Decoding Robust Generalization in Object Recognition Models

Inauguraldissertation
zur Erlangung des akademischen Grades
eines Doktors der Naturwissenschaften
der Universität Mannheim

vorgelegt von

Paul Gavrikov

aus Freiburg im Breisgau

Mannheim, 2025

Dekan:	Prof. Dr. Claus Hertling, Universität Mannheim
Referent:	Prof. Dr.-Ing. Janis Keuper, Hochschule Offenburg und Universität Mannheim
Korreferent:	Prof. Dr. Seong Joon Oh, Eberhard Karls Universität Tübingen
Korreferent:	Prof. Dr. Rainer Gemulla, Universität Mannheim

Tag der mündlichen Prüfung: 08.07.2025

“The first principle is that you must not fool yourself — and you are the easiest person to fool.”

— *Richard P. Feynman*

Erklärung zum Einsatz von Generativen Textmodellen

In der Erstellung dieser Arbeit wurden die generative Textmodelle (auch bekannt als *large language models* (LLMs)) *OpenAI ChatGPT*, *Google Gemini*, *LanguageTool*, sowie *Grammarly* eingesetzt, um die schriftliche Präsentation dieser Thesis zu verbessern. In diesem Zusammenhang wurden einzelne und bereits formulierte Sätze und Textpassagen sprachlich und grammatikalisch überarbeitet, umformuliert, strukturiert und/oder von diesen Modellen zusammengefasst. Die erstellten Texte wurden zudem manuell geprüft und häufig weiter überarbeitet. Die Modelle wurden nicht dazu eingesetzt, neue Inhalte zu generieren. Insbesondere wurden alle in dieser Thesis eingeführten Methoden, Experimente und Resultate eigenständig – beziehungsweise mit oder von den jeweils gekennzeichneten Autoren – erarbeitet.

Abstract

A key challenge in machine learning, particularly for image classifiers, is the robust generalization to new data – while models often perform well on data drawn from similar distributions as their training set, they struggle with samples exhibiting even slight deviations. Although increasing training data volume is a straightforward solution, it is often expensive, resource-intensive, and sometimes simply impossible. We propose a more structured approach: rather than indiscriminately training on massive datasets, we focus on understanding and leveraging knowledge embedded within existing models. Our goal is to investigate model populations with varying degrees of generalization ability to uncover the underlying mechanisms that contribute to robust performance.

We pursue this goal through two complementary lines of investigation. The first part of this thesis focuses on representations in learned weights: We analyze the representations encoded in the learned weights of Convolutional Neural Networks (CNNs), specifically focusing on convolutional filters. Through several studies, we identify points of representational divergence across different CNN models and explore the factors influencing these differences, with a particular emphasis on the impact of adversarial training – a state-of-the-art regularization technique to achieve robustness to adversarial perturbations. Based on these findings, we propose a novel, simple, and computationally efficient regularization technique for convolution filters in standard training that enhances model robustness against adversarial attacks and other forms of covariate shift.

The second part of this thesis focuses on visual perception biases. We investigate the visual biases exhibited by models, specifically comparing them to human perception by measuring alignment. We demonstrate how adversarial regularization affects these biases and correlate several well-known biases with generalization performance. Our results challenge previous claims that simply aligning individual visual biases holistically improves model generalization. Instead, we find that bias alignment often only improves predictions under specific types of covariate shift. Finally, we examine the propagation of visual information within large vision-language models (VLMs), revealing that simple natural language prompts can effectively steer inherent visual perception biases.

Taken together, this thesis offers valuable insights into the learned representations of object recognition models. By enhancing our understanding of generalization mechanisms, we establish promising techniques such as filter regularization for discriminative models and test-time bias steering in VLMs. Furthermore, we delineate the limitations of existing approaches and identify less promising directions, such as simply regularizing biases during the training of discriminative models.

Zusammenfassung

Eine zentrale Herausforderung im maschinellen Lernen – insbesondere bei Bildklassifikatoren – besteht in der robusten Generalisierung auf neue Daten. Während diese Modelle häufig gute Leistungen auf Daten erbringen, die aus Verteilungen stammen, die ihrem Trainingsdatensatz ähneln, haben sie Schwierigkeiten mit Beispielen, die auch nur geringfügig abweichen. Obwohl eine Vergrößerung des Trainingsdatensatzes eine naheliegende Lösung darstellt, ist dieser Ansatz oft zu teuer, ressourcenintensiv und manchmal schlichtweg unmöglich. Wir schlagen einen strukturierteren Ansatz vor: Anstatt scheinbar wahllos auf massiven Datensätzen zu trainieren, konzentrieren wir uns darauf, das in bestehenden Modellen eingebettete Wissen zu verstehen und zu nutzen. Unser Ziel ist es, Modellpopulationen mit unterschiedlichen Ausprägungen der Generalisierungsfähigkeit zu untersuchen, um die zugrunde liegenden Mechanismen aufzudecken, die zu einer robusten Leistung beitragen.

Dieses Ziel verfolgen wir durch zwei komplementäre Forschungsansätze. Der erste Teil dieser Arbeit konzentriert sich auf Repräsentationen in den gelernten Gewichten: Wir analysieren die in den Gewichten von Convolutional Neural Networks (CNNs) kodierten Repräsentationen, wobei unser besonderer Fokus auf den Faltungsfiltern liegt. Durch mehrere Studien identifizieren wir Punkte repräsentativer Divergenz zwischen verschiedenen CNN-Modellen und untersuchen die Faktoren, die diese Unterschiede beeinflussen – mit besonderem Augenmerk auf den Einfluss des “Adversarial Trainings”, einer effektiven Regularisierungstechnik, die Robustheit gegenüber “feindlichen Angriffen” (Adversarial Attacks) erreicht. Basierend auf diesen Erkenntnissen schlagen wir eine neuartige, einfache und recheneffiziente Regularisierungstechnik für Faltungsfilter im Standardtraining vor, die die Robustheit von Modellen gegenüber feindlichen Angriffen und anderen Formen der Kovariaten-Verschiebung (Covariate Shift) erhöht.

Der zweite Teil dieser Arbeit widmet sich visuellen Wahrnehmungsverzerrungen. Wir untersuchen die von den Modellen gezeigten visuellen Verzerrungen, indem wir diese gezielt mit der menschlichen Wahrnehmung vergleichen und deren Übereinstimmung messen. Dabei zeigen wir, wie die Regularisierung durch “Adversarial Training” diese Verzerrungen beeinflusst, und stellen Korrelationen zwischen mehreren bekannten Verzerrungen und der Generalisierungsleistung her. Unsere Ergebnisse stellen frühere Behauptungen in Frage, wonach allein die Korrektur von einzelnen visuellen Verzerrungen die Generalisierung von Modellen verbessern soll. Stattdessen stellen wir fest, dass eine Angleichung der Verzerrungen häufig nur unter bestimmten Formen der Kovariaten-Verschiebung zu verbesserten Vorhersagen führt. Abschließend untersuchen wir die Verarbeitung von visueller Informationen innerhalb großer Bild-Sprach-Modelle (VLMs) und zeigen, dass einfache Prompts in natürlicher Sprache die inhärenten visuellen Wahrnehmungsverzerrungen effektiv steuern können.

Zusammenfassend bietet diese Arbeit wertvolle Einblicke in die gelernten Repräsentationen von Objekterkennungsmodellen. Durch ein vertieftes Verständnis der Generalisierungsmechanismen etablieren wir vielversprechende Techniken wie die Filterregularisierung für diskriminative Modelle und die Wahrnehmungsverzerrungs-Steuerung in VLMs zur Testzeit. Darüber hinaus skizzieren wir die Grenzen bestehender Ansätze und identifizieren weniger vielversprechende Richtungen, etwa das bloße Regularisieren von Verzerrungen während des Trainings diskriminativer Modelle.

Acknowledgements

This thesis would not have been completed if it was not for the support of many people over the last few years, and I want to extend my gratitude to everyone who helped in whatever capacity they could!

I would like to begin by expressing my heartfelt gratitude to my supervisor, Prof. Dr.-Ing. Janis Keuper. Your invaluable guidance, unwavering support, and the freedom you gave me in shaping my own research have been fundamental to my development. I truly appreciated the opportunities you provided—attending conferences and summer schools allowed me to forge lasting connections and explore new ideas (and, of course, you never let me forget to add colorbars everywhere!).

I am also grateful to Prof. Dr. Seong Joon Oh for serving as my co-examiner and to Prof. Dr.-Ing. Margret Keuper, whose relentless stream of ideas and meticulous feedback enriched my work. I still remember how you encouraged us to submit *CNN Filter DB* to CVPR – without your insight and support, achieving my Oral would not have been possible.

My thanks also go to my students (chronologically) – Lucie Cayment, Jakob Groll, Johannes Eichin, David Hoof, and Fabian Junker. Watching you all progress in your own journeys has been one of the most rewarding aspects of my work, and I hope I managed to pass on a bit of wisdom along the way.

I must also acknowledge my wonderful collaborators (alphabetically) – Shashank Agnihotri, Julia Grabinski, Robert Geirhos, Bianca Lamm, Jovita Lukasik, Steffen Jung, M. Jehanzeb Mirza, and many others – whose contributions, from refining drafts to running experiments (and even sharing in the frustrations of setbacks), made this journey all the more memorable. I am incredibly lucky to call some of you my friends. I would like to single out Jovita for her tireless dedication during those long nights of writing, experimenting, and perfecting every detail. I also appreciate the valuable inputs I received from Jovita, Julia, Shashank, and Patrick Müller on my thesis drafts.

I am deeply thankful to my family for their enduring love and support, and for those timely reminders that eventually pushed me to start writing this thesis. Without you, I might still be procrastinating!

And finally, to my partner Julia – your love and encouragement have been a constant source of strength through the ups and downs. Thank you for your patience and support. This journey, and this work, is dedicated to you.

Contents

Chapter 1

Introduction

Could Frank Rosenblatt have envisioned the vast array of artificial intelligence (AI) models that have emerged from his pioneering work on the *Perceptron* (Rosenblatt, 1958)? Step by step, with the invention of backpropagation (Rumelhart et al., 1986), breakthroughs in network architecture design (LeCun et al., 1989a; Hochreiter & Schmidhuber, 1997; Krizhevsky et al., 2012; He et al., 2016; Vaswani et al., 2017; Dosovitskiy et al., 2021), learning methods (Rumelhart et al., 1986; de Sa, 1993; Hinton et al., 2006, 2015; Radford et al., 2021), alongside many other techniques, humanity has developed impressive models based on neural networks for numerous narrow problems (*e.g.*, (Krizhevsky et al., 2012; Graves et al., 2013; Sutskever et al., 2014)) and recently also increasingly general solutions across data modalities (*e.g.*, Reed et al. (2022); OpenAI (2024); Gemini Team (2023); Anthropic (2024)). Of particular success are computer vision models that can see and recognize objects or, more abstractly speaking, patterns – often even surpassing human performance at doing so.

A key driver of this progress and the differentiator to classical machine learning (ML) is that, in deep learning, input features are not just weighted by their importance but also implicitly learned from (large amounts of) raw data. This is especially useful for unstructured data like images, where features are not intuitively apparent to humans and are hard to formalize. However, this convenience comes at the cost of opaqueness and a lack of control. As feature extraction is optimized through backpropagation on the loss function (Rumelhart et al., 1986; LeCun et al., 1998a), there is an inherent risk to extract and utilize features that are (locally) optimal concerning the training data but not for the underlying real-world data - *i.e.*, **overfitting** to **shortcut features** (Geirhos et al., 2020a). In such cases, the model would learn to prioritize (or even exclusively utilize) spurious, task-irrelevant, and training distribution-specific shortcut features instead of utilizing useful, semantic, and task-relevant core features. Given that training, validation, and test datasets are assumed to be drawn from the same distribution (i.i.d.), it may not even be detectable that the model has overfitted and will not be able to **generalize** to new data.

As an example, we may want to train a simple image classification model to detect cats in images. In deep learning fashion, we would collect a reasonable amount of images of cats and apply some kind of gradient-based learning technique. If, however, all cat photos are taken in daylight conditions and are photos of actual living cats, our model may overfit this specific distribution of features. While this would suffice to classify i.i.d. samples from the given dataset, this model may fail to generalize to real-world scenarios, where the photo quality may be poor, the lighting is different, there is some influence of weather, or the cat is an artistic rendition such as a painting as in Figure 1.1. It is already challenging to think of and model all **feature distribution shifts** at test time for small-scale problems like this one, and may quickly appear hopeless as we aim to develop general vision models with orders of magnitudes more objects to recognize – each potentially having their own kinds of expected distribution shifts.

One example of a larger-scale vision problem is the popular *ImageNet Large Scale Visual Recognition*

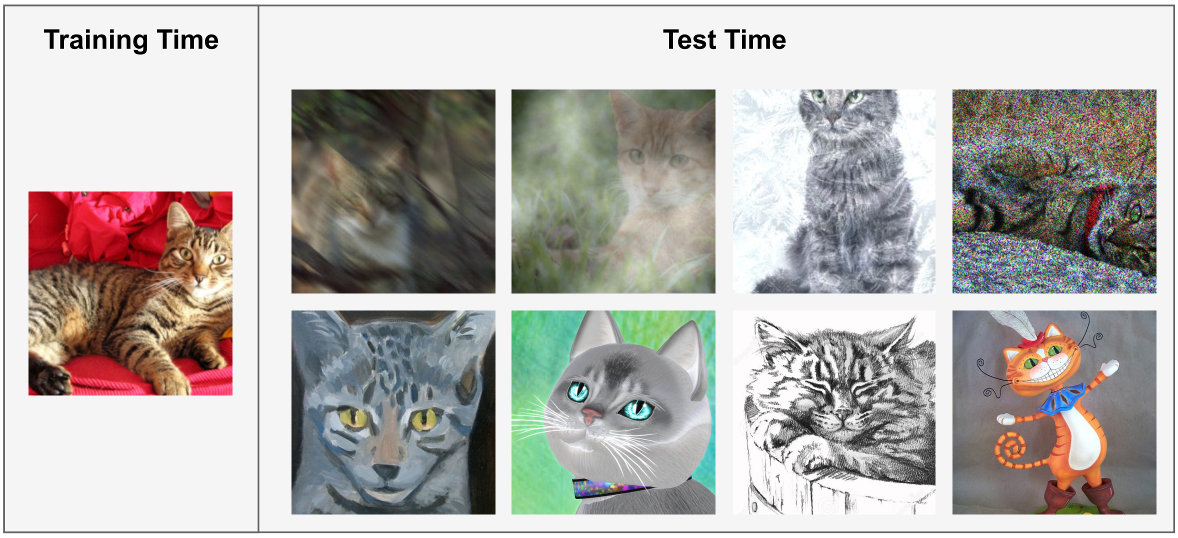


Figure 1.1: The quest for model generalization. We want to train models that recognize all cats shown here, but the features describing a cat observed at training time may diverge from the observed features at test time. Here, we observe clear daylight photos of cats during training, but at test time, we encounter a variety of images: some are blurry, some are artistic representations, some are taken in different weather conditions, and some are toys. **Image Source:** (Deng et al., 2009; Hendrycks & Dietterich, 2019; Hendrycks et al., 2021a; Wang et al., 2019)

Challenge (ILSVRC) 2012 (Deng et al., 2009; Russakovsky et al., 2015). To this day, the 1,000-way classification challenge on a dataset of 1.3M images serves as an important benchmark in image classification. While the community made tremendous progress on top-1 accuracy improvements on the test dataset in the past decade (Krizhevsky et al., 2012; Simonyan & Zisserman, 2015; He et al., 2016; Huang et al., 2017; Dosovitskiy et al., 2021; Zhai et al., 2022a; Srivastava & Sharma, 2024) and already surpassed (estimated) human accuracy in 2015 (Karpathy, 2014; He et al., 2015) it became quickly evident that even the best models are not **robust** – even small modifications of test samples leads to a rapid decline in performance.

For instance, the decision rules are sensitive to barely noticeable additive perturbations of the input data, leading to misclassification with high confidence (**adversarial attacks**) (Biggio et al., 2013; Szegedy et al., 2014b). These adversarial perturbations are often expected to be noise-like and lack semantic meaning for humans,[1] however, even intuitively easier-to-grasp input alterations, such as simple geometrical transformations (Kanbak et al., 2018), viewpoint (Dong et al., 2022) or background changes (Rosenfeld et al., 2018; Xiao et al., 2021a), or common corruptions observed in photography (Hendrycks & Dietterich, 2019; Kar et al., 2022) can strongly deteriorate the performance of trained models.

A straightforward solution to address most robustness issues is to increase (*i.e.*, **scale**) the number of training samples – essentially, memorize the distribution until there's nothing left to learn. Indeed, this strategy has driven much of the progress in recent years (Sutton, 2019; Branwen, 2020). However, it is extremely expensive, potentially wasteful, and relies on the assumption of abundant amounts of training data that never runs out and scaling laws that never saturate. Yet, data will eventually run out (Villalobos et al., 2024), and as Ilya Sutskever said at NeurIPS 2024: *"Data is the fossil fuel of AI"*.

In many domains, scaling data can be challenging, as readily available datasets may be limited, and

[1] This is a common misconception and not entirely true, as a recent study showed that human subjects could also be slightly influenced by these perturbations (Veerabadran et al., 2023). The majority of attacks, however, are semantically meaningless.

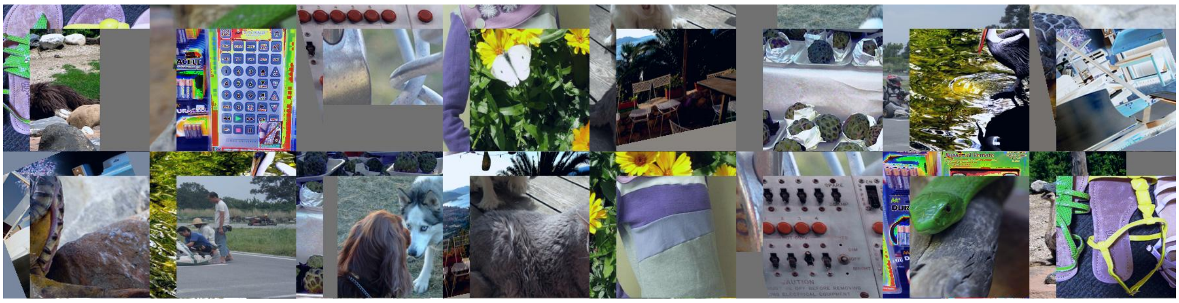


Figure 1.2: Augmentation as a form of regularization. We demonstrate the effect of augmentation on an *ImageNet* batch of 16 samples, using a mix of various transformations under a modern training regime (Wightman et al., 2021). **Image Source:** (Deng et al., 2009)

acquiring new samples can be prohibitively expensive or even impossible. This is particularly evident in medical imaging, where data collection often relies on costly specialized equipment rather than simple cameras. Additionally, accurate labeling requires the expertise of highly trained medical professionals, such as radiologists or pathologists, further complicating and increasing the cost of data acquisition. Further privacy concerns and strict regulations (*e.g.*, HIPAA) make it challenging to build large-scale datasets for developing (or testing) robust AI models in medical vision tasks.

An orthogonal approach to improve generalization is **regularization** – the induction of prior assumptions about the ideal model. Regularization can take various forms and is part of most training pipelines. Often the ideal posterior is not known and likely impossible to estimate, so explicit regularization is not possible, and techniques will always be coarse and implicit approximations. One common form of regularization in machine learning is weight decay, which can be seen as *Occam's razor*, assuming that simpler solutions generalize better. Examples of more specific regularization are data augmentation techniques (shown in Figure 1.2), that (synthetically) transform samples at training time to increase the de facto number of samples and to model desired invariances to specific transformations of the data. Specifically crafted worst-case augmentations to maximize the loss (**adversarial training** (Madry et al., 2018)), are also amongst the only successful methods to increase robustness to adversarial attacks.

This thesis investigates the impact of implicit regularization on the learned representations of trained models, with a central focus on understanding its role in generalization. Implicit regularization influences the model's final weights or behavior without being explicitly defined in the loss function. Our primary goal is to decipher how this implicit regularization shapes the learned representations and, leveraging these insights, to develop more effective and explicit regularization strategies. Specifically, we will mostly focus on adversarial training and robustness.

Rather than conducting a case study of individual models, our approach emphasizes the discovery of generalizable patterns that transcend specific model and training configurations. We analyze *populations* of models, trained under varying conditions, to identify statistical trends and regularities in their learned representations. Our central hypothesis is that these cross-model patterns reveal deeper, more fundamental mechanisms for generalization than those observable in individual models. By focusing on these population-level trends, we aim to uncover intrinsic properties of the learning process that contribute to robust generalization performance. This stands in contrast to explanations that rely on specific model architectures or training hyperparameters, which may not generalize across different settings.

Building upon our focus on population-level analysis of implicit regularization and its impact on generalization, our study covers two angles. First, we examine the learned parameters (**weight space**) itself, investigating how implicit regularization manifests in the statistical properties of the learned weights across populations of models.

Second, we adopt a behavioral perspective, exploring how models with improved generalization perceive input data and what features they may be biased toward. This involves analyzing the decision rules learned by these models and comparing them to human visual perception, aiming to understand the representational differences (**alignment**) contributing to robustness. By comparing model behavior to human vision, we aim to gain insights into the nature of robust representations.

In both these lines of inquiry – the statistical analysis of weight spaces and the behavioral study of biases in model perception – our ultimate goal is to translate our findings into actionable insights. These insights will inform the design of more effective explicit regularization techniques, allowing us to directly control and enhance the generalization capabilities of machine learning models based on the principles uncovered through our population-based analysis of implicit regularization.

Ultimately, this research seeks to advance knowledge about how generalization is represented, enabling the design of more principled and effective regularization techniques for improved generalization in object recognition models.

1.1 Contribution Overview

In this section, we will discuss the contributions grouped by subtopics, which may span multiple chapters. For an outline of the thesis, dissecting the contribution of each chapter, see Section 1.2.

1.1.1 Measuring Representational Distances

Interpreting the individual decision rules encoded within the learned weights of deep neural networks, particularly in deeper layers, is challenging due to their compositional nature. Consequently, research comparing the similarity (or distance) between neural representations commonly focuses on the activation space, effectively studying the similarity of intermediate features (see Sucholutsky et al. (2024) for an overview). Established methods include Representational Similarity Analysis (RSA) (Kriegeskorte et al., 2008; Mehrer et al., 2020) and Centered Kernel Alignment (CKA) (Kornblith et al., 2019a). Other approaches involve activation clustering (Dravid et al., 2023), manual inspection of feature visualizations (Olah et al., 2020b; Geirhos et al., 2024), and clustering of learned weights (Babaiee et al., 2024b,c,a).

Our contributions In Chapter 3, we introduce a novel metric explicitly designed to compare weight representations in Convolutional Neural Networks (CNNs), based on the distribution of learned convolution filter patterns. We apply this metric to analyze representational differences across a diverse range of CNN-based models on global and local scales (Chapter 3) and to understand the impact of adversarial regularization (Chapter 4).

Furthermore, we propose comparing representations using quantitative statistics of learned weights. In Chapter 3, Chapter 4, and Chapter 5, we introduce several metrics tailored to convolutional filters, capturing the diversity, orthogonality, sparsity, and frequency distribution of learned filter patterns. In Chapter 6, we broaden this scope by proposing a more abstract metric that gauges the criticality of any layer with learnable parameters.

