde{e}(t)-\eta\left[0,\sqrt{2},0\right]^\top$	$\lvert A(t)\rvert < \lvert B(t)\rvert$
n/a	0	1	0	n/a	n/a
12	0	-1	1	$\tilde{e}(t)-\eta\left[0,-\sqrt{2},0\right]^\top$	$\lvert A(t)\rvert < \lvert B(t)\rvert$
n/a	0	-1	-1	n/a	n/a
n/a	0	-1	0	n/a	n/a
n/a	0	0	1	n/a	n/a
n/a	0	0	-1	n/a	n/a
13	0	0	0	Optimal	$\lvert A(t)\rvert = \lvert B(t)\rvert = 0$

two conditions imply that $B(t) < 0$ and $|A(t)| < |B(t)|$, and this means that sign$[A(t) - B(t)]$ must be positive. This makes the entry with sign$[A(t) - B(t)] = -1$ invalid, as shown in the fifth row in Table 4.9. We denote those entries with invalid sign combinations as n/a.

Notice in (4.16) that the weight adaptation under signGD depends on the dynamics of $A(t)$ and $B(t)$, and they are functions of $\tilde{e}_0(t)$ and $\tilde{e}_1(t)$ respectively, so let us first focus on understanding the weight adaptation at the first two frequency bases.

There are 13 possible updates in Table 4.9. Notice that the non-zero updates are always in the direction to reduce $|\tilde{e}_0(t)|$ and $|\tilde{e}_1(t)|$, and the step size depends on the magnitude of $|A(t)|$ and $|B(t)|$.To simplify the analysis, let us focus on the updates on A and B instead. For example, in updates 1 and 8, decreasing $|\tilde{e}_0|$ by $\sqrt{3}\eta$ is equivalent to decreasing $|A|$ by $\tilde{\sigma}_0^2\eta$.

Now take note of the limited number of update magnitudes for $|A|$, specifically $\tilde{\sigma}_0^2\eta$, $\frac{2\tilde{\sigma}_0^2\eta}{3}$, and $\frac{\tilde{\sigma}_0^2\eta}{3}$, which correspond to updating $|\tilde{e}_0|$ by $\sqrt{3}\eta$, $\frac{2\sqrt{3}}{3}\eta$, and $\frac{\sqrt{3}}{3}\eta$, respectively. Similarly, $|B|$ has only two update magnitudes, namely $\tilde{\sigma}_1^2\eta$ and $\frac{\tilde{\sigma}_1^2\eta}{2}$, which correspond to updating $|\tilde{e}_1|$ with $\sqrt{2}\eta$ and $\frac{\sqrt{2}}{2}\eta$,

respectively. This observation leads to the following proposition.

Proposition 4.D.1. *Suppose the initial weight $w(0) \sim \mu$ are sampled from probability density μ, then neither A nor B ($\tilde{e}_0$ nor $\tilde{e}_1$) can be reduced to exactly 0 almost surely.*

Proof. Due to the limited number of update magnitudes, reducing $|A|$ and $|B|$ to 0 requires their initial value to be exactly some integer multiplication of those updates. However, with the initial weight sampled from probability density μ, the probability of the initial values of A and B being the exact integer multiple of the possible update is 0. □

Next, we introduce the following lemma to understand the dynamics of $A(t)$.

Lemma 4.D.2. *Consider the update rule $x(t+1) = x(t) - \text{sign}[x(t)]\Delta(t)$, where $x \in \mathbb{R}$, $\Delta(t) \in \{\Delta_1, \Delta_2, \ldots, \Delta_{\max}\}$ and $0 < \Delta_1 < \Delta_2 < \cdots < \Delta_{\max}$. Then there exists t such that $|x(t)| \leq \Delta_{\max}$. Moreover, whenever $|x(t)| \leq \Delta_{\max}$, the rest of the sequence stays $\Delta_{\max}$-bounded, i.e., $|x(t')| \leq \Delta_{\max}$ for all $t' \geq t$.*

Proof. In the following, we provide a proof for the case when $x(0) > 0$. A proof with $x(0) < 0$ can be done in a similar way. The proof can be divided into two parts.

1. Let us denote the sequence of $\{x(0), x(1), \ldots, x(t)\}$ by $\{x(t)\}$. First, we prove that there exists a t such that $|x(t)| \leq \Delta_{\max}$.

If $x(t) > \Delta_{\max}$, then $x(t+1) = x(t) - \Delta(t) \geq x(t) - \Delta_{\max} > 0$.

Consider $\{x(t)\}$ with $x(t') > \Delta_{\max}$ for all $t' \in \{0, 1, \ldots, t\}$, we have $x(t'+1) = x(t') - \Delta(t') < x(t')$. This means that for any $x(t) > \Delta_{\max}$, the sequence $\{x(t)\}$ is decreasing.

We prove, by contradiction, that there exists a t such that $|x(t)| < \Delta_{\max}$. Suppose that such a t does not exist, then one of the two cases must happen.

1. $x(t) > \Delta_{\max}$ for all t.
2. $\exists k$ such that $x(0) > x(1) > \cdots > x(k) > \Delta_{\max}$, but $x(k+1) < -\Delta_{\max}$.

For case 1, since $x(t) > \Delta_{\max}$, we know that $\{x(t)\}$ is decreasing and bounded from below, so we have $x(t) \to x^* \geq \Delta_{\max}$ as $t \to \infty$. This means that $\lim_{t\to\infty} x(t) = \lim_{t\to\infty} x(t+1) = x^*$.

Using the update rule, we have

$$\begin{aligned}\lim_{t\to\infty} x(t+1) &= \lim_{t\to\infty} x(t) - \Delta(t)\,\text{sign}(x(t)) \\ &= \lim_{t\to\infty} x(t) - \Delta(t) \\ &= x^* - \Delta(t),\end{aligned}$$

or $x^* = x^* - \Delta(t)$, which is impossible because $\Delta(t) > 0$.

For case 2, by the assumption of the case, we have $x(k+1) = x(k) - \Delta(k) < -\Delta_{\max}$, which is not possible because $x(k) > \Delta_{\max}$.

The same approach can be applied to prove the case when x is initialized with a negative value, i.e., $x(0) < 0$.

The first part of the proof shows that there exists a t such that $|x(t)| \leq \Delta_{\max}$.

2. Next, we show that for any t such that $|x(t)| \leq \Delta_{\max}$, we have $|x(t+1)| \leq \Delta_{\max}$.

When $0 \leq x(t) \leq \Delta_{\max}$, we have

$$x(t+1) = x(t) - \Delta(t) \geq -\Delta_{\max} \qquad \text{and} \qquad x(t+1) = x(t) - \Delta(t) \leq x(t) \leq \Delta_{\max}.$$

When $-\Delta_{\max} \leq x(t) \leq 0$, we have

$$x(t+1) = x(t) + \Delta(t) \leq \Delta_{\max} \qquad \text{and} \qquad x(t+1) = x(t) + \Delta(t) \geq x(t) \geq -\Delta_{\max}.$$

This means that $-\Delta_{\max} \leq x(t+1) \leq \Delta_{\max}$, and this results holds for any t such that $|x(t)| \leq \Delta_{\max}$.

To combine the two parts of the proof, consider the first of such t, i.e., $t = \min \{\, t \,:\, |x(t)| \leq \Delta_{\max} \,\}$. We can prove, by mathematical induction, that $|x(t')| \leq \Delta_{\max}$ for all $t' \geq t$. □

The following proposition describes the behavior of $A(t)$ under signGD.

Proposition 4.D.3. *There exists t such that $|A(t')| \leq \tilde{\sigma}_0^2 \eta$ for all $t' > t$.*

Proof. Table 4.9 shows that there is always a non-zero update in the direction to reduce $|A(t)|$, so we can define the dynamics of $A(t)$ as

$$A(t+1) = A(t) - \text{sign}[A(t)]\Delta(t),$$

where $\Delta(t) \in \left\{ \frac{\tilde{\sigma}_0^2 \eta}{3}, \frac{2\tilde{\sigma}_0^2 \eta}{3}, \tilde{\sigma}_0^2 \eta \right\}$. Lemma 4.D.2 with $\Delta_{\max} = \tilde{\sigma}_0^2 \eta$ proves the proposition. □

Proposition 4.D.3 implies that once $|A|$ drops below $\tilde{\sigma}_0^2 \eta$, it remains below $\tilde{\sigma}_0^2 \eta$ for all future iterations. Combining Proposition 4.D.3 with the update directions of A in Table 4.9, we know that A will begin oscillating around zero. However, there are some limitations of Proposition 4.D.3. First, we do not know when exactly the oscillation starts: whether it starts immediately following the first iteration when $|A| \leq \tilde{\sigma}_0^2 \eta$ or from some iterations after it. Second, the characteristics of this oscillation (periodic or non-periodic) are unknown. Answers to these questions can improve our understanding of the behavior of A, and later become particularly useful in developing the asymptotic signGD solution of $\tilde{e}_2$, which is important because it leads to the adversarial risk of the signGD solution.

Because the update for $B(t)$ can be zero when $|A(t)| > |B(t)|$, Lemma 4.D.2 is not suitable to understand the dynamics of B, as the lemma requires that all step sizes be greater than zero. Nevertheless, Proposition 4.D.3 allows us to narrow down the range of A and we can partition the set of all possible values of A and B. By analyzing the dynamics of A and B in those partitions, we can develop the standard and adversarial population risk of the asymptotic signGD solution under a constant learning rate η.

Let us first divide the set of values of (A, B) into partitions based on the value of $|A|$, and then divide those partitions into smaller subpartitions based on the relative magnitude of $|A|$ and $|B|$. Such a partitioning process is illustrated in Figure 4.8.

- $R_1 = \left\{ (A, B) \,:\, \frac{2\tilde{\sigma}_0^2 \eta}{3} < |A| < \tilde{\sigma}_0^2 \eta \text{ and } B \in (-\infty, \infty) \right\}$,
  - $R_{11} = \{\, (A, B) \,:\, (A, B) \in R_1 \text{ and } |A| < |B| \,\}$,
  - $R_{12} = \{\, (A, B) \,:\, (A, B) \in R_1 \text{ and } |A| > |B| \,\}$,

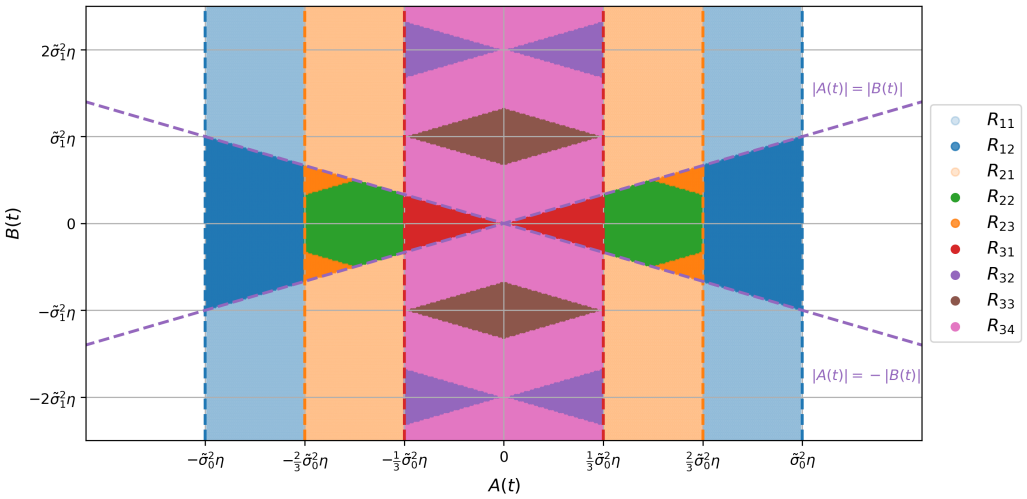


Figure 4.8: **Analyzing the dynamics of A and B by partitioning the set of values of (A, B) in $[-\tilde{\sigma}_0^2\eta, \tilde{\sigma}_0^2\eta] \times \mathbb{R}$.** Such a set is first divided into partitions R_1, R_2 and R_3 based on the value of $|A|$. Then, we consider $R_s = \{R_{22}, R_{31}, R_{32}, R_{33}, R_{34}\}$ as the stationary subpartitions, because once $(A(t), B(t)) \in R_s$, the sequence remains in the stationary subpartitions. Also, we consider $R_t = \{R_{11}, R_{12}, R_{21}, R_{23}\}$ as the transient subpartitions, because any $(A(t), B(t)) \in R_t$ will soon enter one of the stationary subpartitions, that is, there exists $t' \geq t$ such that $(A(t'), B(t')) \in R_s$. We consider $\tilde{\sigma}_0^2 = \tilde{\sigma}_1^2 = 1$ and $\eta = 1$ in this figure.

- $R_2 = \left\{ (A, B) : \frac{\tilde{\sigma}_0^2\eta}{3} < |A| < \frac{2\tilde{\sigma}_0^2\eta}{3} \text{ and } B \in (-\infty, \infty) \right\}$,
  - $R_{21} = \{ (A, B) : (A, B) \in R_2 \text{ and } |A| < |B| \}$,
  - $R_{22} = \left\{ (A, B) : (A, B) \in R_2 \text{ and } |A| > |B| \text{ and } \left|A + \tilde{\sigma}_0^2\eta\right| > |B| \text{ and } \left|A - \tilde{\sigma}_0^2\eta\right| > |B| \right\}$,
  - $R_{23} = \left\{ (A, B) : (A, B) \in R_2 \text{ and } |A| > |B| \text{ and } \left(\left|A + \tilde{\sigma}_0^2\eta\right| < |B| \text{ or } \left|A - \tilde{\sigma}_0^2\eta\right| < |B|\right) \right\}$,
- $R_3 = \left\{ (A, B) : |A| < \frac{\tilde{\sigma}_0^2\eta}{3} \text{ and } B \in (-\infty, \infty) \right\}$,
  - $R_{31} = \{ (A, B) : (A, B) \in R_3 \text{ and } |A| > |B| \}$,
  - $R_{32} = \left\{ (A, B) : \bigcup_{k \in \mathbb{Z}_{\text{even}} - \{0\}} \left\{ (A, B) \in R_3 \text{ and } |A| > \left|B + k\tilde{\sigma}_1^2\eta\right| \right\} \right\}$,
  - $R_{33} = \left\{ (A, B) : \bigcup_{k \in \mathbb{Z}_{\text{odd}}} \left\{ (A, B) \in R_3 \text{ and } |A| + \left|B + k\tilde{\sigma}_1^2\eta\right| < \frac{\tilde{\sigma}_0^2\eta}{3} \right\} \right\}$,
  - $R_{34} = R_3 - (R_{31} \cup R_{32} \cup R_{33})$,

where $\mathbb{Z}_{\text{odd}}$ and $\mathbb{Z}_{\text{even}}$ are the set of odd and even integers, respectively.

There are nine non-overlapping subpartitions. We call R_{22}, R_{31}, R_{32}, R_{33} and R_{34} the *stationary* subpartitions and denote $R_s = \{R_{22}, R_{31}, R_{32}, R_{33}, R_{34}\}$. They are called stationary subpartitions because once $(A(t), B(t)) \in R_s$, the sequence remains in the stationary subpartition. On the other hand, we call R_{11}, R_{12}, R_{21}, R_{23} the *transient* subpartitions and denote $R_t = \{R_{11}, R_{12}, R_{21}, R_{23}\}$. They are called the transient subpartitions because any $(A(t), B(t)) \in R_t$ will soon enter one of the stationary subpartitions, that is, there exists $t' \geq t$ such that $(A(t'), B(t')) \in R_t$. The dynamics of A and B can be summarized in the following proposition.

Proposition 4.D.4.

Transient subpartitions: *For each of $R \in R_t$, consider t such that $(A(t), B(t)) \in R$, then there*

exists $t' > t$ such that $(A(t'), B(t')) \in R_s$. The transition of $(A(t), B(t))$ from R_t to R_s happens at most 3 iterations after t; specifically, it corresponds to the scenario of $(A(t), B(t)) \in R_{11}$, $(A(t+1), B(t+1)) \in R_{23}$, $(A(t+2), B(t+2)) \in R_{21}$, and finally $(A(t+3), B(t+3)) \in R_3$.

Stationary subpartitions: *For each of $R \in R_s$, consider t such that $(A(t), B(t)) \in R$. For any $t' \geq t$, $|A(t')| \leq \frac{2\tilde{\sigma}_0^2\eta}{3}$ and $A(t')$ shows 2-periodic behavior switching between positive and negative signs, that is, for any $i \in \mathbb{Z}_{\geq 0}$, we have $A(t+2i) = A(t)$ and $\operatorname{sign}(A(t+2i)) = \operatorname{sign}(A(t)) = -\operatorname{sign}(A(t+2i+1))$. To be more specific about each stationary subpartition, we have*

1. *For each of $R \in \{R_{22}, R_{31}\}$, consider t such that $(A(t), B(t)) \in R$. For any $t' \geq t$, $|B(t')| \leq \tilde{\sigma}_1^2\eta$ and $B(t')$ remains constant, that is, $B(t') = B(t)$.*
2. *For each of $R \in \{R_{32}, R_{33}\}$, consider t such that $(A(t), B(t)) \in R$.*
   (a) *There exists $\bar{t} > t$ such that $(A(\bar{t}), B(\bar{t})) \in R_{31}$. Denote the smallest $\bar{t}$ as $\bar{t}^*$.*
   (b) *For any $\bar{t}^* > t' \geq t$, we have $|B(t'+1)| = |B(t')| - \operatorname{sign}[B(t')]$.*
   (c) *For any $t' \geq \bar{t}^*$, $|B(t')| \leq \tilde{\sigma}_1^2\eta$ and $B(t')$ remains constant, that is, $B(t') = B(t)$.*
3. *Consider t such that $(A(t), B(t)) \in R_{34}$.*
   (a) *For any $t' \geq t$, $(A(t'), B(t'))$ remains in R_{34}.*
   (b) *There exists $\bar{t} > t$ such that for any $\bar{t} > t' \geq t$, the sign of $B(t')$ remains constant.*
   (c) *For any $t' \geq \bar{t}$, $|B(t')| \leq \tilde{\sigma}_1^2\eta$ and $B(t')$ shows 2-periodic behavior switching between positive and negative signs, that is, for any $i \in \mathbb{Z}_{\geq 0}$, we have $B(\bar{t}+2i) = B(\bar{t})$ and $\operatorname{sign}(B(\bar{t}+2i)) = \operatorname{sign}(B(\bar{t})) = -\operatorname{sign}(B(\bar{t}+2i+1))$.*

Proof. From Proposition 4.D.3, we know that from an arbitrary $(A(0), B(0))$, $|A|$ will drop below $\tilde{\sigma}_0^2\eta$ under the signGD update, which means that (A, B) must enter one of the subpartitions. This allows us to continue analyzing the behavior of $(A(t), B(t))$ by assuming it enters one of the subpartitions at iteration t.

Analysis of R_{11}: For any $(A(t), B(t))$ in R_{11}, we know that $A(t+1) = A(t) - \operatorname{sign}[A(t)]\frac{\tilde{\sigma}_0^2\eta}{3}$. This means that $\frac{\tilde{\sigma}_0^2\eta}{3} < |A(t+1)| < \frac{2\tilde{\sigma}_0^2\eta}{3}$, so $(A(t+1), B(t+1))$ is in R_2 and we can study its dynamics using R_2.

Analysis of R_{12}: For any $(A(t), B(t))$ in R_{12}, we know that $A(t+1) = A(t) - \operatorname{sign}[A(t)]\tilde{\sigma}_0^2\eta$. This means that $|A(t+1)| < \frac{\tilde{\sigma}_0^2\eta}{3}$. Therefore, $(A(t+1), B(t+1))$ is in R_3 and we can analyze its dynamics using R_3.

Analysis of R_{21}: For any $(A(t), B(t))$ in R_{21}, we know that $A(t+1) = A(t) - \operatorname{sign}[A(t)]\frac{\tilde{\sigma}_0^2\eta}{3}$. This means that $|A(t+1)| < \frac{\tilde{\sigma}_0^2\eta}{3}$. Therefore, $(A(t+1), B(t+1))$ is in R_3 and we can analyze its dynamics using R_3.

Dynamics of (A, B) in R_{22} and R_{23}: For any $(A(t), B(t))$ in R_{22} and R_{23}, we have $A(t+1) = A(t) - \operatorname{sign}[A(t)]\tilde{\sigma}_0^2\eta$ and $B(t+1) = B(t)$, so we know that $\frac{\tilde{\sigma}_0^2\eta}{3} < |A(t+1)| < \frac{2\tilde{\sigma}_0^2\eta}{3}$.

Analysis of R_{22}: For any $(A(t), B(t))$ in R_{22}, we have $|A(t+1)| > |B(t+1)|$. This means that $(A(t+1), B(t+1))$ remains in R_{22}, and we have $A(t+2) = A(t+1) - \operatorname{sign}[A(t+1)]\tilde{\sigma}_0^2\eta$ and $B(t+2) = B(t+1)$, which means that $(A(t+2), B(t+2))$ returns to the starting position at $(A(t), B(t))$. In fact, for any $t' \geq t$, $A(t')$ shows 2-periodic behavior switching between positive and negative signs and $B(t')$ remains constant, that is, for any $i \in \mathbb{Z}_{\geq 0}$, we have $A(t+2i) = A(t)$, $\operatorname{sign}(A(t+2i)) = \operatorname{sign}(A(t)) = -\operatorname{sign}(A(t+2i+1))$, and $B(t+2i+1) = B(t+2i) = B(t)$.

Analysis of R_{23}: For any $(A(t), B(t))$ in R_{23}, by the definition of subpartition, we have that $|A(t+1)| < |B(t+1)|$, so $(A(t+1), B(t+1))$ is in R_{21}. This means that $(A(t+2), B(t+2))$ is in R_3 and we can analyze its dynamics using partition R_3.

Analysis of R_{31}: For any $(A(t), B(t))$ in R_{31}, we know that $A(t+1) = A(t) - \text{sign}[A(t)]\tilde{\sigma}_0^2\eta$ and $B(t+1) = B(t)$. This means that $\frac{2\tilde{\sigma}_0^2\eta}{3} < |A(t+1)| < \tilde{\sigma}_0^2\eta$. Since $|A(t+1)| > |B(t+1)| = |B(t)|$, we have that $B(t+2) = B(t+1)$ and $A(t+2) = A(t+1) - \text{sign}[A(t+1)]\tilde{\sigma}_0^2\eta$, which means that $(A(t+2), B(t+2))$ returns to the starting position at $(A(t), B(t))$. Therefore, for any $t' \geq t$, $A(t')$ shows 2-periodic behavior switching between positive and negative signs and $B(t')$ remains constant, that is, for any $i \in \mathbb{Z}_{\geq 0}$, we have $A(t+2i) = A(t)$, $\text{sign}(A(t+2i)) = \text{sign}(A(t)) = -\text{sign}(A(t+2i+1))$, and $B(t+2i+1) = B(t+2i) = B(t)$.

Dynamics of A in R_{32}, R_{33} and R_{34}: The behavior of A in R_{32}, R_{33} and R_{34} is the same. For any $(A(t), B(t))$ in $\{R_{32}, R_{33}, R_{34}\}$, we know that

$$A(t+1) = A(t) - \text{sign}[A(t)]\frac{\tilde{\sigma}_0^2\eta}{3} \quad \text{and} \quad B(t+1) = B(t) - \text{sign}[B(t)]\tilde{\sigma}_1^2\eta. \tag{4.21}$$

This means that $|A(t+1)| < \frac{\tilde{\sigma}_0^2\eta}{3}$, so $(A(t+1), B(t+1))$ remains in R_3. For any $t' \geq t$, $A(t')$ shows 2-periodic behavior switching between positive and negative signs, that is, for any $i \in \mathbb{Z}_{\geq 0}$, we have

$$A(t+2i) = A(t) \quad \text{and} \quad \text{sign}(A(t+2i)) = \text{sign}(A(t)) = -\text{sign}(A(t+2i+1)).$$

The behavior of B is different across the three subpartitions, so we analyze them separately.

Analysis of R_{32}: Because all subpartitions are non-overlapping, for any $(A(t), B(t))$ in R_{32}, there exists a unique $k \in \mathbb{Z}_{\text{even}} - \{0\}$ such that $A(t)$ and $B(t)$ satisfies $|A(t)| > \left|B(t) + k\tilde{\sigma}_1^2\eta\right|$. Next, we show that starting from any $(A(t), B(t))$ in R_{32}, after $|k|$ iterations of signGD update, we have $|A(t+|k|)| > |B(t+|k|)|$, which means that $(A(t+|k|), B(t+|k|))$ is in R_{31}.

This can be proved by showing that $|A(t+|k|)| = |A(t)|$ and $|B(t+|k|)| = \left|B(t) + k\tilde{\sigma}_1^2\eta\right|$. For any $t' \geq t$, $A(t')$ shows 2-periodic behavior, and because k is an even number, we have $|A(t+|k|)| = |A(t)|$.

Since $\left|B(t) + k\tilde{\sigma}_1^2\eta\right| > 0$ and $B(t+1) = B(t) - \text{sign}[B(t)]\tilde{\sigma}_1^2\eta$, we know that the sign of B remains the same for the next $|k| - 1$ updates. This means that

$$B(t+|k|) = B(t) - \sum_{i=0}^{|k|-1} \text{sign}[B(t+i)]\tilde{\sigma}_1^2\eta = B(t) - |k|\,\text{sign}[B(t)]\tilde{\sigma}_1^2\eta.$$

By the definition of the subpartition, we have $\frac{\tilde{\sigma}_0^2\eta}{3} > \left|B(t) + k\tilde{\sigma}_1^2\eta\right|$; and this is true if and only if $B(t)$ and k have opposite signs because k is a non-zero even integer. Therefore, we have

$$\begin{aligned}
|B(t+|k|)| &= \left|B(t) - |k|\,\text{sign}[B(t)]\tilde{\sigma}_1^2\eta\right| \\
&= \left|B(t) - |k|\,(-\text{sign}[k])\tilde{\sigma}_1^2\eta\right| \\
&= \left|B(t) + |k|\,\text{sign}[k]\tilde{\sigma}_1^2\eta\right| \\
&= \left|B(t) + k\tilde{\sigma}_1^2\eta\right|.
\end{aligned}$$

Analysis of R_{33}: Similarly, for any $(A(t), B(t))$ in R_{33}, there exists a unique $k \in \mathbb{Z}_{\text{odd}}$ such

that $A(t)$ and $B(t)$ satisfies $|A(t)| + \left|B(t) + k\tilde{\sigma}_1^2\eta\right| < \frac{\tilde{\sigma}_0^2\eta}{3}$. Again, we show that starting from any $(A(t), B(t))$ in R_{33}, after $|k|$ iterations of signGD update, $(A(t+|k|), B(t+|k|))$ is in R_{31}. This can be proved by showing that $\frac{\tilde{\sigma}_0^2\eta}{3} - |A(t)| \leq |A(t+|k|)|$ and $\left|B(t) + k\tilde{\sigma}_1^2\eta\right| = |B(t+|k|)|$.

First, by using the same analysis of $\left|B(t) + k\tilde{\sigma}_1^2\eta\right|$ in R_{32}, we have $\left|B(t) + k\tilde{\sigma}_1^2\eta\right| = |B(t+|k|)|$. Next, the behavior of $A(t)$ follows (4.21), which means that for any $t' \geq t$, $A(t')$ shows 2-periodic behavior, and because k is an odd number, we have $|A(t+|k|) - A(t)| = |A(t+1) - A(t)| = \frac{\tilde{\sigma}_0^2\eta}{3}$.

Also, we have $|A(t+|k|) - A(t)| \leq |A(t+|k|)| + |A(t)|$, which means that $\frac{\tilde{\sigma}_0^2\eta}{3} \leq |A(t+|k|)| + |A(t)|$, or $\frac{\tilde{\sigma}_0^2\eta}{3} - |A(t)| \leq |A(t+|k|)|$. Combining with the definition of the subpartition, we have

$$|B(t) + |k|| < \frac{\tilde{\sigma}_0^2\eta}{3} - |A(t)| \leq |A(t+|k|)|\,.$$

Therefore, starting from any $(A(t), B(t))$ in R_{33}, after $|k|$ iterations of signGD updates, we have $|A(t+|k|)| > |B(t+|k|)|$, which together with the fact that $|A(t+|k|)| < \frac{\tilde{\sigma}_0^2\eta}{3}$ as shown before, implies that $(A(t+|k|), B(t+|k|))$ is in R_{31}.

Analysis of R_{34}: Finally, we prove, by contradiction, that for any $(A(t), B(t))$ in R_{34}, there is no $t' > t$ such that $(A(t'), B(t'))$ in R_{31}. Suppose that $(A(t'), B(t'))$ enters R_{31}, then $k \triangleq t' - t$ must be either an odd number or an even number. By the definition of the subpartition, if k is a non-zero even number, it means that $(A(t), B(t))$ must be in R_{32}; whereas if k is an odd number, it means that $(A(t), B(t))$ must be in R_{33}. Neither is possible since all subpartitions are non-overlapping, so for any $(A(t), B(t))$ in R_{34}, $(A(t'), B(t'))$ remains in R_{34} for all $t' \geq t$. This means that there will always be a non-zero update on B, and this allows us to apply Lemma 4.D.2. For any $(A(t), B(t))$ in R_{34}, there exists t such that $|B(t')| \leq \tilde{\sigma}_1^2\eta$ for all $t' \geq t$. □

Combining Proposition 4.D.1 and the dynamics of (A, B) in the stationary subpartitions described in Proposition 4.D.4, we have the following remark.

Remark 4.D.5. *The asymptotic solution of A oscillates in $[-\frac{2\tilde{\sigma}_0^2\eta}{3}, \frac{2\tilde{\sigma}_0^2\eta}{3}]$, which is a tighter bound compared to the one in Proposition 4.D.3. The asymptotic solution of B either remains constant in $[-\tilde{\sigma}_1^2\eta, \tilde{\sigma}_1^2\eta]$ (1 and 2c in Proposition 4.D.4) or oscillates in $[-\tilde{\sigma}_1^2\eta, \tilde{\sigma}_1^2\eta]$ (3c in Proposition 4.D.4). Since $A(t)$ and $B(t)$ denote $\frac{\sqrt{3}}{3}\tilde{\sigma}_0^2\tilde{e}_0(t)$ and $\frac{\sqrt{2}}{2}\tilde{\sigma}_1^2\tilde{e}_1(t)$, respectively, this means that $\limsup_{t\to\infty} |\tilde{e}_0(t)| = \frac{2\sqrt{3}}{3}\eta$, and $\limsup_{t\to\infty} |\tilde{e}_1(t)| = \sqrt{2}\eta$,*

From the dynamics of (A, B) in the transient subpartitions described in Proposition 4.D.4, we have the following corollary.

Corollary 4.D.6. *Suppose that $\tilde{e}_0$ enters $[-\sqrt{3}\eta, \sqrt{3}\eta]$ at iteration t, then A starts exhibiting a 2-periodic oscillation at most 3 iterations after t. This means that the maximum difference between the number of positive and negative A's after iteration t is 2:*

$$\left|\sum_{i=t+1}^{\infty} \mathbb{I}\{\text{sign}[A(i)] = 1\} - \mathbb{I}\{\text{sign}[A(i)] = -1\}\right| \leq 2.$$

Dynamics of $\tilde{e}_2$ under signGD

We are now ready to analyze the dynamics of $\tilde{e}_2$ through the behavior of $\tilde{e}_0$ and $\tilde{e}_1$. Particularly, we demonstrate that the final value of $|\tilde{e}_2|$ is affected by the magnitude of $|\tilde{e}_0(0)|$ and $|\tilde{e}_1(0)|$. First,

notice that the update direction along $\tilde{e}_2$ follows the opposite of the sign of $\tilde{e}_0$, and at every iteration when $|A(t)| \leq |B(t)|$, there is a non-zero weight adaptation for $\tilde{e}_2$. Once the oscillation begins for $\tilde{e}_0$, the dynamics of $\tilde{e}_0$ and $\tilde{e}_2$ become similar. Consider T as the first iteration when $|\tilde{e}_0|$ drops below $\sqrt{3}\eta$. We then have

$$\begin{aligned}
\limsup_{t\to\infty} |\tilde{e}_2(t)| &= \left| \tilde{e}_2(T) + \eta \sum_{t=T+1}^{\infty} \left\{ \mathbb{I}\left\{|A(t)| < |B(t)|\right\} \frac{-\sqrt{6}}{3} + \mathbb{I}\left\{|A(t)| = |B(t)|\right\} \frac{-\sqrt{6}}{6} \right\} \operatorname{sign}[\tilde{e}_0(t)] \right| \\
&\leq |\tilde{e}_2(T)| + \frac{\sqrt{6}}{3}\eta \left| \sum_{t=T+1}^{\infty} \operatorname{sign}[\tilde{e}_0(t)] \right| \\
&\leq |\tilde{e}_2(T)| + \frac{2\sqrt{6}}{3}\eta, \qquad (4.22)
\end{aligned}$$

where we use Corollary 4.D.6 to upper bound the absolute value of the summation of the sign of $\tilde{e}_0$ after the T-th iteration in the last inequality. This means that after T iterations, $\tilde{e}_2$ stays in an $O(\eta)$ neighborhood of $\tilde{e}_2(T)$; in other words, $\tilde{w}_2$ stays in an $O(\eta)$ neighborhood of $\tilde{w}_2(T)$. Also, notice that (4.22) does not include $\mathbb{I}\left\{|A(t)| > |B(t)|\right\}$ since $\tilde{e}_2$ is updated only when $|A(t)| \leq |B(t)|$, as shown in Table 4.9.

Define $\Delta\tilde{w}_2$ as the sum of all the updates in $\tilde{w}_2$ up to the T-th iteration:

$$\Delta\tilde{w}_2 \triangleq \eta \sum_{t=0}^{T-1} \left\{ \mathbb{I}\left\{|A(t)| < |B(t)|\right\} \frac{-\sqrt{6}}{3} + \mathbb{I}\left\{|A(t)| = |B(t)|\right\} \frac{-\sqrt{6}}{6} \right\} \operatorname{sign}[\tilde{e}_0(t)],$$

which leads to

$$\limsup_{t\to\infty} |\tilde{w}_2(t)| = |\tilde{w}_2(T) + O(\eta)| = |\tilde{w}_2(0) + \Delta\tilde{w}_2 + O(\eta)|, \qquad (4.23)$$

where $\tilde{w}_2(0)$ is the weight at initialization.

Putting (4.23) together with Remark 4.D.5, the asymptotic solution found by signGD is

$$\tilde{\boldsymbol{w}}^{\text{signGD}} = \left[\tilde{w}_0^*, \tilde{w}_1^*, \tilde{w}_2(0) + \Delta\tilde{w}_2\right]^\top + O(\eta).$$

From the perspective of training under the standard risk, the signGD solution is close to the optimum. Specifically, its standard risk is

$$\begin{aligned}
\mathcal{R}_{\text{s}}(\tilde{\boldsymbol{w}}^{\text{signGD}}) &= \mathbb{E}\left[\ell(\tilde{X}, Y; \tilde{\boldsymbol{w}}^{\text{signGD}})\right] \\
&= \frac{1}{2}\mathbb{E}\left[\left\langle \tilde{X}\,,\, \tilde{\boldsymbol{w}}^{\text{signGD}} - \tilde{w}^* \right\rangle^2\right] \\
&= \frac{1}{2}\left(\mathbb{E}\left[\tilde{X}_0^2\right](\tilde{\boldsymbol{w}}_0^{\text{signGD}} - \tilde{w}_0^*)^2 + \mathbb{E}\left[\tilde{X}_1^2\right](\tilde{\boldsymbol{w}}_1^{\text{signGD}} - \tilde{w}_1^*)^2\right) \qquad (4.24) \\
&= \frac{1}{2}\left(\tilde{\sigma}_0^2 O(\eta^2) + \tilde{\sigma}_1^2 O(\eta^2)\right) \\
&= O((\tilde{\sigma}_0^2 + \tilde{\sigma}_1^2)\eta^2),
\end{aligned}$$

where $\mathbb{E}\left[\tilde{X}_0\tilde{X}_1\right] = 0$ in (4.24) due to the diagonality of $\tilde{\Sigma}$. Note that the standard risk of the GD solution is exactly zero; and by choosing a small learning rate η, the standard risk of the signGD solution can be close to zero as well. However, their adversarial risks are very different. Specifically,

the adversarial risk of the asymptotic signGD solution is

$$\mathcal{R}_{\mathrm{a}}(\tilde{\boldsymbol{w}}^{\mathrm{signGD}}) = \frac{\epsilon^2}{2}||\tilde{\boldsymbol{w}}^{\mathrm{signGD}}||_2^2 = \frac{\epsilon^2}{2}\left\{\tilde{w}_0^{*2} + \tilde{w}_1^{*2} + (\tilde{w}_2(0) + \Delta\tilde{w}_2)^2 + O(\eta^2)\right\}.$$

Consider a sufficiently small learning rate: $\eta \ll \min\{\tilde{w}_0^*, \tilde{w}_1^*, \tilde{w}_2(0) + \Delta\tilde{w}_2\}$. This means that the contribution from $O(\eta^2)$ in $\mathcal{R}_{\mathrm{a}}(\tilde{\boldsymbol{w}}^{\mathrm{signGD}})$ is negligible. Then the adversarial risk of the signGD solution becomes

$$\mathcal{R}_{\mathrm{a}}(\tilde{\boldsymbol{w}}^{\mathrm{signGD}}) = \frac{\epsilon^2}{2}\left\{\tilde{w}_0^{*2} + \tilde{w}_1^{*2} + (\tilde{w}_2(0) + \Delta\tilde{w}_2)^2\right\}. \tag{4.25}$$

We can compare it with the adversarial risk of the asymptotic solution found by GD under the same setup:

$$\mathcal{R}_{\mathrm{a}}(\tilde{\boldsymbol{w}}^{\mathrm{GD}}) = \frac{\epsilon^2}{2}\left\{\tilde{w}_0^{*2} + \tilde{w}_1^{*2} + \tilde{w}_2^2(0)\right\}. \tag{4.26}$$

The main difference between the two adversarial risks in (4.25) and (4.26) is the difference in weights learned at the irrelevant frequency. Since their use of irrelevant frequency in the data is under-constrained, neither algorithm can reduce $\tilde{w}_2$ to zero, thereby neither solution is the most robust standard risk minimizer. The GD solution is sensitive to weight initialization. To understand the $\Delta\tilde{w}_2$ term in the signGD solution, first recall that T denotes the first iteration when $|\tilde{e}_0|$ drops below $\sqrt{3}\eta$ (or $|A|$ drops below $\tilde{\sigma}_0^2\eta$), and from Corollary 4.D.6 we know that $\tilde{w}_0$ starts oscillation at most 3 iterations after T. Recall in (4.23) that $O(\eta)$ has been utilized to account for the maximum sign variations, this means that we can consider oscillations which begin immediately after the T-th update. Suppose that η is small so the sign of $\tilde{e}_0$ would not change before the oscillation starts, then we have

$$\begin{aligned}
|\Delta\tilde{w}_2| &= \left|\eta\sum_{t=0}^{T-1}\left\{\mathbb{I}\left\{|A(t)| < |B(t)|\right\}\frac{-\sqrt{6}}{3} + \mathbb{I}\left\{|A(t)| = |B(t)|\right\}\frac{-\sqrt{6}}{6}\right\}\mathrm{sign}[\tilde{e}_0(t)]\right| \\
&= \left|\eta\sum_{t=0}^{T-1}\left\{\mathbb{I}\left\{|A(t)| < |B(t)|\right\}\frac{-\sqrt{6}}{3} + \mathbb{I}\left\{|A(t)| = |B(t)|\right\}\frac{-\sqrt{6}}{6}\right\}\right|,
\end{aligned}$$

which leads to

$$|\Delta\tilde{w}_2| = C\eta\sum_{t=0}^{T-1}\mathbb{I}\left\{|A(t)| \le |B(t)|\right\}, \tag{4.27}$$

where C denotes some value between $\frac{\sqrt{6}}{6}$ and $\frac{\sqrt{6}}{3}$, which correspond to always using the smaller and the larger updates, respectively.

Dynamics of $|\Delta\tilde{w}_2|$ under signGD

There are two factors that can affect the magnitude of $\sum_{t=0}^{T-1}\mathbb{I}\left\{|A(t)| \le |B(t)|\right\}$ in (4.27): 1) the relative magnitudes between $\tilde{\sigma}_0^2$ and $\tilde{\sigma}_1^2$, and 2) the initial values of $|\tilde{e}_0|$ and $|\tilde{e}_1|$, or equivalently, the initial values of $|A|$ and $|B|$. To analyze this, we divide the set of values of $(|A(t)|, |B(t)|)$ into several partitions: the set of $[0, \tilde{\sigma}_0^2\eta] \times \mathbb{R}$ and $\mathbb{R} \times [0, \tilde{\sigma}_0^2\eta]$ is partitioned into P_1 and P_2, and the set of $[\tilde{\sigma}_0^2\eta, \infty) \times [\tilde{\sigma}_0^2\eta, \infty)$ is partitioned differently based on the value of $\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2}$. Consider a line that travels through the point of $(\tilde{\sigma}_0^2\eta, \tilde{\sigma}_0^2\eta)$ and has a slope of $3\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2}$. The ratio between $\tilde{\sigma}_0^2$ and $\tilde{\sigma}_1^2$ is particularly

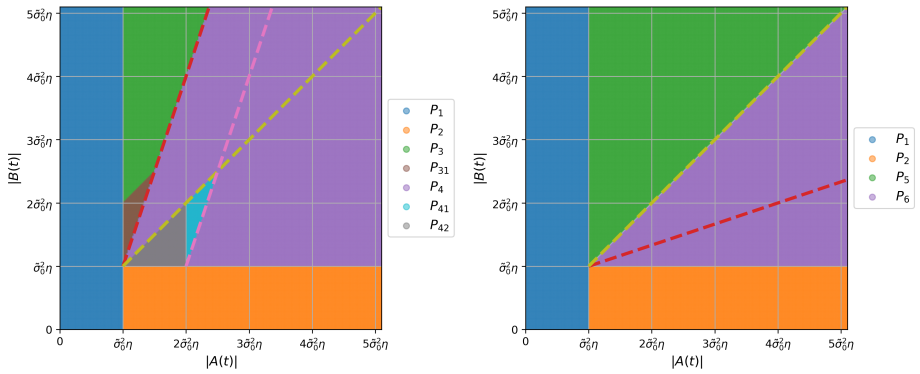


Figure 4.9: **Analyzing the value of $\sum_{t=0}^{T-1}\mathbb{I}\{|A(t)|\le|B(t)|\}$ in (4.27) by partitioning the set of values of $(|A(t)|,|B(t)|)$, and the relative magnitude between $\tilde{\sigma}_0^2$ and $\tilde{\sigma}_1^2$ determines the partitions on which the analysis is based.** Specifically, the analysis is based on partitions P_1, P_2, P_3 and P_4, when $\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2}>\frac{1}{3}$ (left), and on P_1, P_2, P_5 and P_6, when $\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2}<\frac{1}{3}$ (right). The three smaller subpartitions are subsets of the main partition, i.e., $P_{31}\subset P_3$ and $P_{41},P_{42}\subset P_4$, and they are used in the analysis of P_4. The value of $\sum_{t=0}^{T-1}\mathbb{I}\{|A(t)|\le|B(t)|\}$ when $(|A(0)|,|B(0)|)$ is initialized in each partition is summarized in Proposition 4.D.7. The two plots are created with $\tilde{\sigma}_0^2=\tilde{\sigma}_1^2$ (left) and $\tilde{\sigma}_0^2=9\tilde{\sigma}_1^2$ (right), respectively. Note that those values are chosen for illustration purposes and do not affect the generality of the result. In both plots, the red dashed line corresponds to $|B(t)|=3\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2}|A(t)|-(3\tilde{\sigma}_1^2-\tilde{\sigma}_0^2)\eta$ for $|A(t)|\in(\tilde{\sigma}_0^2\eta,\infty)$, and the yellow dashed line corresponds to $|B(t)|=|A(t)|$. The pink dashed line is parallel to the red dashed line with a horizontal gap of $\tilde{\sigma}_0^2\eta$.

useful in analyzing $|\Delta\tilde{w}_2|$ because understanding the position of $(|A(0)|,|B(0|)$ relative to such a line can lead to the value of $|B(T-1)|$, that is, the value of $|B|$ before the oscillation of $|A|$ begins. Since $|B|$ is updated only when $|A|\le|B|$, we have $\sum_{t=0}^{T-1}\mathbb{I}\{|A(t)|\le|B(t)|\}=\frac{|B(0)|-|B(T-1)|}{\tilde{\sigma}_1^2\eta}$. The definitions of partitions are

- $P_1=\{\,(A,B)\,:\,|A|<\tilde{\sigma}_0^2\eta\,\}$,
- $P_2=\{\,(A,B)\,:\,|A|>\tilde{\sigma}_0^2\eta \text{ and } |B|<\tilde{\sigma}_0^2\eta\,\}$,
- When $\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2}>\frac{1}{3}$,
  - $P_3=\left\{\,(A,B)\,:\,\tilde{\sigma}_0^2\eta<|A|<\frac{\tilde{\sigma}_0^2}{3\tilde{\sigma}_1^2}(|B|+(3\tilde{\sigma}_1^2-\tilde{\sigma}_0^2)\eta)\,\right\}$,
    - $P_{31}=\{\,(A,B)\,:\,(A,B)\in P_3 \text{ and } |A|+\tilde{\sigma}_0^2\eta>|B|\,\}$,
  - $P_4=\left\{\,(A,B)\,:\,\tilde{\sigma}_0^2\eta<|B|<\frac{3\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2}|A|-(3\tilde{\sigma}_1^2-\tilde{\sigma}_0^2)\eta\,\right\}$,
    - $P_{41}=\{\,(A,B)\,:\,(A,B)\in P_4 \text{ and } |B|<|A|<2\tilde{\sigma}_0^2\eta\,\}$,
    - $P_{42}=\left\{\,(A,B)\,:\,(A,B)\in P_4 \text{ and } |A|>|B| \text{ and } 2\tilde{\sigma}_0^2\eta<|A|<\frac{|B|+(3\tilde{\sigma}_1^2-\tilde{\sigma}_0^2)\eta}{3\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2}}+\tilde{\sigma}_0^2\eta\,\right\}$,
- When $\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2}<\frac{1}{3}$,
  - $P_5=\{\,(A,B)\,:\,\tilde{\sigma}_0^2\eta<|A|<|B|\,\}$,
  - $P_6=\{\,(A,B)\,:\,\tilde{\sigma}_0^2\eta<|B|<|A|\,\}$.

An illustration of partitions is provided in Figure 4.9, where the two plots demonstrate the two different ways of dividing the set of $[\tilde{\sigma}_0^2\eta, \infty) \times [\tilde{\sigma}_0^2\eta, \infty)$ based on the value of $\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2}$. The connection between the values of $(|A(0)|, |B(0)|)$ and the size of $\sum_{t=0}^{T-1} \mathbb{I}\{|A(t)| \leq |B(t)|\}$ is summarized in the next proposition.

Proposition 4.D.7. *Denote T as the iteration when $|\tilde{e}_0|$ drops below $\sqrt{3}\eta$. The value of $\sum_{t=0}^{T-1} \mathbb{I}\{|A(t)| \leq |B(t)|\}$ depends on the relative magnitude between $\tilde{\sigma}_0^2$ and $\tilde{\sigma}_1^2$, and the initial values of $|A|$ and $|B|$. Specifically, we have*

$$\sum_{t=0}^{T-1} \mathbb{I}\{|A(t)| \leq |B(t)|\} = \begin{cases} 0 & \text{if} \quad (|A(0)|, |B(0)|) \in (P_1 \bigcup P_2) \\ T & \text{if} \quad \frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2} > \frac{1}{3} \quad \text{and} \quad (|A(0)|, |B(0)|) \in P_3 \\ \frac{|B(0)|}{\tilde{\sigma}_1^2\eta} + [\frac{-2\tilde{\sigma}_0^2}{\tilde{\sigma}_1^2}, \frac{-\tilde{\sigma}_0^2}{\tilde{\sigma}_1^2}] & \text{if} \quad \frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2} > \frac{1}{3} \quad \text{and} \quad (|A(0)|, |B(0)|) \in P_4 \\ T & \text{if} \quad \frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2} < \frac{1}{3} \quad \text{and} \quad (|A(0)|, |B(0)|) \in P_5 \\ \frac{|B(0)| - \tilde{\sigma}_0^2\eta}{\frac{1}{3}\tilde{\sigma}_0^2\eta} & \text{if} \quad \frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2} < \frac{1}{3} \quad \text{and} \quad (|A(0)|, |B(0)|) \in P_6. \end{cases}$$

Proof.
We divide the analysis into two main parts: when $\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2} > \frac{1}{3}$ and $\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2} < \frac{1}{3}$, corresponding to the left and right figures in Figure 4.9. For each case, we analyze the behavior of (A, B) within the partition.

Analysis of P_1: For any $(|A(0)|, |B(0)|)$ in P_1, since $|A(0)|$ is already below $\tilde{\sigma}_0^2\eta$, we have $T = 1$ because A remains in P_1. This means that $\sum_{t=0}^{T-1} \mathbb{I}\{|A(t)| \leq |B(t)|\} = 0$.

Analysis of P_2: For any $(|A(0)|, |B(0)|)$ in P_2, $|A|$ decreases until it drops below $\tilde{\sigma}_0^2\eta$, while $|B|$ remains the same. This means that $|A|$ remains smaller than $|B|$, so we have $\mathbb{I}\{|A(t)| \leq |B(t)|\} = 0$ for all $t \in \{0, \ldots, T-1\}$. Therefore, we have $\sum_{t=0}^{T-1} \mathbb{I}\{|A(t)| \leq |B(t)|\} = 0$.

Next, the partitions of the set of $[\tilde{\sigma}_0^2\eta, \infty] \times [\tilde{\sigma}_0^2\eta, \infty]$ are defined differently based on the values of $\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2}$ compared to $\frac{1}{3}$. This is because when $\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2} > \frac{1}{3}$, it is possible for any $(|A(t)|, |B(t)|)$ satisfying $|A(t)| < |B(t)|$, there exists $t' > t$ such that $|A(t')| > |B(t')|$. In other words, $(|A|, |B|)$ can oscillate above and below the line defined by $|A| = |B|$, and this makes analyzing (4.27) difficult. However, when $\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2} < \frac{1}{3}$, any $(|A(t)|, |B(t)|)$ that satisfies $|A(t)| < |B(t)|$ will stay above the line defined by $|A| = |B|$, and this means that $|A|$ will always get updated by $\frac{\tilde{\sigma}_0^2\eta}{3}$ and $|B|$ will always get updated by $\tilde{\sigma}_1^2\eta$. Because of this different behavior, we analyze these two cases separately by defining different partitions. This corresponds to the left and right figures in Figure 4.9. When $\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2} > \frac{1}{3}$, the set of $[\tilde{\sigma}_0^2\eta, \infty] \times [\tilde{\sigma}_0^2\eta, \infty]$ is partitioned into P_3 and P_4.

Analysis of P_3: By definition, any $(|A(t)|, |B(t)|)$ in P_3 satisfies $3\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2}|A(t)| < |B(t)| + (3\tilde{\sigma}_1^2 - \tilde{\sigma}_0^2)\eta$. Starting from any $(|A(0)|, |B(0)|)$ in P_3, the values of $|A|$ and $|B|$ decrease at a rate of $\frac{1}{3}\tilde{\sigma}_0^2\eta$ and $\tilde{\sigma}_1^2\eta$, respectively, and this means that two sides of the inequality decrease at the same rate. Hence, the sequence $(|A(t)|, |B(t)|)$ remains in P_3 for all $0 \leq t < T-1$. This means that $\mathbb{I}\{|A(t)| \leq |B(t)|\} = 1$ for all $t \in \{0, \ldots, T-1\}$. Therefore, we have $\sum_{t=0}^{T-1} \mathbb{I}\{|A(t)| \leq |B(t)|\} = T$.

Analysis of P_4: Since $(|A(T-1)|, |B(T-1)|)$ must be in P_1, we can understand the value of $\sum_{t=0}^{T-1} \mathbb{I}\{|A(t)| \leq |B(t)|\}$ by considering how any $(|A(0)|, |B(0)|)$ in P_4 is transitioned to $(|A(T-1)|, |B(T-1)|)$ in P_1. Also, starting from any $(|A(t)|, |B(t)|)$ in P_4, we know that $|A(t)| - |A(t+1)| > 0$ and $|B(t)| - |B(t+1)| \geq 0$; hence, the transition from P_4 to P_1 must be described in one of the following scenarios.

Transition to P_2 then to P_1: In this case, the value of $|B|$ must first drop below $\tilde{\sigma}_0^2\eta$. Since $|B|$ decreases only when $|A| \leq |B|$, this means that, regardless of the initial value of $|A|$, the same number of updates is required to reduce $|B(0)|$ to $\tilde{\sigma}_0^2\eta$, which is $\frac{|B(0)|-\tilde{\sigma}_0^2\eta}{\tilde{\sigma}_1^2\eta}$, and in each update, the condition $\mathbb{I}\{|A(t)| \leq |B(t)|\}$ is satisfied.

Transition to P_1 directly: For any $(|A(t)|, |B(t)|)$ in P_4 that satisfies $|B(t)| > |A(t)|$ (above the yellow dashed line in Figure 4.9), since the values of $|A|$ and $|B|$ decrease at a rate of $\frac{1}{3}\tilde{\sigma}_0^2\eta$ and $\tilde{\sigma}_1^2\eta$, respectively, $(|A(t+1)|, |B(t+1)|)$ cannot cross the red dashed line which has a slope of $3\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2}$. Now let us consider any $(|A(t)|, |B(t)|)$ in P_4 that satisfies $|B(t)| < |A(t)|$ (below the yellow dashed line). In this case, $|A|$ decreases by $\tilde{\sigma}_0^2\eta$, and the only scenario where $(|A(T-1)|, |B(T-1)|)$ ends up in P_1 is when $\tilde{\sigma}_0^2\eta < |A(T-2)| < 2\tilde{\sigma}_0^2\eta$. That is, $(|A(T-2)|, |B(T-2)|)$ is in P_{42}. When this happens, we have $\tilde{\sigma}_0^2\eta < |B(T-2)| < 2\tilde{\sigma}_0^2\eta$; and because there is no update in $|B(T-2)|$, we have $\tilde{\sigma}_0^2\eta < |B(T-1)| < 2\tilde{\sigma}_0^2\eta$. Therefore, we have $\sum_{t=0}^{T-1}\mathbb{I}\{|A(t)| \leq |B(t)|\} \in [\frac{|B(0)|-2\tilde{\sigma}_0^2\eta}{\tilde{\sigma}_1^2\eta}, \frac{|B(0)|-\tilde{\sigma}_0^2\eta}{\tilde{\sigma}_1^2\eta}]$.

Transition to P_3 then to P_1: Let us first consider the transition from P_4 to P_3. Consider t' such that $(|A(t)|, |B(t)|)$ is in P_4 for $0 \leq t < t'$ and $(|A(t')|, |B(t')|)$ is in P_3. Following the above analysis (direct transition to P_1), we know that $(|A(t'-1)|, |B(t'-1)|)$ must satisfy $|A(t'-1)| > |B(t'-1)|$, where $t'-1$ is the iteration before transitioning to P_3. Also, we know that $|A(t'-1)| > 2\tilde{\sigma}_0^2\eta$ otherwise $(|A(t')|, |B(t')|)$ would be in P_1. The last condition for such a transition to happen is that the horizontal distance from $|B(t'-1)|$ to the line of $|B| = 3\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2}|A| - (3\tilde{\sigma}_1^2 - \tilde{\sigma}_0^2)\eta$ (the red dashed line) must be smaller than $\tilde{\sigma}_0^2\eta$. That is, $(|A(t'-1)|, |B(t'-1)|)$ is in P_{41}, and $(|A(t')|, |B(t')|)$ is in P_{31}. After the transition to P_3, the values of $|A|$ and $|B|$ decrease at a rate of $\frac{1}{3}\tilde{\sigma}_0^2\eta$ and $\tilde{\sigma}_1^2\eta$, respectively, and $|B(T-1)|$ has a range of $[\tilde{\sigma}_0\eta, 2\tilde{\sigma}_0\eta]$. Therefore, we have $\sum_{t=0}^{T-1}\mathbb{I}\{|A(t)| \leq |B(t)|\} \in [\frac{|B(0)|-2\tilde{\sigma}_0^2\eta}{\tilde{\sigma}_1^2\eta}, \frac{|B(0)|-\tilde{\sigma}_0^2\eta}{\tilde{\sigma}_1^2\eta}]$.

When $\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2} < \frac{1}{3}$, the set of $[\tilde{\sigma}_0^2\eta, \infty) \times [\tilde{\sigma}_0^2\eta, \infty)$ is partitioned into P_5 and P_6, as shown in the right figure of Figure 4.9.

Analysis of P_5: Starting from any $(|A(0)|, |B(0)|)$ in P_5, the values of $|A|$ and $|B|$ decrease at a rate of $\frac{1}{3}\tilde{\sigma}_0^2\eta$ and $\tilde{\sigma}_1^2\eta$, respectively. However, since $\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2} < \frac{1}{3}$, there will not be any $0 \leq t \leq T-1$ where $|A(t)| > |B(t)|$. This means that $\mathbb{I}\{|A(t)| \leq |B(t)|\} = 1$ for all $t \in \{0, \ldots, T-1\}$. Therefore, we have $\sum_{t=0}^{T-1}\mathbb{I}\{|A(t)| \leq |B(t)|\} = T$.

Analysis of P_6: Starting from any $(|A(0)|, |B(0)|)$ in P_6, the values of $|A|$ decreases until it becomes smaller than $|B(0)|$. Suppose that this happens at iteration t', that is, $|A(t')| < |B(0)|$. Starting from $(|A(t')|, |B(t')|)$ in P_5, $|A|$ starts to decrease by $\frac{\tilde{\sigma}_0^2\eta}{3}$ and $|B|$ starts to decrease by $\tilde{\sigma}_1^2\eta$. Since $\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}^2} < \frac{1}{3}$, this means that $(|A(t)|, |B(t)|)$ stays in P_5 for $t \in \{t', \ldots, T-2\}$, until it goes to P_1 when $|A(T-1)| < \tilde{\sigma}_0^2\eta$. Therefore, we know that the total change in $|A|$ since t'-th iteration is $|A(t')| - \tilde{\sigma}_0^2\eta = |B(0)| - \tilde{\sigma}_0^2\eta$. Since $|A|$ can only be updated by the amount of $\frac{\tilde{\sigma}_0^2\eta}{3}$ in P_5, we have $\sum_{t=0}^{T-1}\mathbb{I}\{|A(t)| \leq |B(t)|\} = \frac{|B(0)|-\tilde{\sigma}_0^2\eta}{\frac{1}{3}\tilde{\sigma}_0^2\eta}$. □

We now use this analysis on the behavior of $\sum_{t=0}^{T-1}\mathbb{I}\{|A(t)| \leq |B(t)|\}$ to compute $|\Delta\tilde{w}_2|$, which plays a role in the adversarial risk of signGD, as shown in (4.25). For the initial values of $(|A|, |B|)$ to be in P_1 and P_2, the initial errors must be small. However, consider a dataset with a strong task-relevant correlation between the relevant frequency component of the data and the target, a realistic scenario as we discussed in Section 4.3.2. In this case, $|\tilde{w}_0^*|$ and $|\tilde{w}_1^*|$ can be large. Additionally, with a weight initialization around zero, such as in methods by He et al., 2015 and Glorot et al., 2010,

the initial error $|\tilde{e}_0(0)|$ and $|\tilde{e}_1(0)|$ can be large and close to $|\tilde{w}_0^*|$ and $|\tilde{w}_1^*|$ when $|\tilde{w}_0^*| \gg |\tilde{w}_0(0)|$ and $|\tilde{w}_1^*| \gg |\tilde{w}_1(0)|$. Because of this, it is less likely for the initial values of $|A(0)|$ and $|B(0)|$ to be in the P_1 partition in Proposition 4.D.7.

Moreover, it is discussed in Section 4.3.1 that the distribution of spectral energy heavily concentrates at the low end of the frequency spectrum and decays quickly towards higher frequencies. Since $\tilde{\sigma}_i^2$ is interpreted as the expected energy of a random variable at the i-th frequency, it is reasonable to expect that $\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2} < \frac{1}{3}$ and this allows us to further narrow down to initialization of $(|A|, |B|)$ in P_5 and P_6.

The proportional relationship between the size of (4.27) and the magnitude of $|\tilde{e}_0|$ and $|\tilde{e}_1|$ when $\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2} < \frac{1}{3}$ and $(|A|, |B|)$ is initialized in P_5 or P_6 can be described in the following proposition.

Proposition 4.D.8. *Suppose that the ratio between $\tilde{\sigma}_0^2$ and $\tilde{\sigma}_1^2$ satisfies $\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2} < \frac{1}{3}$. The magnitude of $|\Delta\tilde{w}_2|$ depends on the initial values of $|\tilde{e}_0|$ and $|\tilde{e}_1|$, and the resulting $|A(0)|$ and $|B(0)|$. Specifically, we have*

$$|\Delta\tilde{w}_2| = \begin{cases} \sqrt{3}C\,|\tilde{e}_0(0)| & \text{if} \quad |A(0)| < |B(0)| \\ \frac{3\sqrt{2}\tilde{\sigma}_1^2}{2\tilde{\sigma}_0^2} C\,|\tilde{e}_1(0)| & \text{if} \quad |A(0)| > |B(0)|, \end{cases} \tag{4.28}$$

where $C \in [\frac{\sqrt{6}}{6}, \frac{\sqrt{6}}{3}]$ and we neglect the contribution from η.