1.1.2 Learned Representations under Adversarial Training

Adversarial training (AT) against pixel-wise perturbation attacks was amongst the first proposed solutions to improve generalization under adversarial perturbations (Goodfellow et al., 2015) and remains one of the most successful methods at achieving adversarial robustness to date (Madry et al., 2018; Peng et al., 2023; Bartoldson et al., 2024). More specifically, it improves robustness against the threat model during training. In some sense, this form of regularization turns out-of-distribution test (for a normal model) into in-distribution training examples. This results in a tendency to overfit the training threat – meaning that exposure to adversarial perturbations generated by a different attack, norm, or budget, etc. may result in no effective improvement in robustness or even in the decrease of robustness (Rice et al., 2020).

While multiple works seek to improve adversarial training, only a few attempt to understand its implications (both positive and negative). For instance, prior work has shown that adversarial models transfer better (Salman et al., 2020), are better-calibrated (Grabinski et al., 2022a), are more shape-biased (Zhang & Zhu, 2019; Chen et al., 2022a; Geirhos et al., 2021), or are more interpretable (Tsipras et al., 2019).

Our contributions We extend the findings about adversarially-trained models in two areas. We provide the first thorough and large-scale study of the learned weight space in Chapter 4, showing that adversarially-trained CNNs contain more diverse filter patterns that are additionally more orthogonal and less sparse. Specifically, for L^{∞}-norm bounded training, we also discover filters in the first layer that refute the common assumption that all CNNs learn a similar first-layer representation independent of the objective (Yosinski et al., 2014). We additionally show that robust models show a stronger tendency to develop low-frequency filters, especially in early layers, excluding the first layer in Chapter 5. Finally, we also show that AT models utilize more of the network capacity in Chapter 6.

From a bias perspective, we also enrich previous literature studying the shape bias (Zhang & Zhu, 2019; Chen et al., 2022a; Geirhos et al., 2021) by a more thorough investigation in Chapter 7: We investigate the shape bias on a broader scale of models and under increasing perturbation strength in training. We are able to show that the increase in shape bias is persistent, even for transformer-based models and under different norms. We also investigate other aspects of alignment and show that adversarially-trained models increase their error consistency toward humans. Finally, we provide a more nuanced analysis of OOD performance, showing that AT increases robustness to perturbations having a similar spectral distribution but, on average, decreases performance, especially if the performance of the normally-trained models strongly exceeded human performance.

1.1.3 Low-Frequency Regularization

Regularization is an umbrella term for techniques that improve the generalization performance of a model at test time (Goodfellow et al., 2016). Multiple techniques have been proposed to improve robust generalization, including augmentation techniques (Hendrycks et al., 2020; Tokozume et al., 2018; Zhang et al., 2018; Cubuk et al., 2019; Yun et al., 2019; Hendrycks et al., 2021a), and adversarial training (Madry et al., 2018) can be seen as an extreme form of augmentation that trains the model on worst-case perturbations of input samples.

Our contributions Based on our observations that adversarially-trained models learn more low-frequent filters in Chapter 5, we derive a simple regularization that imposes a frequency penalty on learned convolution weights during normal training. While this form of regularization is not able to match the performance of adversarial training, it strongly improves the adversarial robustness against small ϵ-perturbations, as well as robustness to common corruptions (Hendrycks & Dietterich, 2019), and increases shape bias (Geirhos et al., 2019). Compared to previous techniques aiming at regularizing or attenuating high-frequency signals, our implementation has minimal overhead, both during training and inference.

1.1.4 Connection between ImageNet Biases and Generalization

Since the release of the first successful neural network `AlexNet` in 2012 (Krizhevsky et al., 2012), newer models have strongly increased in accuracy on the *ImageNet* challenge. However, while most modern models even exceed human prediction accuracy (He et al., 2015), they are by far not as good in generalization on altered samples (Geirhos et al., 2018) and fail in many cases (Kanbak et al., 2018; Geirhos et al., 2018; Hendrycks & Dietterich, 2019; Dong et al., 2022; Rosenfeld et al., 2018; Xiao et al., 2021a). A common explanation for the failure to generalize is that models learn non-representative "shortcut" features (Geirhos et al., 2020a) that are sufficient to classify the train and i.i.d. test set but do not represent the learned objects well under covariate shifts. A few works identified specific biases and went as far as to suggest that improving these biases in the direction of human perception would repair robustness (Geirhos et al., 2019; Wang et al., 2020a; Subramanian et al., 2023).

Our contributions We provide a large-scale study sanity-checking these claims (specifically (Geirhos et al., 2019; Wang et al., 2020a; Subramanian et al., 2023)) to understand if regularization techniques that aim to improve or align a specific bias are expected to correlate with generalization in Chapter 8. Under controlled conditions, we find that none of the investigated biases correlates with a holistic view of generalization – if at all, biases only generalize with specific forms of generalization or under specific forms of regularization like adversarial training. Surprisingly, misaligning models can sometimes even improve performance.

1.1.5 Visual Biases in Multi-Modal Models

Although research on interpretability lags behind the development of new models and training methods, biases in uni-modal models are relatively well understood. For instance, vision models trained on *ImageNet* are known to exhibit biases towards texture (Geirhos et al., 2018), watermarks (Li et al., 2023d), and background cues (Rosenfeld et al., 2018). Similarly, large language models (LLMs) have their own biases, such as a preference for "high-value" options (Sivaprasad et al., 2024). The emergence of multi-modal models, particularly vision-language models (VLMs) like `LLaVA` (Liu et al., 2024), which combine an LLM with a vision encoder, introduces new complexities to bias analysis. The fusion of modality-specific tokens can amplify existing uni-modal biases, generate novel cross-modal biases arising from inter-modality interactions, or even mitigate certain biases by integrating complementary information. Therefore, understanding biases in these models requires investigating how information is processed and combined across modalities rather than simply extrapolating from uni-modal findings.

Our contributions As an initial exploration of visual biases in LLM-based VLMs, we investigate the texture/shape bias (Geirhos et al., 2019) across a diverse set of state-of-the-art models in Chapter 9. We analyze the propagation of this visual bias and demonstrate the significant role of the LLM in its formation, typically resulting in a stronger shape bias in VLMs compared to their underlying vision encoder models. This observation leads to a key finding: we can influence a VLM's perception through natural language prompting, effectively "talking the model into seeing the world differently" and steering its visual bias. Furthermore, we demonstrate that this influence extends beyond the texture/shape bias by introducing a new dataset to benchmark the low/high-frequency bias, which we also show can be steered through prompting.

1.2 Outline

Here, we discuss the structure of the thesis, chapter by chapter. Please also see Figure 1.3 for an overview of the main chapters.

Chapter 2, Background presents the most important background information for this thesis.

The main chapters are divided into two parts (Figure 1.3), each focusing on a distinct viewpoint of generalization.

Part I explores generalization in the **weight space of convolutional neural networks (CNNs)** focusing on how we can measure, compare, and regularize generalization. It is organized into the following chapters:

Chapter 3, Convolution Filter Analysis introduces key metrics for quantifying convolution filter statistics and identifying distribution shifts in emerging patterns between two sets of convolution filters. These metrics are applied to *CNN Filter DB*, a database of 3 × 3 convolution filter kernels extracted from 647 CNN models trained on various problems. Interesting insights are uncovered regarding how

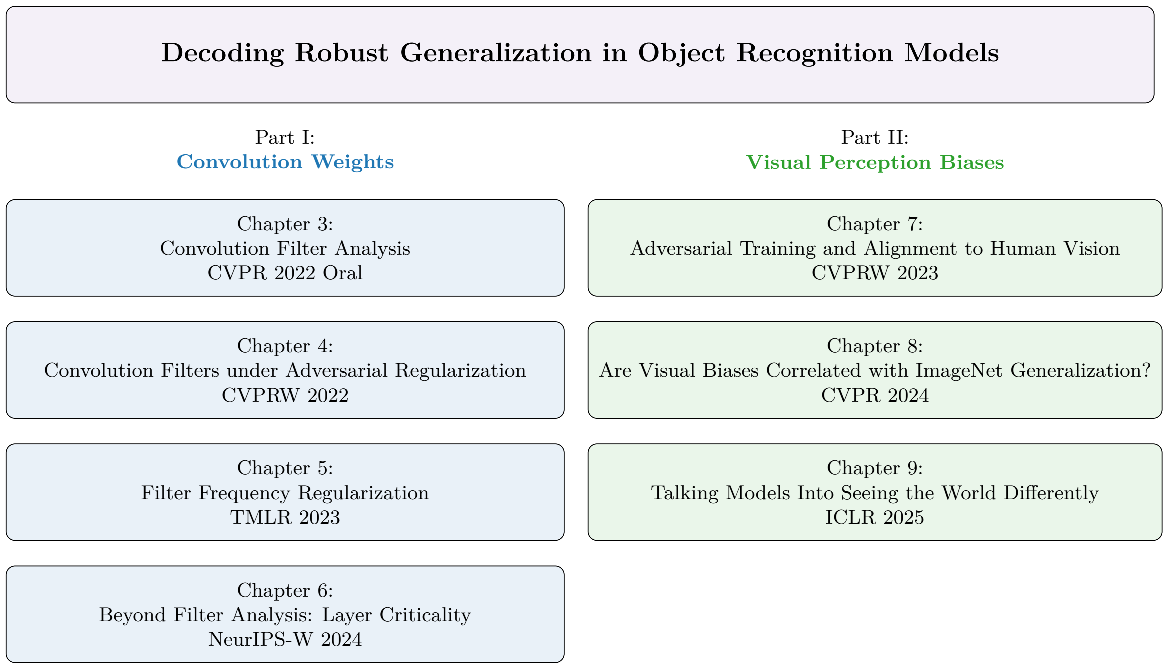

Decoding Robust Generalization in Object Recognition Models
Part I:
Convolution Weights
Part II:
Visual Perception Biases
Chapter 3:
Convolution Filter Analysis
CVPR 2022 Oral
Chapter 7:
Adversarial Training and Alignment to Human Vision
CVPRW 2023
Chapter 4:
Convolution Filters under Adversarial Regularization
CVPRW 2022
Chapter 8:
Are Visual Biases Correlated with ImageNet Generalization?
CVPR 2024
Chapter 5:
Filter Frequency Regularization
TMLR 2023
Chapter 9:
Talking Models Into Seeing the World Differently
ICLR 2025
Chapter 6:
Beyond Filter Analysis: Layer Criticality
NeurIPS-W 2024

Figure 1.3: Thesis Outline.

overparameterization and robustness manifest in weight space and the layers where distribution shifts are most prominent, expanding previously held beliefs about layer specialization.

Chapter 4, Convolution Filters under Adversarial Regularization expands the filter analysis to compare 71 *robust* classification models trained with adversarial regularization against equivalent models trained normally. This comparison aims to uncover patterns that can drive the design of (adversarially-)robust models without relying on adversarial regularization. Findings reveal that robust models develop fewer sparse but more diverse and orthogonal filter patterns throughout the network, particularly in overparameterized settings. Significant differences are observed in the first layer, challenging prior claims about its universal transferability across imaging tasks.

Chapter 5, Filter Frequency Regularization extends our previous study, highlighting a low-frequency bias of robust models that is also visible in convolution filter weights, distinguishing adversarial-regularized models from standard ones. Building on this observation, a regularization technique is proposed to enhance robustness against adversarial attacks and other distribution shifts. The resulting models achieve *native robustness*, which increases a wide spectrum of robustness but avoids dependence on predefined threat models during training and thus alleviates the risk of overfitting them.

Chapter 6, Beyond Filter Analysis: Layer Criticality proposes an alternative exploration of weight space through *critical layers*, revisiting findings by Zhang et al. (2022), which suggest that not all layers equally contribute to a model's learned decision-function. The original methodology is refined to evaluate similarity in prediction behavior rather than relying solely on numerical accuracy comparisons. Applying this metric to a set of 50 models with identical architectures but varying regularization methods reveals significant differences in criticality, particularly in the context of adversarial robustness.

Part II investigates **visual perception biases** by testing the behavior of models and humans against specific features and comprises the following chapters:

Chapter 7, Adversarial Training and Alignment to Human Vision examines the effects of adversarial training on alignment with human visual perception. This study evaluates trends in out-of-distribution generalization, the evolution of the texture/shape bias, and agreement on correct and erroneous predictions of models against human observers. It also offers a frequency-based explanation of generalization in adversarially-trained models.

Chapter 8, Are Visual Biases Correlated with ImageNet Generalization? evaluates the hypothesis that biases can explain failures in model generalization and are thus good priors during training. By fixing architecture and training data (*ImageNet*) and varying only regularization, the relationship between biases and generalization is carefully analyzed, and we conclude that the alignment of individual biases does not guarantee to transfer to improvements of wide spectrum generalization.

Chapter 9, Talking Models Into Seeing the World Differently explores visual biases when facing multi-modal fusion with language in vision-language models (VLMs) through the lens of texture/shape bias. The results reveal that large language models can modify low-level visual information, which allows us to steer perception biases in VLMs simply through language prompts as an alternative to expensive in-training regularization that is necessary for discriminative and uni-modal models.

Chapter 10, Discussion provides a summary and discussion of the thesis findings and concludes the thesis with an outlook on future research directions and a broader view on the topic.

1.3 List of Publications

The following provides a list of publications that have appeared during the PhD program. Publications contributing to this thesis are sorted chronologically. An asterisk * next to author names denotes shared first authorship by equal contribution. We state individual contributions for each author and paper.

Peer-reviewed conference and journal publications

- [Oral] **CNN Filter DB: An Empirical Investigation of Trained Convolutional Filters**
  Paul Gavrikov, *Janis Keuper*
  Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022 (Gavrikov & Keuper, 2022b)

  This article was selected as "Oral" at the Conference on Computer Vision and Pattern Recognition (CVPR 2022); an abstract under the title "An Empirical Investigation of Model-to-Model Distribution Shifts in Trained Convolutional Filters" was accepted at the NeurIPS 2021 Workshop on Distribution Shifts: Connecting Methods and Applications (Gavrikov & Keuper, 2021).

  Project idea: J.K.; model collection, codebase, quality metrics, and analysis: P.G. with input from J.K. (who also provided the idea for SVD-based comparisons); paper writing: P.G. and J.K.; guidance, funding, and infrastructure: J.K.

- **Improving Native CNN Robustness with Filter Frequency Regularization**
  *Jovita Lukasik**, ***Paul Gavrikov****, *Janis Keuper, Margret Keuper*
  Transactions on Machine Learning Research, 2023 (Lukasik et al., 2023)

  Joint first-authorship by J.L. and P.G.; Project idea: M.K. and J.K.; codebase: initially developed by P.G. and expanded by J.L.; decomposition approaches: J.K. (SD) and P.G. (WD); regularization method: J.L. and M.K.; model training and evaluations: J.L. with substantial support of P.G.; coefficient and spectrum analysis and visualizations: P.G.; derivation of connection between shape and low-frequency dominance: M.K.; paper writing: J.L. and P.G. jointly with substantial support of M.K. and input from J.K.; guidance, funding, and infrastructure: M.K. and J.K.

This paper has not been included in the thesis of Jovita Lukasik (Lukasik, 2023).

- **Can Biases in ImageNet Models Explain Generalization?**
  ***Paul Gavrikov**, Janis Keuper*
  Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2024 (Gavrikov & Keuper, 2024)

  Project idea and execution: P.G.; paper writing: P.G. with input from J.K.; guidance, funding, and infrastructure: J.K.

- **Can We Talk Models Into Seeing the World Differently?**
  ***Paul Gavrikov**, Jovita Lukasik, Steffen Jung, Robert Geirhos, M. Jehanzeb Mirza, Margret Keuper, Janis Keuper*
  International Conference on Learning Representations (ICLR), 2025 (Gavrikov et al., 2025)

  An abstract under the title "Are Vision Language Models Texture or Shape Biased and Can We Steer Them?" was accepted at the CVPR 2024 Workshop on What is Next in Multimodal Foundation Models? (Gavrikov et al., 2024b).

  Project idea and lead: P.G.; model collection, codebase, and analysis: P.G. with support of S.J. (GPT-4); conception of OCR analysis (not included in the paper): S.J. and M.K.; idea for VQA analysis: M.J.M.; visualizations: P.G. with support of S.J.; curation of related work: B.L. (preprint) and S.J.; paper writing: P.G. with substantial support of J.L./R.G. and input of all authors; funding: P.G., S.J., M.K., and J.K.; guidance and infrastructure: M.K. and J.K.

Peer-reviewed workshop publications

- **Adversarial Robustness Through the Lens of Convolutional Filters**
  ***Paul Gavrikov**, Janis Keuper*
  Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2022 (Gavrikov & Keuper, 2022a)

  Project idea: J.K.; model collection, codebase, and analysis: P.G.; paper writing: P.G. with input of J.K.; guidance, funding, and infrastructure: J.K.

- **Does Medical Imaging learn different Convolution Filters?**
  ***Paul Gavrikov**, Janis Keuper*
  NeurIPS 2022 Workshop on Medical Imaging, 2022 (Gavrikov & Keuper, 2022c)

  Project idea: P.G. and J.K.; analysis: P.G.; paper writing: P.G. with input from J.K.; guidance, funding, and infrastructure: J.K.

- **An Extended Study of Human-Like Behavior Under Adversarial Training**
  ***Paul Gavrikov**, Janis Keuper, Margret Keuper*
  Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2023 (Gavrikov et al., 2023)

  Project idea and execution: P.G.; model collection, codebase, and analysis: P.G.; idea for spectrum analysis: M.K.; paper writing: P.G. with substantial input from M.K.; guidance, funding, and infrastructure: J.K. and M.K.

- **How Do Training Methods Influence the Utilization of Vision Models?**
  ***Paul Gavrikov**, Shashank Agnihotri, Margret Keuper, Janis Keuper*
  NeurIPS Workshop on Interpretable AI: Past, Present and Future, 2024 (Gavrikov et al., 2024a)

  Project idea and execution: P.G.; paper writing: P.G. with input from S.A./M.K./J.K.; guidance, funding, and infrastructure: J.K. and M.K.

Other publications not contributing to this thesis

- **GLOV: Guided Large Language Models as Implicit Optimizers for Vision Language Models**
  M. Jehanzeb Mirza, Mengjie Zhao, Zhuoyuan Mao, Sivan Doveh, Wei Lin, ***Paul Gavrikov****, Michael Dorkenwald, Shiqi Yang, Saurav Jha, Hiromi Wakaki, Yuki Mitsufuji, Horst Possegger, Rogerio Feris, Leonid Karlinsky, James R. Glass*
  In submission to Transactions on Machine Learning Research, 2024 (Mirza et al., 2024)
- **VisualTorch: Streamlining Visualization for PyTorch Neural Network Architectures**
  Willy Fitra Hendria, ***Paul Gavrikov***
  Journal of Open Source Software (JOSS), 2024 (Hendria & Gavrikov, 2024)
- **On the Interplay of Convolutional Padding and Adversarial Robustness**
  Paul Gavrikov*, Janis Keuper*
  Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops, 2023 (Gavrikov & Keuper, 2023a)
- **Robust Models are less Over-Confident**
  Julia Grabinski, ***Paul Gavrikov****, Janis Keuper, Margret Keuper*
  Advances in Neural Information Processing Systems (NeurIPS), 2022 (Grabinski et al., 2022a)

  An abstract under the same title was accepted at the ICML 2022 Workshop on New Frontiers in Adversarial Machine Learning (Grabinski et al., 2022b).

1.4 Notation

In this section, we define the notion that we use in this thesis. We largely follow the notation defined in Goodfellow et al. (2016), which we reproduce here for completeness, with some modifications, deletions and additions.

a	A scalar (integer or real)
$\boldsymbol{a}$	A vector
$\boldsymbol{A}$	A matrix
$\mathsf{\textbf{A}}$	A tensor
$\boldsymbol{I}_n$	Identity matrix with n rows and n columns
$\boldsymbol{I}$	Identity matrix with dimensionality implied by context
$\mathbb{A}$	A set
$\mathbb{R}$	The set of real numbers
$\{0,1\}$	The set containing 0 and 1
$\{0,1,\dots,n\}$	The set of all integers between 0 and n
$[a,b]$	The real interval including a and b
$(a,b]$	The real interval excluding a but including b
$\mathbb{A}\backslash\mathbb{B}$	Set subtraction, *i.e.*, the set containing the elements of $\mathbb{A}$ that are not in $\mathbb{B}$
$\mathbb{A}\cup\mathbb{B}$	Set union, *i.e.*, the set containing the all elements of $\mathbb{A}$ and $\mathbb{B}$
$\mathbb{A}\cap\mathbb{B}$	Set intersection, *i.e.*, the set containing the elements that are in $\mathbb{A}$ and $\mathbb{B}$
$\varnothing$	Empty set $\{\}$
a_i	Element i of vector $\boldsymbol{a}$, with indexing starting at 1

a_{-i}	All elements of vector $\boldsymbol{a}$ except for element i
$A_{i,j}$	Element i,j of matrix $\boldsymbol{A}$
$\boldsymbol{A}_{i,:}$	Row i of matrix $\boldsymbol{A}$, we may omit trailing : for brevity
$\boldsymbol{A}_{i:j}$	Rows i - j (excluding j) of matrix $\boldsymbol{A}$
$\boldsymbol{A}_{:,i}$	Column i of matrix $\boldsymbol{A}$
$\mathsf{A}_{i,j,k}$	Element (i,j,k) of a 3-D tensor $\mathbf{\mathsf{A}}$
$\mathbf{\mathsf{A}}_{:,:,i}$	2-D slice of a 3-D tensor
a_i	Element i of the random vector $\mathbf{a}$
$\boldsymbol{A}^\top$	Transpose of matrix $\boldsymbol{A}$
$\boldsymbol{A} \odot \boldsymbol{B}$	Element-wise (Hadamard) product of $\boldsymbol{A}$ and $\boldsymbol{B}$
$\nabla_{\boldsymbol{x}} y$	Gradient of y with respect to $\boldsymbol{x}$
$\int f(\boldsymbol{x}) d\boldsymbol{x}$	Definite integral over the entire domain of $\boldsymbol{x}$
$P(\mathrm{a})$	A probability distribution over a discrete variable
$p(\mathrm{a})$	A probability distribution over a continuous variable, or over a variable whose type has not been specified
$\mathrm{a} \sim P$	Random variable a has distribution P
$\mathbb{E}_{\mathrm{x} \sim P}[f(x)]$ or $\mathbb{E} f(x)$	Expectation of $f(x)$ with respect to $P(\mathrm{x})$
$D_{\mathrm{KL}}(P \Vert Q)$	Kullback-Leibler divergence of P from Q
$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma^2})$	Gaussian distribution with mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\sigma^2}$
$\mathcal{U}(\boldsymbol{a}, \boldsymbol{b})$	Uniform distribution with range $[\boldsymbol{a}, \boldsymbol{b}]$
$\mathcal{O}(n)$	Big-O asymptotic notation with order of n complexity
i.i.d.	Independent and identically distributed
$f(\boldsymbol{x}; \boldsymbol{\theta})$	A function of $\boldsymbol{x}$ parametrized by $\boldsymbol{\theta}$. (Sometimes we write $f(\boldsymbol{x})$ and omit the argument $\boldsymbol{\theta}$ to lighten notation)
$\log(x)$	Natural logarithm of x
$\mathrm{ReLU}(x)$	Rectified Linear Unit (ReLU), $\max(0, x)$
$\vert\vert\boldsymbol{x}\vert\vert_p$	L^p norm of $\boldsymbol{x}$
$\vert\vert\boldsymbol{x}\vert\vert$	L^2 norm of $\boldsymbol{x}$
$\vert x \vert$	Cardinality, *i.e.*, the number of elements in x, if x is a vector, matrix, tensor, or set.
$\mathbb{I}(\mathrm{condition})$	Indicator function is 1 if the condition is true, 0 otherwise
$\ell(\hat{\boldsymbol{y}}, \boldsymbol{y})$	A generic loss function l between the model prediction $\hat{\boldsymbol{y}}$ and the true target y
$\mathrm{clip}(x, a, b)$	Clips all elements in x to $[a, b]$ range
$\mathrm{sign}(x)$	Returns the sign of x, *i.e.*, -1 for negative values, 1 for positive values, and 0 otherwise.
$\mathrm{Concat}(x^{(1)}, ..., x^{(n)})$	Concatenates all elements $x^{(i)}$ into a higher-dimensional structure.

p_{data}	The data generating distribution
$\hat{p}_{\text{data}}$	The empirical distribution defined by the training set
$\mathbb{X}$	A set of training examples
$\boldsymbol{x}^{(i)}$	The i-th example (input) from a dataset
$y^{(i)}$ or $\boldsymbol{y}^{(i)}$	The target associated with $\boldsymbol{x}^{(i)}$ for supervised learning

Chapter 2

Background

The background chapter provides an overview of the foundational concepts and methodologies relevant to the research presented in this thesis, establishing the context and framing the key ideas necessary for understanding the subsequent chapters.