Proof. From Proposition 4.D.7, under the assumption that $\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2} < \frac{1}{3}$, we have $\sum_{t=0}^{T-1} \mathbb{I}\{|A(t)| \le |B(t)|\} = T$ when $(|A(0)|, |B(0)|) \in P_5$, and this means that $|\Delta\tilde{w}_2| = C\eta T$ from (4.27). This also implies that for $t \in \{0, \dots, T-1\}$, we have $|A(t)| < |B(t)|$ and $|A(t)| = |A(0)| - \frac{t}{3}\tilde{\sigma}_0^2\eta$.

Since T is defined as the number of iteration required to reduce $|A(0)|$ to $\tilde{\sigma}_0^2\eta$, T is $\frac{|A(0)| - \tilde{\sigma}_0^2\eta}{\frac{1}{3}\tilde{\sigma}_0^2\eta}$, and we have

$$|\Delta\tilde{w}_2| = C\eta T = C\eta \frac{|A(0)| - \tilde{\sigma}_0^2\eta}{\frac{1}{3}\tilde{\sigma}_0^2\eta} = 3C \frac{\frac{\sqrt{3}}{3}\tilde{\sigma}_0^2\,|\tilde{e}_0(0)| - \tilde{\sigma}_0^2\eta}{\tilde{\sigma}_0^2} = C(\sqrt{3}\,|\tilde{e}_0(0)| - 3\eta).$$

From Proposition 4.D.7, when $\frac{\tilde{\sigma}_1^2}{\tilde{\sigma}_0^2} < \frac{1}{3}$ and $(|A(0)|, |B(0)|)$ is in P_6, we have

$$|\Delta\tilde{w}_2| = C\eta \frac{|B(0)| - \tilde{\sigma}_0^2\eta}{\frac{1}{3}\tilde{\sigma}_0^2\eta} = 3C \frac{\frac{\sqrt{2}}{2}\tilde{\sigma}_1^2\,|\tilde{e}_1(0)| - \tilde{\sigma}_0^2\eta}{\tilde{\sigma}_0^2} = C(\frac{3\sqrt{2}\tilde{\sigma}_1^2}{2\tilde{\sigma}_0^2}\,|\tilde{e}_1(0)| - 3\eta).$$

□

Since the initial error $|\tilde{e}_0(0)|$ and $|\tilde{e}_1(0)|$ are close to $|\tilde{w}_0^*|$ and $|\tilde{w}_1^*|$, (4.28) can be written as

$$|\Delta\tilde{w}_2| \approx \begin{cases} \sqrt{3}C\,|\tilde{w}_0^*| & \text{if} \quad |A(0)| < |B(0)| \\ \frac{3\sqrt{2}\tilde{\sigma}_1^2}{2\tilde{\sigma}_0^2} C\,|\tilde{w}_1^*| & \text{if} \quad |A(0)| > |B(0)| \end{cases} \tag{4.29}$$

Now we can consider the ratio between the adversarial risk of the standard risk minimizers found by GD (4.26) and signGD (4.25) with a three-dimensional input space. We observe that the solution found by signGD is more sensitive to perturbations compared to the GD solution:

$$\frac{\mathcal{R}_{\mathrm{a}}(\tilde{\boldsymbol{w}}^{\mathrm{signGD}})}{\mathcal{R}_{\mathrm{a}}(\tilde{\boldsymbol{w}}^{\mathrm{GD}})} = \frac{\tilde{w}_0^{*2} + \tilde{w}_1^{*2} + (\tilde{w}_2(0) + \Delta\tilde{w}_2)^2}{\tilde{w}_0^{*2} + \tilde{w}_1^{*2} + \tilde{w}_2^2(0)} \approx 1 + \frac{\Delta\tilde{w}_2^2}{\tilde{w}_0^{*2} + \tilde{w}_1^{*2}},$$

where we neglect the contribution from $\tilde{w}_2(0)$ in the approximation since we have assumed that the values of $|\tilde{w}_0^*|$ and $|\tilde{w}_1^*|$ are large compared to the initialized weight $|\tilde{w}(0)_2|$. This leads to

$$\frac{\mathcal{R}_{\mathrm{a}}(\tilde{\boldsymbol{w}}^{\mathrm{signGD}})}{\mathcal{R}_{\mathrm{a}}(\tilde{\boldsymbol{w}}^{\mathrm{GD}})} \approx \begin{cases} 1 + C_3 \frac{\tilde{w}_0^{*2}}{\tilde{w}_0^{*2}+\tilde{w}_1^{*2}} & \text{if} \quad |A(0)| < |B(0)| \\ 1 + C_4 \frac{\tilde{w}_1^{*2}}{\tilde{w}_0^{*2}+\tilde{w}_1^{*2}} & \text{if} \quad |A(0)| > |B(0)| \end{cases},$$

where $\frac{1}{2} \leq C_3 \leq 2$ and $\frac{3}{4}\frac{\tilde{\sigma}_1^4}{\tilde{\sigma}_0^4} \leq C_4 \leq 3\frac{\tilde{\sigma}_1^4}{\tilde{\sigma}_0^4}$.

4.D.3 From Irrelevant Frequencies to Spatially Redundant Dimensions

We have demonstrated that when the use of irrelevant frequency is under-constrained, optimizing the standard training objective can lead to solutions with zero standard risk but are sensitive to perturbations. This section offers a spatial interpretation of the findings, where we illustrate that signals with irrelevant frequencies contain spatially redundant dimensions when transformed into the spatial domain. Both interpretations can be used to explain the vulnerability of the solutions.

To illustrate the concept of redundancy in the spatial domain, consider the synthetic dataset with the distribution defined in Section 4.4.2 and the data has a structure of $\left\{(\tilde{X}_0, \tilde{X}_1, 0)\right\}$ in the frequency domain. Taking the DCT transformation of $\tilde{X}$, we see that the spatial representation of the same dataset is

$$\left\{(\sqrt{\frac{1}{3}}\tilde{X}_0 + \sqrt{\frac{1}{2}}\tilde{X}_1, \sqrt{\frac{1}{3}}\tilde{X}_0, \sqrt{\frac{1}{3}}\tilde{X}_0 - \sqrt{\frac{1}{2}}\tilde{X}_1)\right\},$$

where $\tilde{X}_0$ and $\tilde{X}_1$ are random variables with frequency interpretations. In the spatial domain, redundancy refers to the existence of dimensions that are highly correlated with each other. The example mentioned above illustrates that the presence of a single irrelevant frequency in the data distribution corresponds to the existence of one redundant dimension in the spatial domain. Specifically, within this three-dimensional dataset, it is possible to express any dimension as a linear combination of the values at the other two dimensions.

This translation between spectral irrelevance and spatial redundancy can also be observed in the learned weight. Consider a standard risk minimizer $\tilde{w}^* = (\tilde{w}_0^*, \tilde{w}_1^*, 0)$, whose frequency-domain representation is

$$w^* = (\sqrt{\frac{1}{3}}w_0^* + \sqrt{\frac{1}{2}}w_1^*, \sqrt{\frac{1}{3}}w_0^*, \sqrt{\frac{1}{3}}w_0^* - \sqrt{\frac{1}{2}}w_1^*).$$

Because of the irrelevance from $\tilde{w}_2$, there are multiple other standard risk minimizers. In the spatial domain, this means $w^* + \tilde{w}_2\vec{w}_2$ with $\vec{w}_2 = (\sqrt{\frac{1}{6}}, -\sqrt{\frac{2}{3}}, \sqrt{\frac{1}{6}})$ and any choice of $\tilde{w}_2 \in \mathbb{R}$ is still a valid standard risk minimizer.[4] When the model trained by signGD has a large weight at $\tilde{w}_2$, this implies a large $\tilde{w}_2$ for the weight in the spatial domain. Because $\vec{w}_2$ and w^* are orthogonal, we have $\|w^* + \tilde{w}_2\vec{w}_2\|_2 = \|w^*\|_2 + |\tilde{w}_2|$, therefore, the weight norm increases as $\tilde{w}_2$ gets large, and from (4.10), models are more vulnerable.

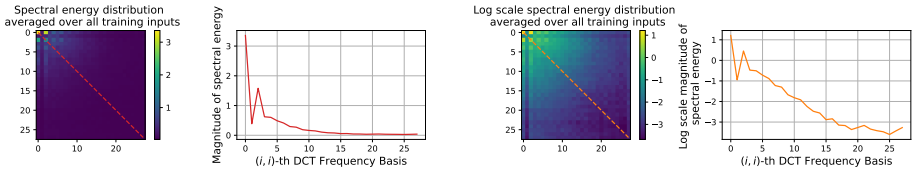
It is important to realize that having irrelevant frequencies is merely a sufficient condition for having spatially redundant features, but is not a necessary condition. For example, rearranging the dimensions of x and w^* in the above example still preserves the spatial redundancy in the dataset,

[4]The $(\sqrt{\frac{1}{6}}, -\sqrt{\frac{2}{3}}, \sqrt{\frac{1}{6}})$ vector is the DCT basis for the $\tilde{w}_2$ term, i.e., $C^\top(0, 0, \tilde{w}_2) = \tilde{w}_2(\sqrt{\frac{1}{6}}, -\sqrt{\frac{2}{3}}, \sqrt{\frac{1}{6}})$.

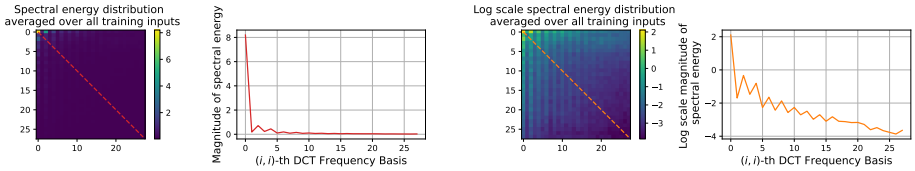
and there are still infinitely many standard risk minimizers. However, it no longer guarantees zero entries in $\tilde{x}$ and $\tilde{w}^*$.

4.E Additional Figures

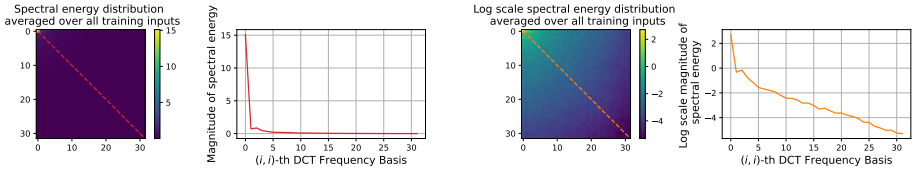
In Figure 4.2, we visualize the energy distribution for CIFAR-100 and Imagenette, with each dataset illustrated through four plots. The first plot's (i, j) coordinate indicates the average amplitude, $\frac{1}{N}\sum_{n=1}^{N}|\tilde{x}_{n;(i,j)}|$, for the (i, j)-th basis across all N training images, where $\tilde{x}_n$ represents the DCT transformation of the n-th image, x_n, and $\tilde{x}_{n;(i,j)}$ represents the amplitude of the (i, j)-th basis in the n-th sample. The second plot focuses on the diagonal elements of the first, showing the average amplitude values, $\left\{\frac{1}{N}\sum_{n=1}^{N}|\tilde{x}_{n;(i,i)}|\right\}_{i=0,\ldots,d-1}$. The two plots are then drawn on a logarithmic scale to highlight the pronounced concentration of energy around the low-frequency harmonics, while the amplitudes for higher-frequency harmonics diminish significantly.



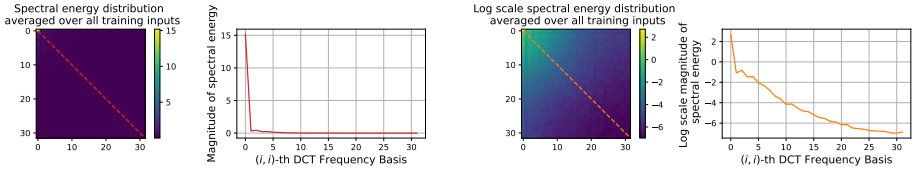
a. MNIST



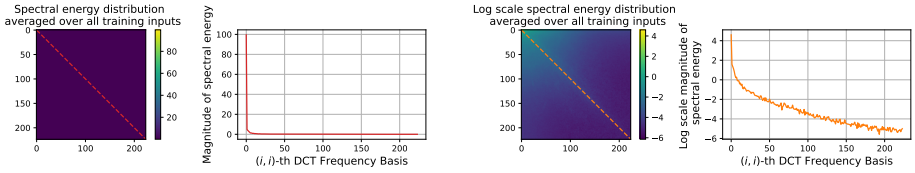
b. Fashion-MNIST



c. CIFAR-10

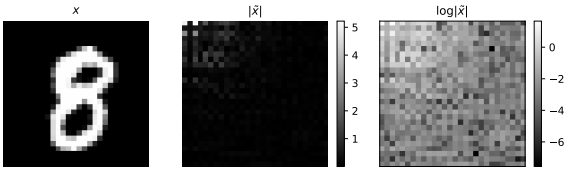


d. SVHN

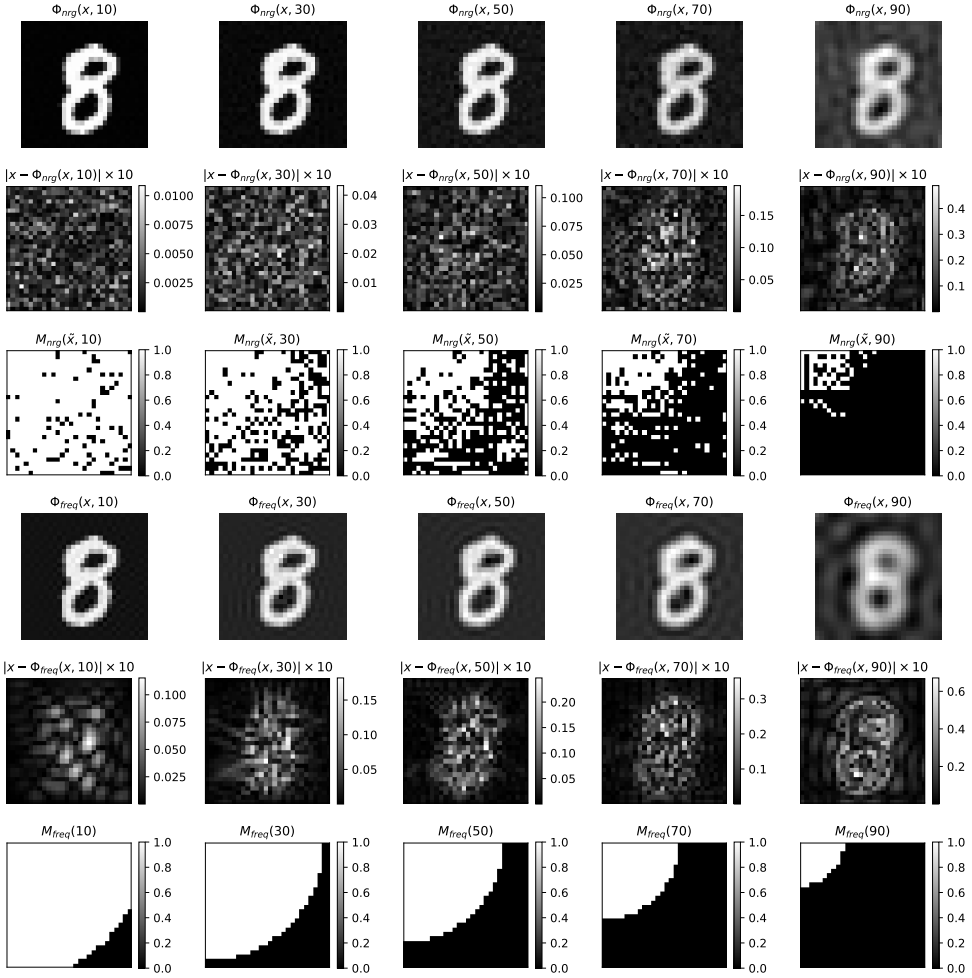


e. Caltech-101

Figure 4.10: **Illustration of the spectral energy distribution in natural data.** Distribution of the spectral energy heavily concentrates at low frequencies and decays exponentially towards higher frequencies.

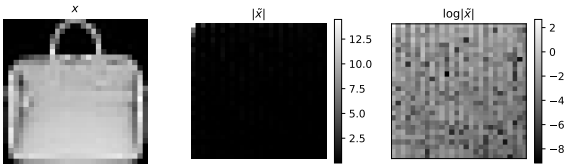


a. Original Image.

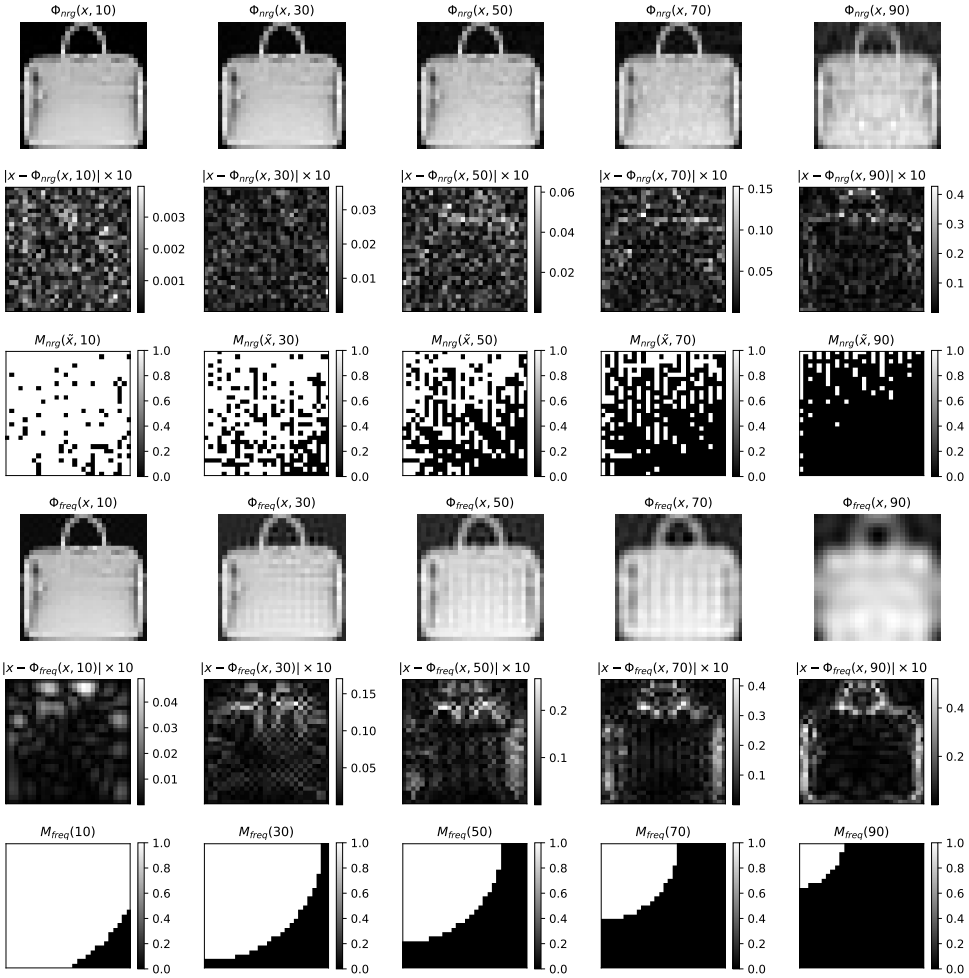


b. Modified Image.

Figure 4.11: **Examples of modified images used in Observation I. (MNIST)** We use a threshold value of $\texttt{threshold} = \{10, 30, 50, 70, 90\}$ to modify images based on its magnitude of DCT basis and their frequency basis. In a), we show the original image x and the magnitude of its DCT basis $|\tilde{x}|$ in both linear and log scale. In b), we show images modified by removing DCT basis vectors whose magnitudes are in the bottom `threshold` percentage (row 1), the differences between the modified images and the original image (row 2), the binary mask used to remove the DCT basis: black means removed (row 3), images modified by removing high-frequency DCT basis vectors (row 4), the differences between the modified images and the original image (row 5) and the binary mask used to remove the DCT basis: black means removed (row 6). Notice that the masks in row 6 only depends on the dimension of the images, whereas the masks in row 3 differs from images to images.

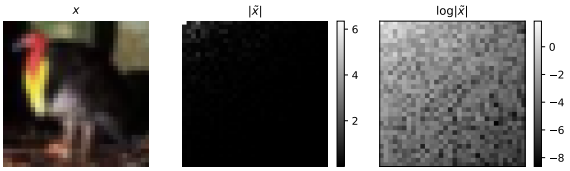


a. Original Image.

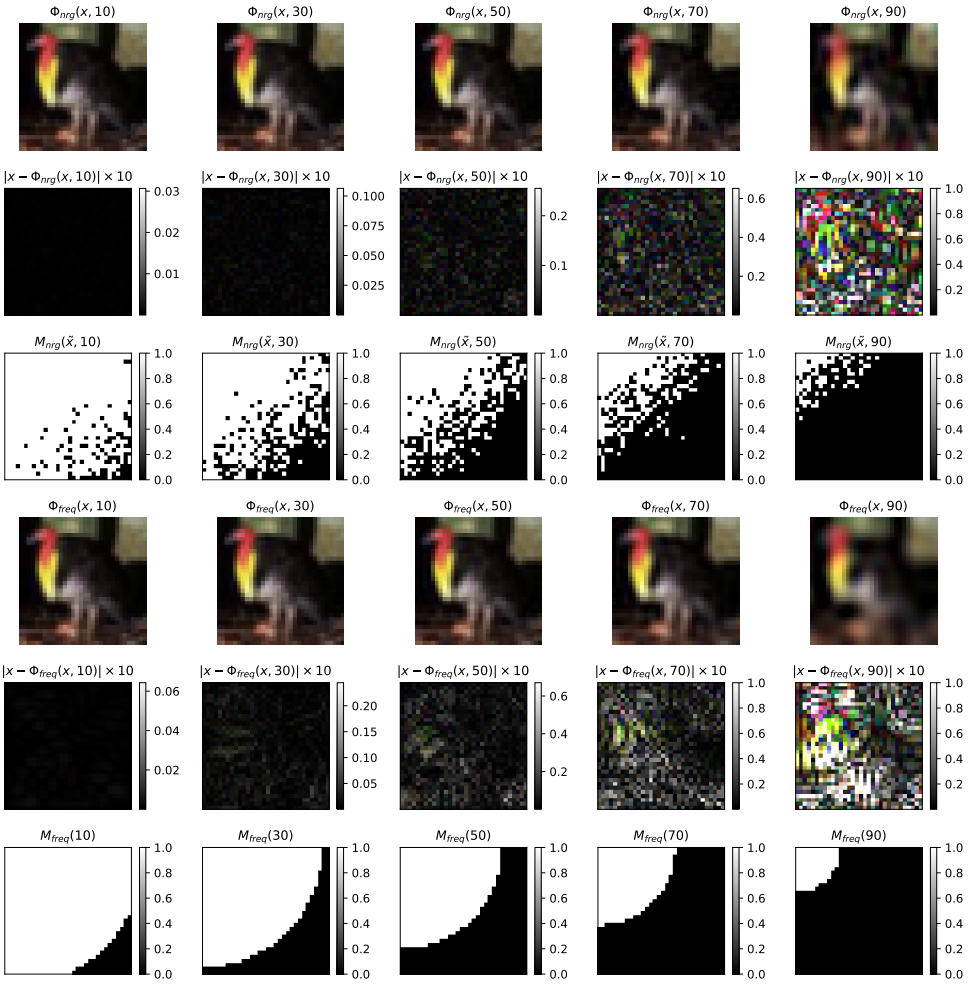


b. Modified Image.

Figure 4.12: **Examples of modified images used in Observation I. (Fashion-MNIST)** We use a threshold value of `threshold` $= \{10, 30, 50, 70, 90\}$ to modify images based on its magnitude of DCT basis and their frequency basis. In a), we show the original image x and the magnitude of its DCT basis $|\tilde{x}|$ in both linear and log scale. In b), we show images modified by removing DCT basis vectors whose magnitudes are in the bottom `threshold` percentage (row 1), the differences between the modified images and the original image (row 2), the binary mask used to remove the DCT basis: black means removed (row 3), images modified by removing high-frequency DCT basis vectors (row 4), the differences between the modified images and the original image (row 5) and the binary mask used to remove the DCT basis: black means removed (row 6). Notice that the masks in row 6 only depends on the dimension of the images, whereas the masks in row 3 differs from images to images.

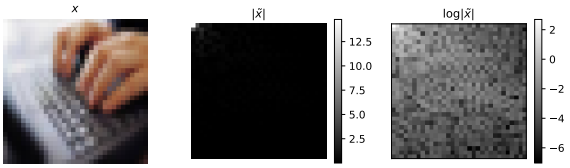


a. Original Image.

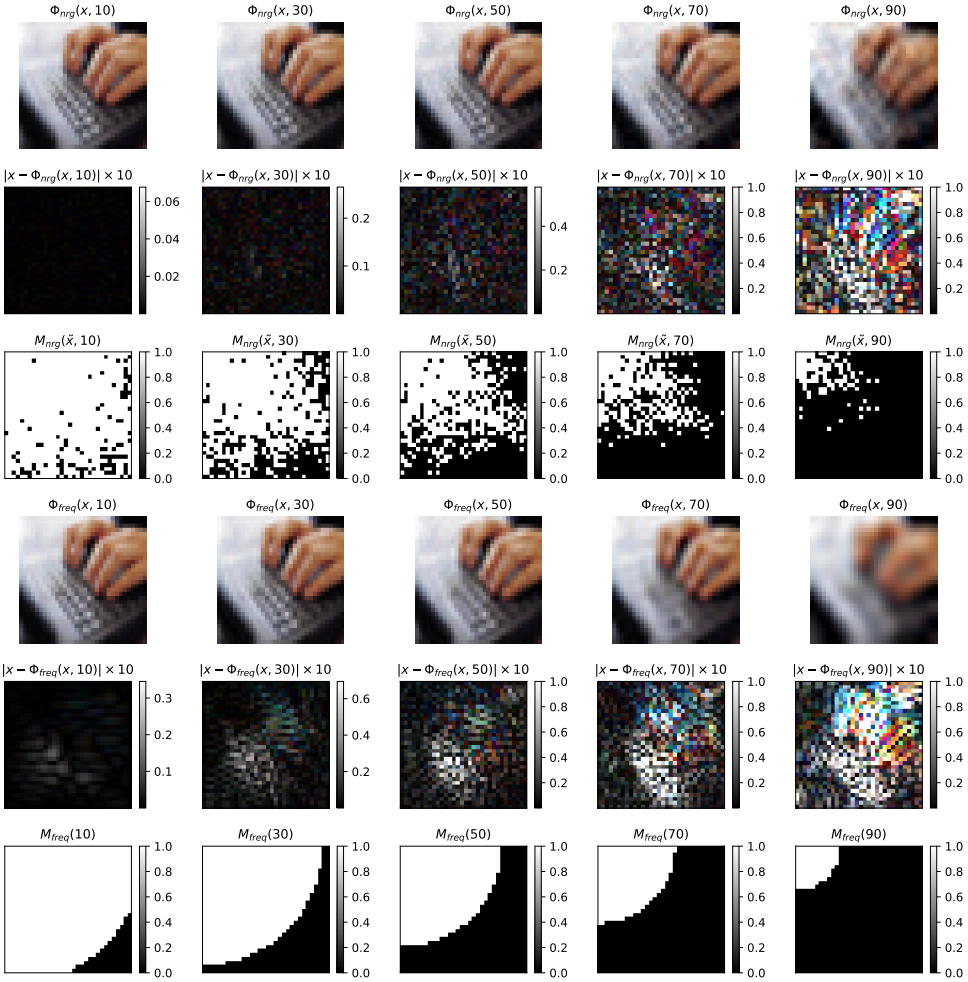


b. Modified Image.

Figure 4.13: **Examples of modified images used in Observation I. (CIFAR-10)** We use a threshold value of $\texttt{threshold} = \{10, 30, 50, 70, 90\}$ to modify images based on its magnitude of DCT basis and their frequency basis. In a), we show the original image x and the magnitude of its DCT basis $|\tilde{x}|$ in both linear and log scale. In b), we show images modified by removing DCT basis vectors whose magnitudes are in the bottom `threshold` percentage (row 1), the differences between the modified images and the original image (row 2), the binary mask used to remove the DCT basis: black means removed (row 3), images modified by removing high-frequency DCT basis vectors (row 4), the differences between the modified images and the original image (row 5) and the binary mask used to remove the DCT basis: black means removed (row 6). Notice that the masks in row 6 only depends on the dimension of the images, whereas the masks in row 3 differs from images to images.

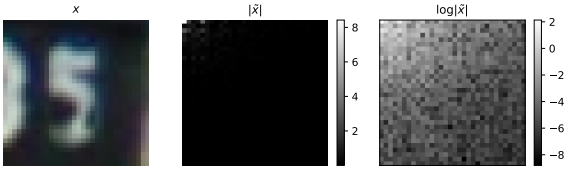


a. Original Image.

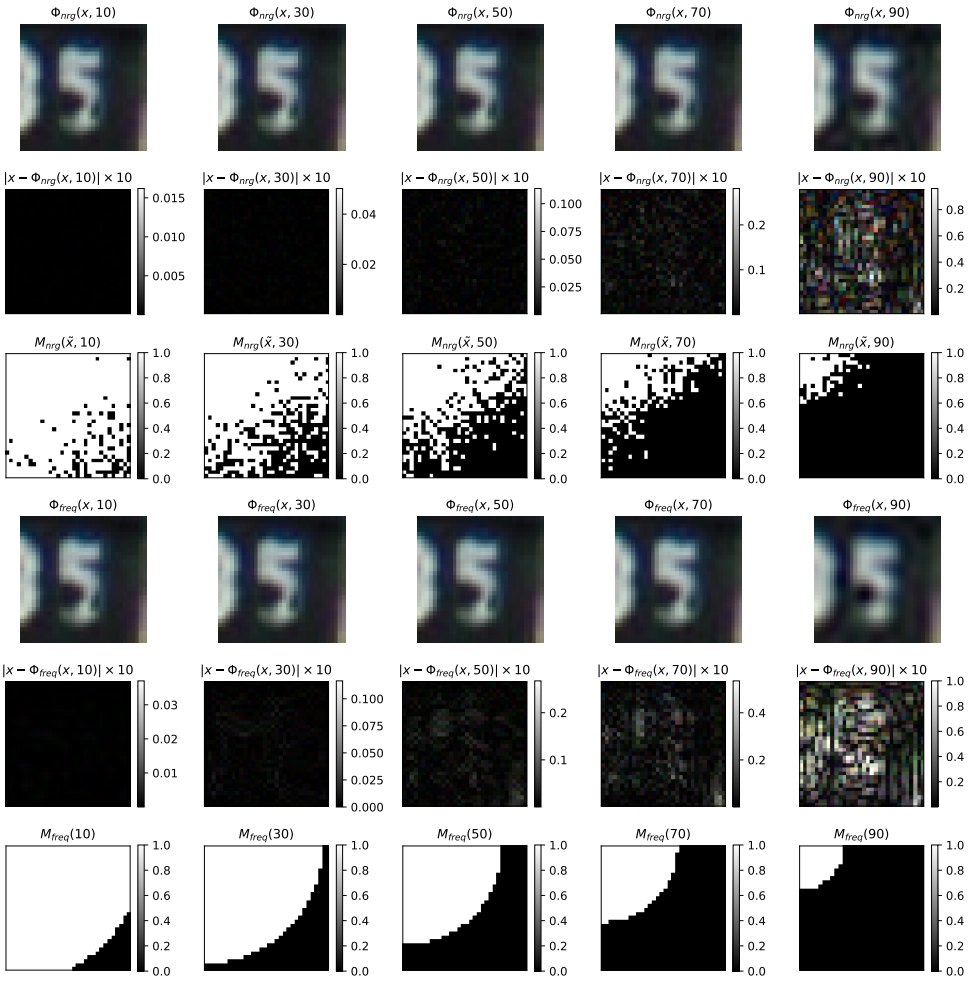


b. Modified Image.