We will begin with the introduction of Fourier transforms in Section 2.1, which is an integral part of this thesis. Then we will introduce relevant fundamentals of deep learning: Multi-Layer Perceptrons (MLPs) in Section 2.2, Convolutional Neural Networks (CNNs) in Section 2.3, and Vision Transformers (ViTs) in Section 2.4. Following these sections, we offer an introduction to the task of image classification and outline its role in deep learning in Section 2.5. Afterward, we will introduce the issue of generalization in deep learning in Section 2.6, with a specific focus on distribution shifts in Section 2.7, and robustness in Section 2.8. Finally, we will conclude this chapter with details about feature biases in the context of visual perception tasks in Section 2.9.

2.1 Fourier Transform

In this thesis, we will frequently convert signals (*e.g.*, images) from their source domain to the frequency domain for analysis. To accomplish this, we will utilize the Fourier transform (Fourier, 1822) and briefly introduce its definition based on Fisher et al. (1997); Pinsky (2008); Szeliski (2022), and application to images here.

Let $f(x)$ be a continuous signal in the time domain. The Fourier-transformed signal $F(u)$ is defined as

$$F(u) = \int f(x) \cdot e^{-i2\pi ux} \, dx, \tag{2.1}$$

and the corresponding inverse, which transforms the signal back to the time domain, is given by

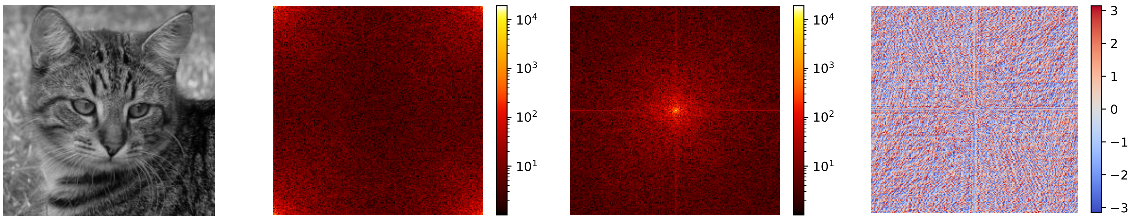
$$f(x) = \int F(u) \cdot e^{i2\pi ux} \, du. \tag{2.2}$$

Since we primarily apply the Fourier transform to images, we extend these definitions to the two-dimensional case:

$$F(u,v) = \iint f(x,y) \cdot e^{-i2\pi(ux+vy)} \, dx \, dy, \tag{2.3}$$

with the corresponding inverse given by

$$f(x,y) = \iint F(u,v) \cdot e^{i2\pi(ux+vy)} \, du \, dv. \tag{2.4}$$



(a) Source **(b)** Magnitude **(c)** Magnitude (shifted) **(d)** Phase (shifted)

Figure 2.1: Discrete Fourier transform of an image. (a) Original not-transformed image, (b) Fourier magnitude spectrum (log-scaled), (c) shifted Fourier magnitude spectrum (log-scaled), (d) shifted Fourier phase spectrum. **Image Source:** (Deng et al., 2009) (cat)

To handle discrete 2D inputs, we use the **discrete Fourier Transform (DFT)**. Using matrix notation, the 2D-DFT $\mathcal{F}$ of an input matrix $\boldsymbol{X} \in \mathbb{R}^{n \times m}$ is defined as $\mathcal{F}(\boldsymbol{X}) = \tilde{\boldsymbol{X}}$, where

$$\tilde{\boldsymbol{X}}_{u,v} = \sum_{x=1}^{n} \sum_{y=1}^{m} \boldsymbol{X}_{x,y} \cdot e^{-i2\pi\left(\frac{(u-1)(x-1)}{n} + \frac{(v-1)(y-1)}{m}\right)} \text{ for } u \in \{1, \dots, n\}, v \in \{1, \dots, m\}. \tag{2.5}$$

The inverse transform $\mathcal{F}^{-1}(\tilde{\boldsymbol{X}}) = \mathcal{F}^{-1}(\mathcal{F}(\boldsymbol{X})) = \boldsymbol{X}$ is given by

$$\boldsymbol{X}_{x,y} = \frac{1}{nm} \sum_{u=1}^{n} \sum_{v=1}^{m} \tilde{\boldsymbol{X}}_{u,v} \cdot e^{i2\pi\left(\frac{(u-1)(x-1)}{n} + \frac{(v-1)(y-1)}{m}\right)} \text{ for } x \in \{1, \dots, n\}, y \in \{1, \dots, m\}. \tag{2.6}$$

Note that while $\boldsymbol{X}$ is real-valued, its Fourier transform $\tilde{\boldsymbol{X}}$ will be complex. Thus, when analyzing the Fourier transform of an image $\boldsymbol{X}$ (see Figure 2.1 for an example), we typically focus on the **magnitude** (or power) spectrum, given by $|\mathcal{F}(\boldsymbol{X})|$, which corresponds to the absolute real part of the transform. This spectrum captures the geometric structure of the original image (Fisher et al., 1997), as illustrated in Figure 2.1b.

For visualization purposes, the spectrum is often shifted so that the first coefficient $\tilde{\mathbf{X}}_{1,1}$, which represents the image mean (or *DC-offset*), is centered. The coefficients are then arranged to increase in frequency from the center outward (compare Figure 2.1b with Figure 2.1c). In our example, we can see that vertical and horizontal frequencies dominate, and generally, most information is contained in lower frequencies. This holds true for most "natural" images (Ruderman, 1994).

The **phase** spectrum (shown in Figure 2.1d), containing the complex part of the Fourier transform, is less interpretable. However, retaining the phase spectrum is essential during the inverse transformation to fully reconstruct the original input (Fisher et al., 1997).

It is worth noting that the time complexity of the DFT for a sequence of length n is $\mathcal{O}(n^2)$. Hence, in practice, a family of more efficient algorithms known as the **Fast Fourier Transform (FFT)** (Cooley & Tukey, 1965) is commonly used. These algorithms achieve computational efficiency by exploiting underlying symmetries, reducing the time complexity to $\mathcal{O}(n \log n)$ (Szeliski, 2022). We will not introduce further details but imply FFT when utilizing Fourier transforms.

2.2 Multi-Layer Perceptrons (MLPs)

In deep learning, we utilize artificial neural networks to learn complex representations from data. This idea is loosely inspired by the organization of the (human) brain.

In the simplest case, such a network consists of a single artificial **neuron** or **perceptron** (Rosenblatt, 1958). The output $\hat{y}$ of the neuron is then given by a linear combination of c_{in} input features $\boldsymbol{x} \in \mathbb{R}^{c_{\text{in}}}$

that are activated by some **activation function** $\sigma(\cdot)$

$$\hat{y} = \sigma(\boldsymbol{x}\boldsymbol{w} + b), \tag{2.7}$$

where $\boldsymbol{w} \in \mathbb{R}^{c_{\text{in}}}$ and $b \in \mathbb{R}$ are the learnable parameters **weights** and **biases**, respectively. Historical choices for $\sigma(\cdot)$ were the tangens-hyperbolicus $\sigma(x) = \tanh(x)$ or the logistic $\sigma(x) = \frac{1}{1+e^{-x}}$ function. However, they are rarely used in modern neural networks. We will discuss modern activations further on.

To model multiple outputs – *i.e.*, to increase the number of neurons from 1 to c_{out}, we can represent our weight as matrix $\boldsymbol{W} \in \mathbb{R}^{c_{\text{out}} \times c_{\text{in}}}$ and the bias as vector $\boldsymbol{b} \in \mathbb{R}^{c_{\text{out}}}$ and call this a **layer**. The transformation then becomes:

$$\hat{\boldsymbol{y}} = \sigma(\boldsymbol{x}\boldsymbol{W}^\top + \boldsymbol{b}). \tag{2.8}$$

Building on this foundation, a single layer of perceptrons can be extended to multiple layers of neurons, giving rise to what is known as a **multi-layer perceptron (MLP)**. In an MLP, neurons are arranged in sequential layers, with each layer's output becoming the input to the next. This architecture allows MLPs to model more complex relationships by stacking several layers, each layer l having its own set of weights $\boldsymbol{W}^{(l)}$, biases $\boldsymbol{b}^{(l)}$, and activation functions $\sigma^{(l)}$:

$$\boldsymbol{x}^{(l+1)} = \sigma^{(l)}\left(\boldsymbol{x}^{(l)}\left(\boldsymbol{W}^{(l)}\right)^\top + \boldsymbol{b}^{(l)}\right). \tag{2.9}$$

This structure makes MLPs **fully connected** networks, meaning each neuron in a layer connects to every neuron in the subsequent layer. The first, last, and intermediate layers in MLPs are called **input, output, and hidden layers**, respectively. These networks may also be called feed-forward networks,[1] as the information flows in one direction from inputs to outputs. An important property of MLPs is that they can approximate any function if they are sufficiently large, as stated by the *universal function theorems* (Hornik et al., 1989; Cybenko, 1989).

Activation Functions

Activation functions can introduce non-linearity into an otherwise linear transformation, enabling neural networks like MLPs to learn more complex relationships. Without (non-linear) activation functions, stacking multiple layers would still yield a linear function. By applying a non-linear activation after each layer, MLPs can approximate a much broader range of functions, capturing intricate patterns in data that linear models cannot.

A popular choice across tasks in gradient-based learning is the **Rectified Linear Unit (ReLU)** activation function (popularized by Nair & Hinton (2010)):

$$\text{ReLU}(x) = \max(0, x). \tag{2.10}$$

It improves the training speed compared to tangens-hyperbolicus, due to faster derivatives (Krizhevsky, 2009), but also improves training stability by addressing the **vanishing gradient problem** (Bengio et al., 1994). This follows from its desirable property that the gradient is either 1 or 0 (rectified), never scaling gradients up or down. However, this property also leads to a new problem coined the **dying ReLU problem**, where once a neuron outputs zero, its gradient also becomes zero, and it effectively "dies" because it stops updating its weights during training. The frequency of dead neurons increases with depth in ReLU networks (Lu et al., 2020). This issue can be mitigated by allowing a small, non-zero gradient for negative activations – the corresponding activation function is termed **Leaky ReLU** (Maas et al., 2013) and defined as:

$$\text{LeakyReLU}(x) = \begin{cases} x, & x \geq 0 \\ \alpha x, & x < 0 \end{cases}. \tag{2.11}$$

[1] Other network types such as recurrent neural networks (Hopfield, 1982; Hochreiter & Schmidhuber, 1997) are not discussed in this thesis.

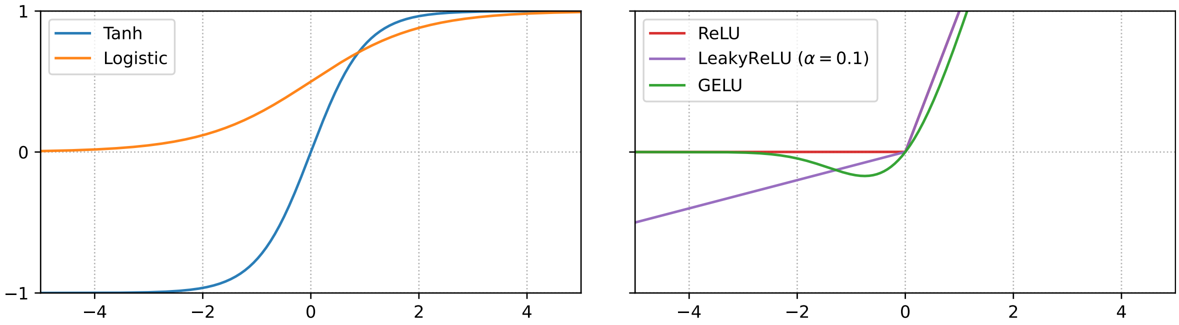


Figure 2.2: Activation functions. We visualize *Tanh*, *Logistic*, *ReLU*, *PreLU*, and *GELU* activation functions in the $[-5, 5]$ range.

It can be fruitful to change α into a learnable parameter, which is then called a **Parameterized ReLU (PReLU)** (He et al., 2015). However, gradient-based learning (Rumelhart et al., 1986) with any of these activation functions suffers from instability, as the functions are not differentiable at 0. A solution to this was given by the **Gaussian Error Linear Unit (GELU)** (Hendrycks & Gimpel, 2016), which is the standard choice in modern networks such as `ConvNeXt's` (Liu et al., 2022) or `ViT's` (Dosovitskiy et al., 2021).

$$\mathrm{GELU}(x) = x \cdot \frac{1}{2}\left(1 + \mathrm{erf}(x/\sqrt{2})\right), \tag{2.12}$$

where $\mathrm{erf}(\cdot)$ is the Gauss error function. For computational reasons, the function can be approximated by

$$\mathrm{GELU}(x) \approx 0.5 \cdot x \cdot \left(\tanh\left(\sqrt{2/\pi}\right) \cdot \left(x + 0.044715 \cdot x^3\right)\right). \tag{2.13}$$

2.3 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are a class of deep learning models specifically designed to process structured grid-like data, such as images (Goodfellow et al., 2016). Originating from ideas inspired by the visual cortex of animals (Hubel & Wiesel, 1959), CNNs have a layered architecture that enables them to automatically and adaptively learn spatial hierarchies of features, from low-level edges and textures to high-level patterns and objects (Zeiler & Fergus, 2013; Yosinski et al., 2014).

Each CNN layer applies a series of small, trainable filters that convolve over input data, detecting localized features and creating feature maps that highlight important elements of the input. Unlike traditional fully connected networks, CNNs are highly efficient for spatial processing because they reduce the need for excessive parameters by sharing weights across spatial dimensions (Goodfellow et al., 2016).

As CNNs are central to this thesis, we will introduce them in more detail. We will start this section with an introduction to the convolution operation in Section 2.3.1, then we will expand this to the definition of a layer in Section 2.3.2, and briefly discuss pooling in Section 2.3.3. Afterward, we will introduce various CNN architectures in Section 2.3.4, before wrapping up this section with a primer on learned convolution filters in Section 2.3.5.

2.3.1 Definition of the Convolution Operation

In the continuous space, the convolution operation between two functions f, g denoted by an asterisk $f * g$ at a point t is defined by

$$(f * g)(t) := \int f(a)g(t-a)\,da. \tag{2.14}$$

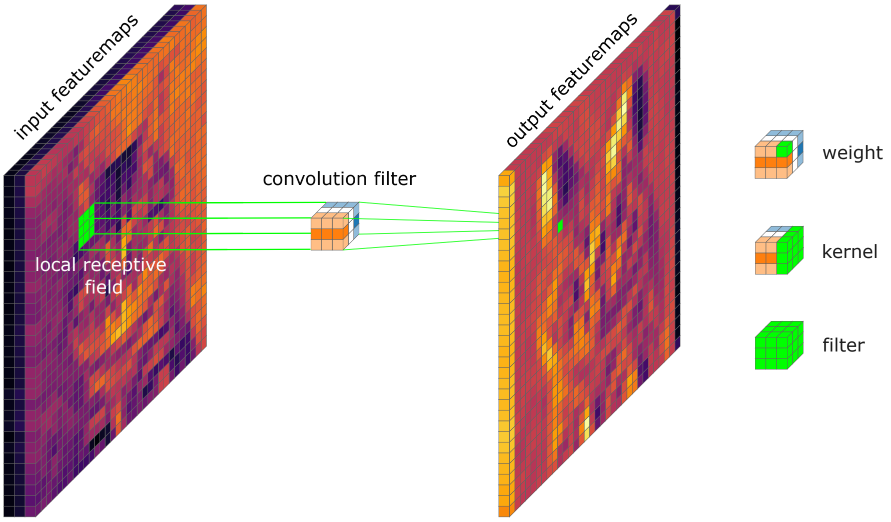


Figure 2.3: Function of a 2D convolution on multi-dimensional inputs. A convolution filter slides over the input and computes the dot product between the filter and the input under the local receptive field. We call a filter a stack of kernels and a kernel a 2D slice of the filter, respectively. A single element of the filter is called weight. A convolution layer in a neural network will typically contain multiple filters and, thus, produce an equivalent amount of feature maps (not shown here).

As this thesis largely deals with 2D image inputs, we expand the notation to two dimensions and the discrete case. First, let us assume a single channel (*e.g.*, a gray-scale image) image $\boldsymbol{X} \in \mathbb{R}^{w \times h}$ with resolution $w \times h$, and a convolution kernel (or short: **kernel**) $\boldsymbol{K} \in \mathbb{R}^{k_1 \times k_2}$ with shape $k_1 \times k_2$. Then the output (or **feature map**) $\boldsymbol{Y} = \boldsymbol{X} * \boldsymbol{K}$ for a given 2D index (i, j) is defined by[2]

$$\boldsymbol{Y}_{i,j} = \sum_m \sum_n \boldsymbol{X}_{i,j} \boldsymbol{K}_{i+m,j+n} \text{ for } i \in \{1, \dots, w\}, j \in \{1, \dots, h\}. \tag{2.15}$$

Theoretically, the kernel $\boldsymbol{K}$ can be of any shape and is not limited to being rectangular. However, in practice, it is often even square, meaning $k_1 = k_2$. For the rest of this thesis, if we refer to the kernel size as k, we imply a square $k \times k$-shaped kernel.

To process multichannel images $\mathbf{X} \in \mathbb{R}^{c \times w \times h}$ (*e.g.*, $c = 3$ for an image in the RGB space), we utilize a **filter** $\mathbf{F} \in \mathbb{R}^{c \times k \times k}$ which can be seen as a stack of kernels. For the special case of $c = 1$ the kernel is also a filter. Then, $\mathbf{Y} = \mathbf{X} * \mathbf{F}$ for a given 2D index (i, j) is defined by

$$\mathbf{Y}_{i,j} = \sum_l \sum_m \sum_n \mathbf{X}_{l,i,j} \mathbf{K}_{l,i+m,j+n} \text{ for } i \in \{1, \dots, w\}, j \in \{1, \dots, h\}. \tag{2.16}$$

The time complexity of a naive implementation of the 2D convolution is $\mathcal{O}(c \times w \times h \times k^2)$ and visualized in Figure 2.3.

2.3.2 The Convolution Layer

Similar to linear layers in MLPs, convolutional layers will utilize multiple filters in parallel to process input feature maps into multiple output feature maps. Additionally, each filter's output may be offset by a learnable bias term. Formally, we still deal with $\mathbf{X} \in \mathbb{R}^{c \times w \times h}$ inputs, but now have a learnable layer weight $\mathbf{W} \in \mathbb{R}^{c_{\text{out}} \times c_{\text{in}} \times k \times k}$ and an optional bias $\mathbf{B} \in \mathbb{R}^{c_{\text{out}}}$ term[3], where c_{out} is the number of

[2] Although the mathematical definition refers to cross-correlation, convolutions in machine learning are generally implemented using cross-correlation. Consequently, the terms are often used interchangeably.

[3] Bias terms are often omitted in modern CNNs when paired with normalization layers (such as batch-normalization), as these layers can effectively handle offset corrections.

Table 2.1: Space and time complexity comparison between different convolutions types. We ignore all parameters (*e.g.*, padding, dilation, stride, bias) except the kernel size and dimensions of input and output features.

	Complexity	
Method	**Time**	**Space**
Standard	$\mathcal{O}(c_{\text{out}} \times c_{\text{in}} \times w \times h \times k^2)$	$\mathcal{O}(c_{\text{out}} \times c_{\text{in}} \times k^2)$
Depthwise Separable	$\mathcal{O}(c_{\text{in}} \times w \times h \times (k^2 + c_{\text{out}}))$	$\mathcal{O}(c_{\text{in}} \times (k^2 + c_{\text{out}}))$
Grouped (g groups)	$\mathcal{O}\left(\frac{c_{\text{out}} \times c_{\text{in}} \times w \times h \times k^2}{g}\right)$	$\mathcal{O}\left(\frac{c_{\text{out}} \times c_{\text{in}} \times k^2}{g}\right)$

output features, c_{in} the number of input features. The other parameters follow the previous definition. The output $\mathbf{Y}$ of such a layer is then defined as

$$\mathbf{Y}_j = (\mathbf{B}_j +) \sum_{i=1}^{c_{\text{in}}} \mathbf{W}_{j,i} * \mathbf{X}_i, \text{ for } j \in \{1, \dots, c_{\text{out}}\}. \tag{2.17}$$

The final time complexity of a naive implementation of a 2D convolution layer is then $\mathcal{O}(c_{\text{out}} \times c_{\text{in}} \times w \times h \times k^2)$, with a space complexity of $\mathcal{O}(c_{\text{out}} \times c_{\text{in}} \times k^2)$. For clarity, we will omit the bias term in the following chapters.

Depthwise Separability of the Convolution Operation

It is possible to reduce the complexity of the "normal" convolution operation by separating it among an axis. A common separation in deep learning is across the channel dimension (depth) called **depthwise-separable convolutions** (Howard et al., 2017; Chollet, 2017). Instead of learning a 3D filter that operates on the entire input tensor's depth, only one filter (kernel) is learned per input channel:

$$\mathbf{Y}_j^{(l)} = \mathbf{W}_j^{(l)} * \mathbf{X}_j^{(l)}, \text{ for } j \in \{1, \dots, c_{\text{in}}\}. \tag{2.18}$$

Unlike in normal convolutions, the channels are processed independently of each other, resulting in no exchanged information between channels. To exchange information between channels, these intermediate outputs $\mathbf{Y}^{(l)}$ are further processed by a linear combination into the final outputs $\mathbf{Y}^{(l+1)}$, which is often implemented by a pointwise convolution (a convolution with a kernel size of 1):

$$\mathbf{Y}_j^{(l+1)} = \sum_{i=1}^{c_{\text{in}}} \mathbf{W}_{j,i}^{(l+1)} \cdot \mathbf{Y}_i^{(l)}, \text{ for } j \in \{1, \dots, c_{\text{out}}\}. \tag{2.19}$$

In total, this changes the time complexity $\mathcal{O}(w \times h \times c_{\text{in}} \times (k^2 + c_{\text{out}}))$ and space complexity to $\mathcal{O}(c_{\text{in}} \times (k^2 + c_{\text{out}}))$.