Figure 4.14: **Examples of modified images used in Observation I. (CIFAR-100)** We use a threshold value of `threshold` $= \{10, 30, 50, 70, 90\}$ to modify images based on its magnitude of DCT basis and their frequency basis. In a), we show the original image x and the magnitude of its DCT basis $|\tilde{x}|$ in both linear and log scale. In b), we show images modified by removing DCT basis vectors whose magnitudes are in the bottom `threshold` percentage (row 1), the differences between the modified images and the original image (row 2), the binary mask used to remove the DCT basis: black means removed (row 3), images modified by removing high-frequency DCT basis vectors (row 4), the differences between the modified images and the original image (row 5) and the binary mask used to remove the DCT basis: black means removed (row 6). Notice that the masks in row 6 only depends on the dimension of the images, whereas the masks in row 3 differs from images to images.

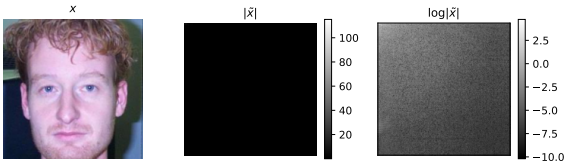


a. Original Image.

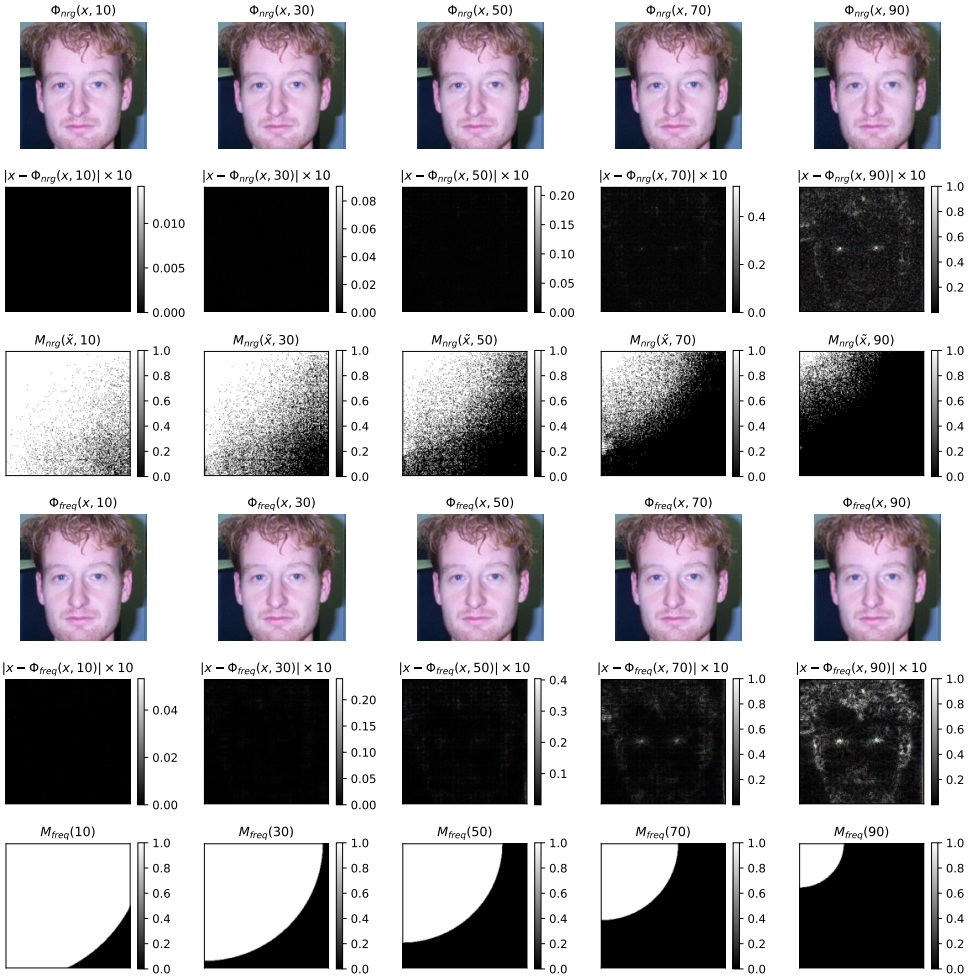


b. Modified Image.

Figure 4.15: **Examples of modified images used in Observation I. (SVHN)** We use a threshold value of $\texttt{threshold} = \{10, 30, 50, 70, 90\}$ to modify images based on its magnitude of DCT basis and their frequency basis. In a), we show the original image x and the magnitude of its DCT basis $|\tilde{x}|$ in both linear and log scale. In b), we show images modified by removing DCT basis vectors whose magnitudes are in the bottom $\texttt{threshold}$ percentage (row 1), the differences between the modified images and the original image (row 2), the binary mask used to remove the DCT basis: black means removed (row 3), images modified by removing high-frequency DCT basis vectors (row 4), the differences between the modified images and the original image (row 5) and the binary mask used to remove the DCT basis: black means removed (row 6). Notice that the masks in row 6 only depends on the dimension of the images, whereas the masks in row 3 differs from images to images.



a. Original Image.



b. Modified Image.

Figure 4.16: **Examples of modified images used in Observation I. (Caltech-101)** We use a threshold value of `threshold` $= \{10, 30, 50, 70, 90\}$ to modify images based on its magnitude of DCT basis and their frequency basis. In a), we show the original image x and the magnitude of its DCT basis $|\tilde{x}|$ in both linear and log scale. In b), we show images modified by removing DCT basis vectors whose magnitudes are in the bottom `threshold` percentage (row 1), the differences between the modified images and the original image (row 2), the binary mask used to remove the DCT basis: black means removed (row 3), images modified by removing high-frequency DCT basis vectors (row 4), the differences between the modified images and the original image (row 5) and the binary mask used to remove the DCT basis: black means removed (row 6). Notice that the masks in row 6 only depends on the dimension of the images, whereas the masks in row 3 differs from images to images.

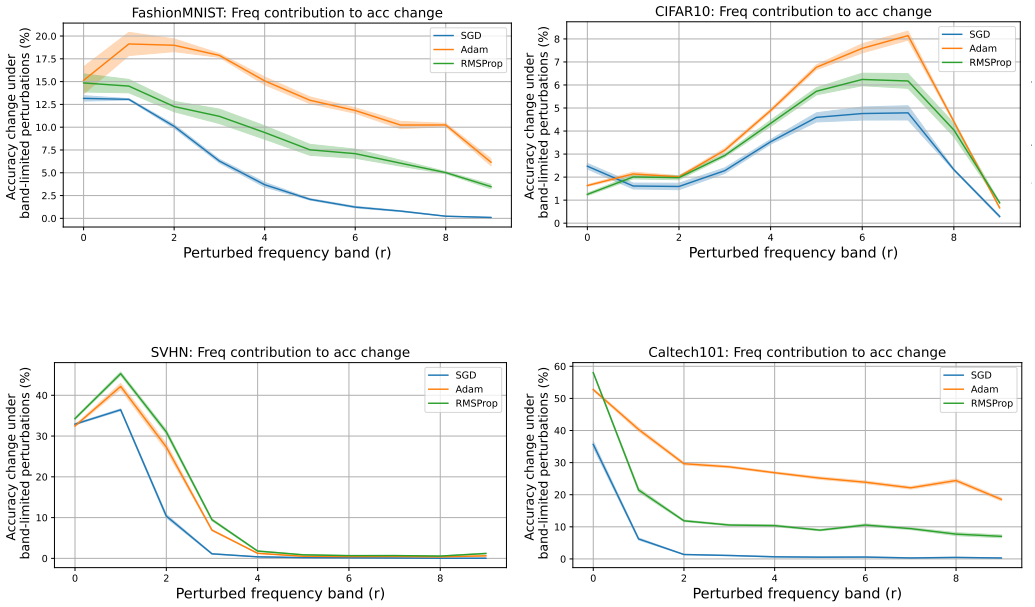


Figure 4.17: **The effect of band-limited Gaussian perturbations on the model (additional figures).** Perturbations from the lowest band, i.e., $\Delta x^{(0)}$, have a similar effect on all the models, despite being trained by different algorithms and exhibiting different robustness properties. On the other hand, models' responses vary significantly when the perturbation focuses on higher frequency bands.

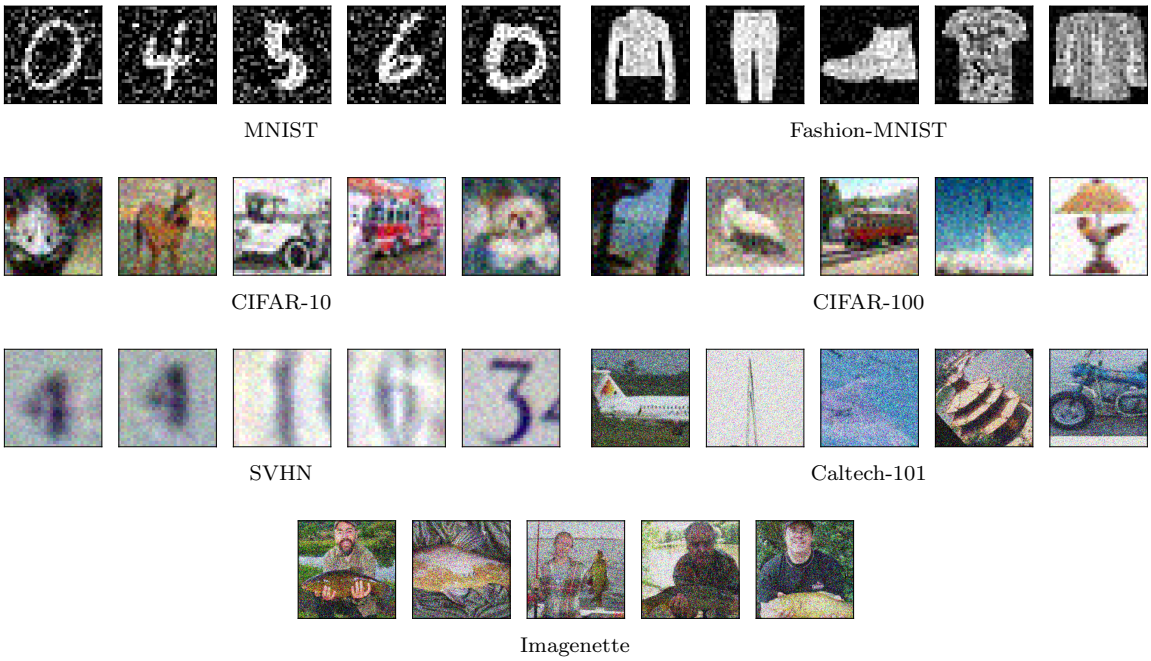


Figure 4.18: **Images perturbed by additive Gaussian white noise with different variance.** For each dataset, we select the largest variance value from Table 4.3.

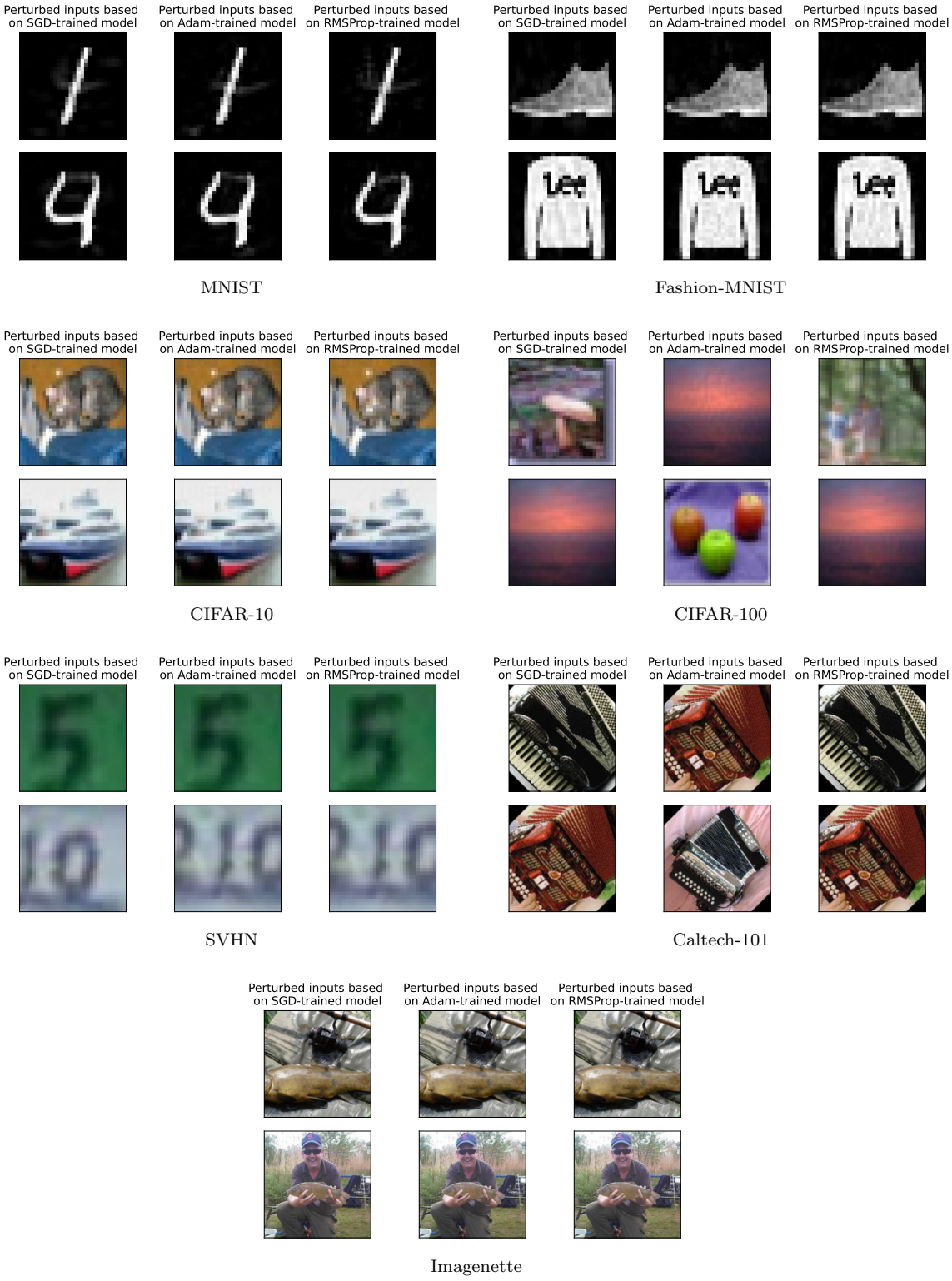


Figure 4.19: **Images perturbed by ℓ_2-norm bounded adversarial perturbation (Croce et al., 2020).** We select the largest ϵ value from Table 4.3 to generate ℓ_2 bounded perturbations for images in each dataset. We also compare perturbations generated using models trained by different algorithms.

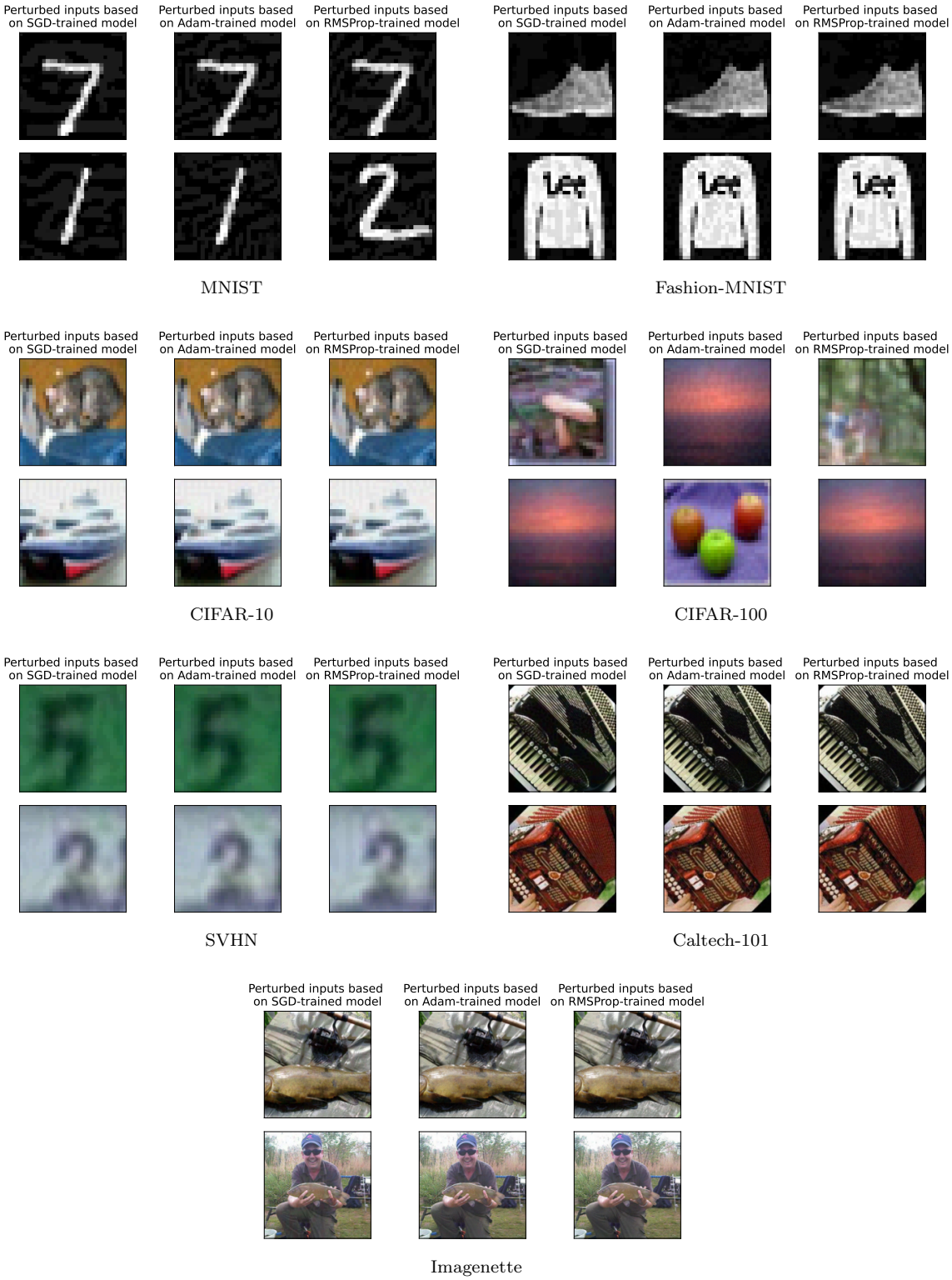


Figure 4.20: **Images perturbed by ℓ_∞-norm bounded adversarial perturbation (Croce et al., 2020).** We select the largest ϵ value from Table 4.3 to generate ℓ_∞ bounded perturbations for images in each dataset. We also compare perturbations generated using models trained by different algorithms.

Chapter 5

Understanding and Improving the Hierarchical Adversarial Robustness of DNNs

5.1 Introduction

In the previous chapter, we demonstrated that dataset properties play a crucial role in shaping model robustness during training by interacting with the optimization process. In this chapter, we extend this idea to the inference stage, showing that dataset properties can also influence the perceived robustness of models.

As datasets grow in size and complexity, their hierarchical structure and the relationships between classes become increasingly significant (Krizhevsky et al., 2012). In evaluating adversarial robustness, the primary metric is often the model's classification accuracy on perturbed inputs (Szegedy et al., 2014). However, a key aspect often overlooked in the current literature is that while adversarial examples can be harmful, not all misclassifications have equal consequences. The current paradigm assumes that all errors caused by attacks are equally problematic, but in many contexts, this is not the case. For example, misclassifying a pedestrian as a shrub is significantly more problematic than misclassifying a traffic cone as a shrub. While both instances technically involve misclassification, the former error carries far more severe consequences, potentially leading to harm or fatal accidents.

To address this, we introduce the concept of hierarchical adversarial robustness, which accounts for variations in the impact of misclassification. Many real-world datasets exhibit a natural hierarchy in their class structure, where fine-grained categories (which we call **leaf-**classes) can be grouped into broader **meta-**classes. For example, in an autonomous driving dataset (Geiger et al., 2013; Cordts et al., 2016), leaf classes might include pedestrians, bicycles, and cars, all belonging to a meta-class representing non-stationary objects. We define hierarchical adversarial examples as those causing misclassification within the same meta-class, such as mistaking a pedestrian for an automobile.

In this chapter, we explore the hierarchical adversarial robustness of deep neural networks (DNNs) from both attack and defense perspectives. On the attack side, we demonstrate that most PGD-

This chapter is based on our work in Improving Hierarchical Adversarial Robustness of Deep Neural Networks.

perturbed inputs remain correctly classified at the meta-class level, highlighting the need for tailored attacks. To this end, we propose a hierarchical attack algorithm specifically designed to generate such examples. On the defense side, we introduce a Hierarchical Adversarially Robust (HAR) network to improve robustness. The HAR network splits the traditional end-to-end learning task into a primary meta-class classification task, complemented by several leaf-class classification tasks. Each task is trained independently with adversarial defenses, using only data relevant to its respective group. Our empirical results show that this ensemble-based architecture significantly enhances hierarchical adversarial robustness compared to standard approaches.

5.1.1 Contributions

Through this work, we make the following contributions:

- We introduce the concept of hierarchical adversarial examples: a special case of the standard adversarial examples which causes mistakes at the meta-class level (Section 5.2).
- Our result shows that perturbations generated by PGD result in misclassifications that mostly still fall within the same meta-class level, meaning that they are not effective in degrading models' hierarchical adversarial robustness (Section 5.3). This finding motivates us to develop an attack method to generate hierarchical adversarial examples (Section 5.4).
- We demonstrate that the proposed hierarchical attack provides a more accurate empirical representation of the hierarchical adversarial robustness of models (Section 5.6).
- We propose an ensemble-based approach, termed the Hierarchical Adversarially Robust (HAR) network, to improve the hierarchical adversarial robustness of DNNs (Section 5.5).
- Our results demonstrate that HAR networks exhibit significantly improved robustness against both ℓ_∞ and ℓ_2 perturbations on the CIFAR-10 and CIFAR-100 datasets (Section 5.6).

5.2 Hierarchical Adversarial Robustness

An important development accompanying the advancement of DNN is increasing complexity of datasets, particularly in terms of the number of classes. For example, we have progressed from the 10-class MNIST dataset to the 1000-class ImageNet dataset. As dataset complexity grows, often these datasets can be logically divided into several meta-classes, each encompassing multiple leaf classes. Note that we use the term label and class interchangeably. In Table 5.1, we illustrate the hierarchical structure of classes in both CIFAR-10 and CIFAR-100. In our experiments, we also consider a medium-sized subset of CIFAR-100, designated as CIFAR-5x5, consisting of 25 leaf classes categorized into 5 meta-classes.

Flat classification, such as the standard multi-class classification task, is the most common form of classification in machine learning. In this setting, the goal is to predict a single label from a set of mutually exclusive classes. Hierarchical classification, on the other hand, involves a two-stage process where an input image is first categorized into meta-classes and subsequently into leaf classes. For a detailed discussion on hierarchical classification, we refer the reader to the survey by Silla et al. (2011), in which the terms "meta-class" and "leaf class" are also introduced. More recent work leveraging the hierarchical structure of datasets falls under the domain of extreme

Table 5.1: **Hierarchical structure of classes within the CIFAR-10 and CIFAR-100 dataset.** We examine predefined class hierarchies in the CIFAR-10 and CIFAR-100 datasets. In addition, we analyze a medium-sized subset of CIFAR-100, designated as CIFAR-5x5, consisting of 25 leaf classes categorized into 5 meta-classes.

	Meta-class	Leaf class
CIFAR-10	Animals	bird, cat, deer, dog, frog, horse
	Vehicles	airplane, automobile, ship, truck
CIFAR-5x5	Fish	aquarium fish, flatfish, ray, shark, trout
	Vehicles	bicycle, bus, motorcycle, pickup truck, train
	People	baby, boy, girl, man, woman
	Trees	maple, oak, palm, pine, willow
	Insects	bee, beetle, butterfly, caterpillar, cockroach
CIFAR-100	Aquatic mammals	beaver, dolphin, otter, seal, whale
	Fish	aquarium fish, flatfish, ray, shark, trout
	Flowers	orchids, poppies, roses, sunflowers, tulips
	Food containers	bottles, bowls, cans, cups, plates
	Fruit and vegetables	apples, mushrooms, oranges, pears, sweet peppers
	Household electrical devices	clock, computer keyboard, lamp, telephone, television
	Household furniture	bed, chair, couch, table, wardrobe
	Insects	bee, beetle, butterfly, caterpillar, cockroach
	Large carnivores	bear, leopard, lion, tiger, wolf
	Large man-made outdoor things	bridge, castle, house, road, skyscraper
	Large natural outdoor scenes	cloud, forest, mountain, plain, sea
	Large omnivores and herbivores	camel, cattle, chimpanzee, elephant, kangaroo
	Medium-sized mammals	fox, porcupine, possum, raccoon, skunk
	Non-insect invertebrates	crab, lobster, snail, spider, worm
	People	baby, boy, girl, man, woman
	Reptiles	crocodile, dinosaur, lizard, snake, turtle
	Small mammals	hamster, mouse, rabbit, shrew, squirrel
	Trees	maple, oak, palm, pine, willow
	Vehicles 1	bicycle, bus, motorcycle, pickup truck, train
	Vehicles 2	lawn-mower, rocket, streetcar, tank, tractor

classification (Buvanesh et al., 2022; Gupta et al., 2021), which focuses on classification problems with thousands or millions of labels (Bhatia et al., 2016).

Class hierarchies can be formed in different ways. One common approach is to assign meta-classes based on taxonomy, as seen in datasets like CIFAR-10 and CIFAR-100, where classes are grouped based on visual similarities. A more practical scenario is an application-driven meta-class design. For instance, in autonomous driving, meta-classes might distinguish between "objects a vehicle must avoid" and "objects not critical for collision avoidance". Alternatively, meta-classes can also be learned strategically to optimize the performance in a specific downstream task (Deng et al., 2011). A closely related study, Chandrasekaran et al. (2019), also examines robustness in a hierarchical setting, but its hierarchy is derived from feature-space clustering and the work focuses only on improving robustness. On the other hand, we investigate hierarchical robustness from both the adversary's and defender's perspectives, assuming a predefined class hierarchy.

Previous research on adversarial robustness, including both defenses or attacks, typically as-

sumes that all misclassifications are equally detrimental. However, this assumption fails to capture the varying degrees of impact that different misclassifications can have in practice. For example, in an autonomous driving system, mistaking a perturbed image of a traffic sign for a pedestrian poses a significant safety risk, whereas confusing a bus with a pickup truck might have negligible consequences for the system's operation. Moreover, misclassifications across different meta-classes can raise significant ethical concerns, particularly in datasets with hierarchical structures involving sensitive attributes such as ethnicity, gender, disability, and age groups.

Mistakes across meta-classes lead to much more severe consequences compared to mistakes within meta-classes. Given this distinction, we introduce the term *hierarchical adversarial examples*. These are a subset of adversarial examples characterized by misclassifications that occur between leaf classes originating from different meta-class.

Let us first define hierarchical adversarial examples to differentiate them from standard adversarial examples. Consider a neural network for classification $f(x) : \mathbb{R}^d \to \mathbb{R}^c$ with a softmax as its last layer, where d represents the input dimension and c the number of classes. The predicted class for a given input x is determined by $\arg\max_i f(x)_i$.

Consider a two-level hierarchical classification task where leaf classes are grouped into meta-classes. While this hierarchy can be extended to include higher-level meta-classes, we will focus on a simpler two-level structure for clarity and didactic purposes. The dataset is represented as $\{x, y, z\}_n$, where each entry includes an image x, its leaf class y, and corresponding meta-class z, with $y \in z$ indicating that leaf class y is part of meta-class z. In other words, each meta-class z is a set comprising all the leaf classes that belong to it. Using CIFAR-10 from Table 5.1 as an example, we have $z_{\text{animal}} = \{C_{\text{bird}}, C_{\text{cat}}, \ldots, C_{\text{horse}}\}$ as the animal meta-class and $z_{\text{vehicle}} = \{C_{\text{airplane}}, C_{\text{automobile}}, \ldots, C_{\text{truck}}\}$ as the vehicle meta-class. In this chapter, we focus on non-overlapping meta-classes for simplicity, i.e., $z_i \cap z_j = \emptyset$ for $i \neq j$. Given an input data x, let its ground-truth leaf and meta-class be y^* and z^* respectively. In this framework, a hierarchical adversarial example is defined by meeting all the following criteria.

- The unperturbed data x is correctly classified by the classifier: $\arg\max_i F(x)_i = y^*$.
- The perturbed data $x' = x + \delta$ is perceptually indistinguishable from the original data x.
- The perturbed data x' is incorrectly classified: $\arg\max_i F(x')_i = y'$ where $y' \neq y^*$.
- The incorrectly assigned label y' belongs to a different meta-class, i.e., $y' \notin z^*$.

Note that the first three criteria outlined suffice to define standard adversarial examples, while hierarchical adversarial examples represent a specific subset of all adversarial examples by having the fourth property. We also highlight that measuring perceptual distance can be challenging (Li et al., 2003), thus the second property is often replaced by limiting that the adversary can only modify any input x to $x + \delta$ with $\delta \in \Delta$, as previously discussed in Chapter 2. In this chapter, we focus on ℓ_∞- and ℓ_2-norm perturbations.

5.3 Untargeted PGD's Ineffectiveness in Degrading Hierarchical Adversarial Robustness

To motivate the development of a new attack algorithm for generating hierarchical adversarial examples, we begin with an empirical analysis of the errors made by models on both clean and perturbed

Table 5.2: **Percentage of the misclassified inputs that are still correctly classified at the meta-class level (%).** We calculate the percentage of inputs misclassified at the leaf level yet still correctly classified at the meta-class level. An important observation is that for images misclassified at the leaf class level, whether they are perturbed or unperturbed does not significantly change the likelihood of correct classification at the meta-class level. This shows that untargeted PGD perturbations are ineffective in degrading hierarchical robustness. All results presented in this chapter are obtained from the average of three independent runs.