Actual implementations often mix non-linearities between the depthwise and pointwise convolutions and are, thus, not truly linear convolutions (Howard et al., 2017; Tan & Le, 2020; Liu et al., 2022) – in the sense that they can express transformations that a normal convolution would not be able to.

Grouped convolutions (Krizhevsky et al., 2012; Xie et al., 2017) generalize this concept of separating input channels. Instead of fully separating the depth into c_{in} individual 2D slices as in depthwise-separable convolutions (where the number of groups $g = c_{\text{in}}$), grouped convolutions operate on subsets of input channels ($g < c_{\text{in}}$). Standard convolutions can be viewed as a special case where $g = 1$. Because grouped convolutions already mix different channels locally within groups (assuming $g < c_{\text{in}}$), a consecutive pointwise convolution is often omitted. Please note that the resulting operation is not equivalent to the standard convolution. A full comparison of space and time complexity of all introduced methods is shown in Table 2.1.

Hyperparameters of Convolution Layers

Beyond the number of input and output channels, convolution layers (*e.g.*, as implemented in *PyTorch* (Paszke et al., 2019)) are parameterized by additional hyperparameters, each influencing how the kernel interacts with the input data:

- **Kernel Size**: Defines the spatial dimensions (width and height) of the convolutional filter. A larger kernel size allows the network to capture more context from the input, but also increases computational cost and the number of learnable parameters.
- **Padding**: Determines the amount of padding added around the input tensor's border. Padding is typically used to control the spatial dimensions of the output feature map, preserving input size when desired (known as "same" padding) or reducing it when not padded (known as "valid" padding). The actual method of filling is often controlled by an additional "padding mode" parameter and may be data-dependent (*e.g.*, mirroring the image across the axes) or independent (*e.g.*, filling with zeros).
- **Stride**: Specifies the step size of the convolution as it moves across the input tensor – with a default value of 1. Increasing the stride models a lossy downsampling of the feature map by reducing its spatial dimensions, as it effectively skips over certain input positions.
- **Dilation**: Controls the spacing between individual weights within the kernel, allowing for "gaps" in the kernel – with a default of 1 (no dilation). Dilation enlarges the effective receptive field without increasing the number of learnable parameters, making it particularly useful for capturing long-range dependencies in the inputs.
- **Groups**: Divide the input and output channels into separate groups that are convolved independently. Grouped convolutions allow for reduced computation and can encourage more efficient use of parameters.
- **Bias**: Adds a constant value to each element of the convolved output. This allows the layer to model affine transformations, increasing its representational power, but is often omitted in newer models, as batch-normalization has a similar effect (Ioffe & Szegedy, 2015).

2.3.3 Pooling

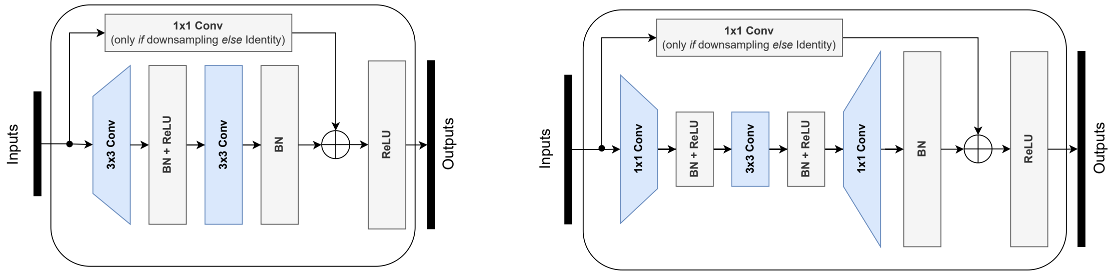
Pooling is another critical operation in CNNs, used to progressively reduce the spatial dimensions of feature maps while preserving essential information. By applying pooling after convolutional layers, CNNs achieve several benefits: reduced computational load and an increased receptive field, which helps capture features at multiple scales and introduces a degree of local translation invariance (Goodfellow et al., 2016).

Pooling operations perform downsampling by aggregating values within local regions of the input feature maps. This can be seen as a specialized form of a convolution, where the kernel models an aggregation function (*e.g.*, maximum or average) instead of a weighted sum. The size of this local region is defined by the pooling kernel size, and the stride determines how the kernel is shifted across the input, similar to convolutional layers. While max and average pooling layers are the most common (LeCun et al., 1989a; Krizhevsky et al., 2012; Simonyan et al., 2014; He et al., 2016; Huang et al., 2017), more complex and even learnable pooling techniques exist and sometimes do not operate under locality constraints (Chen et al., 2023b).

2.3.4 Network Architectures

The `Neocognitron` (Fukushima, 1988) laid the conceptual groundwork for CNNs. It was inspired by the human visual cortex and employed a hierarchical, layered architecture of fixed feature-detecting neurons to recognize patterns in images.

The evolution of convolutional neural networks (CNNs) for image recognition began with LeCun et al. (1989a), who introduced the core idea of learnable convolutional filters trained via backpropagation to



(a) Basic-Block

(b) Bottleneck-Block

Figure 2.4: Building blocks of the `ResNet` network. We show (a) the *Basic-Block* used in shallower models and the (b) *Bottleneck-Block* used in deeper models (He et al., 2016). The height of the blocks (outside the downsampling branch) visualizes the number of channels.

detect handwritten digits in zip codes. Their `LeNet-1` architecture employed two 5 × 5 convolutional layers and two linear layers with sigmoid activations. This was followed by LeCun et al. (1998a)'s `LeNet-5`, which refined the architecture by adding subsampling (pooling) layers after the convolutional layers and increasing the number of linear layers to three. `LeNet-5` demonstrated the effectiveness of CNNs for real-world applications, driving interest in deep learning for pattern recognition.

The big breakthrough moment for deep learning is often attributed to *ImageNet* (Deng et al., 2009), and in particular, its 2012 challenge winner `AlexNet` (Krizhevsky et al., 2012). `AlexNet` was the first neural network to win the annular challenge, exceeding the runner-up's accuracy by 10.8%. The network architecture expanded upon `LeNet` with more layers, ReLU activations, and dropout (Srivastava et al., 2014) for regularization. The network also mixed kernel sizes: the first layer uses 11 × 11, the second layer 5 × 5, and all other layers 3 × 3 kernels. Importantly, the network was amongst the first successful models to be trained with GPU acceleration. This success sparked a new wave of interest in deep learning and multiple architectural changes in the coming years.

Simonyan & Zisserman (2015) introduced `VGG`, which demonstrated that stacking smaller 3 × 3 kernels could yield faster networks compared to networks with larger kernels like `AlexNet` while matching or exceeding their performance. `VGG`'s architecture, which originally included 16 or 19 layers, allowed for deeper networks while keeping parameter count manageable.

Lin et al. (2014) introduced *Networks-in-Networks* or pointwise (1 × 1) convolutions, which enabled more efficient mixing of channels and can be seen as a linear operation on the depth axis. This innovation paved the way for compact yet expressive model architectures and enabled more flexible manipulation of feature maps.

Szegedy et al. (2014a) introduced the `Inception v1` architecture (also known as `GoogLeNet`), which utilized parallel paths with varying kernel sizes in each layer to capture information at multiple scales. This design enabled more efficient computation and increased model capacity, specifically in model depth, compared to `VGG`. The next step in the design of `Inception` was the introduction of batch-normalization layers (Ioffe & Szegedy, 2015), which resulted in the next iteration known as `Inception v2`. Further modifications to the architecture, as well as the use of auxiliary losses, resulted in `Inception v3` (Szegedy et al., 2016b). The `Inception` architecture was also amongst the first to switch to a single linear layer head for classification tasks (also see Section 2.5.1).

Another breakthrough was the invention of `ResNet` (He et al., 2016), which introduced residual skip connections, allowing the network to pass an (almost) unmodified input across layers and thereby reducing the vanishing gradient problem. Instead, of learning a transformation of the input signal $\boldsymbol{x}^{(l+1)} = g\left(\boldsymbol{x}^{(l)}\right)$, a residual is learned:

$$\boldsymbol{x}^{(l+1)} = g\left(\boldsymbol{x}^{(l)}\right) + \boldsymbol{x}^{(l)}. \tag{2.20}$$

The introduction of residual connections allowed for the training of extremely deep networks, reaching up to 1202 layers and significantly advanced image classification performance. `ResNet`'s architecture differs from `Inception`'s parallel branch design, instead opting for a simpler, sequential arrangement of residual blocks (shown in Figure 2.4), reminiscent of `VGG` but without intermediate pooling.

The network begins with a stem consisting of a 7×7 convolution followed by normalization, rectification, and pooling. Then, the feature maps are processed through four stages, where each consists of multiple residual blocks. The first residual block in each stage downsamples feature maps and increases the number of channels. Smaller `ResNets` utilize Basic-Blocks, each containing two 3×3 convolutional layers. Increases in the number of channels (accompanied by reductions in spatial resolution) are handled by the first convolution within these blocks. Larger `ResNets` utilize Bottleneck-Blocks for improved efficiency. These blocks employ a pointwise convolution to reduce the number of channels, followed by a 3×3 convolution, and then another pointwise convolution to expand the channels back (and potentially increase them further, mirroring the channel expansion of Basic-Blocks). Global average pooling is performed on the final feature maps. For lower resolution images, He et al. (2016) proposes `ResNets` that only contain three stages and other changes in order to reduce the number of downsampling operations.

The concept of residual connections is often described as enabling a direct and uninterrupted flow of information. While this is a useful intuition, in `ResNet`, this flow is only partially direct. The initial stem, which includes convolutional and pooling operations, as well as the downsampling layers within the residual blocks and the final global average pooling, introduce transformations that affect the information flow. However, the residual connections still provide a significant improvement in gradient flow compared to plain networks.

He et al. (2015) further optimized the gradient flow by using a more sophisticated initialization of parameters and optimized activations and eventually surpassed the estimated 5.1% ImageNet top-5 error of humans (Karpathy, 2014; Russakovsky et al., 2015).

A different line of work attempted to decrease the latency and reduce the parameters of neural networks for mobile applications – (Howard et al., 2017) introduced `MobileNet`, which employed depthwise-separable convolutions and inverted bottleneck layers. This design enabled the creation of lightweight, efficient networks without compromising accuracy significantly. The key idea here is to replace traditional convolutions with depthwise-separable ones as discussed in Section 2.3.2 and to increase the number of channels within the residual block instead of reducing them as in traditional Bottleneck-Blocks. Chollet (2017) also brought the idea of depthwise-separable convolutions to non-resource-constrained models.

`DenseNet` (Huang et al., 2017) extended the idea of residual connections introduced by `ResNet` beyond individual blocks. `DenseNets` concatenate the feature maps of all preceding layers in a block to the current layer's input. This dense connectivity pattern allows each layer to directly access the feature maps learned by all earlier layers, promoting feature reuse and reducing the vanishing gradient problem. This approach leads to improved performance with fewer parameters compared to traditional convolutional neural networks but increases computational and memory costs. Yet, the prevalence of standard skip connections over dense skip connections in newer CNN architectures suggests a preference for cost (Tan & Le, 2020; Trockman & Kolter, 2023; Liu et al., 2022).

`EfficientNet` (Tan & Le, 2020) represents a significant advancement in designing CNNs with both accuracy and computational efficiency in mind. Unlike prior architectures that focused primarily on deeper or wider networks, `EfficientNets` employ a compound scaling method that balances depth (number of layers), width (number of channels per layer), and resolution (input image size). This scaling strategy first attempts to identify an optimal baseline model, `EfficientNet-B0`, which is then scaled up to create a family of models (`EfficientNet-B1-7`) for different resource levels. The compound scaling formula, designed through neural architecture search (NAS), optimizes the network's structure by adjusting the three dimensions in a balanced way. This ensures that each layer of the network grows proportionally as the model scales up, which prevents local over- or underfitting while maximizing efficiency. Later work proposed further optimizations to this architecture (Tan & Le, 2021).

ConvNeXt (Liu et al., 2022) proposed a modern family of CNNs in light of the transformer architectures that had become popular in vision tasks. Based on the ResNet architecture, the authors increased kernel size to 7 × 7, incorporated transformer-inspired patch preprocessing, GELU activations, and multiple other smaller changes, demonstrating that CNNs could still be competitive with transformer models. Further optimizations of the resulting model were obtained by introducing Global Response Normalization Layers and the use of self-supervised pretraining (Woo et al., 2023).

Modern trends Recently, there has been a trend to increase kernel sizes to improve receptive field and model expressiveness, as demonstrated by ConvNeXt (Liu et al., 2022) and other subsequent models (Ding et al., 2022; Liu et al., 2023d). Additionally, some CNNs (*e.g.*, (Trockman & Kolter, 2023; Liu et al., 2022)) have begun adopting transformer-like patch preprocessing (see Section 2.4 for details), blurring the line between CNNs and transformers for improved flexibility and performance in complex visual tasks.

2.3.5 Understanding Learned CNNs Filters

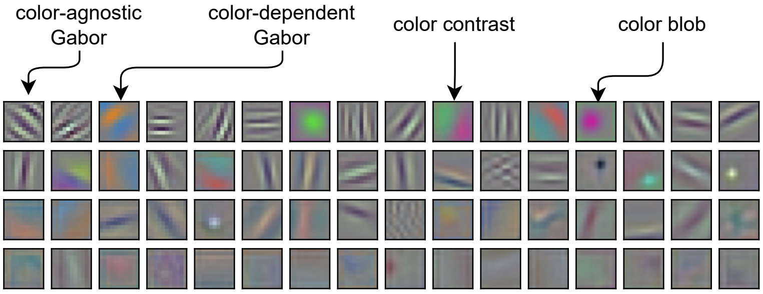


Figure 2.5: Convolution filters in the first layer. Here we show 11 × 11 filters extracted from the first convolution layer of an *ImageNet*-trained AlexNet (Krizhevsky et al., 2012).

Unlike MLPs, the spatial dimension of convolution filters allows us to *sometimes* interpret the function of a (learned) filter. For instance, it is well-established that the first layer of any CNN learns a fairly universal set of filters (shown in Figure 2.5), including color-agnostic and color-dependent Gabor filters processing texture information (Gabor, 1946) (also found in the V1 of the human visual cortex (Olshausen & Field, 1997)), filters sensitive to color contrast, color blobs activating to specific colors, amongst some others. Yosinski et al. (2014) suggest that these first-layer filters are not specific to task, data, or network architecture, exhibiting transferability across various models. In contrast, deeper layers exhibit increasing specialization with depth. This is further confirmed by the analysis of (Zeiler & Fergus, 2013), which employed *Deconvolutional Networks* (Zeiler et al., 2010) to understand features of higher-layer filters.

Unfortunately, a key limitation of filter interpretability is that while the first layer is easily interpretable due to its direct operation on the input image (see Figure 2.5), deeper layers operate on mixed representations from all preceding layers, obscuring the interpretability of encoded transformation. Consequently, the number of interpretability studies focusing on deep filters is relatively limited. One notable, thorough study of deeper layers is the exhaustive analysis of filters, connections, and their organization in an *ImageNet*-trained InceptionV1 model (Szegedy et al., 2016a) presented in (Olah et al., 2020a,b; Cammarata et al., 2020; Olah et al., 2020c; Schubert et al., 2021; Cammarata et al., 2021; Voss et al., 2021a,b; Petrov et al., 2021). However, the authors primarily utilized feature visualizations (Olah et al., 2017) for the interpretation of deeper layers.

A separate research direction focuses on a quantitive analysis of the “goodness” filters, often for pruning purposes. This encompasses methods ranging from simple magnitude-based kernel or filter importance estimation (Anwar et al., 2017) to more sophisticated information criteria based on pattern diversity (Li et al., 2019).

Understanding the structure of filters without interpreting them can also be of interest to improve initialization, aiming to enhance performance or reduce the number of necessary learned parameters. This fundamentally corresponds to a prior in weight space, offering an alternative to the commonly practiced i.i.d. parameter initialization. To this end, Trockman et al. (2023) studied the covariance structure of depthwise filters in the `ConvMixer` model (Trockman & Kolter, 2023) and proposed a performance-enhancing initialization scheme. Similarly, (Babaiee et al., 2024b) clustered filters of depthwise-separable convolutions, showing that filters are often derived from clusters of a few patterns.

2.4 Vision Transformers (ViTs)

While CNNs were the dominant paradigm for vision tasks in the past decade, transformer-based architectures have often exceeded the performance in multiple tasks such as image classification, object detection, semantic segmentation (Dosovitskiy et al., 2021; Liu et al., 2021b), and learning universal representations suitable for multiple down-stream tasks (Fang et al., 2023; Wang et al., 2023a; Oquab et al., 2024).

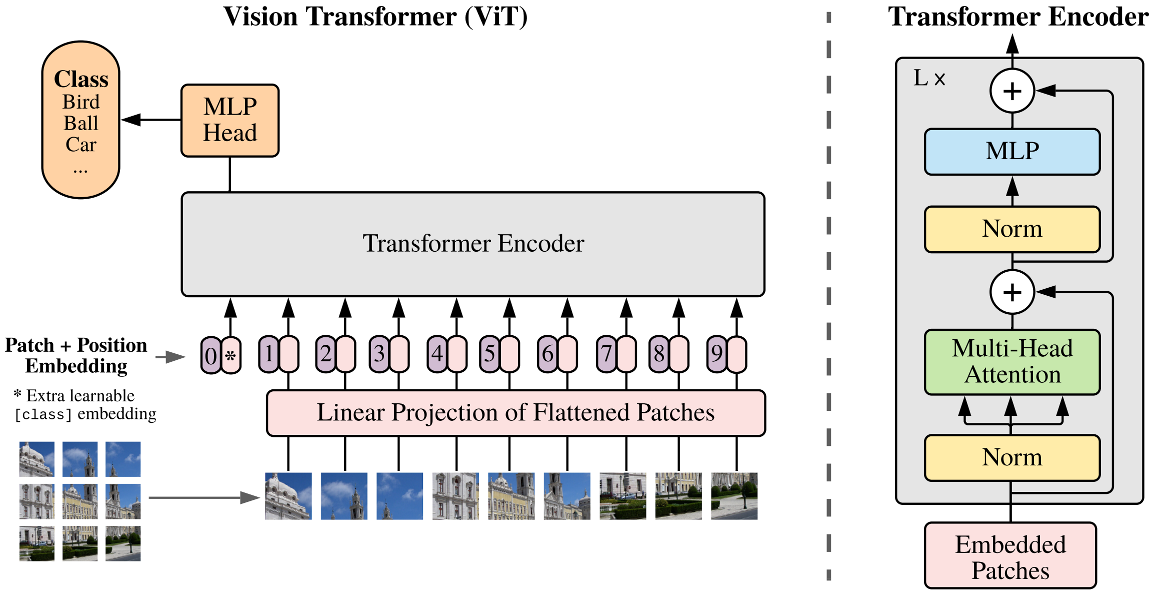


Figure 2.6: `ViT` architecture. Images are turned into patches, flattened and projected to high-dimensional tokens, concatenated with position encoding, and processed by multiple transformer encoder layers. **Image Source:** (Dosovitskiy et al., 2021)

`Vision Transformers (ViTs)` (Dosovitskiy et al., 2021) shown in Figure 2.6 reimagine image processing by replacing convolutions in favor of transformer-based **self-attention** mechanisms (Vaswani et al., 2017). While CNNs rely on localized kernels to progressively learn features from input images, `ViTs` process images as a sequence of patches (*tokens*), processing them similarly to tokens in a sentence.

This is achieved by first transforming an input image $\mathbf{X} \in \mathbb{R}^{c \times w \times h}$ into a flattened sequence $\boldsymbol{X}^p \in \mathbb{R}^{n \times (p^2 \cdot c)}$ of n patches of $p \times p$ size, and then linearly projecting it into a "token" $\boldsymbol{X}^t \in \mathbb{R}^{n \times d}$ with a latent vector size d. The dimensionality of the token and the number of tokens remains unchanged throughout all layers of the `ViT` – which is another key differentiator for classical CNNs.

Attention operations are invariant to the patch order, so a position embedding localizing the patch token in the original sample is appended for spatial inputs like images.

For classification tasks, the original ViT design also included a special learnable [class] token that is prepended to the image token sequence and eventually transformed into a prediction by a linear layer – the remaining tokens were not utilized. Newer models do not rely on a dedicated [class] token and instead discriminate over an aggregated representation of all tokens. However, the addition of separate input-independent tokens can help to make ViTs more interpretable (Darcet et al., 2024).

At the heart of the vision transformer is the self-attention operation, which allows ViTs to capture both short- and long-range dependencies within an image – even in early layers and without the need for explicitly hierarchical feature extraction – making them often more robust than CNNs (Naseer et al., 2021). Given an input sequence $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ consisting of n tokens with dimensionality d, we linearly project this by $\boldsymbol{W}^Q, \boldsymbol{W}^K, \boldsymbol{W}^V \in \mathbb{R}^{d \times d_h}$ into queries $\boldsymbol{Q} \in \mathbb{R}^{n \times d_h}$, keys $\boldsymbol{K} \in \mathbb{R}^{n \times d_h}$, and values $\boldsymbol{V} \in \mathbb{R}^{n \times d_h}$, with a hidden dimension d_h. Self-attention then computes a weighted sum of the values, where the weights (*attention map*) are determined by the dot-product of the query and the keys, scaled by the square root of the hidden dimension d_h, and normalized by the Softmax function (see Equation 2.24):

$$\begin{aligned}\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V} &= \boldsymbol{X}\boldsymbol{W}^Q, \boldsymbol{X}\boldsymbol{W}^K, \boldsymbol{X}\boldsymbol{W}^V \\ \text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) &= \underbrace{\text{Softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d_h}}\right)}_{\text{Attention Map}} \boldsymbol{V}\end{aligned} \tag{2.21}$$

The feature maps are generated by multiple **attention heads**. This process is called **multi-head self-attention (MHA)**, where every head serves as a neuron applying a (unique) attention transformation to the input. The resulting outputs are concatenated and linearly projected by $\boldsymbol{W}^O \in \mathbb{R}^{h \cdot d_h \times d}$ back to d dimensional outputs.

$$\begin{aligned}\text{MHA}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) &= \text{Concat}(\text{head}_1, \ldots, \text{head}_h)\boldsymbol{W}^O \\ \text{where head}_i &= \text{Attention}(\boldsymbol{X}\boldsymbol{W}_i^Q, \boldsymbol{X}\boldsymbol{W}_i^K, \boldsymbol{X}\boldsymbol{W}_i^V)\end{aligned} \tag{2.22}$$

Convolutions in transformers Vision transformers are not free of convolutions. The original ViT model utilized a strided convolution with a kernel size of p and a stride of p to partition the input image into patches. These patches were then linearly projected into token embeddings by d filters. Later refinements, as seen in ViT$_C$ (Xiao et al., 2021b), replaced this initial "patchification" step with iterative downsampling through multiple convolution layers. A subsequent pointwise convolution was employed to match the latent vector dimensionality. This change improved optimization stability, accuracy (Xiao et al., 2021b), and robustness (Singh et al., 2023).

2.5 Image Classification

Image classification is the task of assigning an image to one or more predefined classes or labels. In the simplest case, binary classification, we aim to categorize an image into one of two classes. For instance, we might want to classify an image as either a "cat" or a "dog."

When dealing with more than two classes, we have two primary approaches: single-label and multi-label classification. In single-label classification, we assign a single label to an image, even if it could potentially belong to multiple classes. For example, an image of a "cat wearing a hat" would be classified solely as a "cat." In contrast, multi-label classification allows us to assign multiple labels to an image, such as "cat" and "hat" in the previous example. While multi-label classification offers a more nuanced approach, most research, including this thesis, focuses on single-label classification.

In the following subsections, we will introduce a few ways to classify the image with different networks that are relevant to this thesis. Conceptually, all of these methods involve generating a **logit** vector, which can be seen as a list of scores, one for each label. The label with the highest score is then selected as the predicted label. Formally, given a network f, this can be expressed as

$$\arg\max_i f(\boldsymbol{x}). \tag{2.23}$$

Often, we are not only interested in the top prediction but also the corresponding confidence for the prediction (and confidence in alternate predictions). To achieve this, we can convert our logit vector $\boldsymbol{z} = f(\boldsymbol{x})$ with $\boldsymbol{z} \in \mathbb{R}^n$, where n is the number of class labels, into (pseudo-)probabilities using the Softmax function, such that outputs lie in the $[0, 1]$ range and sum to 1:

$$\text{Softmax}(\boldsymbol{z}, \tau) = \begin{pmatrix} \hat{\boldsymbol{y}}_1 \\ ... \\ \hat{\boldsymbol{y}}_n \end{pmatrix}, \text{ where } \hat{\boldsymbol{y}}_i = \frac{e^{\boldsymbol{z}_i/\tau}}{\sum_{j=1}^n e^{\boldsymbol{z}_j/\tau}}. \tag{2.24}$$

The Softmax function's entropy, a measure of its uncertainty (Shannon, 1948), can be adjusted using a temperature scale τ, with a default value of 1. Reducing τ below 1 results in a probability distribution with lower entropy, effectively making the model's predictions more peaked and confident, assigning higher probabilities to the most likely classes and lower probabilities to less likely ones. Instead, increasing τ beyond 1 smoothens the distribution and decreases the confidence for each class in favor of a more uniform representation.

When we refer to probabilities predicted by some model, we imply the Softmax-processed logits with temperature $\tau = 1$, unless otherwise stated.

2.5.1 Linear Classification of Image Encoder Representations

Conceptually, a discriminative neural network for image classification can be separated into a feature-extraction part ϕ_{enc} generating a representation of the input sample in a high-dimensional space (features), and a task-specific classification head doing the actual classification (Goodfellow et al., 2016). Older networks like `AlexNet` (Krizhevsky et al., 2012) or `VGG` (Simonyan & Zisserman, 2015) used MLPs with non-linear and additional regularization such as dropout (Srivastava et al., 2014) in the head to transform the representation from feature space into class-wise predictions scores (the logits).

Modern networks will typically generate more disentangled feature spaces that are separated by a single linear layer:

$$\mathbf{W} \cdot \phi_{\text{enc}}(\mathbf{X}) + \boldsymbol{b}, \tag{2.25}$$

where $\phi_{\text{enc}}(\mathbf{X}) \in \mathbb{R}^d$ generates a d-dimensional representation, that is modelled into logits for all n classes, by a learnable weight $\mathbf{W} \in \mathbb{R}^{n \times d}$ and a bias $\boldsymbol{b} \in \mathbb{R}^n$.

2.5.2 Classification in Joint-Embedding Spaces

In **joint-embedding classification**, a model is learned to represent two or more modalities, such as images and text, in a shared embedding space (Chen et al., 2020c; Radford et al., 2021; Jia et al., 2021). At inference time, a distance metric defines the logits by measuring the distance between an input in one modality (*e.g.*, an image) and a given set of plausible labels in the other (*e.g.*, text). This method is also called **zero-shot classification**, as the model is not confined to the same set of prediction labels as in traditional classification, and both the number of labels and the labels themselves can be selected at inference time.

The cosine distance is the most common metric to extract a logit vector from a joint-embedding space. Given an image encoder ϕ_{img} and a text encoder ϕ_{txt}, we obtain the logit z describing the distance between a given image $\boldsymbol{x}_{\text{img}}$ and a prompt $\boldsymbol{x}_{\text{prompt}}$ by:

$$z = \frac{\boldsymbol{z}_{\text{img}} \cdot \boldsymbol{z}_{\text{txt}}}{\|\boldsymbol{z}_{\text{img}}\|_2 \|\boldsymbol{z}_{\text{txt}}\|_2}, \text{ with } \boldsymbol{z}_{\text{img}} = \phi_{\text{img}}(\boldsymbol{x}_{\text{img}}) \text{ and } \boldsymbol{z}_{\text{txt}} = \phi_{\text{txt}}(\boldsymbol{x}_{\text{prompt}}). \tag{2.26}$$

The prompt is often a combination of the class label and some surrounding text or template, like `"a photo of a <class>"`, where `<class>` denotes the class label (Radford et al., 2021).

However, a key limitation of this classification approach is that not every label may be equally well represented in the embedding space. For instance, even semantically similar or close labels could be represented in different regions of the embedding space, leading to different performances under classification. One relatively simple solution is to use an average text embedding obtained from an ensemble of m prompts $\boldsymbol{x}_{\text{prompt}}^{(i)}$ (Radford et al., 2021). Then,

$$\boldsymbol{z}_{\text{txt}} = \frac{1}{n} \sum_{i=1}^{m} \frac{\phi_{\text{txt}}\left(\boldsymbol{x}_{\text{prompt}}^{(i)}\right)}{\|\phi_{\text{txt}}\left(\boldsymbol{x}_{\text{prompt}}^{(i)}\right)\|_2}. \tag{2.27}$$

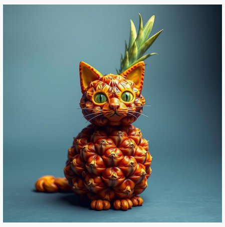
Both prompt templates and ensembles can be further optimized for the downstream task at hand, which is an active research line under the umbrella term of *prompt engineering* (*e.g.*, see (Zhou et al., 2022; Roth et al., 2023; Mirza et al., 2024)).

2.5.3 Classification with Multi-Modal LLMs

Large language models (LLM) support a multitude of tasks, exceeding simple classification tasks, however we can also turn them into simple classifiers. Conceptually, we could prompt a multi-modal LLM to classify a given image. However, the autoregressive nature of LLMs generates an open-vocabulary answer that we have to map to a class prediction. This mapping would be difficult to implement, as the response might be in any format. Additionally, we would like a response as a logit vector, as before, to be able to model confidence. We propose two options to tackle this.

Visual Question Answering with Multiple-Choice Answers

Input



```
Which option best describes the image?
A. cat
B. pineapple
Answer with the option's letter from the given choices
directly.
```

Output

```
A
```

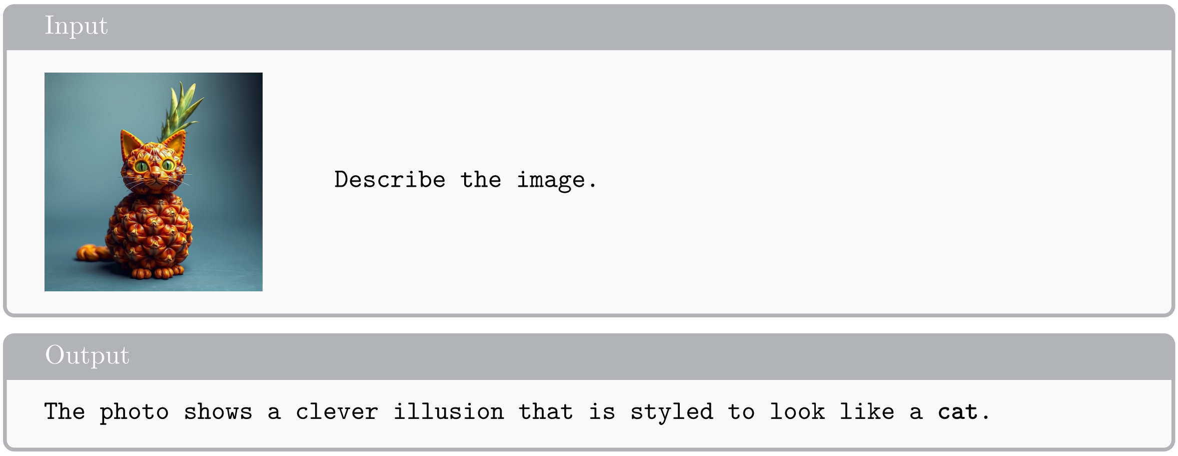
Figure 2.7: Classification via VQA. An example of an (ideal) conversation for multi-modal multiple-choice VQA prompt.

In our first option, which is inspired by *Visual Question Answering (VQA)* (Antol et al., 2015), we instruct the model to select an option from a given set of labels that best describes the given input, without an option for no answer or "other" (shown in Figure 2.7). Further, we instruct the model to only respond with the predicted option key and nothing else to better extract the response. This instruction is conceptually equivalent to a single-label, multi-class classification.

If the number of options is smaller than the token vocabulary, we can map each option to one token – *i.e.*, use the token as an answer key. Firstly, this has the advantage of faster inference – we can instruct the LLM to respond only with one token and stop generation after a single forward pass, but only if we trust that the LLM has followed the instruction to respond only with the option key and nothing else.

Secondly, and even more importantly, it allows us to model confidence in each option. Typically, obtaining a confidence judgment from an LLM is non-trivial; on the one hand, simply asking for its confidence in the prompt is unlikely to return a grounded response. On the other hand, we cannot use the conditional probability of the generated sequence as the confidence in the answer either, as there is a large number of semantically equivalent responses, and modeling all of them seems impossible or, at the least, impractical.

However, by artificially restricting the outputs to a single token, we can treat the logits of valid responses as probabilities for each option (and ignore all other token logits).



Input

`Describe the image.`

Output

`The photo shows a clever illusion that is styled to look like a cat.`

Figure 2.8: Classification of LLM generated descriptions. An example of an (ideal) conversation for a multi-modal description prompt.

Embedding of Image Captions

Alternatively, we can simply instruct the LLM to provide a description or caption of the input sample (shown in Figure 2.8) and then use a text encoder model to calculate the distance between the response and each label in the candidate set, similar to zero-shot classification with CLIP in Section 2.5.2.

Formally, given the response $\boldsymbol{x}_{\text{desc}}$, a candidate label $\boldsymbol{x}_{\text{label}}$, and a text embedding model ϕ_{txt}, we can obtain the logit by

$$z = \frac{\phi_{\text{txt}}(\boldsymbol{x}_{\text{desc}}) \cdot \phi_{\text{txt}}(\boldsymbol{x}_{\text{label}})}{\|\phi_{\text{txt}}(\boldsymbol{x}_{\text{desc}})\|_2 \|\phi_{\text{txt}}(\boldsymbol{x}_{\text{label}})\|_2}. \tag{2.28}$$

2.5.4 Image Classification Datasets

This section introduces a few widely-used image classification datasets relevant to this thesis.

MNIST (LeCun et al., 1998b) is a dataset of 28 ×28 pixel grayscale images of handwritten digits, which have been highly preprocessed to center the digits and remove any background. The dataset contains 10 distinct classes representing the digits 0 through 9 and is split into 60,000 training and 10,000 test images, balanced across classes.

SVHN (Netzer et al., 2011) can be seen as an out-of-distribution variant of *MNIST*, where 32 ×32 pixel images also show digits from 0 to 9 but are cropped from photos of house numbers and are, thus, harder to classify due to additional variance in style and color. The dataset is split into 73,257 training, 531,131 extra training, and 26,032 test images but is not balanced across classes.

CIFAR-10 (Krizhevsky et al., 2009) is a subset of the web-crawled *80 Million Tiny Images* dataset (Torralba et al., 2008) containing 32 ×32 pixel color images of natural objects in 10 classes: airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks. Objects in *CIFAR-10* are centered in each image, and the dataset is split into 50,000 training and 10,000 test images with a balanced distribution across classes.

CIFAR-100 (Krizhevsky et al., 2009) is a separate subset of the *80 Million Tiny Images* dataset (Torralba et al., 2008), featuring 100 fine-grained classes grouped into 20 broader categories. While *CIFAR-10* and *CIFAR-100* share some similar objects, *CIFAR-100* presents a greater challenge due to its: (1) higher number of classes, (2) finer-grained labels, and (3) fewer samples per class. *CIFAR-100* also has a class-balanced split of 50,000 training and 10,000 test images.

ImageNet (Deng et al., 2009) is a large, unbalanced dataset containing 14,197,122 images across 21,841 synsets (or classes) of "natural objects" categorized by nouns in the *WordNet* hierarchy (Fellbaum, 1998). This full dataset is sometimes referred to as *ImageNet-21k* or *ImageNet-22k* (where the suffix indicates the approximate number of classes). Preprocessed versions of the *Fall 2011* and *Winter 2021* releases, which reduce the number of classes to roughly 11,000, are also available (Ridnik et al., 2021) and are sometimes labeled as *ImageNet-21k*, *-21k-P* for "preprocessed", or *-11k*. These large-scale versions of ImageNet are rarely used as benchmarks due to their massive size and lack of standardized train/validation/test splits, though they have been used to pre-train multiple state-of-the-art (SOTA) models (Dosovitskiy et al., 2021; Liu et al., 2021b, 2022).

When we refer to *ImageNet* or *ImageNet-1k* in this thesis, we imply the official 1,000-class subset from the *ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2012* (Russakovsky et al., 2015). This dataset mostly contains photographs of natural objects, including some fine-grained classes (*e.g.*, around 120 dog breeds). *ImageNet-1k* has become the standard benchmark for large-scale image classification. It consists of 1.28 million training images (unbalanced across classes, with mostly 500–1,000 images per class), 50,000 balanced validation images, and 100,000 test images (with unknown balance) with various resolutions. While labels are available for the training and validation sets, test set labels are withheld, and the official test server is now inactive. As a result, the validation set is commonly used as the community's de facto test set.

The impact of *ImageNet* has led to numerous studies examining the correlation between *ImageNet* performance and performance on other tasks (Recht et al., 2019; Shankar et al., 2021; Kornblith et al., 2019b; Miller et al., 2021), the creation of datasets testing the generalization capabilities of *ImageNet*-trained classifiers (Geirhos et al., 2018; Hendrycks & Dietterich, 2019; Wang et al., 2019; Geirhos et al., 2019; Hendrycks et al., 2021a,b; Mintun et al., 2021; Li et al., 2023d), and even the collection of new test sets following the original methodology, such as *ImageNet-v2* (Recht et al., 2019). Notably, while models experience a performance drop on *ImageNet-v2*, human performance remains unaffected (Shankar et al., 2020).

However, due to its vast size, some *ImageNet* samples contain imperfect annotations. Issues include incorrect labels, images with multiple valid classes but only one annotated label, and some overlapping or hierarchical relationships among classes (Yun et al., 2021), which have led to new annotation efforts of the test *ImageNet-ReaL* (Beyer et al., 2020) and train sets (Yun et al., 2021).

Training models on large datasets like *ImageNet-1k* can be computationally demanding. To address this, several alternatives have been proposed to reduce the number of classes, samples, or resolutions. Examples include *Imagenette/ImageWoof* (Howard, 2019) and *ImageNet-32* (Chrabaszcz et al., 2017). In this thesis, we will occasionally utilize *TinyImageNet* (Le & Yang, 2015), a subset of *ImageNet* containing 200 classes. The images are downsampled to 64 ×64 pixels, with class-balanced splits comprising 125,000 training samples, 25,000 validation samples, and 25,000 test samples.

2.6 Generalization

To train a machine learning model, we collect a dataset $\mathbb{X} = \{(\boldsymbol{x}^{(1)}, y^{(1)}), \ldots, (\boldsymbol{x}^{(n)}, y^{(n)})\}$ containing (potentially noisy) samples drawn from the true data distribution p_{data}. The dataset is then randomly split into disjoint training and test sets $\mathbb{X} = \mathbb{X}_{\text{train}} \cup \mathbb{X}_{\text{test}}$ with $\mathbb{X}_{\text{train}} \cap \mathbb{X}_{\text{test}} = \varnothing$.

During training, we then minimize the empirical risk on the training set $\mathbb{X}_{\text{train}}$ with n_{train} training samples, expressed by a function $\ell(\hat{y}, y)$ that models the loss (or error) between the predicted label distribution $\hat{y}$ and true label y (Goodfellow et al., 2016):

$$\min_{\boldsymbol{\theta}} \frac{1}{n_{\text{train}}} \sum_{i=1}^{n_{\text{train}}} \ell\left(f\left(\boldsymbol{x}_{\text{train}}^{(i)}; \boldsymbol{\theta}\right), y_{\text{train}}^{(i)}\right). \tag{2.29}$$

This process minimizes the loss on the training data, but of course, we also a model that is able to **generalize**, *i.e.*, we want it to perform well on new and unseen data as well. We quantify the **test error** or **generalization error** (Goodfellow et al., 2016) on the held-out test data by

$$\frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \ell\left(f\left(\boldsymbol{x}_{\text{test}}^{(i)}; \boldsymbol{\theta}\right), y_{\text{test}}^{(i)}\right). \tag{2.30}$$

Statistical learning theory taught us that we only have a chance to reduce train and test error at the same time if both train and test set are i.i.d.– meaning that the samples in each dataset are independent and drawn from the same underlying distribution (Bishop, 1995; Goodfellow et al., 2016).

However, the distribution $\hat{p}_{\text{data}}$ modeled by our collected dataset only approximates the true distribution and may diverge from it in many ways. In that case, the empirical risk we obtain at test time may severely underestimate the true risk

$$\mathbb{E}_{(\boldsymbol{x},y)\sim\hat{p}_{\text{data}}}[\ell(f(\boldsymbol{x}; \boldsymbol{\theta}), y)] \ll \mathbb{E}_{(\boldsymbol{x},y)\sim p_{\text{data}}}[\ell(f(\boldsymbol{x}; \boldsymbol{\theta}), y)]. \tag{2.31}$$

If that were to be the case, the model would poorly generalize to new data because it **overfitted** the training feature statistics. On the other hand, we may also obtain an **underfitted** model that achieves a high loss on both the empirical (including training data) and true data distribution by learning an oversimplified decision rule (Goodfellow et al., 2016).

2.6.1 Regularization

To avoid overfitting our training distribution, we can employ techniques known as **regularization**. Goodfellow et al. (2016) defines regularization as "any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error". In the following, we will give a non-exhaustive list of exemplary regularization techniques that are important for this thesis and refer the reader to Goodfellow et al. (2016, Chapter 7) for more details.

- **Weight Penalties:** Weight penalties are a common regularization technique in machine learning, where the loss function is modified to include a cost term based on the learnable parameters. This is expressed as $J(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta}) = \ell(f(\boldsymbol{x}; \boldsymbol{\theta}), \boldsymbol{y};) + \lambda\Omega(\boldsymbol{\theta})$, where $\Omega(\boldsymbol{\theta})$ is the cost function scaled by a hyperparameter λ. Commonly, the term is designed to discourage large or complex parameter values, effectively reducing the complexity of the learned algorithm. Choices for $\Omega(\boldsymbol{\theta})$ include the L^1-norm (*lasso*), which promotes sparsity by encouraging many weights to become zero, and the L^2-norm (*ridge*), which discourages large weights without necessarily enforcing sparsity, or a combination of both (*elastic net*). In ML lingo, the L^2-norm penalty is also often known as *weight decay*, albeit that weight decay is directly implemented in the weight update rule, which introduces issues with optimizers such as *Adam* (Loshchilov & Hutter, 2019). The weight penalty is, however, not constrained to norm penalties, for instance, we will introduce a more sophisticated weight penalty that leverages frequency-domain properties in Chapter 5.

- **Dropout:** Randomly removing units during training reduces the risk of overfitting by ensuring that the model does not become overly dependent on specific units or features (Srivastava et al., 2014). A "unit" can broadly refer to a single element in a feature map, a channel, or even the output of an entire layer. Dropout can be effectively implemented by multiplying the activations with a randomly generated mask that zeroes out certain elements with probability p. This approach effectively models a dynamic ensemble of subnetworks during training, improving generalization. Dropout often complements other regularization techniques, such as weight decay or parameter norm constraints, but has also been shown to outperform these in individual comparisons (Srivastava et al., 2014; Goodfellow et al., 2016).

- **Image Augmentation:** Scaling the number of training data (along other axes) is the best way to improve generalization (Goodfellow et al., 2016; Sutton, 2019; Branwen, 2020). However, if training data is limited, the number of de facto samples can be synthetically enlarged by various label-preserving transformations known as augmentations. These can range from simple forms like applying noise (Sietsma & Dow, 1991), crops and horizontal flips (Krizhevsky et al., 2012), mixtures of multiple transformations (Hendrycks et al., 2020), to learned transformation policies (Cubuk et al., 2019), combination of samples (Tokozume et al., 2018; Zhang et al., 2018; Yun et al., 2019), and model-based approaches (Hendrycks et al., 2021a). Incorporating a specific transformation into the data transformation process (at different intensities) trains the model to become invariant to that transformation, meaning it learns to maintain consistent predictions regardless of whether the transformation is present in the test data. For instance, if we apply a brightness augmentation during training, we expect our model to become invariant to brightness changes at test time (under reasonable conditions).

- **Early Stopping:** Under empirical risk minimization, training a neural network for more epochs generally keeps decreasing the training loss (given appropriate optimization hyperparameters). However, while the test loss often decreases initially as well, it may eventually increase if the model begins to overfit (Goodfellow et al., 2016). Early stopping mitigates overfitting by halting the training process when a predefined condition is met, such as no improvement in validation loss for a number of consecutive epochs.

- **Adversarial Training:** Training on data that is perturbed to maximize training loss (Szegedy et al., 2014b; Goodfellow et al., 2015; Madry et al., 2018), which can be seen as a worst-case data augmentation. We will offer a more nuanced introduction to this technique in Section 2.8.2.

- **Inductive Bias:** The inductive bias of each neural network architecture can also be seen as regularization, as it imposes constraints on how the data is processed. For instance, in CNNs, the locality of the convolution operation and its weight-sharing property serve as a strong regularization as compared to regular MLPs (Goodfellow et al., 2016).

2.7 Distribution Shifts

From a data perspective, the root cause of performance differences between train and test time is a **distribution shift** in feature space (Storkey, 2008; Quiñonero-Candela et al., 2008; Moreno-Torres et al., 2012). Here, we use "features" loosely to refer to any information present in the input samples. In images, this could range from low-level details like pixel color intensities and frequency distributions to higher-level attributes, or even entire objects or scene compositions.

Statistically speaking, we are modeling the conditional distribution $p(Y|Z)$ between the feature space Z and the label space Y based on the joint distribution $p_{train}(Z, Y)$ from our training set. However, our test distribution p_{test} may diverge: $p_{test}(Z, Y) \neq p_{train}(Z, Y)$. We can assume that an increased divergence between p_{train} and p_{test} decreases the model performance by common metrics. In this section, we aim to formalize the conditions that may lead to distribution shifts.

Recall that we can factorize a joint distribution p using conditional probabilities:

$$p(Z, Y) = p(Z|Y) \cdot \underbrace{p(Y)}_{\text{label shift}} = \overbrace{p(Y|Z)}^{\text{concept shift}} \cdot \underbrace{p(Z).}_{\text{covariate shift}} \tag{2.32}$$

Based on Equation 2.32, we can now identify the shift that each term is introducing. Note that, in theory, multiple shifts could co-occur. However, for simplicity, we consider mutually exclusive shifts.