Method	CIFAR-10		CIFAR-100	
	Clean	PGD20	Clean	PGD20
Standard	84.46	80.00	33.71	36.80
PGD10	83.75	75.56	35.57	39.06
TRADES	77.39	77.14	34.84	38.22

images. In Table 5.2, we show the conditional probability that an image is correctly at the meta-class level, given that it has already been misclassified at the leaf-class level:

$$\mathbb{P}(\text{correct meta-class}|\text{incorrect leaf class}),$$

where a larger value indicates that more mistakes occur **within** the same meta-class, suggesting greater hierarchical robustness in the model. We analyze both unperturbed inputs and those perturbed by untargeted ℓ_∞-norm constrained PGD20 perturbations with $\epsilon = \frac{8}{255}$. The evaluation includes three types of models: those trained with unperturbed data (Standard), those adversarially trained with PGD10 perturbations (Madry et al., 2018) (PGD10), and those trained using TRADES (Zhang et al., 2019) (TRADES).

Two important observations emerge from the results presented in Table 5.2. First, compared to the likelihood of being classified under a random meta-class, which stands at 50% for CIFAR-10 and 5% for CIFAR-100, misclassifications are more likely to be within the same meta-class. Second, for images misclassified at the leaf class level, whether they are perturbed or unperturbed does not significantly change the likelihood of correct classification at the meta-class level. For instance, in a TRADES-robustified model trained on CIFAR-10, approximately 77% of misclassified images, whether clean or PGD-perturbed, are still correctly classified at the meta-class level. This result still shows that given two misclassified clean and PGD-perturbed inputs, the chances of misclassification at the meta-class level is similar. These findings motivate us to develop a new approach for generating hierarchical adversarial perturbations.

5.4 Generating Hierarchical Adversarial Perturbations

In Section 2.2 and 2.3, we discussed FGSM, BIM and PGD attacks, focusing on formulations for their untargeted versions. In these algorithms, the specific class to which the perturbed input is misclassified is irrelevant, as long as it is different from the true class. The similarity across these attacks is their objective to find a perturbation that maximizes the loss function w.r.t. the true

label:

$$\arg\max_{\delta} \ell(x+\delta, y^*),$$

where y^* is the true label of the input image x.

Targeted attacks, on the other hand, are pertinent in scenarios where the specific misclassified class matters. For these attacks, the objective is:

$$\arg\min_{\delta} \ell(x+\delta, y),$$

where $y \neq y^*$ is the specified target label, different from the ground truth label y^*.

Results from the previous section show that untargeted PGD attacks are not effective at generating adversarial examples that cause misclassifications at the meta-class level. To overcome this limitation, we introduce a variant of the PGD algorithm specifically designed to find hierarchical adversarial examples. Let x'_t denote the perturbed input at iteration t, we define:

$$x'_{t+1} = \Pi_{B_\infty(x,\epsilon)} \{x'_t - \alpha \operatorname{sign}(\nabla_x \ell(x'_t, \hat{y}))\} \tag{5.1}$$

where $\hat{y}$ is a target class chosen from a different meta-class, i.e., $\hat{y} \notin z^*$. We use the same random initialization as used in the vanilla PGD: $x'_0 = \Pi_{B_\infty(x,\epsilon)}\{x + \epsilon\delta\}$, where each dimension of δ is independently sampled from a uniform distribution $\mathcal{U}(-1, 1)$.

An important factor in generating hierarchical adversarial examples is the choice of the target leaf class. We adopt the two target selection strategies proposed in Carlini et al. (2017). The first is an *average-case* approach, where the target class is randomly selected from all eligible leaf classes. This method does not require additional computational resources beyond those used for vanilla PGD, although it may not always result in successful attacks. In our work, we consider a *worst-case* approach that iterates through all candidate leaf classes as potential targets. This attack proceeds until it either achieves a successful misclassification or exhausts all candidate classes.

The procedure for generating ℓ_∞-norm constrained hierarchical adversarial examples is summarized in Algorithm 5.1. We see that lines 2–5 closely resemble the standard PGD algorithm, with the key difference being the iterative target class selection implemented through the for loop in line 1. The algorithm terminates either upon successfully generating a hierarchical adversarial example (line 11) or when all candidate leaf classes have been exhausted (line 7).

5.5 Hierarchical Adversarial Robust Network

The visual separability between meta-classes can be highly uneven, making some meta-classes more challenging to distinguish than others. This variation motivates the use of an ensemble of models in a "local classifier per node" approach, leveraging dedicated classifiers for specific leaf classes within each meta-class (Silla et al., 2011). The primary advantage of this approach is its use of local information to develop specialized classifiers for individual leaf classes within each meta-class.

Algorithm 5.1: Generating ℓ_∞-norm constrained hierarchical adversarial examples.

Input : A pair of input data (x, y^*), where the leaf class y^* belongs to the meta-class z^*; $S = \{y \mid y \notin z^*\}$, a set comprising all leaf classes that do not belong to the meta-class z^*; a model $F(\cdot)$; a loss function $\ell(\cdot)$; an ℓ_∞-norm constraint ϵ; the number of PGD iterations k; PGD step-size α.

for $\hat{y} \in S$ **do**
  $x'_0 \leftarrow \Pi_{B_\infty(x,\epsilon)}\{x + \epsilon\delta\}$, where $\delta \sim \mathcal{U}([-1, 1]^d)$
  for $t = 0, \ldots, k-1$ **do**
    $x'_{t+1} = \Pi_{B_\infty(x,\epsilon)}\left\{x'_t - \alpha \operatorname{sign}\left(\nabla_x \ell(x'_t, \hat{y})\right)\right\}$
  end
  if $\arg\max_i F(x'_k)_i = \hat{y}$ **then**
    Terminate (successful attack)
  else
    $S \leftarrow S \setminus \hat{y}$
    if S *is empty* **then**
      Terminate (failed attack)
    end
  end
end

5.5.1 Network Design

Building on this concept, we propose a Hierarchical Adversarial Robust (HAR) network that decomposes the end-to-end learning task into two distinct parts. First, a single classifier is used for classifying meta-classes. Subsequently, within each meta-class, a separate classifier is used for the leaf classes. This architecture improves the hierarchical adversarial robustness of the network by allowing independent training of each component using adversarial defense techniques. The final probability distribution for all leaf classes is calculated based on Bayes' Theorem. The HAR network benefits from a robustified meta-classifier that improves robustness across meta-classes, and multiple robustified leaf classifiers that improve robustness among visually similar leaf classes.

This design mirrors the hierarchical structure of the dataset, where the leaf classes are grouped into meta-classes. Using Bayes' Theorem, the probability for leaf classes can be computed as follows:

$$P(y \mid x) = P(y \mid x, z)P(z \mid x).$$

Here, $P(y \mid x, z)$ represents the output of the leaf classifier, and $P(z \mid x)$ represents the output of the meta-classifier. This formulation allows us to compute the probability distribution across all leaf classes by combining the predictions from the respective classifiers.

5.5.2 Inference

Given an input x, let $\mathrm{M}(x)$ denote the output of the meta-classifier, where $\mathrm{M}(x)_i$ represents the probability of the i-th meta-class. Similarly, let $\mathrm{L}_i(x)$ denote the output of the i-th leaf classifier, with $\mathrm{L}_i(x)_j$ indicating the probability of the j-th leaf class within the i-th meta-class. Assuming that the meta-classes are non-overlapping (Section 5.2), the output of the network is given by

$$F(x) = [\mathrm{M}(x)_1\mathrm{L}_1(x)_1, \ldots, \mathrm{M}(x)_1\mathrm{L}_1(x)_{k_1}, \ldots, \mathrm{M}(x)_C\mathrm{L}_C(x)_1, \ldots, \mathrm{M}(x)_C\mathrm{L}_C(x)_{k_C}], \tag{5.2}$$

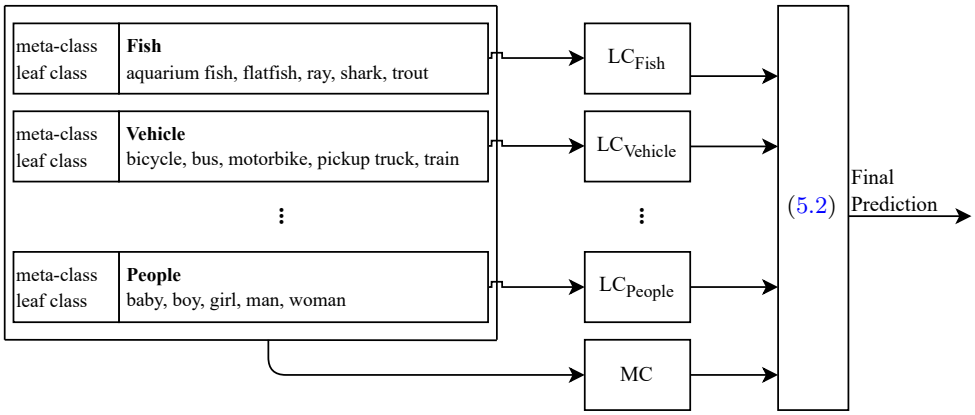


Figure 5.1: **Overview of the HAR Network.** For clarity, we use 'MC' to denote the meta-classifier and 'LC' for the leaf-classifier in the diagram. The meta-classifier is trained to predict the meta-classes, while the leaf classifiers are trained to predict the leaf classes within their respective meta-classes. The final prediction of the HAR network is computed using (5.2), which integrates outputs from the MC and the respective LCs.

where C represents the total number of meta-classes and k_C represents the number of leaf classes within the C-th meta-class.

One advantage of this modular network design is that it allows for training each network using adversarial defense techniques tailored to improving robustness specific to their respective classes. In particular, we use a robustified meta-classifier designed specifically to improve robustness between meta-classes, which contributes to an overall improvement in hierarchical adversarial robustness as we demonstrate in our experiments.

5.5.3 Training

Each model within the HAR network is trained independently. The meta-classifier is trained using the entire dataset, labeled with meta-classes, represented as $\{x, z\}$. Subsequently, the leaf-classifiers are trained using their corresponding subsets of the dataset, represented as $\{x, y\}$. This modular structure allows for parallel training of the different components of the HAR network.

5.6 Experiments

In this section, we perform a series of experiments to evaluate the hierarchical adversarial robustness of the HAR network. We compare its hierarchical robustness against flat models when subjected to both the standard PGD attack and our proposed hierarchical PGD attack. We focus on evaluations based on ℓ_∞-norm constrained attacks on the CIFAR-100 dataset. Discussions and evaluations related to ℓ_2-norm constrained perturbations, as well as results for the smaller CIFAR-10 and CIFAR-5x5 datasets, are provided in the appendix, with results that are consistent with those from the ℓ_∞-norm evaluations.

5.6.1 Experiment Setup

Baseline flat models: We focus on ResNet10 and ResNet50 (He et al., 2016a). To establish baselines, we train ResNet50 using unperturbed data (Standard), 10-step untargeted PGD perturbations (PGD10), and using TRADES (Zhang et al., 2019). In addition, we explore a targeted variant of the PGD adversarial training (PGD10-T), where for a given training pair (x, y) with $y \in z^*$, perturbations are computed using a targeted PGD10 attack with targets uniformly sampled from leaf classes outside the meta-class $\{y \mid y \notin z^*\}$.

HAR models: For the HAR model, we use ResNet10 for both meta-classifier and leaf classifiers. We opt for a lower-capacity ResNet10 in the HAR setup to minimize the difference in the number of parameters compared to a single flat ResNet50. A detailed comparison of trainable parameters is provided in Appendix 5.A. During the training of HAR models, all component networks (meta-classifier and leaf classifiers) undergo the same adversarial defense approach. For instance, a HAR network trained with TRADES comprises one meta-classifier and twenty leaf classifiers, all trained using TRADES.

Training details: All models are trained for a total of 200 epochs, with an initial learning rate of 0.1. The learning rate decays by an order of magnitude at epoch 100 and 150. We use a mini-batch size of 128 for testing and training. We use SGD with momentum of 0.9 and a weight decay of 2e-4. For TRADES, a hyperparameter sweep was conducted on the validation set to select the model with the highest accuracy against untargeted ℓ_∞ bounded PGD20 attacks. This optimization configuration applies to both the flat models and all component models within the HAR network.

Metric: The results in this section are based on the accuracy at the meta-class level, where a larger value indicates better robustness. For our proposed hierarchical attack, the reported accuracy reflects the percentage of test data for which the attack fails to change the meta-class prediction to the desired target, even after iterating through all eligible target labels. In other words, it remains correct at the meta-class level. Given the large number of classes, we follow Dong et al. (2018) and perform the hierarchical attack on 1000 randomly selected test data. All ℓ_∞ adversarial examples used for all evaluations are generated with $\epsilon = 8/255$ and a step size of $2/255$ (pixel values are normalized to $[0, 1]$).

5.6.2 Evaluation of Hierarchical Adversarial Robustness Using Hierarchical PGD

We evaluate the hierarchical adversarial robustness of the HAR network against untargeted PGD perturbations and the proposed hierarchical PGD perturbations, and compare with flat models. The results are summarized in Table 5.3.

We first evaluate the effectiveness of the proposed hierarchical PGD attack. The results indicate that this attack is more effective in degrading the hierarchical robustness of all tested models. For instance, standard trained flat models, which maintain about 24% accuracy under vanilla untargeted PGD perturbations, see their accuracy drop to 0% with the proposed hierarchical perturbation. Similarly, adversarially trained flat models also exhibit a significant accuracy decline when subjected to this attack. Therefore, a method that can accurately measure the hierarchical adversarial robustness of models is needed, and our proposed attack offers a promising approach in this direction.

Table 5.3: **Evaluation of meta-class accuracy for flat models and HAR models on the original CIFAR-100 test dataset, under ℓ_∞-norm constrained untargeted and proposed hierarchical PGD attacks (%).** All perturbations are bounded by $\epsilon = 8/255$. We evaluate both flat models and HAR models under various training paradigms, with the definitions of all baseline flat models provided in Section 5.6.1. Results show that the proposed hierarchical attack significantly degrades the hierarchical robustness of all tested models, with the HAR network demonstrating greater hierarchical robustness compared to flat models.

Method		Clean	Untargeted		Hierarchical	
			PGD20	PGD200	PGD20	PGD200
Flat	Standard	82.57	24.89	24.94	0.00	0.00
	PGD10	69.81	37.80	37.02	22.60	21.98
	PGD10-T	75.02	41.30	40.84	22.10	21.00
	TRADES	67.67	41.47	41.27	27.00	26.80
HAR	Standard	81.24	29.25	29.27	4.00	3.30
	PGD10	66.23	30.53	29.93	25.80	24.40
	TRADES	62.49	32.86	32.36	30.20	29.90

5.6.3 Improving Hierarchical Adversarial Robustness with HAR

Next, we evaluate the hierarchical robustness of the HAR network in comparison to flat models. The results demonstrate that the hierarchical robustness of the HAR network is significantly superior to that of the flat models, a finding consistent across all models tested. Additionally, we observe that training with targeted PGD10 perturbations does not yield a significant improvement over conventional adversarial training with untargeted PGD10 perturbations. These results underscore the effectiveness of the HAR network design in bolstering the hierarchical adversarial robustness of models.

Under the white-box threat model, attackers with complete knowledge of the HAR network's internal structure can generate perturbations specifically targeting the meta-classifier. During our evaluations, we investigate whether untargeted PGD adversaries, based on the meta-classifier, produce stronger hierarchical adversarial examples compared to those generated using the entire HAR model. To generate these perturbations, we apply the standard untargeted PGD attack on the meta-classifier with the objective:

$$\max_{\delta} \ell(M(x+\delta), z^*),$$

where M is the meta-classifier and z^* is the ground true meta-class of input x. We call the resulting perturbation as PGD-Meta. We compare it with PGD perturbations generated using the entire HAR network, referred to as PGD-HAR, and with the proposed hierarchical PGD attack, as shown in Table 5.4. The result indicates that the perturbations generated using the meta-classifier are weaker attacks compared to those generated using the full HAR network. This suggests that the hierarchical adversarial robustness of the HAR network is not solely attributable to the robustness of the meta-classifier, but rather to the combined robustness of the entire network configuration.

Table 5.4: **Evaluation of meta-class accuracy for the HAR models under ℓ_∞-norm untargeted PGD perturbations generated based on the meta-classifier (%).** Perturbations generated based on the meta-classifier are denoted as PGD-meta. We also compare results from Table 5.3: PGD-HAR, which represents perturbations generated using the entire HAR network, and Hierarchical PGD, which refers to the proposed hierarchical PGD attack.

Method	PGD-HAR	Hierarchical PGD	PGD-Meta
Standard	29.27	3.30	0.00
PGD10	29.93	24.40	29.96
TRADES	32.36	29.90	29.38

5.7 Conclusions

Not all mistakes caused by adversarial perturbations have equal consequences. In this study, we introduced a novel concept called hierarchical adversarial robustness. For datasets where classes can be grouped into meta-classes, we defined hierarchical adversarial examples as those leading to misclassification at the meta-class level. In terms of attacks, we demonstrated that untargeted PGD attacks are ineffective at generating hierarchical adversarial perturbations. Consequently, we proposed a hierarchical PGD attack specifically designed to generate such examples. On the defense side, we developed the HAR model to improve hierarchical adversarial robustness, utilizing an ensemble approach with a meta-classifier and multiple leaf classifiers, each trained independently using adversarial defense techniques. Our empirical results show that the HAR model significantly increases hierarchical adversarial robustness.

5.7.1 Challenges and Limitations

Efficient Hierarchical Attack: In Section 5.4, we adopt the target selection strategies from Carlini et al. (2017): an average-case approach, which randomly selects a leaf class as the target, and a worst-case approach, which iterates through all valid leaf classes as targets until all options are exhausted. We employ the worst-case approach, which can be computationally expensive, especially when the number of leaf classes is large. This underscores the need for developing a more efficient hierarchical attack algorithm capable of identifying the target leaf class without exhaustively exploring all options.

Efficient Model Design: One drawback of the proposed HAR model is the poor scalability with the number of meta-classes. As the number of meta-classes increases, so does the number of leaf classifiers, leading to greater model complexity. This can significantly increase both training time and computational resource demands. Although our method enables potential parallel training of each component model, developing a more efficient design that can handle a large number of meta-classes without compromising performance remains an interesting avenue for future research.

Table 5.5: **Evaluation of meta-class accuracy for flat models and HAR models on the original CIFAR-10 and CIFAR-5x5 test datasets, under ℓ_∞-norm constrained untargeted and proposed hierarchical PGD attacks (%).** All perturbations are bounded by $\epsilon = 8/255$.

Dataset	Method		Clean	Untargeted PGD	Hierarchical PGD
CIFAR-10	Flat	Standard	98.93	79.73	0.00
		PGD10	97.49	87.40	72.00
		TRADES	95.69	88.15	79.76
	HAR	Standard	98.44	65.46	0.00
		PGD10	95.57	86.38	79.52
		TRADES	94.49	86.07	81.23
CIFAR-5x5	Flat	Standard	92.68	65.52	0.00
		PGD10	86.08	70.68	42.20
		TRADES	84.16	69.60	48.04
	HAR	Standard	94.36	56.72	0.11
		PGD10	89.16	63.32	52.32
		TRADES	84.96	66.36	58.96

Appendices

5.A Comparison of Trainable Parameters

In our evaluations with CIFAR-100, we use ResNet50 for the vanilla models and multiple ResNet10 for the HAR network. We opt for a lower-capacity ResNet10 in the HAR setup to minimize the difference in the number of parameters compared to a single flat ResNet50. While achieving an exact match in parameter count is challenging with CIFAR-100, given its large number of meta-classes, our goal is to mitigate concerns that improved hierarchical adversarial robustness might solely stem from increased network complexity.

5.B Results on CIFAR-10 and CIFAR-5x5 with ℓ_∞-norm Constrained Perturbations

To further address concerns regarding parameter count, we conduct evaluations on two smaller datasets: the CIFAR-10 dataset, which includes 2 meta-classes and 10 leaf classes, and the medium-sized CIFAR-5x5 dataset, featuring 5 meta-classes and 25 leaf classes. For both, we use ResNet50 for flat models and multiple ResNet10 for the HAR network, resulting in the HAR network having fewer parameters than the flat model. The results are summarized in Table 5.5, where we compare the hierarchical robustness of flat models and the HAR network against ℓ_∞-norm bounded untargeted PGD and the proposed hierarchical PGD attack. These results align with findings from the CIFAR-100 experiments, further demonstrating the HAR network's effectiveness in improved hierarchical adversarial robustness.

Table 5.6: **Evaluation of meta-class accuracy for flat models and HAR models on the original CIFAR-100 test dataset, under ℓ_2-norm constrained untargeted and proposed hierarchical PGD attacks (%).** All perturbations are bounded by $\epsilon = 0.5$. We evaluate flat models and HAR models under various training paradigms.

Method		Clean	Untargeted PGD50	Hierarchical PGD50
Flat	Standard	82.57	32.17	0.10
	PGD10	74.85	53.64	42.90
	PGD10-T	78.58	52.97	38.50
HAR	PGD10	73.31	48.78	**44.00**

Table 5.7: **Evaluation of leaf-class accuracy for flat models and HAR models on the CIFAR-10 test dataset.**

	Standard	PGD10	TRADE
Flat	92.54	87.14	84.92
HAR	92.19	84.64	83.90

5.C Results on CIFAR-100 with ℓ_2-norm Constrained Perturbations

Similar to the setup with ℓ_∞-norm constrained perturbations discussed in Section 5.6, we compare the HAR network with flat baseline models robustified against ℓ_2 attacks. We were unable to achieve reasonable robustness results with TRADES against ℓ_2 attacks, likely because the hyperparameter sweep was based on ℓ_∞ results, and no additional sweep was performed for ℓ_2 perturbations. Consequently, the optimal value for β in TRADES is unsuitable in the ℓ_2 setting. For this reason, we omit TRADES from this section.

The results are summarized in Table 5.6. Overall, these findings are consistent with those from the ℓ_∞-norm attacks. The proposed hierarchical attack significantly degrades the hierarchical robustness of all tested models, while the HAR model exhibits greater robustness compared to flat models.

5.D Comparison of Clean Test Accuracy on CIFAR-10

Finally, we report the clean test accuracy of the flat baselines and the proposed HAR networks on CIFAR-10. All models follow the training configurations in Section 5.6. We observe a slight decrease in the clean accuracy with the proposed HAR network, following the accuracy-robustness trade-off discussed earlier in Section 2.5.

Chapter 6

Improving Adversarial Transferability via Model Alignment

6.1 Introduction

In the previous chapters, we demonstrated that seemingly indistinguishable perturbations to an input can yield drastically different outputs from neural networks, highlighting the brittleness of these models. In practice, adversaries often do not access to the target model's architecture or parameters. However, as discussed in Section 2.6, what makes adversarial examples particularly concerning is their transferability across models—i.e., the ability of an example generated for one model to fool another.

In this chapter, we propose a model alignment technique to improve the transferability of adversarial examples. At first glance, one might question: *Why not concentrate on building defensive methods to reduce vulnerability?* Developing methods to generate more effective adversarial examples is just as important as devising strategies to improve the robustness of neural networks. As of now, the top three robustification techniques (Wang et al., 2023; Bai et al., 2023; Peng et al., 2023) listed on RobustBench (Croce et al., 2021) all rely on some variants of adversarial training: augmenting the training data with adversarial examples (Goodfellow et al., 2015; Madry et al., 2018).

This highlights the importance of understanding and improving adversarial example generation. Our work focuses on one particular aspect of this direction: generating adversarial examples that exhibit higher transferability. Specifically, *Our goal is to transform any source model into one from which attacks generate more transferable perturbations.*

One possible explanation for the transferability of adversarial examples is that these perturbations exploit similar features present in both the source and target models (Ilyas et al., 2019). To see this, let us first consider the hypothesis that neural networks capture two distinct types of features from data: semantic features and human-imperceptible features. This hypothesis has been proposed and empirically supported in Chapter 4 and studies such as Wang et al. (2020a) and Dong et al. (2019a). We provide a summary of their findings as follows. First, models learn semantic features that align with human perception. The extraction of such features is similar across different models, reflecting a shared understanding of the semantics. Second, models learn human-imperceptible

This chapter is based on our work in Improving Adversarial Transferability via Model Alignment.

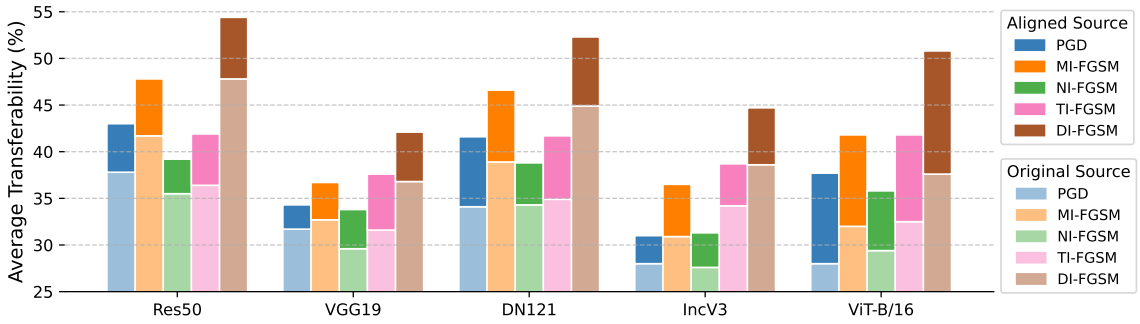


Figure 6.1: **Attacking the aligned source model for more transferable perturbations.** We compare the transferability of ℓ_∞-norm bounded perturbations ($\epsilon = 4/255$) generated using the source model before and after performing model alignment. The result highlights the **compatibility** of model alignment with a wide range of attacks, as perturbations generated from the aligned source model become more transferable. Here, the source model is aligned using a witness model from the same architecture but is initialized and trained independently. Results are averaged over all target models.

features, and their learning is model-specific. For example, Chapter 4 theoretically demonstrated that the use of these features varies based on models' initialization and their optimization process. Wang et al. (2020a) discussed how model architecture can result in model-specific interpretations of these features.

Features and vulnerabilities are highly correlated, as vulnerabilities often arise from the exploitation of specific features—patterns or attributes in the data that models rely on to make predictions (Ilyas et al., 2019). In the context of transferability, the degree of similarity between those exploited features in the source and target model is crucial. That is, the more similar the exploited features between models, the more likely it is that the perturbation will successfully transfer. To support this, Liu et al. (2016) empirically showed that different models have similar decision boundaries (from learning similar features), thus enabling some perturbation to be transferable across different models. However, some perturbations exploit features that are source-model-specific.Qin et al. (2022) empirically demonstrated that when maximizing the cross entropy loss to find adversarial examples, some perturbations fail to transfer because they correspond to sharp local maxima in the input space, specific to the source model. These perturbations exploit features that are unique to the source model and are not shared with the target model. Motivated by these observations, we propose a model alignment technique to modify the source model to encourage a similar feature extraction as other, independently trained models, which we refer to as *witness models*.

During the alignment process, the parameters of the source model are fine-tuned to minimize an alignment loss. This alignment loss measures the divergence in the predictions between the source and the witness model. Through this alignment process, the source model learns to focus on a set of features that are similarly extracted by the witness model. This allows attack algorithms to more effectively exploit features common across models, leading to more transferable perturbations.

Model alignment complements, rather than competes with, other attack algorithms. This synergy underscores our method's key advantage: its broad compatibility with a wide range of attack algorithms, as highlighted in Figure 6.1. Extensive experiments on various combinations of archi-

tectures, transferability on individual target models, and results with other attacks are included in Section 6.5.

6.1.1 Contributions

Our contributions can be summarized as follows:

- We present a model alignment method to fine-tune the source model by minimizing an alignment loss which measures the difference in the output between the source model and the witness model.
- We analyze the effect of the proposed alignment method, demonstrating that perturbations generated from the source model exploit more semantic features that are shared across different models.
- To understand the effect of model alignment, we conduct a geometric analysis to study the changes in the loss landscape resulting from this process.
- Extensive experiments on ImageNet (Krizhevsky et al., 2012), using CNNs and Vision Transformers (ViTs) (Dosovitskiy et al., 2021; Liu et al., 2021), demonstrate that perturbations generated from aligned source models exhibit significantly higher transferability than those from the original source model. We demonstrate that our alignment technique is compatible with a wide range of attacks.

6.2 Background

In this section, we provide a brief overview of approaches to generate more transferable perturbations. For a more comprehensive review of research related to adversarial transferability, we direct the reader to a recent survey (Gu et al., 2023).

6.2.1 Generating Transferable Perturbations

Existing work on improving the transferability of adversarial examples can be categorized into four groups: data-augmentation-based methods (Xie et al., 2019; Dong et al., 2019b; Zou et al., 2020; Wu et al., 2021; Li et al., 2020b; Byun et al., 2022; Wang et al., 2021c), optimization-based methods (Dong et al., 2018; Lin et al., 2020; Wang et al., 2021b; Zhao et al., 2021; Zhang et al., 2022; Xiao et al., 2021; Li et al., 2020a), model-modification-based methods (Benz et al., 2021b; Wu et al., 2020a; Guo et al., 2020b; Wang et al., 2024), and ensemble-based methods (Liu et al., 2016; Gubri et al., 2022; Qian et al., 2023; Li et al., 2022b).

Data augmentation-based methods: Data augmentation methods are ubiquitously used in training DNNs to prevent overfitting. Advanced augmentation techniques (Zhang et al., 2018; Yun et al., 2019; Cubuk et al., 2019; Ma et al., 2022a) have become crucial to achieving state-of-the-art generalization performance on large-scale datasets like ImageNet. Building on this concept, several works have proposed the incorporation of various forms of data augmentation into attack algorithms. This integration aims to prevent adversarial examples from overfitting to the source model, thereby improving transferability.

Optimization-based methods: Lin et al. (2020) drew a parallel between generating transferable adversarial examples and training neural networks. In this analogy, source models are the training data, adversarial perturbations are model parameters, and the target model is the testing data. Thus, transferability of adversarial examples is akin to model generalization. As such, optimization-based approaches, such as momentum (Dong et al., 2018; Lin et al., 2020; Zou et al., 2020) and variance tuning (Wang et al., 2021b; Xiong et al., 2022), that were initially proposed to improve the generalization of neural networks can be leveraged to improve the transferability of the adversarial examples.

Model modification-based methods: Several studies have proposed methods to improve the transferability of adversarial examples by modifying the source model. For example, Benz et al. (2021b) demonstrated that perturbations generated from a source model without batch normalization (Ioffe et al., 2015) are more transferable than those from models equipped with batch normalization. Methods like Linear Backpropagation (LinBP) (Guo et al., 2020b) and Backward Propagation Attack (BPA) (Wang et al., 2024) focus on non-linear activations and modify the derivative of ReLU. Wu et al. (2020a) showed that increasing gradients from skip connections over residual units can significantly increase transferability.