A **covariate shift** (Shimodaira, 2000; Gretton et al., 2009) occurs when the marginal distribution of the input features, Z, changes between the training and testing time, such that $p_{\text{test}}(Z) \neq p_{\text{train}}(Z)$, while the conditional distribution of the target variable given the features remains constant, *i.e.*, $p_{\text{test}}(Y|Z) = p_{\text{train}}(Y|Z)$. This situation assumes that Z is causally related to Y.

In simpler terms, covariate shift implies that while the feature distribution differs in the test set, the relationship between features and outputs is preserved. For example, consider training a classifier to distinguish soda bottles from water bottles. If the training data contains only Coke bottles in the soda class, the model may fail to recognize a Fanta bottle during testing. Although the relationship between features and labels remains the same, the feature distribution shifts because Coke-specific features are missing in the test set, replaced by new Fanta-specific features.

Covariate shift is one of the most extensively studied forms of distributional shift in machine learning and is the central focus of this thesis.

A **label shift** (Zhang et al., 2013; Lipton et al., 2018) refers to a change in the marginal distribution of the labels Y, between the training and testing time, while the conditional distribution of features given the label remains constant. Formally, this implies that $p_{\text{test}}(Y) \neq p_{\text{train}}(Y)$, but $p_{\text{test}}(Z|Y) = p_{\text{train}}(Z|Y)$. Label shift assumes that the labels influence the features, making it the inverse of covariate shift.

To illustrate this, consider the soda versus water classification task from above. Imagine we train the classifier using a dataset where soda bottles are much more common than water bottles, leading the model to assign a high prior probability to the soda label. However, in the test setting, the distribution changes, and water bottles become the majority. Although the visual features distinguishing sodas from water remain the same, the model may struggle to adapt to this new label distribution, resulting in a tendency to over-predict soda due to the training imbalance.

A **concept shift** (Widmer & Kubat, 1996; Tsymbal, 2004) occurs when the conditional distribution of the label given the features, $p(Y|Z)$, changes between the training and testing time, while the feature distributions do not change, *i.e.*, $p_{\text{test}}(Y|Z) \neq p_{\text{train}}(Y|Z)$ but $p_{\text{test}}(Z) = p_{\text{train}}(Z)$. Unlike covariate shift or label shift, concept shift implies that even if the input features are distributed similarly, they now correspond to different labels or interpretations. This scenario often arises when the criteria or meaning of a label changes over time or across different contexts. To remain with the soda analogy, we can observe a change in the label if we survey the term used to describe a soda bottle across different states of the USA (Zhang et al., 2021a).[4] Depending on the region, the "true" label may be pop, soda, coke, or something else,[5] while the features remain exactly the same.

In-distribution vs. Out-of-distribution Samples

In the context of machine learning, the terms in-distribution (ID) and out-of-distribution (OOD) are frequently used but can carry different meanings, depending on the context. Out-of-distribution may refer to a range of scenarios, from small covariate shifts to cases where samples belong to new labels that were not observed during training.

[4] https://popvssoda.com/

[5] We assume that all these labels are part of Y.

To understand this ambiguity, consider a covariate shift, where $p_{\text{test}}(Z) \neq p_{\text{train}}(Z)$. This can happen for three main reasons:

1. Test features lie in the same manifold as train features but are differently distributed.
2. Test and train feature manifolds partially overlap (potentially on long tails), with potentially different distributions within the overlapped regions.
3. Test and train features span disjoint manifolds.

Based on a literal interpretation, we might consider only scenarios (2) and (3) as OOD, since they involve features that either partially or fully fall outside the training distribution. However, what qualifies as OOD data ultimately depends on how we define our reference distribution. As a concrete example, let us focus on *common corruptions (ImageNet-C)* (Hendrycks & Dietterich, 2019), which includes synthetically corrupted versions of the *ImageNet* validation set. These corrupted samples could be viewed as ID since they have been derived from the same *ImageNet* images, but also as OOD because the specific corruptions may not appear – or may appear in different intensities – in the training set. ID is equally difficult to define and to guarantee. Even if we take a balanced split of a dataset into a train and test set, we cannot always guarantee the absence of some level of covariate shift.

In the scope of this thesis, we will refer to covariate-shifted data as OOD data, but the test samples are never novel in the sense that no new labels are introduced in the test data. We will only call test data ID if it is a split coming from the same source dataset (*e.g.*, *ImageNet* train and validation set (Russakovsky et al., 2015)) or was sampled in an (almost) identical way (*e.g.*, *ImageNet* train and *ImageNet-v2* (Recht et al., 2019)) accepting only small levels of covariate shift.

2.8 Robustness

We define robustness as invariance to mild distribution shifts (maintaining the previous notion of out-of-distribution data), particularly covariate shifts. We consider robustness a necessary but insufficient attribute for generalization on new samples from the true data distributions: a model that generalizes optimally will always be robust, but a robust model may not necessarily generalize optimally. Also, in the following chapters, we will sometimes explore specific kinds of robustness, such as robustness to image corruptions (Section 2.8.1) or adversarial attacks (Section 2.8.2), which may be orthogonal to each other.

2.8.1 Corruption Robustness

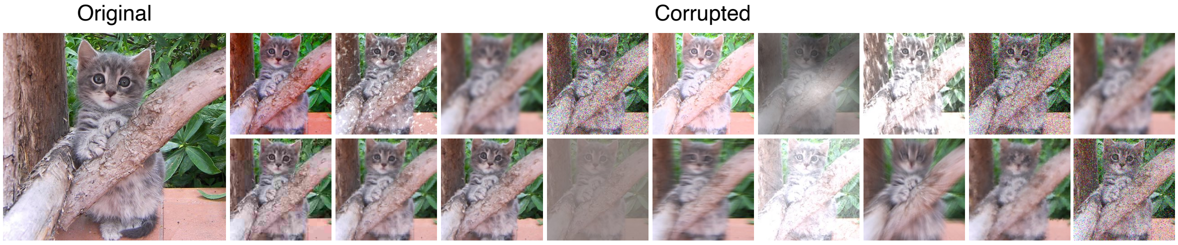


Figure 2.9: Exemplary corruptions to an image. We can expect that if a robust model correctly recognizes the original image on the left, it should also generalize and recognize at least some of the corrupted images on the right side. **Image Source:** (Deng et al., 2009; Hendrycks & Dietterich, 2019)

A basic notion of robustness to covariate shift assumes that a classifier performing well on an i.i.d. test set will maintain similar performance even when that set is *corrupted* (see Figure 2.9 for an example). Various corruption methods have been explored, including random synthetic image transformations (Geirhos et al., 2018, 2019; Hendrycks & Dietterich, 2019; Mintun et al., 2021; Kar et al., 2022), targeted modifications of object attributes (Li et al., 2023c; Zhang et al., 2024), and using real photographs with quality issues (Hendrycks et al., 2021a; Bafghi & Gurari, 2023).

Notably, most classifiers often struggle with corrupted data, even after achieving high accuracy on clean i.i.d. data – a stark contrast to human performance, which remains largely unaffected by these corruptions (Geirhos et al., 2018). While training on specific corruptions (*e.g.*, through data augmentation) can improve performance on test data with similar corruptions, sometimes even surpassing human performance, the resulting models typically fail to generalize to unseen corruptions (Geirhos et al., 2018) or generalize only when the test corruptions are perceptually similar to those used during training (Yin et al., 2019; Mintun et al., 2021).

Defenses

Generally, defenses can be categorized into *domain adaptation* (see Wang & Deng (2018) for a survey) or *domain generalization* (see Zhou et al. (2023) for a survey) approaches.

Domain adaptation aims to align the model to features of the target domain, often requiring access to (supervised or unsupervised) samples from the test domain, potentially at the expense of performance on the source domain (Ganin & Lempitsky, 2015; Sun & Saenko, 2016; Tzeng et al., 2017; Hoffman et al., 2018). Test-time training (TTT) (Sun et al., 2020) represents a promising approach within domain adaptation, adapting to the target domain using an unsupervised training signal during testing.

Conversely, in this thesis, we are interested in models that generalize without the need for adaptation. Domain generalization seeks to learn features that generalize across domains without access to target domain samples. A (narrow) domain generalization strategy for improving robustness to corruptions focuses on designing augmentation techniques that generalize to unseen corruptions during testing (Tokozume et al., 2018; Zhang et al., 2018; Yun et al., 2019; Geirhos et al., 2019; Lopes et al., 2020; Rusak et al., 2020; Hendrycks et al., 2020, 2021a, 2022; Cubuk et al., 2019, 2020; Erichson et al., 2022; Modas et al., 2022).

2.8.2 Adversarial Robustness

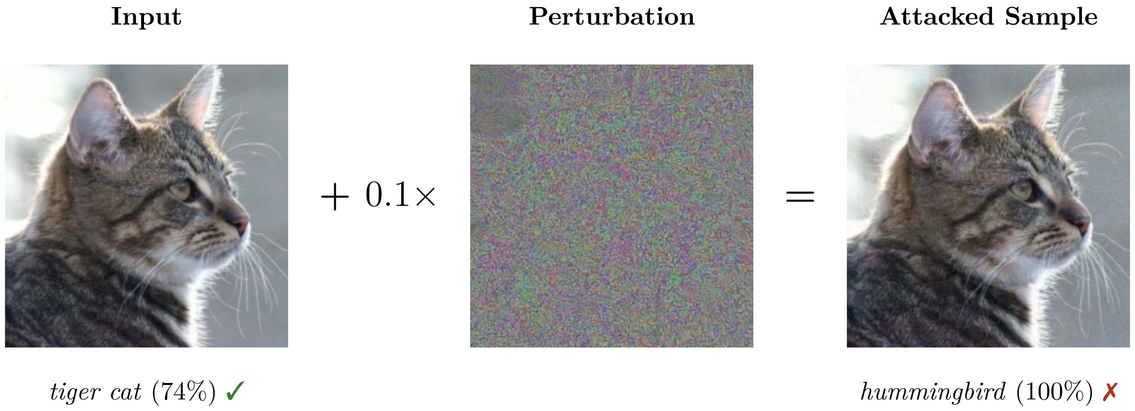


Figure 2.10: An example of an adversarial attack on image classification. A perturbation that appears semantically irrelevant to humans (center; contrast amplified by a factor of 10; generated with an L^∞-PGD-40 attack with $\epsilon = 8/255$) causes an *ImageNet*-trained `ResNet-50` model (He et al., 2016) to misclassify a photo of *tiger cat* as a *hummingbird* with maximum confidence. The figure is inspired by Fig. 1 of (Goodfellow et al., 2015). **Image Source:** (Deng et al., 2009) (cat)

Adversarial data can be seen as a special kind of covariate shift. Instead of "natural" covariate shifts, which may be caused by imbalanced representations of true features, we now actively modify the feature distribution at test time with respect to our model and objective to increase the error.

Applying this technique has demonstrated that learned decision rules by default are not even robust around in-distribution samples – predictions are highly sensitive to small input perturbations (Biggio et al., 2013; Szegedy et al., 2014b), even if they are (almost) imperceptible and semantically meaningless to humans (see Figure 2.10 for an example). In some cases, these perturbations can be as small as a single pixel (Su et al., 2019). The field of **adversarial robustness** is attempting to find (and defend) perturbations of the inputs that would fool a model. For the task of image classification, these perturbations would (but ideally should not) cause the misclassification of samples.

Adversarial Attacks

Adversarial attacks aim to modify an input sample in a way that fools the model to cause a prediction error. Often, a core assumption of these attacks is that an additive perturbation $\boldsymbol{\delta}$ to a sample $\boldsymbol{x}$ fools the attacked model without carrying a semantic meaning to a human observer. For example in classification, if we have a sample $\boldsymbol{x}^{(a)}$ belonging to a class a and a sample $\boldsymbol{x}^{(b)}$ belonging to a class b it would be not surprising that $\arg\max f\left(\boldsymbol{x}^{(a)} + \boldsymbol{\delta}\right) = b$, if $\boldsymbol{\delta} = \left(\boldsymbol{x}^{(b)} - \boldsymbol{x}^{(a)}\right)$, as it actually changes the resulting sample and its features to members of the class b. Instead, we are interested in perturbations that are constrained to not cross the boundary of the "true" class manifold. This property is commonly approximated by restricting perturbation magnitudes by some L^p-norm (Goodfellow et al., 2015).

Formally, pixel-wise L^p-attacks can be defined as follows. Given an input sample $\boldsymbol{x}$ with its corresponding class label y and a loss function ℓ, the adversarial attack seeks to maximize the loss by finding a small perturbation $\boldsymbol{\delta}$ to $\boldsymbol{x}$. This perturbation is constrained within a L^p-norm ball of radius (or "budget") ϵ, centered at $\boldsymbol{x}$. The goal is to keep perturbations small enough (minor changes) to restrict them to imperceptible alterations in input space while causing the model to misclassify the adversarial sample $\boldsymbol{x}_{\text{adv}} = \boldsymbol{x} + \boldsymbol{\delta}$. We can approximate misclassifications by an increase in loss:

$$\max_{\|\boldsymbol{\delta}\|_p \leq \epsilon} \ell\left(f\left(\boldsymbol{x} + \boldsymbol{\delta}; \boldsymbol{\theta}\right), y\right). \tag{2.33}$$

The choice of p in the L^p norm significantly impacts the spatial distribution of perturbations. L^∞-bounded attacks constrain only the maximum change to any single pixel (*e.g.*, by $\pm\epsilon$), allowing any number of pixels to be perturbed. In contrast, L^2-bounded attacks impose a trade-off: perturbing fewer pixels allows for larger changes per pixel (up to $\pm\epsilon$ if limited to a single pixel), while perturbing all n pixels limits each change to $\pm\sqrt{\epsilon^2/n}$. Due to the squared nature of the L^2 norm, these attacks model a trade-off between local exploitation (*i.e.*, stronger local perturbation) and exploration (*i.e.*, weaker but more spatially distributed perturbation). Newer works have also moved beyond L^p perturbations and extended them to functional spaces (Laidlaw & Feizi, 2019), patches (Brown et al., 2018), or other similarity metrics (Wong et al., 2019; Laidlaw et al., 2021). However, in the context of this thesis, whenever referring to adversarial robustness of any kind, we refer to L^p-robustness unless specified differently.

Generally, adversarial attacks can be found in both white- and black-box settings (Liu et al., 2017; Ilyas et al., 2018; Bhagoji et al., 2018; Andriushchenko et al., 2020) where the adversary has or hasn't full access to the model, respectively. Amongst the most effective attacks are white-box attacks that utilize the gradient of the prediction to perturb images in the direction of the highest loss (Goodfellow et al., 2015; Madry et al., 2018; Croce & Hein, 2020b,a). Some attacks may be *targeted*, *i.e.*, attempting to flip the prediction towards a specific label, as opposed to *untargeted* attacks, which can be seen as a relaxed constraint that causes any misclassification. In the following, we will consider untargeted cases unless stated otherwise.

A conceptually simple untargeted white-box attack is the *fast gradient sign method (FGSM)* (Goodfellow et al., 2015). Given a sample $\boldsymbol{x}$ and attack radius ϵ under L^∞-norm, it perturbs the input sample by a single step in the direction of the gradient signs, thus maximizing the error.

$$\boldsymbol{x}_{\text{adv}} = \boldsymbol{x} + \underbrace{\epsilon \cdot \text{sign}(\nabla_{\boldsymbol{x}} \ell\left(f\left(\boldsymbol{x}; \boldsymbol{\theta}\right), y\right))}_{=\boldsymbol{\delta}} \tag{2.34}$$

A limitation of FGSM is its single-iteration approach, which may fail to identify an adversarial region of the decision space. The attack success rate can be increased by using an iterative method, such as *basic iterative method (BIM)* (Kurakin et al., 2017). As an additional benefit, this method does not utilize the full perturbation budget by selecting perturbations at the maximum allowable boundary. BIM takes t FGSM steps with radius α, which are clipped back to the radius should they exhaust the budget.

$$\begin{aligned} \boldsymbol{x}_{\text{adv}}^{(1)} &= \boldsymbol{x} \\ \boldsymbol{\delta}^{(t)} &= \left(\boldsymbol{x}_{\text{adv}}^{(t)} + \alpha \cdot \operatorname{sign}\left(\nabla_{\boldsymbol{x}} \ell \left(f\left(\boldsymbol{x}_{\text{adv}}^{(t)}; \boldsymbol{\theta}\right), y\right)\right)\right) - \boldsymbol{x} \\ \boldsymbol{x}_{\text{adv}}^{(t+1)} &= \boldsymbol{x} + \operatorname{clip}\left(\boldsymbol{\delta}^{(t)}, -\epsilon, \epsilon\right) \end{aligned} \tag{2.35}$$

A common modification of this, known as *projected gradient descent (PGD)*, perturbs the initial sample randomly (Madry et al., 2018):

$$\boldsymbol{x}_{\text{adv}}^{(1)} = \boldsymbol{x} + \boldsymbol{p}, \quad \text{where } \boldsymbol{p} \sim \mathcal{U}(-\epsilon, \epsilon) \tag{2.36}$$

While BIM/PGD may not fully utilize the available perturbation budget, other methods more effectively optimize perturbation magnitude. These include approaches that relax constraints such as the requirement to maximum confidence (Carlini & Wagner, 2016), or boundary attacks (Brendel et al., 2019; Croce & Hein, 2020a).

Whereas adversarial examples are often crafted against a specific model, some adversarial samples can be universal and transferred to other deep learning and classical machine learning architectures alike (Papernot et al., 2016).

Measuring Adversarial Robustness in Image Classification

A commonly reported metric for classification robustness is robust accuracy on some dataset $\mathbb{X}$, *i.e.*, the top-1 classification accuracy under a specified attack (with a maximum budget). Formally, we can define this as

$$\frac{1}{|\mathbb{X}|} \sum_{(\boldsymbol{x}, y) \in \mathbb{X}} \mathbb{I}\left(\arg\max f\left(\sigma\left(\boldsymbol{x}, f\right)\right) = y\right), \tag{2.37}$$

where σ is the function generating the adversarial attack. This will be our quantification of robustness in this thesis.

For completeness, we want to mention a few alternative ways to measure performance under adversarial attacks. For instance, it is possible to measure the robust accuracy against label flips instead of the true label by substituting $y = f(\boldsymbol{x})$ in the indicator function. Furthermore, some works are interested in quantifying the average minimum budget necessary to cause a misclassification, *i.e.*:

$$\frac{1}{|\mathbb{X}|} \sum_{(\boldsymbol{x}, y) \in \mathbb{X}} \min \left\{ \|\boldsymbol{\delta}\|_p \mid \arg\max f(\boldsymbol{x} + \boldsymbol{\delta}) \neq y \right\}. \tag{2.38}$$

Independent of the performance metric, it is necessary to discuss how adversarial robustness should be evaluated. For instance, each attack may leave individual fingerprints that a neural network (defense) can overfit. Thus, a faithful evaluation of adversarial robustness often employs an ensemble of attacks (Carlini et al., 2019), *e.g.*, through *AutoAttack* (Croce & Hein, 2020b). The ensemble applies *APGD-CE*, *APGD-T* (Madry et al., 2018; Croce & Hein, 2020b), *FAB* (Croce & Hein, 2020a), and *Square* (Andriushchenko et al., 2020) attacks to obtain a comparable (because parameter-free) robustness accuracy. *RobustBench* (Croce et al., 2021) provides a benchmark for adversarial defense methods using *AutoAttack* for evaluation and maintains a leaderboard of best-performing models. Robustness is evaluated by $p = 2, \epsilon = 0.5$ on *CIFAR-10*, as well as $p = \infty, \epsilon = 8/255$ on *CIFAR-10/100*, and $p = \infty, \epsilon = 4/255$ on *ImageNet-1k*, respectively. However, these established ϵ-thresholds were disputed as being too large and generating perturbations that can easily be detected (Harder et al., 2021). When evaluating the robustness of adversarially regularized models, we will nonetheless utilize these thresholds to remain compatible with previous works. For the evaluation of normally trained models, we will utilize significantly lower values that we will discuss in the individual chapters.

Adversarial Defenses

Models trained without adversarial defenses can typically not withstand attacks with high ϵ budgets like the ones in *RobustBench* and will collapse below random accuracy. A straight-forward defense to adversarial attacks is *adversarial training (AT)* (Madry et al., 2018), which trains the model on perturbed samples during training and effectively turns out-of-distribution attacks into in-distribution samples. It can be described by the following min-max objective:

$$\min_{\boldsymbol{\theta}} \frac{1}{|\mathbb{X}|} \sum_{(\boldsymbol{x},y)\in\mathbb{X}} \max_{\|\boldsymbol{\delta}\|_p \leq \epsilon} \ell\left(f\left(\boldsymbol{x}+\boldsymbol{\delta};\boldsymbol{\theta}\right),y\right). \tag{2.39}$$

Common choices for training attacks are FGSM or PGD. Unfortunately, adversarial training is also susceptible to overfitting to the attacks employed during its training phase, especially under single-step attacks like FGSM (Wong et al., 2020; Rice et al., 2020; Kim et al., 2021). The most effective method to mitigate this is, surprisingly, also the simplest, namely *early stopping* on the validation data (Zhang et al., 2020). It is worth noting that even with early stopping, adversarial training is very likely to overfit to adversarial robustness, at a cost in clean accuracy (Tsipras et al., 2019) and robustness to other forms of covariate shift (Moayeri et al., 2022; Kireev et al., 2022; Alhamoud et al., 2023).

AT also leads to a significant increase in training cost due to the additional gradient passes necessary to compute perturbations and, thus, alternatives like adversarial purification (Shi et al., 2021; Yoon et al., 2021; Nie et al., 2022; Carlini et al., 2023), randomized smoothing (Cohen et al., 2019), or inductive priors (Wang et al., 2020a; Lukasik et al., 2023) have been proposed. However, most alternatives have been identified as either conceptually invalid (Uesato et al., 2018; Carlini & Wagner, 2017; Athalye et al., 2018), introduce other latency trade-offs (Cohen et al., 2019), or do not match the performance of AT. Thus, AT remains (one of) the state-of-the-art approach for adversarial defense, especially when combined with external (synthetic) data (Rebuffi et al., 2021a; Gowal et al., 2021; Wang et al., 2023d) and/or more sophisticated loss functions (Carlini & Wagner, 2016; Pang et al., 2020a; Cui et al., 2023).

In some scenarios, it may also be sufficient to simply detect and reject adversarial samples instead of strengthening the neural decision rule (*e.g.*, (Feinman et al., 2017; Lee et al., 2018; Ma et al., 2018; Harder et al., 2021)).

2.9 Feature Biases in Visual Perception

In previous sections, we primarily discussed generalization through the lens of data, emphasizing distribution shifts between train and test time. In this section, we shift our focus to the model's perspective to analyze how it may be biased against certain features (Section 2.9.1), then link this back to generalization and robustness (Section 2.9.2), provide methods for the identification of biases (Section 2.9.3), and finally provide some recent examples (Section 2.9.4).

2.9.1 Definition

Consider each input sample as a set of n features represented by $\boldsymbol{x} = \{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(n)}\}$, where we again define features in an abstract sense. We can hypothesize that a model assigns a weight of importance to each feature during inference. These will likely be of varying levels – some features could be highly influential to the decision, receiving significant weight, while others might have minimal or no impact, with small or zero weights. In the latter case, we can say that a model is invariant to the respective feature – a property that can be desirable, for instance, a cat is a cat independent of the background behind it, it's position in an image, or whether it is a close-up shot or not. This differential assignment of importance leads to an implicit ranking of features. It is noteworthy that feature importance can vary from one sample to another, even for samples belonging to the same class.