Among the categories of methods discussed, model alignment is most closely related to model modification-based approaches. However, a key advantage of our approach is its model-agnostic nature: alignment can be applied without changing the model's forward or backward pass, improving any source model's ability to generate more transferable perturbations. In contrast, other methods require changes, such as those seen in LinBP and BPA, or even complete retraining.

Ensemble-based methods: Another line of approaches involves the use of multiple models for generating adversarial examples. Liu et al. (2016) were among the first to propose enhancing transferability by attacking an ensemble of models, with the rationale being that a perturbation capable of fooling multiple models is more likely to deceive the target model. More recently, Gubri et al. (2022) proposed constructing an ensemble of source models by collecting weights along the fine-tuning trajectory of a trained model.

In Section 6.5, we demonstrate that model alignment is fully compatible with all these methods.

6.2.2 Understanding Adversarial Transferability

Several works have focused on understanding the transferability of adversarial perturbations (Zhang et al., 2024; Wu et al., 2020b; Waseda et al., 2023; Zhu et al., 2021), with some analyzing from a geometric perspective (Fawzi et al., 2017; Charles et al., 2019; Zhao et al., 2020; Liu et al., 2016). Liu et al. (2016) empirically demonstrated that adversarial examples with weak transferability often correspond to local maxima in the source model's loss landscape with respect to the input space. They analyzed the position of adversarial examples relative to the decision boundaries of both the source and target models. They found that some perturbations fail to transfer because they are located in tiny pockets within regions corresponding to the ground truth label, which exist only for the source model but not for the target model. Similarly, Gubri et al. (2022) hypothesized that adversarial examples at flat loss maxima of the source model tend to transfer more effectively than those at sharp maxima. Motivated by these insights, in Section 6.4, we extend this geometric perspective by examining how the alignment method influences the loss surface geometry of the source model, particularly in its capacity to generate more transferable adversarial examples.

6.3 Formulation of Model Alignment

We introduce the model alignment as a fine-tuning process and explain why aligned models are better at generating transferable perturbations. In this work, we focus on neural networks used for classification tasks. Let us consider a neural network designed for m-class classification, represented as a series of function compositions:

$$f(x) = (\phi^{[l]} \circ \phi^{[l-1]} \circ \ldots \circ \phi^{[1]})(x),$$

where each $\phi^{[i]}$ represents an operation within the network, which could be a linear transformation (such as a fully connected layer), an activation function, or a pooling operation. The parameters of the neural network are collectively denoted as θ. The intermediate outputs of these operations are often referred to as hidden representations. We denote them by $z^{[i]}$, where $z^{[i]} = \phi^{[i]}(z^{[i-1]})$ for $i = 1, 2, \ldots, l$, and the initial input is $z^{[0]} = x$. Additionally, we incorporate the softmax function into the neural network's definition. Specifically, in the final layer, we have $\phi^{[l]} = \text{softmax}(z^{[l-1]})$, where $z^{[l-1]}$ are called the logits. With this definition, the output of this network, $f(x)$, can be interpreted as a probability distribution over the m classes where each component $f(x)_i$ represents the probability of the input x belonging to class i.

The goal of model alignment is to modify the source model such that it can extract features similar to those of a witness model. In this chapter, we define a witness model as an independently initialized and trained model that differs from the source model in at least one factor, such as architecture, initialization, or training procedure. We denote the parameters of the source model and the witness model as θ_s and θ_w, respectively. Let us consider the following pointwise formulation of the alignment loss:

$$\ell_a(x, \theta_s, \theta_w) = d(z_s^{[q]}(x), z_w^{[q]}(x)), \tag{6.1}$$

where the metric d measures the output difference at layer q between the models.

During alignment, the parameters of the source model are fine-tuned to minimize this alignment loss which captures the differences between models' output. Specifically, when $q = l$, the alignment loss measures the divergence between the probability distributions generated by the two models. In this scenario, the Kullback-Leibler (KL) divergence is particularly suitable, due to its effectiveness in measuring distribution differences and its relative ease of implementation in practice. Our analysis and primary experimental results are based on this setting, where model alignment occurs in the output space. We also explore the alignment in the embedding space (i.e., $q < l$), which is detailed in our ablation study. The pointwise loss defined in (6.1) focuses on aligning the source model with a single witness model. However, alignment can also be extended to multiple witness models to further improve the alignment process. This process involves using a set of witness models, denoted as Θ, with the number of witness models represented by $|\Theta|$. Finally, the update rule for the parameters of the source model based on SGD can be written as

$$\theta_s(t+1) = \theta_s(t) - \eta \frac{1}{|\mathcal{B}|\,|\Theta|} \sum_{x \in \mathcal{B}} \sum_{\theta_w \in \Theta} \nabla \ell_a(x, \theta_s(t), \theta_w),$$

where $\mathcal{B}$ represents the mini-batch.

Table 6.1: **Evaluation of the change in the similarity between the source and witness model after model alignment.** We focus on aligning a Res50 with a Res18 and a ViT-B/16 with a ViT-T/16. We first evaluate the similarity between each original source model and its corresponding witness model. This is then followed by an evaluation between the aligned model and the witness model. To measure the similarity, we use the KL divergence, prediction agreement, and the cosine similarity between the input gradients.

Source/Witness	KL		Prediction Agreement		Input Gradient Cosine Similarity	
	Before	After	Before	After	Before	After
Res50 / Res18	0.63	0.31	80.7%	82.4%	0.043	0.096
ViT-B / ViT-T	0.75	0.24	78.9%	85.0%	0.026	0.063

6.4 Understanding Model Alignment

We now provide an understanding of the model alignment approach. To investigate the effect of alignment, we first evaluate the change in the similarity between the source and witness models. We then analyze the perturbations generated from the source model before and after alignment. Finally, we conduct a geometric analysis to study the changes in the loss surface resulting from the alignment process.

We summarize the key implementation details here and refer readers to Section 6.5.1 for additional information. ResNet50 and ViT-B/16 are used as source models, aligned with ResNet18 and ViT-T/16, respectively, as their witness models. Using the ImageNet training set, the alignment is performed for one epoch on the output layer (i.e., $q = l$ in (6.1)), using KL divergence as the distance metric d. All results presented in this section are averaged over 1000 randomly sampled images from the ImageNet test set. We consider ℓ_∞-norm adversarial examples generated using 20 iterations of PGD with $\epsilon = 4/255$ and $\alpha = 1/255$ (refer to (2.5) in Section 2.3). We use Δx_s and Δx_a to represent the perturbations generated based on the source and aligned models, respectively.

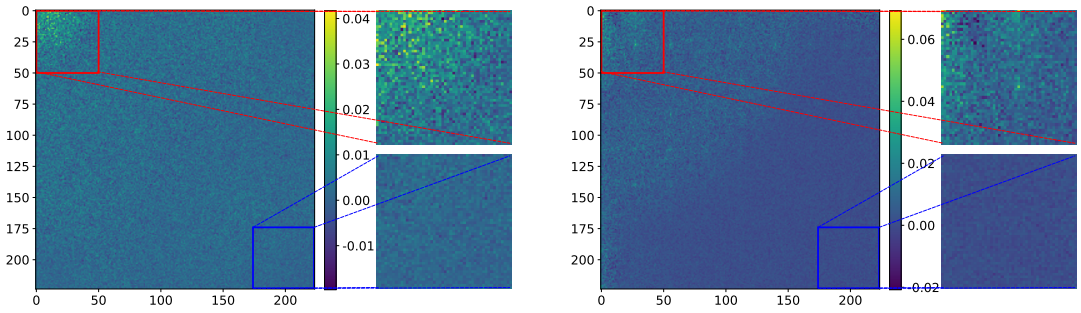
6.4.1 Evaluating Similarity between the Source and Witness Model

To study the similarity between the source and witness models before and after alignment, we examine KL divergence and cosine similarity of input gradients, with the latter being particularly important for its role in the attack processes. Additionally, we measure prediction agreement between the two models:

$$\text{Prediction Agreement} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}\left(\arg\max f_s(x_i) = \arg\max f_w(x_i)\right),$$

where N is the number of samples, $f_s(x)$ and $f_w(x)$ are the outputs of the source and witness models, respectively, and $\mathbb{I}(\cdot)$ is the indicator function. This metric represents the percentage of samples for which the two models agree on the predicted class.

We first evaluate the similarity between each original source model and its corresponding witness model. This is then followed by an evaluation between the aligned model and the witness model. Table 6.1 presents the results, indicating reductions in KL divergence, increased cosine similarity, and improved prediction agreement post-alignment.



a. ResNet50. b. ViT-B/16.

Figure 6.2: **Frequency-domain visualization of the differences in the perturbation generated using the original and aligned model.** We compare the magnitude of the DCT coefficients between the perturbations generated by the two models: $|\text{DCT}(\Delta x_a)| - |\text{DCT}(\Delta x_s)|$. The pronounced brightness in the top-left region of the spectrum indicates that the primary differences lie within the low-frequency range, which is typically associated with semantic features.

6.4.2 Aligned Model Exploits More Semantic Features

To verify that attacks applied to the aligned model exploit more semantic features, we compare the perturbations generated using the original and aligned source models.

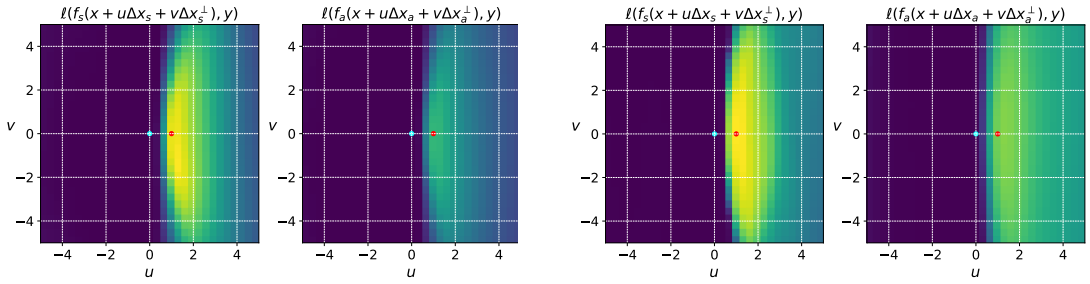
Adversarial perturbations are imperceptible and difficult to characterize in terms of the specific features they exploit in the spatial domain. However, by analyzing perturbations in the frequency domain, we can observe the types of features each model exploits. Semantic features mostly concentrate around the low-frequency end of the spectrum, as demonstrated in Chapter 4 and studies such as Wang et al. (2020a) and Dong et al. (2019a). We consider discrete cosine transform (DCT) (Ahmed et al., 1974) and compare the DCT coefficients of the perturbations generated from the source and the aligned model, denoted as $\text{DCT}(\Delta x_s)$ and $\text{DCT}(\Delta x_a)$, respectively. Since our interest lies in the magnitude of these coefficients, we visualize the difference by computing $|\text{DCT}(\Delta x_a)| - |\text{DCT}(\Delta x_s)|$.

The results are illustrated in Figure 6.2. We observe that the differences between the perturbations are predominantly located in the top-left corner of the DCT spectrum, indicating that they primarily differ in the amount of low-frequency information. This result shows that adversarial perturbations generated from the aligned source model exploit more low-frequency, semantic features compared to those generated from the original source model.

6.4.3 Model Alignment Yields Smoother Loss Surface

Previous works have studied the connection between perturbation from sharp loss maxima in the input space and their poor transferability (Qin et al., 2022; Gubri et al., 2022). We extend this geometric perspective by examining how model alignment affects the loss surface geometry of the source model, particularly in its capacity to generate more transferable adversarial examples.

Let us first focus on the setting where (6.1) measures the KL divergence between the predictions from the two models. During the alignment process, the parameters of the source model are fine-tuned using *soft labels* based on the witness model's outputs. Unlike one-hot labels, which assign a probability of 1 to the target class and 0 to all others, soft labels represent a probability distribution



a. ResNet50. b. ViT-B/16.

Figure 6.3: **Visualization of the loss surface around adversarial perturbations generated from the original and aligned model.** Each plot illustrates the loss surface projected on the plane spanned by the direction of the adversarial perturbation and its orthogonal vector. We examine the loss landscape surrounding a clean data point (cyan marker) and an ℓ_∞-bounded adversarially perturbed data point (red marker), generated from the source model (Δx_s) and the aligned model (Δx_a). Notice that perturbations generated using the original source models are located at sharper loss maxima, whereas the loss surface around the perturbation generated using the aligned model is much flatter.

across all classes.

Training DNNs with soft labels prevents the model from becoming overly confident in its predictions, thereby improving generalization (Szegedy et al., 2016; He et al., 2019; Hinton et al., 2015; Müller et al., 2019). Compared to using hard, one-hot encoded labels, it has been shown that training with soft labels implicitly regularizes the norm of the input Jacobian (Carratino et al., 2022; Zhang et al., 2021), and leads to smoother decision boundaries (Zhang et al., 2018; Verma et al., 2019).

To understand why model alignment can be helpful, we begin by identifying a data point for which the perturbation, when generated based on the original source model, fails to transfer to the target model, but the perturbation generated using the aligned model misleads the target model. In Figure 6.3, we visualize the loss surface surrounding this data point, spanned by two pairs of orthogonal vectors: Δx_s, $\Delta x_s^\perp$, Δx_a and $\Delta x_a^\perp$. Here, $\Delta x_s^\perp$ and $\Delta x_a^\perp$ are randomly selected orthogonal vectors and are also bounded by the same ϵ. In each plot, the center of the plot (cyan marker) represents the clean data point, and the adversarial example is highlighted using the red marker.

We make two observations. First, the bright yellow region around the perturbations generated from the original source model suggests that they are located at sharp loss maxima. However, these same perturbations do not effectively cause a significant increase in loss when applied to the target model. This observation aligns with previous findings indicating that perturbations with poor transferability often correspond to sharp local maxima unique to the source model, which are not present in the target model (Gubri et al., 2022; Qin et al., 2022). Second, the loss surface around perturbations generated from the aligned model is noticeably flatter. This is in line with prior findings that adversarial perturbations from flatter maxima tend to be transferable (Gubri et al., 2022; Qin et al., 2022). It is noteworthy that the PGD attack does not explicitly target flat maxima for generating adversarial examples. This observation leads us to hypothesize that the smoothing effect induced by the alignment process is more global, rather than being confined to specific cases of adversarial examples.

Table 6.2: **Evaluation of the ℓ_2-norm of the gradient ($\|\nabla_x \ell(x+\Delta x, y, \theta)\|_2$) and the largest eigenvalue of the Hessian ($\lambda_{\max}(\nabla_x^2 \ell(x+\Delta x, y, \theta))$) on the original ($\theta_s$) and aligned model ($\theta_a$).** Results are averaged over 1000 randomly selected ImageNet test samples. In addition to the original test data ($\Delta x = 0$), we consider two types of perturbations: $\Delta x \in \{\text{Gaussian}, \text{PGD}\}$. Notice that both the gradient norm and the largest eigenvalue decrease significantly when evaluated on all three types of input, suggesting that the smoothing effect induced by the alignment is more global, rather than being confined to specific cases of adversarial examples.

Source / Witness	Res50 / Res18				ViT-B/16 / ViT-T/16	
Metric	$\|\nabla_x \ell(x+\Delta x, y, \theta)\|_2$		$\lambda_{\max}(\nabla_x^2 \ell(x+\Delta x, y, \theta))$		$\|\nabla_x \ell(x+\Delta x, y, \theta)\|_2$	
Δx	θ_s	θ_a	θ_s	θ_a	θ_s	θ_a
0	0.056	0.032	4.16	1.25	0.060	0.029
Gaussian	0.057	0.032	4.34	1.27	0.063	0.029
PGD	0.453	0.171	0.10	0.05	0.184	0.055

To verify this, we evaluate the change in 1. the ℓ_2-norm of the gradients and 2. the maximum eigenvalue of the Hessian w.r.t. clean, Gaussian-perturbed ($\sigma^2 = 0.01$), and PGD-perturbed inputs. More formally, we compare $\|\nabla_x \ell(x+\Delta x, y, \theta)\|_2$ and $\lambda_{\max}(\nabla_x^2 \ell(x+\Delta x, y, \theta)$ for $\Delta x \in \{0, \text{Gaussian}, \text{PGD}\}$ for $\theta \in \{\theta_s, \theta_a\}$ over 1000 randomly sampled ImageNet test samples. As PyTorch currently does not support computing second-order derivatives for ViT models, we only include results for Res50 for the Hessian and consider both Res50 and ViT-B/16 for the gradient norm.

The results are summarized in Table 6.2. We observe a significant decrease in both the gradient norm and the largest eigenvalue across all inputs, and the reduction is particularly pronounced at PGD-perturbed data points. This decrease is consistent with the previous finding which demonstrated a connection between the use of soft labels and the smoothing of the loss surface (Carratino et al., 2022; Zhang et al., 2021). More importantly, this observation supports our hypothesis that the improved transferability of adversarial examples generated from the aligned model is due to the smoothing of the loss surface.

6.5 Experiments

In this section, we present a series of experiment results to demonstrate the improved ability of a given source model to generate transferable adversarial perturbations. We study factors that can further improve the alignment process and demonstrate the compatibility of our approach with a wide range of attacks.

6.5.1 Experiment Setup

Model: We consider a variety of neural network architectures as the source, witness, and target models. For CNN-based models, we include ResNet18 (Res18), ResNet50 (Res50), ResNet101 (Res101) (He et al., 2016b), VGG19 (Simonyan et al., 2015), DenseNet121 (DN121) (Huang et al., 2017), and Inception-v3 (IncV3) (Szegedy et al., 2016). For ViT-based architectures, our selection includes ViT-T/16, ViT-S/16, ViT-B/16 (Dosovitskiy et al., 2021), and Swin Transformers (SWIN) (Liu et al., 2021). We follow the optimization schedule described in the official Pytorch repository to train all models. Training and model details are included in Appendix 6.A.

Fine-tune: During alignment, all source models are fine-tuned for one epoch using SGD with a momentum of 0.9, sweeping over 3 learning rates. No additional data is used during the process, as fine-tuning relies exclusively on the same training data used to train the source and witness models. We use a fixed batch size of 128 for CNN-based models and 512 for ViT-based models. We use a cosine decay learning rate schedule with a linear warmup. For ViT-based models, we follow (Steiner et al., 2021) and gradients are clipped at a global norm of 1.

These configurations apply to all experiments presented in this chapter. For the main results in Table 6.3, alignment is performed using a single witness model, focusing on the output layer (i.e., $q = l$ in (6.1)) and using KL divergence as the distance d. In the ablation study, we explore multi-witness alignment as well as alignment in the embedding space (i.e., $q < l$), using different embedding-space distillation methods.

Dataset: We follow previous work (Dong et al., 2018; Lin et al., 2020) in which evaluations are based on 1000 images from the ImageNet test set. For a given source and target model, these samples meet the following criteria: they are correctly classified by both models, and the adversarial examples, when generated from each model, lead to misclassifications in their respective originating models. The samples are initially selected uniformly and independently from the test set, and only those meeting the criteria are included.

Attack method: We focus on non-targeted adversarial perturbations constrained by the ℓ_∞-norm. Unless otherwise stated, all perturbations are generated using 20 iterations of PGD with $\epsilon = 4/255$ and $\alpha = 1/255$ (refer to (2.5) in Section 2.3). All target models have an error rate of 100% under white-box PGD attacks. Results with different attack methods and larger values of ϵ are included in the ablation study.

Metric: We measure transferability using the error rate, with a higher rate indicating greater transferability. In the tables of this section, we first present a row labeled 'n/a' to denote the error rate for perturbations generated by the original source model. We then demonstrate the change in transferability after alignment following the $+/-$ sign. A larger change indicates a greater increase in transferability resulting from using the aligned model. All results presented in this section are obtained from the average of three independent runs.

6.5.2 Model Alignment Improves Transferability

To demonstrate the improved transferability of perturbations generated using the aligned source model, we evaluate across a diverse set of neural network architectures for both source and witness models, focusing on the transferability to an extensive array of target models. Table 6.3 shows the original error rates and their changes after using the aligned model. We make two key observations.

First, the model alignment approach can transform any given source model into one from which the PGD attack generates more transferable perturbations. Such an improvement can be observed across various combinations of source and witness models and is evident in all evaluated target models. This clear improvement validates our approach, showing that aligning the source model can lead to the generation of more transferable adversarial perturbations. Notably, both CNN and ViT-based source models can benefit from the alignment process. Recent studies have shown that the transferability between ViTs and CNNs is poor, and many attack algorithms do not generalize well to ViTs (Naseer et al., 2021; Wei et al., 2022; Mahmood et al., 2021). Our model alignment method offers a promising solution to bridge this gap.

Table 6.3: **Aligning the source model can lead to the generation of more transferable adversarial examples (%).**The result demonstrates the increase in error rates when adversarial examples generated from aligned source models are applied to different target models. Our method improves the transferability of adversarial examples generated from a wide range of architectures, including both CNNs and ViTs.

Source	Witness	Target									
		Res18	Res50	Res101	VGG19	DN121	IncV3	ViT-T/16	ViT-S/16	ViT-B/16	SWIN
Res50	n/a	44.52	61.51	53.01	43.47	50.59	36.05	25.58	21.54	18.86	22.65
	Res50	+9.60	+9.90	+8.12	+6.59	+8.98	+4.79	+0.67	+0.65	+0.42	+2.65
	VGG19	+8.30	+7.88	+7.41	+9.17	+9.93	+6.76	+0.75	+0.52	+0.17	+1.97
	DN121	+9.39	+10.82	+8.40	+8.41	+12.41	+7.21	+1.87	+2.02	+0.38	+2.61
	IncV3	+7.69	+7.52	+5.99	+5.56	+9.19	+8.23	+2.29	+0.30	+0.24	+3.01
	ViT-B/16	−9.41	−19.39	−17.48	−10.04	−12.76	−8.49	−3.25	−2.69	−1.14	−4.17
	SWIN	−9.23	−16.21	−14.30	−7.27	−10.02	−5.53	−2.72	−2.93	−2.36	−2.73
VGG19	n/a	33.49	30.10	26.13	79.43	32.07	30.62	23.27	19.85	18.66	23.09
	Res50	+7.19	+4.16	+2.76	+5.19	+7.62	+5.91	+3.45	+1.20	+1.36	+1.12
	VGG19	+4.66	+2.10	+1.77	+7.30	+3.44	+1.74	+2.64	+0.68	+1.39	+0.90
	DN121	+6.35	+3.52	+5.07	+10.60	+10.45	+5.03	+4.09	+2.36	+2.58	+1.31
	IncV3	+4.91	+1.57	+1.66	+3.82	+4.94	+4.33	+2.70	+0.61	+0.61	+1.04
	ViT-B/16	−4.34	−5.68	−5.01	−24.66	−5.11	−3.67	−0.83	−0.37	+0.77	−2.53
	SWIN	−3.24	−5.83	−4.01	−21.91	−6.48	−4.05	−1.11	−0.73	+0.64	−2.96
DN121	n/a	40.20	44.01	37.15	40.84	61.15	32.28	23.68	20.34	18.21	23.56
	Res50	+11.67	+12.96	+8.37	+7.32	+11.68	+5.41	+1.69	+0.46	+1.00	+1.59
	VGG19	+9.02	+6.38	+4.88	+12.34	+7.87	+3.77	+2.72	+0.04	+1.22	+0.94
	DN121	+12.56	+11.60	+10.48	+9.10	+15.47	+6.84	+3.99	+1.59	+1.24	+1.86
	IncV3	+8.08	+10.50	+9.88	+9.91	+12.76	+7.24	+3.81	+1.23	−0.02	+2.79
	ViT-B/16	−3.08	−8.43	−7.78	−6.67	−12.96	−1.78	+2.06	+1.26	+1.22	−2.78
	SWIN	−4.13	−8.57	−5.34	−6.11	−10.21	−4.64	+0.54	−0.16	+1.21	−1.88
IncV3	n/a	30.68	27.49	25.02	33.47	30.98	51.80	22.99	18.77	17.72	20.94
	Res50	+4.92	+6.59	+4.27	+5.49	+5.74	+4.49	+1.71	+2.26	+2.74	+2.23
	VGG19	+4.65	+2.69	+3.10	+5.70	+3.39	+4.36	+1.05	−0.54	+0.38	+0.53
	DN121	+4.50	+6.38	+2.13	+7.35	+6.52	+8.89	+3.54	+0.68	+1.41	+1.45
	IncV3	+1.78	+3.91	+3.70	+3.53	+3.06	+8.75	+1.79	+0.49	+1.41	+1.75
	ViT-B/16	+0.47	−1.13	−2.97	−0.77	−1.35	−8.20	+2.92	+1.64	+2.07	−0.23
	SWIN	−0.49	−3.36	−3.85	−3.29	−4.54	−12.82	+0.09	+0.76	+1.13	−1.34
ViT-B/16	n/a	23.05	19.67	18.30	21.83	21.96	21.52	36.44	44.58	51.27	21.22
	Res50	+17.53	+12.89	+12.23	+14.83	+16.66	+15.76	+43.47	+37.17	+29.16	+21.16
	VGG19	+10.89	+9.49	+9.77	+14.23	+12.65	+10.94	+39.63	+31.65	+21.30	+14.97
	DN121	+17.15	+13.92	+12.28	+15.38	+17.68	+16.96	+47.36	+41.88	+31.22	+21.87
	IncV3	+15.41	+11.87	+11.31	+15.09	+14.23	+16.46	+46.07	+40.39	+28.87	+19.92
	ViT-B/16	+5.14	+2.04	+2.07	+4.79	+2.87	+4.53	+24.04	+23.14	+21.93	+6.34
	SWIN	+6.31	+5.75	+5.92	+8.12	+7.46	+7.28	+39.11	+36.49	+28.61	+14.62

Second, we find that CNN-based source models generally benefit from aligning with other CNN-based models, rather than with ViT-based models. On the other hand, the ViT-B/16 source model can benefit from aligning with witness models from both CNN and ViT families. We provide a possible explanation for this phenomenon. Previous studies have shown that ViTs and CNNs learn distinctively different features (Zhou et al., 2022; Raghu et al., 2021). Specifically, Raghu et al. (2021) focused on early-layer representations learned by the two models and showed that more ResNet layers are required to be modified to match hidden representations of a ViT, compared to the other way around. In the context of model alignment, this suggests an asymmetric behavior in the alignment process; namely, it might be easier to align ViTs with CNNs than to align CNNs with ViTs.

Choosing the witness model: The main goal of Table 6.3 is to demonstrate that the improved transferability is not limited to particular choices of witness models. The result also highlights the effectiveness of a straightforward, model-selection-free self-alignment strategy. This is evident from the consistent improvement observed when the source and witness models share the same architecture, but are initialized and trained independently. The selection of witness models is further explored in our ablation studies.

Regularization to prevent overfitting: A prolonged alignment process may inadvertently cause overfitting to the witness model, thereby diminishing the gains in transferability. To counteract overfitting, regularization methods can be implemented. For example, we apply early stopping and limit the alignment to a single epoch. Moreover, an ensemble of witnesses can be used to avert overfitting to any singular witness model. Results with multiple witness models are discussed in Section 6.5.3. Furthermore, when aligning by minimizing KL divergence, adjusting the temperature scaling within the softmax function can be an effective measure to prevent the exact replication of the witness model's predictions by the source model.

6.5.3 Ablation Studies

We conduct several ablation studies to investigate factors that could further improve the alignment process. Additionally, we demonstrate that our approach is compatible with a wide range of attack algorithms.

Smaller Witness Model Might Boost Learning Shared Features.

Results in Table 6.3 indicate that DN121, which has the fewest parameters, frequently emerges as a more effective witness model. This motivates us to investigate the role of the capacity of the witness model during the alignment. While several factors contribute to a model's capacity, such as its structure, the normalization techniques, and its non-linear activations, our study primarily focuses on the number of parameters as a proxy for capacity. With this in mind, we consider three models each from the ResNet and ViT families.

Table 6.4 summarizes the results of the model capacity analysis, pointing to one key observation. Model alignment is more effective when the witness models have a smaller model capacity. For instance, when using Res101 as the source model, alignment with the lower capacity Res18 as the witness model is more beneficial than alignment with Res50. This matches the previous findings in Table 6.3.

Table 6.4: **Analysis on the impact of witness model capacity on the alignment process (%).** We observe a greater improvement in the transferability of adversarial perturbations when the source model is aligned with witness models of smaller capacity.

Source	Witness	Target									
		Res18	Res50	Res101	VGG19	DN121	IncV3	ViT-T/16	ViT-S/16	ViT-B/16	SWIN
Res18	n/a	64.10	45.85	37.09	42.96	48.57	37.31	27.52	21.12	20.57	21.31
	Res50	−1.94	+0.41	−0.74	+0.82	−1.24	−1.78	+0.26	+1.33	+0.42	+1.48
	Res101	−5.13	−3.92	−2.70	−1.75	−1.39	−0.22	+0.53	+0.25	+0.03	+0.48
Res50	n/a	44.52	61.51	53.01	43.47	50.59	36.05	25.58	21.54	18.86	22.65
	Res18	+35.30	+26.56	+25.20	+24.38	+27.86	+19.18	+5.49	+2.85	+1.61	+9.60
	Res101	+4.27	+3.02	+3.93	+3.61	+4.41	+3.60	+1.86	−0.54	+0.14	+1.36
Res101	n/a	38.84	54.63	62.95	39.24	49.93	34.77	23.01	20.12	19.07	22.90
	Res18	+40.64	+35.11	+27.78	+29.63	+33.20	+28.42	+10.10	+5.70	+2.37	+13.99
	Res50	+14.97	+19.56	+16.97	+13.50	+17.07	+10.41	+4.04	+0.98	+1.44	+4.81
ViT-T/16	n/a	28.77	23.45	21.05	25.41	25.45	25.22	63.89	40.47	29.62	22.97
	ViT-S/16	+0.30	+0.70	+1.02	+2.76	+0.54	+0.69	+7.01	+3.77	+2.92	+0.42
	ViT-B/16	+0.86	−0.18	−1.11	+1.15	+0.47	+0.81	+5.09	+4.32	+3.55	−1.57
ViT-S/16	n/a	24.27	20.48	18.86	21.85	21.90	22.96	47.34	52.09	43.85	23.87
	ViT-T/16	+9.64	+6.65	+6.29	+8.51	+10.66	+7.58	+40.53	+32.27	+26.10	+12.35
	ViT-B/16	+3.10	+1.21	+2.87	+3.75	+3.19	+3.36	+19.07	+17.44	+13.81	+4.73
ViT-B/16	n/a	23.05	19.67	18.30	21.83	21.96	21.52	36.44	44.58	51.27	21.22
	ViT-T/16	+10.13	+9.71	+8.42	+10.71	+11.30	+9.89	+51.72	+44.99	+34.69	+20.01
	ViT-S/16	+7.30	+4.57	+4.53	+4.75	+6.21	+6.34	+36.47	+35.66	+29.43	+12.00

We provide one interpretation of this observation. Smaller models might tend to focus more on learning semantic features for generalization, as they lack the capacity of larger models to learn imperceptible features. Therefore, when a source model is aligned with a smaller model, it is steered towards learning more semantic features that are commonly shared across different models, thereby leading to more transferable perturbations.

More Witness Models, Higher Transferability

Evaluations in Table 6.3 focus on alignment using a single witness model. However, aligning the source model with multiple witness models can encourage learning features extracted by a group of witness models, potentially increasing transferability even further. This approach also serves as an effective strategy to prevent overfitting to a single witness model. Table 6.5 demonstrates the results of using an increasing number of Res18 models to align with Res50. While we do observe that a greater number of witness models tends to result in more transferable perturbations, the improvement is modest. This result suggests that simply quadrupling the number of witness models, without considering their diversity, does not lead to a proportional improvement in transferability. Developing a strategy for selecting optimal number and type of witness models represents an interesting direction for future research.

Alignment in the Embedding Space Can Further Improve Transferability.

Recent advances in knowledge distillation highlight the potential benefits of aligning intermediate representations over aligning outputs (Ma et al., 2022c; Chen et al., 2021; Park et al., 2019). Mo-

Table 6.5: **Aligning the Res50 source model with multiple Res18 as witness models (%).** Using an increasing number of Res18 witness models during the alignment process results in a modest improvement in transferability for adversarial examples generated from the aligned Res50. This result suggests that simply quadrupling the number of witness models, without considering their diversity, does not lead to a proportional improvement in transferability.

# Witness	Target									
	Res18	Res50	Res101	VGG19	DN121	IncV3	ViT-T/16	ViT-S/16	ViT-B/16	SWIN
n/a	44.52	61.51	53.01	43.47	50.59	36.05	25.58	21.54	18.86	22.65
1	+35.30	+26.56	+25.20	+24.38	+27.86	+19.18	+5.49	+2.85	+1.61	+9.60
2	+35.01	+27.83	+24.82	+24.50	+27.88	+19.31	+8.28	+3.82	+1.19	+10.67
3	+34.69	+27.69	+27.39	+24.37	+29.44	+21.51	+8.19	+4.49	+3.27	+11.74
4	+35.57	+27.63	+27.94	+25.12	+28.61	+21.81	+8.29	+5.00	+3.78	+12.15

Table 6.6: **Improving transferability via embedding space alignment (%).** We focus on Res50 and ViT-B/16 as source models, and they are aligned using a Res18 and a ViT-T/16 as the witness model, respectively. Both models demonstrate improved adversarial transferability when alignment is performed directly in the embedding space.

Source	Method	Target									
		Res18	Res50	Res101	VGG19	DN121	IncV3	ViT-T/16	ViT-S/16	ViT-B/16	SWIN
Res50	n/a	44.52	61.51	53.01	43.47	50.59	36.05	25.58	21.54	18.86	22.65
	KL	+35.30	+26.56	+25.20	+24.38	+27.86	+19.18	+5.49	+2.85	+1.61	+9.60
	RKD	+2.95	+2.12	+2.80	+2.78	+3.35	+4.09	−1.26	−0.53	−0.36	+0.21
	EGA	+35.35	+23.56	+27.09	+24.69	+28.94	+22.00	+10.12	+4.62	+3.80	+10.37
	HINT	+5.95	+6.55	+5.27	+5.43	+9.54	+4.34	+2.55	−0.01	+1.24	+2.58
	NCE	+21.51	+19.25	+20.39	+16.06	+19.70	+13.29	+6.78	+3.38	+1.06	+6.73
ViT-B/16	n/a	23.05	19.67	18.30	21.83	21.96	21.52	36.44	44.58	51.27	21.22
	KL	+10.13	+9.71	+8.42	+10.71	+11.30	+9.89	+51.72	+44.99	+34.69	+20.01
	RKD	+13.40	+8.47	+8.79	+11.54	+10.25	+11.37	+46.07	+43.41	+34.02	+17.51
	EGA	+12.55	+10.21	+9.21	+11.39	+9.30	+12.05	+53.08	+44.65	+33.50	+21.40
	HINT	+12.78	+10.02	+9.84	+12.21	+12.07	+12.31	+53.41	+47.05	+38.50	+25.51
	NCE	+8.90	+8.48	+7.16	+9.00	+7.26	+7.92	+45.09	+41.84	+32.57	+15.47

tivated by these findings, our study investigates model alignment on the hidden representations of the source model and the witness model. We focus on Res50 and ViT-B/16 as source models, and they are aligned using a Res18 and a ViT-T/16 as the witness model, respectively. We set q in (6.1) to the layer just before the fully-connected layer. We evaluate four embedding-space distillation methods: Relational Knowledge Distillation (RKD) (Park et al., 2019), Embedding Graph Alignment (EGA) (Ma et al., 2022c), Intermediate-level Hints (HINT) (Romero et al., 2014) and Noise Contrastive Estimation (NCE) (Chen et al., 2021). Those methods correspond to different choices of distance metric d in (6.1).

The results are summarized in Table 6.6, with the second row representing output-space alignment using KL divergence for comparison. We observe that both models can benefit from the embedding-space alignment methods. These promising results pave the way for further exploration of embedding-space model alignment in the context of improving adversarial transferability.

Table 6.7: **Compatibility of model alignment with different attack algorithms (%).** The Res50 and ViT-B/16 source models are aligned with a Res18 and ViT-T/16 witness model, respectively. Each entry begins with the error rate for perturbations generated by the original source model. Then, this is followed by the change in the error rate after model alignment. The results demonstrate that the model alignment process enhances the effectiveness of all attack algorithms.

Source	Attack	Target					
		Res18	VGG19	DN121	IncV3	ViT-B/16	SWIN
Res50	MI	75.54 + 20.23	71.46 + 19.97	79.60 + 16.31	59.08 + 22.22	25.19 + 7.87	37.04 + 16.02
	NI	59.71 + 21.99	56.30 + 19.14	59.56 + 19.90	46.45 + 15.59	26.66 + 3.62	28.62 + 8.41
	VMI	85.26 + 12.70	83.16 + 13.13	88.09 + 9.72	71.98 + 16.68	29.62 + 8.32	49.13 + 15.45
	VNI	86.92 + 11.18	85.64 + 11.36	90.47 + 8.21	73.60 + 16.40	29.15 + 8.58	46.64 + 16.75
	SINI	67.43 + 17.05	61.73 + 11.87	65.82 + 13.34	53.74 + 11.96	26.07 + 3.03	30.59 + 5.29
	TI	71.31 + 21.67	64.87 + 20.13	73.95 + 16.85	53.13 + 19.18	25.58 + 7.59	30.00 + 11.71
	DI	87.95 + 11.03	82.12 + 15.88	90.02 + 9.40	77.78 + 15.45	27.48 + 8.69	43.20 + 21.83
ViT-B/16	MI	33.73 + 15.60	35.24 + 12.06	32.29 + 13.51	33.97 + 11.85	71.48 + 24.13	33.57 + 20.87
	NI	35.10 + 6.93	35.58 + 4.58	31.93 + 7.44	33.67 + 5.70	53.55 + 17.73	30.69 + 8.60
	VMI	35.59 + 18.65	37.82 + 14.72	34.55 + 16.12	34.42 + 15.15	77.20 + 19.14	37.63 + 21.75
	VNI	33.99 + 22.45	37.19 + 17.16	32.59 + 19.92	33.19 + 20.83	80.07 + 17.44	38.19 + 21.89
	SINI	36.68 + 11.40	36.56 + 7.49	33.12 + 9.63	35.87 + 9.70	62.07 + 9.97	33.02 + 10.26
	TI	36.85 + 29.96	32.74 + 24.05	35.48 + 25.98	32.99 + 21.31	67.16 + 27.23	32.50 + 25.52
	DI	41.75 + 30.79	39.03 + 30.42	41.40 + 26.69	38.62 + 30.51	84.10 + 13.70	42.09 + 29.79

Model Alignment is Compatible with Other Transfer-enhancing Methods.

To demonstrate that model alignment is compatible with a wide range of attack algorithms, we extend our analysis to include additional transfer-enhancing attacks. This includes optimization-based methods such as MI-FGSM (Dong et al., 2018), NI-FGSM/SINI-FGSM (Lin et al., 2020) and VMI-FGSM/VNI-FGSM (Wang et al., 2021b). We also include data-augmentation-based methods such as TI-FGSM Dong et al., 2019b and DI-FGSM (Xie et al., 2019). Moreover, we consider a larger ℓ_∞-norm constraint of $\epsilon = 8/255$ with a step size of $\alpha = 2/255$, and double the number of iterations to 40.

The results are summarized in Table 6.7. Each entry begins with the error rate for perturbations generated by the original source model. Then, this is followed by the change in the error rate after model alignment. We observe that all considered attack algorithms can leverage the aligned source model to improve transferability. In addition to the attacks included in the table, we also consider LinBP (Guo et al., 2020b), a model-modification-based attack, which does not require the training of additional models. The implementation of LinBP is readily available for Res50. When using a standard Res50 as the target model, MI-FGSM-LinBP-generated perturbations from the aligned Res50 show a 3.59% increase in transferability compared to those from the original, unaligned Res50.

Model modification-based attacks: In addition to the attacks included in the table, we show that model alignment can be integrated with LinBP (Guo et al., 2020b) and BPA (Wang et al., 2024), two model-modification-based methods. We compare the transferabilty of MI-FGSM-LinBP and PGD-BPA perturbations from the aligned Res50 to those from the original, unaligned Res50. In Table 6.8, we observe increased transferability across all target models, with an average increase of

Table 6.8: **Compatibility of model alignment with model modification-based transfer-enhancing attack algorithms (%).** We focus on Res50 source models, which is aligned with a Res18 as the witness model. The aligned models are denoted as *Res*50∗. Model alignment can be integrated with model modification-based methods.

Attack	Source	Target						
		Res18	Res50	VGG19	DN121	IncV3	ViT-T	SWIN
MI-FGSM-LinBP	Res50	87.89	93.55	84.08	88.57	71.19	43.16	46.19
	Res50*	97.27	96.27	93.07	95.61	86.04	56.25	57.71
PGD-BPA	Res50	51.06	63.98	51.54	58.24	34.02	12.0	11.04
	Res50*	76.76	81.98	70.62	77.98	53.18	19.3	20.06

Table 6.9: **Improved transferability from using aligned models in an ensemble (%).** Perturbations from the aligned model (Res50*) demonstrate higher transferability compared to those from an ensemble of source and witness models.

Source	Target									
	Res18	Res50	Res101	VGG19	DN121	IncV3	ViT-T/16	ViT-S/16	ViT-B/16	SWIN
Res18	87.53	67.411	52.074	58.85	67.12	50.22	31.88	24.65	19.16	30.91
Res50	66.97	85.02	72.14	64.88	75.89	49.56	32.14	25.23	22.58	28.96
Res50*	92.03	95.73	89.86	90.60	92.51	71.42	44.12	33.24	25.51	49.12
Res50 + Res18	91.25	91.56	80.60	75.90	85.36	60.11	19.01	15.95	10.90	27.10
2×Res50*	98.70	99.44	98.32	96.09	98.71	84.65	38.98	25.71	15.42	57.30
2×(Res50 + Res18)	97.69	98.73	96.81	92.95	98.32	78.61	23.87	18.06	12.25	41.77

9.66% and 16.86%, respectively. These results demonstrate that model alignment can be effectively combined with model modification-based attacks to further improve transferability.

Note that the evaluation data selection strategy in Table 6.8 differs slightly from that in Table 6.7, where perturbations are generated from both the source and target models, and transferability evaluations are based on those misclassified by their originating models. Our evaluation considers a wide range of target model architectures. However, since the attack algorithms are only available for limited architectures, making it difficult to generate perturbations from some target models. As such, results for MI-FGSM-LinBP and PGD-BPA are evaluated on 1000 randomly selected inputs and are not included in Table 6.7 with other attacks.

Ensemble attacks: Given that an additional witness model is involved in the attack process, it is important to compare perturbations produced by an aligned model with those generated by an ensemble of the original source and witness model. In Table 6.9, we first demonstrate transferability using the original Res18 and Res50, followed by a Res50 aligned with Res18 (Res50*). Next, we incorporate the ensemble in logits scheme (Dong et al., 2018) into the PGD attack, and consider three scenarios: an ensemble of the source and witness models (Res50+Res18), an ensemble of two aligned models (2×Res50*), and an ensemble of two pairs of source-witness models (2×(Res50+Res18)).

We make two observations. First, perturbations from a single aligned model demonstrate higher transferability compared to those from an ensemble of source and witness models, and an ensemble of two aligned models not only further improves transferability but also surpasses the performance of the four-model ensemble. Beyond the improved transferability, model alignment brings the added

benefits of faster inference times and lower memory requirements, presenting a significant advantage over conventional ensemble approaches. Second, when targeting ViTs, perturbations generated from ResNet ensembles exhibit limited transferability. For instance, using an ensemble of aligned Res50 models underperforms compared to using just a single aligned Res50. This observation is in line with the previous result (Ma et al., 2023b), underscoring the specific challenges in transferring attacks between different architectures.

6.6 Conclusions

In this work, we proposed model alignment as a novel perspective in improving the transferability of adversarial examples. During alignment, the parameters of the source model are fine-tuned to minimize an alignment loss which measures the divergence in the predictions between the source and the witness model. We conduct a geometric analysis to study the changes in the loss landscape resulting from this process to better understand the underlying effect of model alignment. Extensive experiments on the ImageNet dataset demonstrate that perturbations generated from aligned source models exhibit significantly higher transferability than those from the original source model.

6.6.1 Challenges and Limitations

Theoretical Analysis: A limitation of our study is the absence of a theoretical framework to understand the model alignment process, and we consider it as future work. Our analysis is primarily empirical, focusing on observing the perturbation differences and studying the effects of model alignment on the loss surface. Having a theoretical framework to understand our method could be helpful in developing new training strategies, which could directly result in source models that achieve the same outcomes as those attained through the model alignment process.

Limited Improvement on ViTs: We observed that when CNN-based models are aligned using ViTs, the improvement in adversarial transferability across most target models is less significant compared to alignment using CNN-based models. However, the inverse is not true: ViT-based models aligned using CNNs show a substantial improvement in transferability across all target models. Future work could investigate the reasons for this discrepancy and develop alignment strategies specifically designed for cross-architecture alignment.

Appendices

6.A Implementation Details

In the following, we provide additional details on the training configurations used in our experiments. The source code for our method is available at https://github.com/averyma/model-alignment. For all models considered in this work, we adhere to the optimization configurations as detailed in the official PyTorch repository.[1]

CNN-based models: All CNN-based models are trained using the same configurations. Those models include ResNet18 (Res18), ResNet50 (Res50), ResNet101 (Res101), VGG19, DenseNet121

[1] https://github.com/pytorch/vision/tree/main/references/classification

(DN121), and Inception-v3 (IncV3). They are trained for 90 epochs using SGD with an initial learning rate of 0.1 and a momentum coefficient of 0.9. We start training with a 5-epoch learning rate warmup, followed by a cosine decay schedule. The batch size is set at 256.

ViT-based models: The ViT-based models include ViT-T/16, ViT-S/16, ViT-B/16, and Swin Transformers (SWIN). They are all trained for 300 epochs using AdamW. The initial learning rates are set at 0.003 for ViT models and 0.001 for SWIN. For the ViT's, training begins with a 30-epoch learning rate warmup, followed by a cosine decay schedule, whereas for SWIN, the warmup is 20 epochs. The batch size is set at 1024. We use label smoothing during training. To ensure training stability, the global gradient norm is clipped at 1.

Data augmentations: For the training of all CNN-based models, augmentation techniques are random resizing, cropping, and flipping. For the training of all ViT-based models, we further incorporate Mixup (Zhang et al., 2018) and Cutmix (Yun et al., 2019). Only random resizing, cropping and flipping are used during the alignment process. Specifically, RandAugment (Cubuk et al., 2020) is applied to ViT's, while TrivialAugment (Müller et al., 2021) and random erasing (Zhong et al., 2020) are applied to SWIN.

Model definitions: All model definitions are obtained from the torchvision library (maintainers et al., 2016), with the exceptions of IncV3, ViT's, and SWIN, which are obtained from the timm library (Wightman, 2019).

Attack algorithms: All the attack algorithms used in this work are provided by the Torchattacks library (Kim, 2020).

Alignment in the embedding space: When using a witness model with a different architecture than the source model, the dimensions of their hidden representations are likely to be different, i.e., $\dim(z_s^{[q_s]}(x)) \neq \dim(z_w^{[q_w]}(x))$, where q_s and q_w represent the layers just before the fully-connected layer in their respective models. To address this dimensional mismatch, we follow previous work (Park et al., 2019; Ma et al., 2022c; Romero et al., 2014; Chen et al., 2021) and apply a linear projection to z_s so its dimension matches that of z_w. The weights of the linear projection are treated as trainable parameters during the alignment process.

6.B Experiments on Additional Datasets

We focus on the ImageNet dataset in Section 6.5. Here, we supplement our results with experiments on Stanford Cars (Krause et al., 2013) and Food101 (Bossard et al., 2014). On Stanford Cars, SWIN is aligned using Res50. On Food101, Res50 is aligned using ConvNeXt-B (Liu et al., 2022). Results in Table 6.10 show that the improved transferability achieved through model alignment is also evident on other datasets.

6.C Improved Transferability on Defended Models

Evaluations in Section 6.5 are based on models without any defense mechanism. That is, all target models are trained by minimizing the cross-entropy loss on the unperturbed training data. To further demonstrate our approach, we evaluate using a normally trained Res50 with five defense mechanisms, Bit-Red Xu et al., 2018, JPEG Guo et al., 2018, FD Liu et al., 2019, RS Cohen et al., 2019, NRP Naseer et al., 2020. We adhere to the exact configuration of the defense methods

Table 6.10: **Improved transferability via model alignment on other datasets (%).** we supplement our results with experiments on Stanford Cars (Krause et al., 2013) and Food101 (Bossard et al., 2014). Aligned models are denoted using *. Results show that the improved transferability achieved through model alignment is also evident on additional datasets.

Dataset	Source	Target	
		ConvNeXt-B	ViT-B
Stanford Cars	SWIN	68.77	17.6
	SWIN*	88.89	50.67

Dataset	Source	Target	
		SWIN	ViT-B
Food101	Res50	26.67	12.36
	Res50*	31.88	16.16

Table 6.11: **Improved transferability on models equipped with defense methods (%).** We supplement our results by evaluating the transferability of perturbations on models equipped with defense mechanisms. In particular, we evaluate using a normally trained Res50 with five defense mechanisms. We adhere to the exact configuration of the defense methods as described in (Xie et al., 2019). We also consider an adversarially trained Res50 (AT) (Wong et al., 2019a).

Source	Defense						
	n/a	AT	Bit-Red	JPEG	FD	RS	NRP
Res50	61.51	48.06	49.22	38.81	43.77	35.41	33.01
Res50*	88.07	52.15	54.70	53.03	50.59	43.99	41.19

as described in Xie et al. (2019). We also consider an adversarially trained Res50 (AT) (Wong et al., 2019a). Results in Table 6.11 show that although the defense methods can generally reduce transferability, using the aligned model still results in a significant improvement in transferability compared to the original source model.

Chapter 7

Concluding Remarks

In this thesis, we investigated adversarial robustness in deep learning from multiple perspectives. We began by focusing on the optimization process of training neural networks–both in terms of objectives and algorithms. In Chapter 3, we introduced SOAR, a novel training objective that improves adversarial robustness without relying on computationally expensive adversarial data generation, and we empirically validated its effectiveness on image datasets. In Chapter 4, we analyzed the effect of optimization algorithms on model robustness, showing that models trained with SGD are more robust than those trained with adaptive methods such as Adam or RMSProp. As datasets grow larger and more structured, we recognized the importance of hierarchical relationships among classes. Thus, in Chapter 5, we introduced the concept of hierarchical adversarial robustness, proposed methods for generating hierarchical adversarial perturbations, and presented an architectural strategy to defend against them. Finally, we turned to the practical issue of perturbation transferability. In Chapter 6, we proposed a fine-tuning approach called model alignment to enhance the transferability of adversarial perturbations. Altogether, these contributions explore multiple key facets of the adversarial robustness phenomenon, offering insights that advance our knowledge of adversarial robustness and pave a path forward toward more robust deep learning systems.

7.1 Suggestions for Future Research

At the end of each chapter, we discussed the implications of our findings and suggested potential directions for future research. Here, we highlight some of the most promising areas for further investigation.

Understanding the Role of Normalization in Adversarial Robustness

In Chapter 3, we observed that networks with Batch Normalization layers do not benefit from SOAR in terms of adversarial robustness. Similarly, in Chapter 4, we found that models with Batch Normalization exhibit significantly lower accuracy under perturbations compared to those without it. Normalization has become a standard practice in training deep neural networks, with techniques such as Layer Normalization (Ba, 2016) widely adopted in training transformers (Brown et al., 2020; Radford et al., 2021). While some studies have explored the connection between Batch Normalization and model robustness (Galloway et al., 2019), a comprehensive investigation into the

relationship between various normalization techniques and adversarial robustness remains lacking. Similar to our focus in Chapter 4, where we analyzed how optimization—a key element of the training pipeline—affects adversarial robustness, future research could focus on how different normalization strategies influence adversarial robustness.

Analysis Beyond Linear Models

In Chapters 3 and 4, our theoretical analyses relied exclusively on linear models. While these models offer valuable insights, it is important to acknowledge the inherent non-linearity of neural networks. This reliance on linear approximations limits the depth of understanding that can be achieved regarding the complex dynamics of neural networks. Future research could address this gap by leveraging techniques like neural tangent kernels (Jacot et al., 2018), which have been instrumental in theoretically analyzing neural network learning dynamics, and tools such as saliency maps (Simonyan et al., 2013; Ma et al., 2022a), which provide quantitative insights into how models focus on specific parts of the input when making decisions. These approaches could offer a more comprehensive understanding of neural network learning dynamics.

Theoretical Understanding of Model Alignment

In Chapter 6, we proposed model alignment as a novel approach to improving the transferability of adversarial examples. While our empirical results demonstrate the effectiveness of model alignment, a theoretical understanding of the model alignment process is still lacking. Model alignment can benefit from a theoretical framework in three ways. First, a theoretical analysis could provide insights into the underlying mechanisms of model alignment, helping to develop new training strategies that achieve the same outcomes as model alignment without the need for fine-tuning. This is similar to how SOAR was developed by theoretically building the connection between regularized training and data augmentation with adversarial perturbations. Second, a theoretical framework could guide the selection of witness models and the design of alignment strategies tailored to specific architectures. Finally, a theoretical understanding of model alignment provides insight into the limitations of the approach and identifies scenarios where it may not be effective.

Cross-architecture Model Alignment

An interesting observation in Chapter 6 is that model alignment for CNN-based source models is more effective when the witness model is also a CNN, compared to when it is a transformer. This discrepancy highlights the significance of network designs in the success of model alignment. Future research could investigate the role of model architecture in shaping the effectiveness of model alignment and propose alignment strategies specifically tailored to different architectures.

Adversarial Robustness in Generative Models

In this thesis, we focused on adversarial robustness in the context of classification. However, the rise of generative models such as vision language models (VLMs) (Lu et al., 2019; Li et al., 2022a) and large language models (LLMs) (Kenton et al., 2019; Brown et al., 2020; Dubey et al., 2024), has introduced new challenges and opportunities for adversarial robustness research. Unlike traditional

classifiers, generative models operate in more open-ended settings, producing free-form outputs. This means adversaries are no longer limited to inducing misclassifications, but can also generate unsafe or unethical text (Zou et al., 2023; Ma et al., 2025) and images (Zhao et al., 2023; Zhou et al., 2024). As the field evolves, extending adversarial robustness research beyond classification to encompass generative tasks will be essential for ensuring the safe and responsible deployment of these powerful models.

Bibliography

Naman Agarwal, Rohan Anil, Elad Hazan, Tomer Koren, and Cyril Zhang (2020). "Disentangling adaptive gradient methods from learning rates". In: *arXiv preprint arXiv:2002.11803*.

Nasir Ahmed, T Natarajan, and Kamisetty R Rao (1974). "Discrete cosine transform". In: *IEEE transactions on Computers*.

Shun-ichi Amari, Jimmy Ba, Roger Grosse, Xuechen Li, Atsushi Nitanda, Taiji Suzuki, Denny Wu, and Ji Xu (2021). "When does preconditioning help or hurt generalization?" In: *International Conference on Learning Representations (ICLR)*.

Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein (2020). "Square attack: a query-efficient black-box adversarial attack via random search". In: *Proceedings of the European Conference on Computer Vision (ECCV)*.

Apple (2021). *CSAM Detection: Technical Summary*. https://www.apple.com/child-safety/pdf/CSAM_Detection_Technical_Summary.pdf.

Anish Athalye (2021). *NeuralHash Collider*. https://github.com/anishathalye/neural-hash-collider.

Anish Athalye, Nicholas Carlini, and David Wagner (2018). "Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples". In: *International Conference on Machine Learning (ICML)*.

Hagai Attias and Christoph Schreiner (1996). "Temporal low-order statistics of natural sounds". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Jimmy Lei Ba (2016). "Layer normalization". In: *arXiv preprint arXiv:1607.06450*.

Tao Bai, Jinqi Luo, Jun Zhao, Bihan Wen, and Qian Wang (2021). "Recent advances in adversarial training for adversarial robustness". In: *Proceedings of the International Joint Conferences on Artificial Intelligence (IJCAI)*.

Yatong Bai, Brendon G Anderson, Aerin Kim, and Somayeh Sojoudi (2023). "Improving the Accuracy-Robustness Trade-off of Classifiers via Adaptive Smoothing". In: *SIAM Journal on Mathematics of Data Science*.

Baidu (2017). *Apollo*. https://github.com/ApolloAuto/.

Lukas Balles and Philipp Hennig (2018). "Dissecting Adam: The sign, magnitude and variance of stochastic gradients". In: *International Conference on Machine Learning (ICML)*.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston (2009). "Curriculum learning". In: *International Conference on Machine Learning (ICML)*.

Philipp Benz, Chaoning Zhang, Adil Karjauv, and In So Kweon (2021a). "Revisiting batch normalization for improving corruption robustness". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*.

Philipp Benz, Chaoning Zhang, and In So Kweon (2021b). "Batch normalization increases adversarial vulnerability and decreases adversarial transferability: A non-robust feature perspective". In: *Proceedings of the International Conference on Computer Vision (ICCV)*.

Kush Bhatia, Kunal Dahiya, Himanshu Jain, Purushottam Kar, Anshul Mittal, Yashoteja Prabhu, and Manik Varma (2016). *The extreme classification repository: multi-label datasets and code*. http://manikvarma.org/downloads/XC/XMLRepository.html.

Chris M Bishop (1995). "Training with noise is equivalent to Tikhonov regularization". In: *Neural computation*.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool (2014). "Food-101–mining discriminative components with random forests". In: *Proceedings of the European Conference on Computer Vision (ECCV)*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. (2020). "Language models are few-shot learners". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Samuel Burer and Adam N Letchford (2009). "On nonconvex quadratic programming with box constraints". In: *SIAM Journal on Optimization*.

Anirudh Buvanesh, Rahul Chand, Jatin Prakash, Bhawna Paliwal, Mudit Dhawan, Neelabh Madan, Deepesh Hada, Vidit Jain, Sonu Mehta, Yashoteja Prabhu, et al. (2022). "Enhancing Tail Performance in Extreme Classifiers by Label Variance Reduction". In: *International Conference on Learning Representations (ICLR)*.

Junyoung Byun, Seungju Cho, Myung-Joon Kwon, Hee-Seon Kim, and Changick Kim (2022). "Improving the transferability of targeted adversarial examples through object-based diverse input". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Qi-Zhi Cai, Chang Liu, and Dawn Song (2018). "Curriculum adversarial training". In: *Proceedings of the International Joint Conferences on Artificial Intelligence (IJCAI)*.

Nicholas Carlini and David Wagner (2017). "Towards evaluating the robustness of neural networks". In: *Proceedings of the IEEE Symposium on Security and Privacy (SP)*.

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

Luigi Carratino, Moustapha Cissé, Rodolphe Jenatton, and Jean-Philippe Vert (2022). "On mixup regularization". In: *Journal of Machine Learning Research (JMLR)*.

Varun Chandrasekaran, Brian Tang, Nicolas Papernot, Kassem Fawaz, Somesh Jha, and Xi Wu (2019). "Rearchitecting classification frameworks for increased robustness". In: *arXiv preprint arXiv:1905.10900*.

Zachary Charles, Harrison Rosenberg, and Dimitris Papailiopoulos (2019). "A geometric perspective on the transferability of adversarial directions". In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Yanbei Chen, Yongqin Xian, A Koepke, Ying Shan, and Zeynep Akata (2021). "Distilling audio-visual knowledge by compositional contrastive learning". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Shuyu Cheng, Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu (2019). "Improving black-box adversarial attacks with a transfer-based prior". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Lianhua Chi and Xingquan Zhu (2017). "Hashing techniques: A survey and taxonomy". In: *ACM Computing Surveys (CSUR)*.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter (2019). "Certified adversarial robustness via randomized smoothing". In: *International Conference on Machine Learning (ICML)*.

comma.ai (2018). *Openpilot*. https://github.com/ApolloAuto/.

Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele (2016). "The cityscapes dataset for semantic urban scene understanding". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Edoardo Debenedetti, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein (2021). "RobustBench: a standardized adversarial robustness benchmark". In: *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track (Round 2)*.

Francesco Croce and Matthias Hein (2019). "Minimally distorted adversarial examples with a fast adaptive boundary attack". In: *International Conference on Machine Learning (ICML)*.

— (2020). "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks". In: *International Conference on Machine Learning (ICML)*.

Ekin Dogus Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc Le (2019). "AutoAugment: Learning augmentation strategies from data". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ekin Dogus Cubuk, Barret Zoph, Jon Shlens, and Quoc Le (2020). "RandAugment: Practical automated data augmentation with a reduced search space". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Wei Dai, Chia Dai, Shuhui Qu, Juncheng Li, and Samarjit Das (2017). "Very deep convolutional neural networks for raw waveforms". In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Jia Deng, Sanjeev Satheesh, Alexander C Berg, and Fei Li (2011). "Fast and balanced: Efficient label tree learning for large scale object recognition". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Yian Deng and Tingting Mu (2024). "Understanding and improving ensemble adversarial defense". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Francis X Diebold (1998). *Elements of forecasting*. Thomson South-Western.

Gavin Weiguang Ding, Yash Sharma, Kry Yik Chau Lui, and Ruitong Huang (2018). "Max-margin adversarial (mma) training: Direct input space margin maximization through adversarial training". In: *International Conference on Learning Representations (ICLR)*.

Yin Dong, Raphael Gontijo Lopes, Jon Shlens, Ekin Dogus Cubuk, and Justin Gilmer (2019a). "A Fourier perspective on model robustness in computer vision". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li (2018). "Boosting adversarial attacks with momentum". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu (2019b). "Evading defenses to transferable adversarial examples by translation-invariant attacks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby (2021). "An image is worth 16x16 words: Transformers for image recognition at scale". In: *International Conference on Learning Representations (ICLR)*.