We define a **bias** as one or more features that are prioritized relative to others during a prediction. Our definition of biases does not depend on whether the preferred features are genuinely correlated with the label. Rather, it only focuses on the relative importance assigned by the model to each feature – thus, biases could be desirable or destructive. For instance, if we aim to detect objects as in *ImageNet* (Deng et al., 2009), a model that prefers the foreground features over the background has a desirable bias, while the opposite bias would be destructive, as it is not correlated with the task.

We offer the following taxonomy of biases depending on their severity:

- **Sample-specific bias**: The feature importance distribution for a specific sample is biased. Adversarial examples could be seen as highly biased examples, as they modify a sample to increase the perceived importance of superficial features to cause prediction errors (Ilyas et al., 2019).
- **Group-specific bias**: The feature importance distribution for multiple samples (*e.g.*, all samples from one class) is biased. An example of a group-specific bias can be seen when, for instance, an *ImageNet* classifier relies on a watermark to make predictions for specific classes (Li et al., 2023d), with the watermark functioning as the biased feature.
- **Global biases**: The feature importance distribution for all samples is biased. For instance, CNNs trained on *ImageNet* are biased towards texture rather than shape information across all decisions, whereas humans strongly prefer shape (Geirhos et al., 2019) – with texture serving here as the biased feature.

2.9.2 Connection to Generalization

Ultimately, we can link our notion of biases to the model's generalization capability. For example, consider a taxonomy of feature types that a model could potentially learn, as outlined by Geirhos et al. (2020a):

- **Uninformative features**: These show no meaningful correlation with the true data distribution and align only at chance levels.
- **Overfitting features**: These features correlate with the training set labels but fail to generalize to new data from the true distribution.
- **Shortcut features**: These correlate with labels in the i.i.d. training and test sets but fail under distribution shifts (*e.g.*, covariate shifts) or on new samples from the true data distribution.
- **Intended features**: These features maintain consistent correlations with labels across all samples from the true data distribution.

Over a set of multiple samples, uninformative features will typically have zero importance. Overfitting and shortcut features may have non-zero importance on specific empirical datasets, but do not extend beyond them. Intended features, by contrast, have non-zero importance throughout the true data distribution.

A model biased towards unintended, overfitting, or shortcut features is likely to exhibit poor generalization. Conversely, a model that is biased against intended features will generalize more effectively. Identifying the optimal bias is often impossible in practice, and we may not even be able to rule out the existence of multiple equally effective importance distributions. Typically, human evaluators can only identify unwanted biases. For example, biases in *ImageNet* models towards watermarks or backgrounds to detect (foreground) objects are clearly problematic, but defining an optimal set of biases to detect an object is less obvious.

Instead of establishing optimal biases, a fruitful approach can be to measure and compare biases for different observers – for instance, one line of research explores comparing biases between artificial neural networks and human vision, with human perception serving as a proxy for optimal bias (Geirhos et al., 2019; Subramanian et al., 2023). In Part II, we, too, will measure and discuss biases in comparison to human evaluators.

2.9.3 Methods for the Identification of Features/Biases

Now that we have established what a bias is, we want to outline methodologies to identify and quantify bias in neural networks.

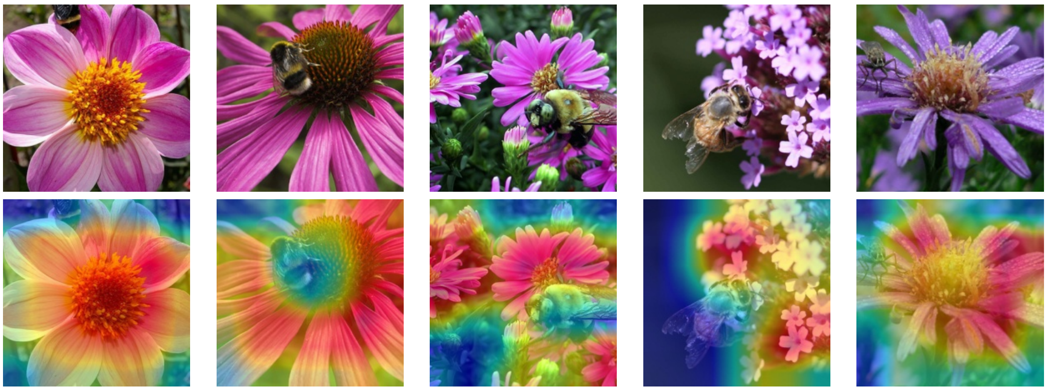


Figure 2.11: Attributions through saliency maps. We show *ImageNet* samples for the class *bee* and the respective saliency maps (blue to red in increasing relevancy), which reveal the model's bias to classify based on the flower, and not the bee itself. **Image Source:** (Deng et al., 2009; Singla & Feizi, 2022)

Feature Attributions

Attributions maps attempt to highlight regions in input images that are relevant for a prediction, for example through gradient propagation (Erhan et al., 2009; Baehrens et al., 2010; Simonyan et al., 2014; Shrikumar et al., 2017) or attention (Serrano & Smith, 2019; Jain & Wallace, 2019; Wiegreffe & Pinter, 2019; Abnar & Zuidema, 2020) (one example is shown in Figure 2.11). In most cases, they will provide a pixel or patch-level importance for a single prediction, where we consider (groups of) pixels/patches to be the features. While attributions primarily highlight regions, they could, in principle, be utilized to detect bias. For example, we could instruct a human or artificial observer to formulate a bias from attributed regions over multiple predictions.

However, the success of this method is limited by the following components: (1) the attribution method must generate a faithful attribution, (2) the observers (which may be humans) must be able to identify the abstract feature from attributed pixels, and (3) the observers must be able to extrapolate a decision rule based over multiple attributed samples.

Unfortunately, it has been shown that many attribution methods are not trustworthy and fail basic sanity checks (Adebayo et al., 2018; Jain & Wallace, 2019), and likewise do the metrics that attempt to establish their trustworthiness (Tomsett et al., 2020). However, even if they were trustworthy, human judgment varies a lot and may not be sufficient to identify a bias from attributions alone. While we do not dispute that attributions can be useful, and indeed have been used to identify class-specific shortcuts (*e.g.*, (Zech et al., 2018; Singla & Feizi, 2022; Neuhaus et al., 2023)), it seems very difficult and impractical to discover global biases only based on attribution methods.

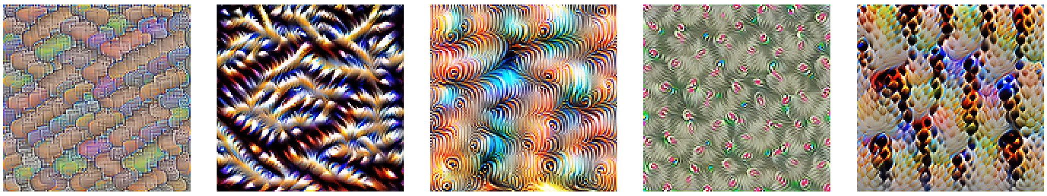


Figure 2.12: Feature visualization. We show 5 neurons of the first residual block in stage 3 of an *ImageNet*-trained `ResNet-50`.

Feature Visualization

The field of mechanistic interpretability (MI) aims to identify and interpret circuits (subgraphs in neural networks) that connect inputs to outputs. In principle, this vision aligns well with our established goal to identify global biases.

For image inputs, a common MI method used to “understand” a function of a neuron is feature visualization – here, inputs are optimized to maximize the output of individual neurons (with some similarity to adversarial attacks) (Nguyen et al., 2016; Mordvintsev et al., 2015; Olah et al., 2017) (Figure 2.12).

However, being gradient-based, these methods suffer from limitations like sensitivity to initialization and local optima. They assume that neurons are mono-semantic (representing one concept) but are often poly-semantic (representing multiple concepts), making interpretation difficult (Olah et al., 2020b; Elhage et al., 2022). Finally, regularization and preconditioning, while improving visual interpretability, introduce biases that can misrepresent the neuron’s true function, potentially leading to misleading conclusions despite looking visually appealing (Olah et al., 2017; Zimmermann et al., 2021; Geirhos et al., 2024).

In summary, while feature visualization remains a valuable tool for exploring neural network representations and has been used to detect biases such as the bias for written text (Goh et al., 2021), its inherent limitations must be carefully considered. Further, the interpretation of visualizations is subject to the same issues that we discussed for attribution methods.

Behavioral Testing

Rather than dissecting the intricacies of internal features, we can adopt a behavioral testing paradigm, treating the network as a “black box” and probing its responses to carefully designed **stimuli**. This approach offers several advantages. It provides a more precise methodology, as specific features within the input stimuli can be explicitly manipulated to directly assess the model’s sensitivity to those features. Furthermore, behavioral testing facilitates easier comparison with humans, as the same stimuli can be presented to human participants, enabling a more intuitive understanding of the model’s learned behavior relative to human perception. The downside is that the feature(s) must be predetermined.

In the following, we will explore two distinct methodologies for behavioral testing: feature-ablated stimuli and hybrid feature stimuli.

Feature ablations We can understand biases as the strong preference for one or multiple features – *i.e.*, if these features would no longer be present in the input, we would expect the performance to be impacted strongly.

Assuming that we have an understanding of what features exist in our dataset, we can create a modified test dataset that ablates a feature: $\mathbb{X}_{\boldsymbol{z}-\text{ablation}} = \{(\boldsymbol{x}\backslash\boldsymbol{z}, y) \mid (\boldsymbol{x}, y) \in \mathbb{D}_{\text{test}}\}$, where we assume that $\boldsymbol{z}$ is the same feature across all samples (*e.g.*, the lowest frequencies as shown in Figure 2.13). Then we

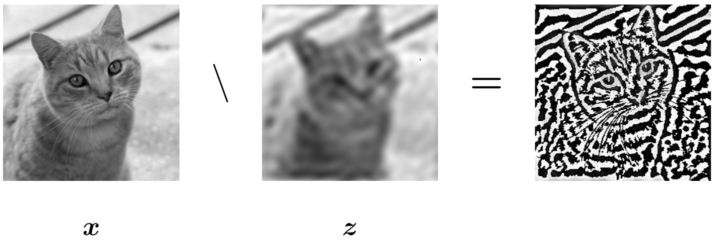


Figure 2.13: Feature ablation. A sample is constructed by removing one or more features. Here, we remove ca. 11% of the low-frequency components (feature $\boldsymbol{z}$) in Fourier space Section 2.1 from a cat image. **Image Source:** Fir0002/Flagstaffotos (cat)

can simply measure the drop in accuracy (or alternatively loss):

$$\mathrm{bias}(\boldsymbol{z}) \approx \mathrm{accuracy}(f, \mathbb{D}_{\mathrm{test}}) - \mathrm{accuracy}(f, \mathbb{D}_{\boldsymbol{z}-\mathrm{ablation}}) \tag{2.40}$$

The higher the drop in accuracy, the higher the importance or bias of the tested feature must be – thus, unlike the previous methods, we can also quantify the importance of the feature.

However, one needs to use caution when performing this type of ablation, as removing features from an image may not always be easy and introduce artifacts (Tomsett et al., 2020). For example, let us consider that we want to test the bias towards background cues of an object classifier. If we crop the background of a photo to test the sensitivity to background cues, we need to consider how we crop and replace it carefully. A coarse crop may destroy foreground features (*e.g.*, hair) or retain background artifacts. Similarly, filling a background with white color for a dark object will introduce stronger edges, which may introduce a confounder to our study that prevents us from systematically isolating the effect of the background.

In this thesis, we will only use feature ablations to test for bias in frequency space, largely avoiding dealing with artifacts. In Chapter 8, we assess bias by masking specific frequency ranges within images. To examine the impact of the *critical band*, we utilize the noise-injection methodology proposed by Subramanian et al. (2023). This approach offers a significant advantage by enabling gradual and controlled feature removal, unlike binary masking.

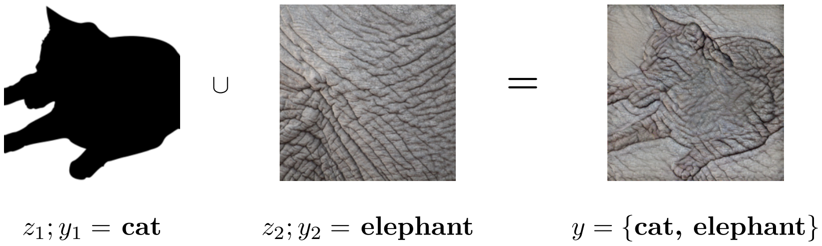


Figure 2.14: Cue-conflicted image. A sample is constructed by two or more features belonging to conflicting classes. Here we show the 2-way texture/shape conflict, where the shape feature z_1 belongs to the label "cat", and the texture feature z_2 belongs to the label "elephant"(Geirhos et al., 2019). **Image Source:** (Deng et al., 2009; Geirhos et al., 2019)

Cue-conflicted images A different kind of ablation is performed through **cue-conflicted images** (as inspired by Oliva et al. (2006)'s Hybrid Images), where we ablate and replace one or more features with equivalent features obtained from a sample of a different *conflicting* class (Figure 2.14). For instance, Geirhos et al. (2019) replaced texture information in *ImageNet* samples with conflicting (*i.e.*, belonging to another class) textures to measure the texture/shape bias, where they assumed that the remainder of the image defines the shape. Effectively, this methodology introduces conflicting feature cues in an image, and the stimulus is now correlated with multiple targets.

Formally, we define a set of stimuli (test dataset) as a $(m+1)$-tuple containing a sample, and m conflicting labels $\mathbb{X}_{\text{stimuli}} = \{(\boldsymbol{x}^{(1)}, y_1^{(1)}, ..., y_m^{(1)}), ..., (\boldsymbol{x}^{(n)}, y_1^{(n)}, ..., y_m^{(n)})\}$. A stimulus $\boldsymbol{x}$ is then created by $\boldsymbol{x} = \mathbb{B} \cup (\bigcup_{i=1}^{m} \boldsymbol{z}_i^{(i))})$ where $\boldsymbol{z}_i^i$ is the i-th feature obtained from a sample with the label y_i, such that $\forall_{(i,j);i \neq j} y_i \neq y_j$. $\mathbb{B}$ is the unmodified set of features.

From an information perspective, the classification of all labels is valid, and applying an unbiased classifier on these stimuli should generate equivalent confidence in any of the valid classes – however, internal biases in the learned representation will often result in a prediction based on one particular feature. We can quantify the relative importance or bias for $\boldsymbol{z}^{(i)}$ by the frequency of predictions of y_i, of all correct predictions:

$$\text{bias}\left(\boldsymbol{z}^{(i)}\right) = \frac{\text{accuracy against } i\text{-th label}}{\text{accuracy against all labels}} = \frac{\sum\limits_{(\boldsymbol{x}, y_1, \ldots, y_m) \in \mathbb{X}_{\text{stimuli}}} \mathbb{I}(\arg\max f(\boldsymbol{x}) = y_i)}{\sum\limits_{(\boldsymbol{x}, y_1, \ldots, y_m) \in \mathbb{X}_{\text{stimuli}}} \mathbb{I}(\arg\max f(\boldsymbol{x}) \in \{y_1, ..., y_m\})}. \tag{2.41}$$

In this thesis, we will exclusively look at cases with $m = 2$ conflicting features and $\mathbb{B} = \varnothing$. Specifically, we will use the cue-conflicted images to test for the texture/shape bias following Geirhos et al. (2019) and introduce new cue-conflicted stimuli to test for low-/high-frequency bias in Chapter 9.

2.9.4 Examples of Biases in Object Recognition Models

This thesis primarily investigates global biases within object recognition, specifically focusing on image classification (Section 2.5) of objects in natural scene contexts. Only a few of these biases are currently known, especially in the context of large-scale object detection problems like *ImageNet* (see Section 2.5.4 for details).

For instance, Geirhos et al. (2019) demonstrated that convolutional neural networks (CNNs) trained on *ImageNet* are highly biased toward textures as opposed to shape cues. This bias means that, for example, an image of a cat with elephant-like skin texture will likely be misclassified as an elephant. Rosenfeld et al. (2018); Xiao et al. (2021a) found that many classifiers also often rely excessively on background information to identify foreground objects. This reliance can lead to misclassification when objects are placed on unusual backgrounds, such as birds positioned on water, as noted by Sagawa et al. (2020). Furthermore, Wang et al. (2020a) revealed that image classifiers, particularly those trained on low-resolution datasets, almost exclusively utilize high-frequency cues, even though the information contained there should intuitively be irrelevant for classification. Relatedly, Subramanian et al. (2023) analyzed the spectral bands that are utilized in object recognition. They showed that artificial neural networks utilize significantly wider spectral bands than humans and thus rely on additional features.

In the context of multi-modal models, Goh et al. (2021) studied the learned features of CLIP models and observed a bias toward textual cues within images. For instance, an apple with a piece of paper reading “iPod” stuck to it would likely be classified as an iPod rather than an apple.

Part I
The CNN Weight Space

In this first part of the thesis, we want to focus on generalization in the learned parameters (*weight space*), specifically in the convolution filters of Convolutional Neural Networks (CNNs) that we introduced in Section 2.3. Before we dive into generalization (Section 2.6), we will define how to characterize and compare CNN filter representations and obtain a general intuition about distribution shifts in weight space through a large-scale study on a heterogeneous CNN model zoo (Chapter 3).

Turning to generalization, we look at a highly successful, yet specialized form of generalization to adversarial attacks (Section 2.8.2): adversarial training (Madry et al., 2018) – a form of regularization based on worst-case data augmentation against explicitly defined threat models. This computationally highly expensive regularization results in implicit changes to the learned representations in weight space, which we try to understand (Chapters 4 to 6), and turn into a computationally more reasonable and explicit form of weight regularization beyond adversarial robustness (Chapter 5).

Finally, we propose a different angle on the weight space beyond filter patterns through layer criticality in Chapter 6.

Chapter 3

Convolution Filter Analysis

Currently, many theoretical and practically relevant questions regarding the transferability of learned representations in Convolutional Neural Networks (CNNs) remain unsolved. While ongoing research efforts are engaging these problems from various angles, in most computer vision-related cases, these approaches can be generalized to investigations of the effects of distribution shifts in image data (see Sections 2.7 and 2.8).

In this thesis, we propose to study the shifts in the learned weights of trained CNN models. Here, we specifically focus on the properties of the distributions of dominantly used 3 × 3 convolution filter kernels. We collect and provide a dataset with over 1.4 billion filter kernels from approximately 650 CNNs, trained with variations in datasets, architectures, and vision tasks. We then propose a methodology to study filter representations and obtain insights about distribution shifts in CNNs on both global (populations of model groups) and local scales (individual model comparisons).

This chapter is based on two publications. The main content is based on **"CNN Filter DB: An Empirical Investigation of Trained Convolutional Filters"**, presented at CVPR, 2022 (Gavrikov & Keuper, 2022b) and has received an Oral award (top 4% of submissions). The study on medical imaging models is based on **"Does Medical Imaging learn different Convolution Filters?"**, presented at the NeurIPS Workshop on Medical Imaging, 2022 (Gavrikov & Keuper, 2022c). As the first author of both papers, Paul Gavrikov collected models, performed the experiments, and created the plots. The analyses were written under supervision and with the input of Janis Keuper.

Code: `https://github.com/paulgavrikov/CNN-Filter-DB`

3.1 Introduction

Despite their overwhelming success in the application to various vision tasks, the practical deployment of convolutional neural networks (CNNs) (see Section 2.3 for an introduction) is still suffering from several inherent drawbacks. Two prominent examples are (1) the dependence on very large amounts of annotated training data (Sun et al., 2017), which is not available for all target domains and is expensive to obtain, and (2) still widely unsolved problems with robustness and generalization abilities of CNNs (Akhtar & Mian, 2018) towards shifts of the input data distributions (see Sections 2.6 to 2.8). One can argue that both problems are strongly related since a common practical solution to (1) is the finetuning (Pan & Yang, 2010; Yosinski et al., 2014; Razavian et al., 2014; Oquab et al., 2014) of pre-trained models by small datasets from the actual target domain. This results in the challenge of finding suitable pre-trained models based on data distributions that are "as close as possible" to the target distributions. Hence, both cases (1+2) imply the need to model and observe distribution shifts in the contexts of CNNs.

In this thesis, we propose not to investigate these shifts in the input (image) domain but rather in

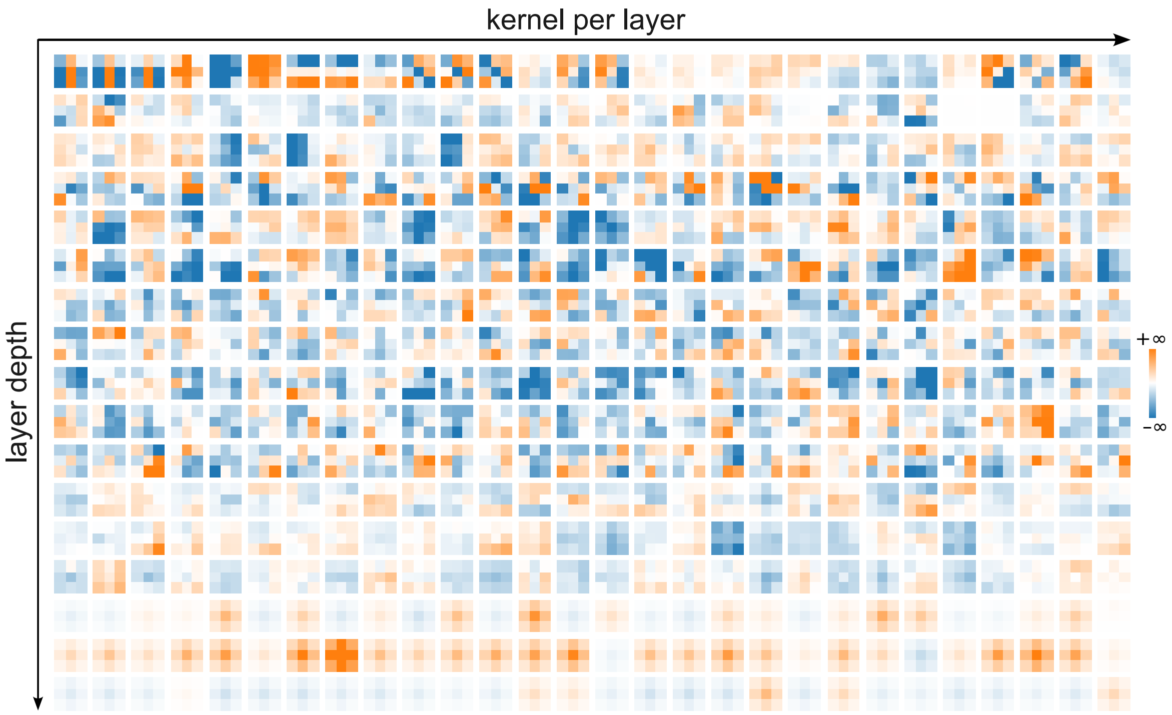


Figure 3.1: Convolution filter kernels. We show the first 28 3 × 3 filter kernels extracted of each convolution layer in a `ResNet-18` trained on *CIFAR-10*. The filters show a clear loss of pattern diversity and increasing sparsity with depth. The color range is normalized by the largest weight in *each* layer to emphasize relative magnitude.

the 2D filter kernel distributions of the CNNs themselves. We argue that the distributions of trained convolutional filters in a CNN, which implicitly reflect the sub-distributions of the input image data, are more suitable and easily accessible representations for this task. This is further backed by previous work that showed a correlation between weight space structure in CNNs and generalization (Unterthiner et al., 2021).