Harris Drucker and Yann Le Cun (1992). "Improving generalization performance using double back-propagation". In: *IEEE Transactions on Neural Networks*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. (2024). "The llama 3 herd of models". In: *arXiv preprint arXiv:2407.21783*.

John Duchi, Elad Hazan, and Yoram Singer (2011). "Adaptive subgradient methods for online learning and stochastic optimization." In: *Journal of Machine Learning Research (JMLR)*.

Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song (2018). "Robust physical-world attacks on deep learning visual classification". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard (2017). "The robustness of deep networks: A geometrical perspective". In: *IEEE Signal Processing Magazine*.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur (2021). "Sharpness-aware minimization for efficiently improving generalization". In: *International Conference on Learning Representations (ICLR)*.

Joseph Fourier (1822). *Théorie analytique de la chaleur (The Analytic Theory of Heat)*.

Angus Galloway, Anna Golubeva, Thomas Tanay, Medhat Moussa, and Graham W Taylor (2019). "Batch normalization is a cause of adversarial vulnerability". In: *arXiv preprint arXiv:1905.02161*.

Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun (2013). "Vision meets robotics: The kitti dataset". In: *International Journal of Robotics Research*.

Justin Gilmer, Luke Metz, Fartash Faghri, Samuel S Schoenholz, Maithra Raghu, Martin Wattenberg, and Ian Goodfellow (2018). "Adversarial spheres". In: *arXiv preprint arXiv:1801.02774*.

Xavier Glorot and Yoshua Bengio (2010). "Understanding the difficulty of training deep feedforward neural networks". In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Zhitao Gong, Wenlu Wang, and Wei-Shinn Ku (2017). "Adversarial and clean data are not twins". In: *arXiv preprint arXiv:1704.04960*.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville (2016). *Deep Learning*. http://www.deeplearningbook.org. MIT Press.

Ian Goodfellow, Jonathon Shlens, and Christian Szegedy (2015). "Explaining and harnessing adversarial examples". In: *International Conference on Learning Representations (ICLR)*.

Ian Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio (2013). "Maxout networks". In: *International Conference on Machine Learning (ICML)*.

Henry Gouk, Eibe Frank, Bernhard Pfahringer, and Michael J Cree (2021). "Regularisation of neural networks by enforcing Lipschitz continuity". In: *Machine Learning*.

Herbert Gross (2005). *Handbook of Optical Systems: Fundamentals of Technical Optics*.

Kathrin Grosse, Praveen Manoharan, Nicolas Papernot, Michael Backes, and Patrick McDaniel (2017). "On the (statistical) detection of adversarial examples". In: *arXiv preprint arXiv:1702.06280*.

Jindong Gu, Xiaojun Jia, Pau de Jorge, Wenqain Yu, Xinwei Liu, Avery Ma, Yuan Xun, Anjun Hu, Ashkan Khakzar, Zhijiang Li, et al. (2023). "A Survey on Transferability of Adversarial Examples across Deep Neural Networks". In: *Transactions on Machine Learning Research (TMLR)*.

Martin Gubri, Maxime Cordy, Mike Papadakis, Yves Le Traon, and Koushik Sen (2022). "LGV: Boosting Adversarial Example Transferability from Large Geometric Vicinity". In: *Proceedings of the European Conference on Computer Vision (ECCV)*.

Suriya Gunasekar, Blake E Woodworth, Srinadh Bhojanapalli, Behnam Neyshabur, and Nati Srebro (2017). "Implicit regularization in matrix factorization". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Chuan Guo, Jared S Frank, and Kilian Q Weinberger (2020a). "Low frequency adversarial perturbation". In: *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*.

Chuan Guo, Jacob R Gardner, Yurong You, Andrew Gordon Wilson, and Kilian Q Weinberger (2019). "Simple black-box adversarial attacks". In: *International Conference on Machine Learning (ICML)*.

Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens Van Der Maaten (2018). "Countering adversarial images using input transformations". In: *International Conference on Learning Representations (ICLR)*.

Yiwen Guo, Qizhang Li, and Hao Chen (2020b). "Backpropagating linearly improves transferability of adversarial examples". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Nilesh Gupta, Sakina Bohra, Yashoteja Prabhu, Saurabh Purohit, and Manik Varma (2021). "Generalized zero-shot extreme multi-label learning". In: *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*.

Trevor Hastie (2009). *The elements of statistical learning: data mining, inference, and prediction*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2015). "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification". In: *Proceedings of the International Conference on Computer Vision (ICCV)*.

— (2016a). "Deep residual learning for image recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

— (2016b). "Identity Mappings in Deep Residual Networks". In: *Proceedings of the European Conference on Computer Vision (ECCV)*.

Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li (2019). "Bag of tricks for image classification with convolutional neural networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky (2012). "Neural networks for machine learning lecture 6a: overview of mini-batch gradient descent". In: *Lecture Notes*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean (2015). "Distilling the knowledge in a neural network". In: *arXiv preprint arXiv:1503.02531*.

Kurt Hornik, Maxwell Stinchcombe, and Halbert White (1989). "Multilayer feedforward networks are universal approximators". In: *Neural Networks*.

Jeremy Howard (2019). *Imagenette*. https://github.com/fastai/imagenette/.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger (2017). "Densely connected convolutional networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Jinggang Huang and David Mumford (1999). "Statistics of natural images and models". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ruitong Huang, Bing Xu, Dale Schuurmans, and Csaba Szepesvári (2015). "Learning with a strong adversary". In: *arXiv preprint arXiv:1511.03034*.

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry (2019). "Adversarial examples are not bugs, they are features". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Sergey Ioffe and Christian Szegedy (2015). "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In: *International Conference on Machine Learning (ICML)*.

Arthur Jacot, Franck Gabriel, and Clément Hongler (2018). "Neural tangent kernel: Convergence and generalization in neural networks". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Ziyu Jiang, Tianlong Chen, Ting Chen, and Zhangyang Wang (2020). "Robust pre-training by adversarial contrastive learning". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Pengfei Jing, Qiyi Tang, Yuefeng Du, Lei Xue, Xiapu Luo, Ting Wang, Sen Nie, and Shi Wu (2021). "Too good to be safe: Tricking lane detection in autonomous driving with crafted perturbations". In: *Proceedings of the USENIX Security Symposium*.

Hamed Karimi, Julie Nutini, and Mark Schmidt (2016). "Linear convergence of gradient and proximal-gradient methods under the Polyak-Łojasiewicz condition". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova (2019). "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Hoki Kim (2020). "Torchattacks: A pytorch repository for adversarial attacks". In: *arXiv preprint arXiv:2010.01950*.

Minseon Kim, Jihoon Tack, and Sung Ju Hwang (2020). "Adversarial self-supervised contrastive learning". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Diederik P Kingma and Jimmy Ba (2015). "Adam: A method for stochastic optimization". In: *International Conference on Learning Representations (ICLR)*.

Stepan Komkov and Aleksandr Petiushko (2021). "Advhat: Real-world adversarial attack on arcface face id system". In: *Proceedings of the International Conference on Pattern Recognition (ICPR)*.

Yassin Kortli, Maher Jridi, Ayman Al Falou, and Mohamed Atri (2020). "Face recognition systems: A survey". In: *Sensors*.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei (2013). "3D object representations for fine-grained categorization". In: *Proceedings of the International Conference on Computer Vision Workshops (ICCVW)*.

Alex Krizhevsky and Geoffrey Hinton (2009). "Learning multiple layers of features from tiny images". In: *Master Thesis*.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton (2012). "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Ram Shankar Siva Kumar, Magnus Nyström, John Lambert, Andrew Marshall, Mario Goertzel, Andi Comissoneru, Matt Swann, and Sharon Xia (2020). "Adversarial machine learning-industry perspectives". In: *IEEE Security and Privacy Workshops (SPW)*.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio (2016). "Adversarial machine learning at scale". In: *International Conference on Learning Representations (ICLR)*.

Yann LeCun (1998). *The MNIST database of handwritten digits*. http://yann.lecun.com/exdb/mnist/.

Hyeungill Lee, Sungyeob Han, and Jungwoo Lee (2017). "Generative adversarial trainer: Defense to adversarial perturbations with gan". In: *arXiv preprint arXiv:1705.03387*.

Beitao Li, Edward Chang, and Yi Wu (2003). "Discovery of a perceptual distance function for measuring image similarity". In: *Multimedia Systems*.

Fei-Fei Li, Rob Fergus, and Pietro Perona (2004). "Learning Generative Visual Models from Few Training Examples: An Incremental Bayesian Approach Tested on 101 Object Categories". In: *Computer Vision and Pattern Recognition Workshop*.

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi (2022a). "BLIP: Bootstrapping language-image pre-training for unified vision-language understanding and generation". In: *International Conference on Machine Learning (ICML)*.

Maosen Li, Cheng Deng, Tengjiao Li, Junchi Yan, Xinbo Gao, and Heng Huang (2020a). "Towards transferable targeted attack". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Qizhang Li, Yiwen Guo, Wangmeng Zuo, and Hao Chen (2022b). "Making Substitute Models More Bayesian Can Enhance Transferability of Adversarial Examples". In: *International Conference on Learning Representations (ICLR)*.

Yanxi Li and Chang Xu (2023). "Trade-off between robustness and accuracy of vision transformers". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yingwei Li, Song Bai, Cihang Xie, Zhenyu Liao, Xiaohui Shen, and Alan Yuille (2020b). "Regional homogeneity: Towards learning transferable universal adversarial perturbations against defenses". In: *Proceedings of the European Conference on Computer Vision (ECCV)*.

Jiadong Lin, Chuanbiao Song, Kun He, Liwei Wang, and John E Hopcroft (2020). "Nesterov accelerated gradient and scale invariance for adversarial attacks". In: *International Conference on Learning Representations (ICLR)*.

Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song (2016). "Delving into Transferable Adversarial Examples and Black-box Attacks". In: *International Conference on Learning Representations (ICLR)*.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo (2021). "Swin transformer: Hierarchical vision transformer using shifted windows". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie (2022). "A convnet for the 2020s". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zihao Liu, Qi Liu, Tao Liu, Nuo Xu, Xue Lin, Yanzhi Wang, and Wujie Wen (2019). "Feature distillation: Dnn-oriented jpeg compression against adversarial examples". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee (2019). "ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Xiao Luo, Haixin Wang, Daqing Wu, Chong Chen, Minghua Deng, Jianqiang Huang, and Xian-Sheng Hua (2023). "A survey on deep hashing methods". In: *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*.

Kaifeng Lyu and Jian Li (2020). "Gradient descent maximizes the margin of homogeneous neural networks". In: *International Conference on Learning Representations (ICLR)*.

Avery Ma, Nikita Dvornik, Ran Zhang, Leila Pishdad, Konstantinos G Derpanis, and Afsaneh Fazly (2022a). "SAGE: Saliency-Guided Mixup with Optimal Rearrangements". In: *British Machine Vision Conference (BMVC)*.

Avery Ma, Fartash Faghri, Nicolas Papernot, and Amir-massoud Farahmand (2020). "SOAR: Second-order adversarial regularization". In: *arXiv preprint arXiv:2004.01832*.

Avery Ma, Amir-massoud Farahmand, Yangchen Pan, Philip Torr, and Jindong Gu (2024). "Improving adversarial transferability via model alignment". In: *Proceedings of the European Conference on Computer Vision (ECCV)*.

Avery Ma, Yangchen Pan, and Amir-massoud Farahmand (2023a). "Understanding the robustness difference between stochastic gradient descent and adaptive gradient methods". In: *Transactions on Machine Learning Research (TMLR)*.

— (2025). "PANDAS: Improving Many-shot Jailbreaking via Positive Affirmation, Negative Demonstration, and Adaptive Sampling". In: *International Conference on Machine Learning (ICML)*.

Avery Ma, Aladin Virmaux, Kevin Scaman, and Juwei Lu (2021). "Improving Hierarchical Adversarial Robustness of Deep Neural Networks". In: *arXiv preprint arXiv:2102.09012*.

Chao Ma, Lei Wu, and Weinan E (2022b). "A qualitative study of the dynamic behavior for adaptive gradient algorithms". In: *Mathematical and Scientific Machine Learning (MSML)*.

Wenshuo Ma, Yidong Li, Xiaofeng Jia, and Wei Xu (2023b). "Transferable adversarial attack for both vision transformers and convolutional networks via momentum integrated gradients". In: *Proceedings of the International Conference on Computer Vision (ICCV)*.

Yuchen Ma, Yanbei Chen, and Zeynep Akata (2022c). "Distilling knowledge from self-supervised teacher by embedding graph alignment". In: *British Machine Vision Conference (BMVC)*.

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

Kaleel Mahmood, Rigel Mahmood, and Marten Van Dijk (2021). "On the robustness of vision transformers to adversarial examples". In: *Proceedings of the International Conference on Computer Vision (ICCV)*.

TorchVision maintainers and contributors (2016). *TorchVision: PyTorch's Computer Vision library*. https://github.com/pytorch/vision.

Stéphane Mallat (1999). *A wavelet tour of signal processing*. Elsevier.

Robert McAulay and Thomas Quatieri (1986). "Speech analysis/synthesis based on a sinusoidal representation". In: *IEEE Transactions on Acoustics, Speech, and Signal Processing*.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, Pascal Frossard, and Stefano Soatto (2017). "Analysis of universal adversarial perturbations". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Jonathan Uesato, and Pascal Frossard (2019). "Robustness via curvature regularization, and vice versa". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Emmanuel Moulay, Vincent Léchappé, and Franck Plestan (2019). "Properties of the sign gradient descent algorithms". In: *Information Sciences*.

Rafael Müller, Simon Kornblith, and Geoffrey E Hinton (2019). "When does label smoothing help?" In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Samuel G Müller and Frank Hutter (2021). "Trivialaugment: Tuning-free yet state-of-the-art data augmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Muzammal Naseer, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Fatih Porikli (2020). "A self-supervised approach for adversarial robustness". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Muzammal Naseer, Kanchana Ranasinghe, Salman Khan, Fahad Shahbaz Khan, and Fatih Porikli (2021). "On improving adversarial transferability of vision transformers". In: *International Conference on Learning Representations (ICLR)*.

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

Alan V Oppenheim, John R Buck, and Ronald W Schafer (2001). *Discrete-time signal processing*. Pearson.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami (2017). "Practical black-box attacks against machine learning". In: *Proceedings of the ACM Asia Conference on Computer and Communications Security (ACM ASIACCS)*.

Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami (2016). "The limitations of deep learning in adversarial settings". In: *IEEE European Symposium on Security and Privacy (EuroS&P)*.

Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho (2019). "Relational knowledge distillation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

ShengYun Peng, Weilin Xu, Cory Cornelius, Matthew Hull, Kevin Li, Rahul Duggal, Mansi Phute, Jason Martin, and Duen Horng Chau (2023). "Robust principles: Architectural design principles for adversarially robust cnns". In: *British Machine Vision Conference (BMVC)*.

William B Pennebaker and Joan L Mitchell (1992). *JPEG: Still image data compression standard*. Springer Science & Business Media.

Mohammad Pezeshki, Oumar Kaba, Yoshua Bengio, Aaron C Courville, Doina Precup, and Guillaume Lajoie (2021). "Gradient starvation: A learning proclivity in neural networks". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Qian Qian and Xiaoyuan Qian (2019). "The implicit bias of adagrad on separable data". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Yaguan Qian, Shuke He, Chenyu Zhao, Jiaqiang Sha, Wei Wang, and Bin Wang (2023). "LEA2: A Lightweight Ensemble Adversarial Attack via Non-overlapping Vulnerable Frequency Regions". In: *Proceedings of the International Conference on Computer Vision (ICCV)*.

Chongli Qin, James Martens, Sven Gowal, Dilip Krishnan, Krishnamurthy Dvijotham, Alhussein Fawzi, Soham De, Robert Stanforth, and Pushmeet Kohli (2019). "Adversarial robustness through local linearization". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Zeyu Qin, Yanbo Fan, Yi Liu, Li Shen, Yong Zhang, Jue Wang, and Baoyuan Wu (2022). "Boosting the transferability of adversarial attacks with reverse adversarial perturbation". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Rahul Rade and Seyed-Mohsen Moosavi-Dezfooli (2022). "Reducing excessive margin to achieve a better accuracy vs. robustness trade-off". In: *International Conference on Learning Representations (ICLR)*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. (2021). "Learning transferable visual models from natural language supervision". In: *International Conference on Machine Learning (ICML)*.

Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy (2021). "Do vision transformers see like convolutional neural networks?" In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Aditi Raghunathan, Sang Michael Xie, Fanny Yang, John C Duchi, and Percy Liang (2020). "Understanding and mitigating the tradeoff between robustness and accuracy". In: *International Conference on Machine Learning (ICML)*.

Sashank J Reddi, Satyen Kale, and Sanjiv Kumar (2018). "On the Convergence of Adam and Beyond". In: *International Conference on Learning Representations (ICLR)*.

Martin Riedmiller and Heinrich Braun (1993). "A direct adaptive method for faster backpropagation learning: The RPROP algorithm". In: *IEEE International Conference on Neural Networks*.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio (2014). "Fitnets: Hints for thin deep nets". In: *arXiv preprint arXiv:1412.6550*.

Daniel L Ruderman (1994). "The statistics of natural images". In: *Network: Computation in Neural Systems*.

Anindya Sarkar, Anirban Sarkar, Sowrya Gali, and Vineeth N Balasubramanian (2021). "Adversarial robustness without adversarial training: A teacher-guided curriculum learning approach". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

A. van der Schaaf and J.H. van Hateren (1996). "Modelling the power spectra of natural images: statistics and information". In: *Vision Research*.

Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry (2018). "Adversarially robust generalization requires more data". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Odelia Schwartz and Eero P Simoncelli (2001). "Natural signal statistics and sensory gain control". In: *Nature Neuroscience*.

Steven H Schwartz (2004). *Visual perception: A clinical orientation*. McGraw-Hill Medical Pub. Division.

Uri Shaham, Yutaro Yamada, and Sahand Negahban (2018). "Understanding adversarial training: Increasing local stability of supervised models through robust optimization". In: *Neurocomputing*.

Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter (2016). "Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition". In: *Proceedings of the ACM Conference on Computer and Communications Security (ACM CCS)*.

Yash Sharma, Gavin Weiguang Ding, and Marcus Brubaker (2019). "On the effectiveness of low frequency perturbations". In: *Proceedings of the International Joint Conferences on Artificial Intelligence (IJCAI)*.

Carlos N Silla and Alex A Freitas (2011). "A survey of hierarchical classification across different application domains". In: *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*.

André Belotto da Silva and Maxime Gazeau (2020). "A General System of Differential Equations to Model First-Order Adaptive Algorithms." In: *Journal of Machine Learning Research (JMLR)*.

Samuel Henrique Silva and Peyman Najafirad (2020). "Opportunities and challenges in deep learning adversarial robustness: A survey". In: *arXiv preprint arXiv:2007.00753*.

Carl-Johann Simon-Gabriel, Yann Ollivier, Leon Bottou, Bernhard Schölkopf, and David Lopez-Paz (2019). "First-order adversarial vulnerability of neural networks and input dimension". In: *International Conference on Machine Learning (ICML)*.

Eero P Simoncelli (1997). "Statistical models for images: Compression, restoration and synthesis". In: *Conference on Signals, Systems, and Computers*.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman (2013). "Deep inside convolutional networks: Visualising image classification models and saliency maps". In: *arXiv preprint arXiv:1312.6034*.

Karen Simonyan and Andrew Zisserman (2014). "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556*.

— (2015). "Very deep convolutiona networks for large-scale image recognition". In: *International Conference on Learning Representations (ICLR)*.

Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman (2018). "Pixeldefend: Leveraging generative models to understand and defend against adversarial examples". In: *International Conference on Learning Representations (ICLR)*.

Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro (2018). "The implicit bias of gradient descent on separable data". In: *Journal of Machine Learning Research (JMLR)*.

Andreas Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer (2021). "How to train your vit? data, augmentation, and regularization in vision transformers". In: *Transactions on Machine Learning Research (TMLR)*.

Andreas Peter Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer (2022). "How to train your ViT? Data, Augmentation, and Regularization in Vision Transformers". In: *Transactions on Machine Learning Research (TMLR)*.

David Stutz, Matthias Hein, and Bernt Schiele (2019). "Disentangling adversarial robustness and generalization". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna (2016). "Rethinking the inception architecture for computer vision". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

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

Tesla (2018). *Autopilot*. https://www.tesla.com/autopilot/.

David J Tolhurst, Yoav Tadmor, and Tang Chao (1992). "Amplitude spectra of natural images". In: *Ophthalmic and Physiological Optics*.

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

Florian Tramèr, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel (2017). "The space of transferable adversarial examples". In: *arXiv preprint arXiv:1704.03453*.

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry (2018). "Robustness May Be at Odds with Accuracy". In: *International Conference on Learning Representations (ICLR)*.

Richard E Turner (2010). "Statistical models for natural sounds". PhD thesis. UCL (University College London).

Jonathan Uesato, Jean-Baptiste Alayrac, Po-Sen Huang, Robert Stanforth, Alhussein Fawzi, and Pushmeet Kohli (2019). "Are labels required for improving adversarial robustness?" In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Fatemeh Vakhshiteh, Ahmad Nickabadi, and Raghavendra Ramachandra (2021). "Adversarial attacks against face recognition: A comprehensive study". In: *IEEE Access*.

Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio (2019). "Manifold Mixup: Better representations by interpolating hidden states". In: *International Conference on Learning Representations (ICLR)*.

Roman Vershynin (2018). *High-dimensional probability*. Cambridge University Press: An Introduction with Applications in Data Science.

Aladin Virmaux and Kevin Scaman (2018). "Lipschitz regularity of deep neural networks: analysis and efficient estimation". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Martin J Wainwright and Eero Simoncelli (1999). "Scale mixtures of Gaussians and the statistics of natural images". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Gregory K Wallace (1991). "The JPEG still picture compression standard". In: *Communications of the ACM*.

Bohan Wang, Qi Meng, Wei Chen, and Tie-Yan Liu (2021a). "The implicit bias for adaptive optimization algorithms on homogeneous neural networks". In: *International Conference on Machine Learning (ICML)*.

Haohan Wang, Xindi Wu, Zeyi Huang, and Eric P Xing (2020a). "High-frequency component helps explain the generalization of convolutional neural networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Haotao Wang, Aston Zhang, Shuai Zheng, Xingjian Shi, Mu Li, and Zhangyang Wang (2022a). "Removing batch normalization boosts adversarial training". In: *International Conference on Machine Learning (ICML)*.

Hongjun Wang and Yisen Wang (2022b). "Self-ensemble adversarial training for improved robustness". In: *International Conference on Learning Representations (ICLR)*.

Xiaosen Wang and Kun He (2021b). "Enhancing the transferability of adversarial attacks through variance tuning". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Xiaosen Wang, Xuanran He, Jingdong Wang, and Kun He (2021c). "Admix: Enhancing the transferability of adversarial attacks". In: *Proceedings of the International Conference on Computer Vision (ICCV)*.

Xiaosen Wang, Kangheng Tong, and Kun He (2024). "Rethinking the Backward Propagation for Adversarial Transferability". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Yisen Wang, Difan Zou, Jinfeng Yi, James Bailey, Xingjun Ma, and Quanquan Gu (2020b). "Improving adversarial robustness requires revisiting misclassified examples". In: *International Conference on Learning Representations (ICLR)*.

Zekai Wang, Tianyu Pang, Chao Du, Min Lin, Weiwei Liu, and Shuicheng Yan (2023). "Better diffusion models further improve adversarial training". In: *International Conference on Machine Learning (ICML)*.

Pete Warden (2018). "Speech commands: A dataset for limited-vocabulary speech recognition". In: *arXiv preprint arXiv:1804.03209*.

Futa Waseda, Sosuke Nishikawa, Trung-Nghia Le, Huy H Nguyen, and Isao Echizen (2023). "Closer look at the transferability of adversarial examples: How they fool different models differently". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*.

Zeming Wei, Jingyu Zhu, and Yihao Zhang (2023). "Sharpness-Aware Minimization Alone can Improve Adversarial Robustness". In: *ICML Workshop on New Frontiers in Adversarial Machine Learning*.

Zhipeng Wei, Jingjing Chen, Micah Goldblum, Zuxuan Wu, Tom Goldstein, and Yu-Gang Jiang (2022). "Towards transferable adversarial attacks on vision transformers". In: *AAAI Conference on Artificial Intelligence (AAAI)*.

Yuxin Wen, Shuai Li, and Kui Jia (2020). "Towards understanding the regularization of adversarial robustness on neural networks". In: *International Conference on Machine Learning (ICML)*.

Ross Wightman (2019). *PyTorch Image Models*. https://github.com/rwightman/pytorch-image-models. DOI: 10.5281/zenodo.4414861.

Ashia C Wilson, Rebecca Roelofs, Mitchell Stern, Nati Srebro, and Benjamin Recht (2017). "The marginal value of adaptive gradient methods in machine learning". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Fabian Woitschek and Georg Schneider (2021). "Physical adversarial attacks on deep neural networks for traffic sign recognition: A feasibility study". In: *2021 IEEE Intelligent Vehicles Symposium (IV)*.

Eric Wong and Zico Kolter (2018). "Provable Defenses against Adversarial Examples via the Convex Outer Adversarial Polytope". In: *International Conference on Machine Learning (ICML)*.

Eric Wong, Leslie Rice, and J Zico Kolter (2019a). "Fast is better than free: Revisiting adversarial training". In: *International Conference on Learning Representations (ICLR)*.

Eric Wong, Frank Schmidt, and Zico Kolter (2019b). "Wasserstein adversarial examples via projected sinkhorn iterations". In: *International Conference on Machine Learning (ICML)*.

Dongxian Wu, Yisen Wang, Shu-Tao Xia, James Bailey, and Xingjun Ma (2020a). "Skip connections matter: On the transferability of adversarial examples generated with resnets". In: *International Conference on Learning Representations (ICLR)*.

Lei Wu and Zhanxing Zhu (2020b). "Towards understanding and improving the transferability of adversarial examples in deep neural networks". In: *Asian Conference on Machine Learning*.

Weibin Wu, Yuxin Su, Michael R Lyu, and Irwin King (2021). "Improving the transferability of adversarial samples with adversarial transformations". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Han Xiao, Kashif Rasul, and Roland Vollgraf (2017). "Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms". In: *arXiv preprint arXiv:1708.07747*.

Zihao Xiao, Xianfeng Gao, Chilin Fu, Yinpeng Dong, Wei Gao, Xiaolu Zhang, Jun Zhou, and Jun Zhu (2021). "Improving transferability of adversarial patches on face recognition with generative models". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan L Yuille (2019). "Improving transferability of adversarial examples with input diversity". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yifeng Xiong, Jiadong Lin, Min Zhang, John E Hopcroft, and Kun He (2022). "Stochastic variance reduced ensemble adversarial attack for boosting the adversarial transferability". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Weilin Xu, David Evans, and Yanjun Qi (2018). "Feature squeezing: Detecting adversarial examples in deep neural networks". In: *Network and Distributed System Security Symposium*.

Xilie Xu, Jingfeng Zhang, Feng Liu, Masashi Sugiyama, and Mohan S Kankanhalli (2024). "Enhancing adversarial contrastive learning via adversarial invariant regularization". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Huanrui Yang, Jingyang Zhang, Hongliang Dong, Nathan Inkawhich, Andrew Gardner, Andrew Touchet, Wesley Wilkes, Heath Berry, and Hai Li (2020). "Dverge: diversifying vulnerabilities for enhanced robust generation of ensembles". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo (2019). "CutMix: Regularization strategy to train strong classifiers with localizable features". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Chaoning Zhang, Philipp Benz, Adil Karjauv, Jae Won Cho, Kang Zhang, and In So Kweon (2022). "Investigating top-k white-box and transferable black-box attack". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan (2019). "Theoretically principled trade-off between robustness and accuracy". In: *International Conference on Machine Learning (ICML)*.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz (2018). "Mixup: Beyond Empirical Risk Minimization". In: *International Conference on Learning Representations (ICLR)*.

Linjun Zhang, Zhun Deng, Kenji Kawaguchi, Amirata Ghorbani, and James Zou (2021). "How Does Mixup Help With Robustness and Generalization?" In: *International Conference on Learning Representations (ICLR)*.

Yechao Zhang, Shengshan Hu, Leo Yu Zhang, Junyu Shi, Minghui Li, Xiaogeng Liu, Wei Wan, and Hai Jin (2024). "Why Does Little Robustness Help? A Further Step Towards Understanding Adversarial Transferability". In: *Proceedings of the IEEE Symposium on Security and Privacy (SP)*.

Pu Zhao, Pin-Yu Chen, Payel Das, Karthikeyan Natesan Ramamurthy, and Xue Lin (2020). "Bridging mode connectivity in loss landscapes and adversarial robustness". In: *International Conference on Learning Representations (ICLR)*.

Yunqing Zhao, Tianyu Pang, Chao Du, Xiao Yang, Chongxuan Li, Ngai-Man Man Cheung, and Min Lin (2023). "On evaluating adversarial robustness of large vision-language models". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Zhengyu Zhao, Zhuoran Liu, and Martha Larson (2021). "On success and simplicity: A second look at transferable targeted attacks". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang (2020). "Random erasing data augmentation". In: *AAAI Conference on Artificial Intelligence (AAAI)*.

Daquan Zhou, Zhiding Yu, Enze Xie, Chaowei Xiao, Animashree Anandkumar, Jiashi Feng, and Jose M Alvarez (2022). "Understanding the robustness in vision transformers". In: *International Conference on Machine Learning (ICML)*.

Yiwei Zhou, Xiaobo Xia, Zhiwei Lin, Bo Han, and Tongliang Liu (2024). "Few-shot adversarial prompt learning on vision-language models". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Zhe Zhou, Di Tang, Xiaofeng Wang, Weili Han, Xiangyu Liu, and Kehuan Zhang (2018). "Invisible mask: Practical attacks on face recognition with infrared". In: *arXiv preprint arXiv:1803.04683*.

Haoran Zhu, Boyuan Chen, and Carter Yang (2023). "Understanding Why ViT Trains Badly on Small Datasets: An Intuitive Perspective". In: *arXiv preprint arXiv:2302.03751*.

Yao Zhu, Jiacheng Sun, and Zhenguo Li (2021). "Rethinking adversarial transferability from a data distribution perspective". In: *International Conference on Learning Representations (ICLR)*.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J Zico Kolter, and Matt Fredrikson (2023). "Universal and transferable adversarial attacks on aligned language models". In: *arXiv preprint arXiv:2307.15043*.

Junhua Zou, Zhisong Pan, Junyang Qiu, Xin Liu, Ting Rui, and Wei Li (2020). "Improving the transferability of adversarial examples with resized-diverse-inputs, diversity-ensemble and region fitting". In: *Proceedings of the European Conference on Computer Vision (ECCV)*.