In order to foster systematic investigations of learned filters, we collect and now publicly provide a dataset of over 1.4 billion filters with metadata from hundreds of trained CNNs, using a wide range of datasets, architectures, and vision tasks.

On this data source, we conduct an analysis and report a series of novel insights into representations of widely used CNN models. Based on the methods introduced in this chapter, we show that many publicly provided models suffer from degeneration – *i.e.*, some of their filters or entire layers do not model "useful" transformations. For instance, we show that overparameterization leads to sparse and/or non-diverse filters in normal models (as can be seen in Figure 3.1), while robust training increases filter diversity and reduces sparsity. Our results also show that learned filters do not significantly differ across models trained for various tasks, except for extreme outliers such as GAN-Discriminators. Models trained on datasets of different visual categories do not significantly shift either. We find that most shifts in the studied models are due to degeneration rather than an actual difference in structure. Therefore, our results suggest that pre-training can be performed independently of the actual target data, and only the amount of training data and its diversity matter. This is in line with recent findings that models can be pre-trained even with non-natural images (Kataoka et al., 2020). For classification models, we show that the most variance in learned filters is fairly spread out throughout model depth. In contrast, object/face detection models only show significant variance in the early layers. At least from a weight perspective, this overhauls the common notion that the most specialized filters are always found in the last layers.

We summarize our contributions as follows:

- We publish a diverse database of over 1.4B 3 × 3 convolution filters alongside relevant meta-information of the extracted filters and models.
- We present a data-agnostic method based on sparsity and entropy of filter patterns to find "degenerated" convolution layers due to overparameterization or non-convergence of trained CNN models.
- We perform an analysis of distribution shifts in filters over various groups, providing insights that the formed filters are fairly similar across a wide range of examined groups.
- We show that model-to-model shifts are, contrary to the predominant opinion, not only seen in deeper layers.

3.2 Background

We are unaware of any systematic, large-scale analysis of learned filters across a wide range of datasets, architectures, and tasks such as the one performed in this chapter. However, there are, of course, several partially overlapping aspects of our analysis that have been covered in related works:

Filter analysis An extensive analysis of features, connections, and their organization extracted from trained `InceptionV1` (Szegedy et al., 2016a) models was presented in (Olah et al., 2020a,b; Cammarata et al., 2020; Olah et al., 2020c; Schubert et al., 2021; Cammarata et al., 2021; Voss et al., 2021a,b; Petrov et al., 2021). The authors claim different CNNs will form similar features and circuits, even when trained for different tasks. We will obtain similar findings about the learned filters.

Transfer learning Our overall approach in this thesis and chapter can be seen as a form of transfer learning, where we transfer knowledge about the generalization of model populations into new models. A survey on transfer learning for image classification CNNs can be found in (Hussain et al., 2019), and general surveys for other tasks and domains are available in (Pan & Yang, 2010; Zhuang et al., 2020). Yosinski et al. (2014) studied learned filter representations in *ImageNet-1k* (Deng et al., 2009) classification models and presented the first approaches towards transfer learning. They argued that different CNNs will form similar filters in early layers, which will mostly resemble Gabor filters and color blobs. In contrast, deeper layers will capture the specifics of the dataset by forming increasingly specialized filters. However, multiple works have shown that overall, learned representations transfer well to other problems (Zeiler & Fergus, 2013; Khaligh-Razavi & Kriegeskorte, 2014; Donahue et al., 2014) which seem to contrast the findings of Yosinski et al. (2014).

Technically, more similar to our work is the work of Aygun et al. (2017) capturing convolution filter pattern distributions with Gaussian Mixture Models to achieve cross-architecture transfer learning. Additionally, Tayyab & Mahalanobis (2019) demonstrated that convolution filters can be replaced by a fixed filter basis blended into the final weight by 1 × 1 convolution layers. Effectively, these works indicate that the filter space is low-rank. While these works were mostly motivated by achieving efficiency, we are rather interested in deeper insights into learned representations.

Pruning criteria Although we do not attempt pruning, our work overlaps with pruning techniques as they commonly rely on estimation criteria to understand which parameters to compress. These either rely on data-driven computation of a forward-pass (Alain & Bengio, 2018; Luo et al., 2017; He et al., 2017; Lin et al., 2018, 2019), or backward-propagation (Yu et al., 2018; Ding et al., 2019), or estimate importance solely based on the numerical weight (typically any L^p-norm) of the parameters (Li et al., 2019; Han et al., 2015; Li et al., 2017; He et al., 2019, 2018). We will also use a simple magnitude-based approach to estimate sparsity, but additionally introduce a method based on the entropy of singular values.

3.3 CNN Filter DB

To study learned filters, we construct *CNN Filter DB*, a database of learned filter kernels extracted from a large and heterogeneous CNN model zoo.

3.3.1 Collecting Filter Kernels

We collect a total of **647 publicly available CNN models** from *Pytorch Image Models (timm)* (Wightman, 2019), *RobustBench* (Croce et al., 2021), *torchvision* (Paszke et al., 2019) and other sources that have been pre-trained for various 2D visual tasks. In order to provide a heterogeneous and diverse representation of convolution filters "in the wild," we retrieved pre-trained models for 11 different tasks: *Classification, GAN-Generator, Segmentation, Object Detection, Style Transfer, Depth Estimation, Face Detection, Super Resolution, GAN-Discriminator, Face Recognition, Auto-Encoder.* We also recorded various metadata such as depth and frequency of included operations for each model and manually categorized the variety of used training sets into 16 visually distinctive groups: *plants, natural, art, map, handwriting, medical CT, medical MRI, depth, faces, textures, fractals, seismic, astronomy, thermal, medical X-ray, cars.* This classification, while potentially subjective, provides a pragmatic heuristic for the organization of our model zoo. In total, the models were trained on 71 different datasets, with the dominant subset being formed by *image classification* models trained on *ImageNet-1k* (Deng et al., 2009) (355 models).

Additionally, we have purposely train low-resolution variants of `AlexNet` (Krizhevsky et al., 2012), `DenseNet-121/161/169` (Huang et al., 2017), `ResNet-9/14/18/34/50/101/152` (He et al., 2016), `VGG-11/13/16/19` (Simonyan & Zisserman, 2015), `MobileNet v2` (Sandler et al., 2018), `Inception v3` (Szegedy et al., 2016b), and `GoogLeNet` (Szegedy et al., 2016a) models for image classification on simple datasets such as *CIFAR-10/100* (Krizhevsky et al., 2009), *MNIST* (LeCun et al., 1998b), *Kuzushiji-MNIST (KMNIST)* (Clanuwat et al., 2018), and *Fashion-MNIST* (Xiao et al., 2017) in order to study the effect of overparameterization on learned filters. Please see Section 2.3.4 for details about these and other CNN architectures.

All collected models were trained with full 32-bit precision.[1] Please see Gavrikov & Keuper (2022b) for an overview of all models. Afterward, we converted the collected models into the ONNX format (Bai et al., 2019), which allows a streamlined filter extraction without framework dependencies. Finally, we extract filter kernels from the ONNX zoo. We only focus on the widely used filters from regular convolution layers with a kernel size of 3 × 3 to keep the computational load manageable.

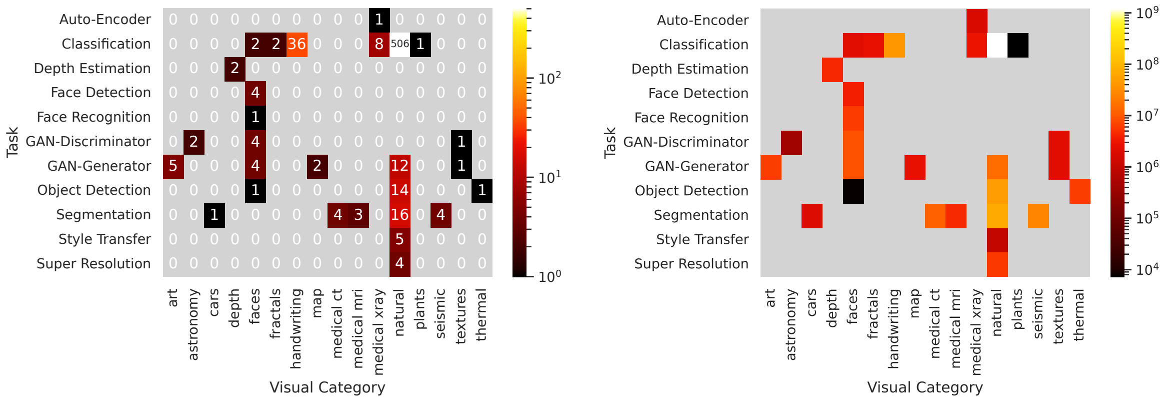
3.3.2 Dataset Statistics

We provide *CNN Filter DB* as a ca. 100 GB large HDF5 file, which contains unprocessed 3 × 3 filters along with meta information. In total, 1,464,797,156 filter kernels from 21,436 layers and 647 models have been obtained for our dataset. Every kernel is accompanied by metadata about its source model and its position in the model architecture. Heatmaps for aggregated frequency of filters/models by task and visual category are shown in Figure 3.2.

3.4 Methodology of our Study

In this section, we will introduce the methodology to analyze and compare convolution filters. Our analysis is based on **principal component analysis (PCA)** achieved through singular value decomposition (SVD) (Jolliffe, 1986) and magnitude statistics. We will briefly provide a general introduction to PCA/SVD before applying it to our specific use case and introducing our magnitude-based analysis.

[1] Although, experiments in Section 3.6.1 indicate that mixed/reduced precision training (Micikevicius et al., 2018) does not affect distribution shifts beyond noise.



(a) Model frequency **(b)** Kernel frequency

Figure 3.2: *CNN Filter DB* statistics. Number of available (a) models and (b) filter kernels for each task/visual category of the training dataset(s)-combinations in *CNN Filter DB*. Gray areas denote no available samples.

Recall that we can apply a SVD transforming a matrix $\boldsymbol{X} \in \mathbb{R}^{n \times m}$ with n samples and m features into

$$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^\top (+\mu), \tag{3.1}$$

where $\boldsymbol{U}$ is a $n \times n$ orthogonal matrix, $\boldsymbol{S}$ is a $n \times m$ diagonal scaling matrix, and $\boldsymbol{V}^\top$ is a $m \times m$ rotation matrix (Banerjee & Roy, 2014). We call the row vector $\boldsymbol{C}_{i,:}$ with $\boldsymbol{C} = \boldsymbol{U}\boldsymbol{S}$ the **coefficient vector** for the i-th sample in $\boldsymbol{X}$. The diagonal entries $\boldsymbol{S}_{i,i}, i = 1, \dots, \min(n, m)$ form the **singular values**. We define them as being arranged in a decreasing order. Further, we will assume that $n > m$, resulting in m singular values, and the last $n - m$ rows of $\boldsymbol{S}$ being zero. The row vectors $\boldsymbol{V}^\top_{i,:}$ are called **principal components** or **basis vectors** and form an orthonormal basis. Commonly, $\boldsymbol{X}$ is normalized to feature-wise zero mean before the decomposition by computing the offset μ to ensure that the principal components capture the directions of maximum variance in the data.

The explained variance of the i-th each principal component is given by $\boldsymbol{S}^2_{i,i}/(n-1)$. In the context of machine learning, we are often not interested in the absolute variance, but its ratio to the total variance η (**variance ratio**), which can be obtained by

$$\eta_i = \frac{\boldsymbol{S}^2_{i,i}}{\sum_{j=1}^{m} \boldsymbol{S}^2_{j,j}}. \tag{3.2}$$

Once a basis has been determined, we can project new samples $\boldsymbol{X}^{'}$ onto it to compute their coefficients $\boldsymbol{C}^{'} = (\boldsymbol{X}^{'} - \mu)\left(\boldsymbol{V}^\top\right)^\top = (\boldsymbol{X}^{'} - \mu)\boldsymbol{V}$, or, equivalently, project coefficients $\boldsymbol{C}^{'}$ back to the original space by $\boldsymbol{C}^{'}\boldsymbol{V}^\top(+\mu) = \boldsymbol{X}^{'}$.

3.4.1 Identifying Principal Patterns in Filter Groups

Given a group of filters – *e.g.*, which may be all filters from one or multiple layers or models – we first reshape each $(c_\text{in} \times k \times k)$-shaped filter into a $(c_\text{in} \times k^2)$-shaped matrix, and then stack all filter matrices into one large group matrix containing n kernels $\boldsymbol{X} \in \mathbb{R}^{n \times k^2}$, where $n = \sum_i c^{(i)}_\text{in}$. Recall that we limit our analysis to $k = 3$ kernels.

If our group contains filters from different layers, they may be different in magnitude – for instance, deeper layers are usually initialized with lower magnitudes to avoid vanishing gradients (Glorot & Bengio, 2010; He et al., 2015). To introduce invariance to this, we rescale $\boldsymbol{X}$ kernel-wise to uniform norm (except zero kernels):

$$\boldsymbol{X}_{i,:} = \begin{cases} \boldsymbol{X}_{i,:}/\|\boldsymbol{X}_{i,:}\|_\infty, & \text{if } \|\boldsymbol{X}_{i,:}\|_\infty \neq 0 \\ \boldsymbol{X}_{i,:} & \text{else} \end{cases} \tag{3.3}$$

Then, we apply the PCA methodology from Equation 3.1 to obtain the basis $\boldsymbol{V}^\top$. Reshaping $\boldsymbol{V}^\top$ into a ($k^2 \times k \times k$)-shaped tensor gives us the k^2 principal patterns (basis vectors) – *i.e.*, the patterns from which the filters are constructed. Every i-th principal pattern is accompanied by the variance ratio η_i, which can be understood as the importance of this pattern to the filter group. Note that this methodology is invariant to the kernel magnitude and also the order in the filter, layer, or model.

3.4.2 Measuring Layer-wise Filter Quality

In our study, we want to measure the quality of trained filters. Specifically, given a convolution layer weight $\mathbf{W} \in \mathbb{R}^{c_{\text{out}} \times c_{\text{in}} \times k \times k}$ with c_{out} filters (along the first) axes, we are interested in gauging the "goodness" of that layer.

Determining this may prove challenging, and we revert to an assumption based on the *Lottery Ticket Hypothesis (LTH)* (Frankle & Carbin, 2019). The hypothesis states that each network has a specific amount of weights that saturates its ability to transform a given dataset into a well-separable feature space. Exceeding this number will result in a partitioning of the network into multiple interconnected submodels, of which most (the *losing* subnetworks) will not contribute to the final prediction. We hypothesize that these losing subnetworks can be seen in the form of "degenerated" filters in CNNs. In like manner, an insufficient amount of training samples or training epochs will also lead to degenerated filters.

We characterize the following possible types of degeneration.

1. *High sparsity:* Filters are dominantly close to zero and therefore produce quasi-zero feature maps (Li et al., 2019). Resulting feature maps from these filters carry no vital information and can be discarded (pruned) due to their low magnitude.
2. *Low diversity in structure:* Filters are structurally similar to each other and, therefore, redundant. They produce similar feature maps in different scales and could be represented by a subset of present filters.
3. *Randomness:* Filter weights are conditionally independent of their neighbors. This indicates that no or not sufficient training was performed, and filters are performing random transformations.

Sparsity

We propose to measure sparsity degeneration in a layer by the ratio of sparse filter kernel in $\mathbf{W}$. A kernel $\mathbf{W}_{i,j,:,:}$ is considered sparse if all its entries are near zero – or, analogously, the L^∞-norm of a kernel falls below a certain threshold ϵ_0. We define our first metric, **sparsity**, by

$$\text{Sparsity}(\mathbf{W}) = \frac{|\{(i,j) \in \{1,\dots,c_{\text{out}}\} \times \{1,\dots,c_{\text{in}}\} : \|\mathbf{W}_{i,j,:,:}\|_\infty \leq \epsilon_0(\mathbf{W})\}|}{c_{\text{out}} c_{\text{in}}}. \tag{3.4}$$

Empirically, we set ϵ_0 based on the magnitude of the largest kernel in the layer

$$\epsilon_0(\mathbf{W}) = 10^{-2} \max_{i,j} \|\mathbf{W}_{i,j,:,:}\|_\infty. \tag{3.5}$$

(Principal Pattern) Variance Entropy

To detect the other types of degeneration, we introduce a metric termed **(principal pattern) variance entropy** based on Shannon (1948) Entropy. Therefore, we reshape the tensor $\mathbf{W}$ into a ($c_{\text{out}} \cdot c_{\text{in}} \times k^2$)-shaped matrix and apply PCA to it as in Equation 3.1. Then, we measure the entropy of the explained variance ratio η of each principal component Equation 3.1.

$$\mathrm{VarianceEntropy}(\mathbf{W}) = -\sum_{i=1}^{k^2} \eta_i \log_{10} \eta_i \tag{3.6}$$

If the entropy is close to zero, this indicates one strong principal pattern from which most of the kernels can be reconstructed, and the weight, therefore, shows a low filter diversity degeneration. On the other hand, a large entropy indicates a (close to) uniform distribution of the variance across patterns (*i.e.*, no actually principal pattern exists) and, thus, a randomness of the kernels. It should be noted that the entropy only becomes expressive if the number of samples exceeds the number of features, *i.e.*, $c_{\text{out}} \cdot c_{\text{in}} \gg k^2$ in our case. Thus, our variance entropy metric can only be used for comparing layers with approximately the same number of kernels $n = c_{\text{in}} \cdot c_{\text{out}}$ or needs to be normalized by a "randomness" threshold.

To obtain such a threshold, we perform multiple experiments in which we initialize convolution filters of different sizes from a standard normal distribution and fit a sigmoid T_H to the minimum results obtained for entropy.

$$T_H(n) = \frac{L}{1 + e^{-k(\log_2(n) - x_0)}} + b \tag{3.7}$$

Specifically, we draw $n = 2^1, \ldots, 2^{21}$ filters with 3×3 shape from a standard normal distribution and calculate the layer entropy. We repeat this process 1,000 times for each n and fit a sigmoid to the lowest entropy we have observed for each n. Figure 3.3 shows the obtained samples alongside the fitted sigmoid T_H. We obtain the following values $L = 1.26$, $x_0 = 2.30$, $k = 0.89$, $b = -0.31$ and call any layer with $\mathrm{VarianceEntropy}(\mathbf{W}) > T_H(c_{\text{in}} \cdot c_{\text{out}})$ random. These values are specific for 3×3 kernels and will not be valid for other kernel sizes.

This metric is related to sparsity, as sparse layers can be seen as a specific form of low-diversity degeneration, whereas sparsity and randomness are mutually exclusive.

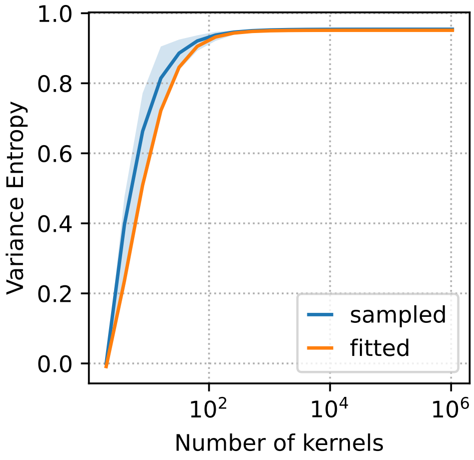


Figure 3.3: Randomness entropy threshold. Sampled entropy for randomly initialized convolution layers with n kernels and sigmoid T_H fitted to the lower bound.

3.4.3 Measuring Distribution Shifts between Filter Groups

Additionally, to the analysis of principal patterns and measuring the quality through our metrics, we want to measure the "distance" between two groups of filters. For instance, we may compare all filters from two layers, models, or even entire populations of models. Framing this as a distribution shift between pattern distributions allows us to use the same preprocessing of reshaping, stacking, and scaling each group into two matrices $\boldsymbol{X}^p, \boldsymbol{X}^q$. Note that the sample size between both tensors may be different.

We can then project both tensors to a common basis $\boldsymbol{V}^\top$ giving us the coefficients $\boldsymbol{C}^p = \boldsymbol{X}^p \cdot \boldsymbol{V}^\top, \boldsymbol{C}^q = \boldsymbol{X}^q \cdot \boldsymbol{V}^\top$. In our case, we use the basis that we obtain from a PCA applied to all scaled kernels in our database. Then we can represent each group by a density histogram (*i.e.*, a discrete distribution) p_i for each principal component $V_i^\top$. The histogram range is defined by the minimum and maximum values of all coefficients in our dataset and is divided into 70 uniform bins.

We measure the divergence between two distributions by the symmetric, non-negative variant of Kullback & Leibler (1951), which we denote as ($D_{\text{KL-SYM}}$).

$$\begin{aligned} D_{\text{KL}}(p \| q) &= \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)} \\ D_{\text{KL-SYM}}(p \| q) &= D_{\text{KL}}(p \| q) + D_{\text{KL}}(q \| p) \end{aligned} \tag{3.8}$$

Finally, the shift D between two filter sets is given by the sum of the divergence of the coefficient distributions p_i, q_i along every principal component index i. The sum is weighted by the ratio of variance η_i explained by the i-th principal component of the common base $\boldsymbol{V}^\top$.

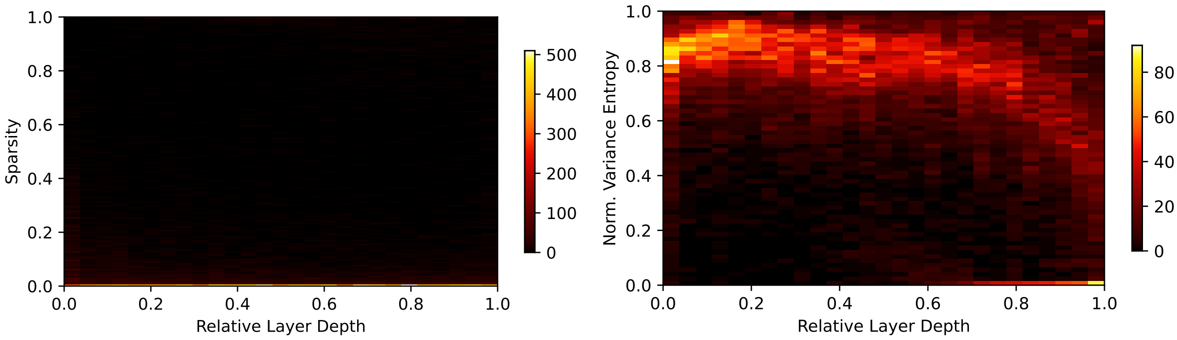
$$D(p\|q) = \sum_i \eta_i \cdot D_{\mathrm{KL-SYM}}\left(p_i \parallel q_i\right) \tag{3.9}$$

To avoid undefined expressions, all probability distributions p are set to hold $\forall x \in \mathcal{X} : p(x) \geq \epsilon$ for some small ϵ.

3.5 Analysis of Convolution Layer Degeneration in CNNs

In this section, we first discuss our quality metrics and then study relative degeneration in three settings: *global* (all filters), *overparameterization* (model with an excessive capacity), and *adversarial robustness* (models trained with adversarial training as introduced in Section 2.8.2).

3.5.1 (Global) Filter Quality



(a) Global distribution of sparsity

(b) Global distribution of variance entropy

Figure 3.4: Distribution of layer quality in CNN Filter DB. We measure the quality for each layer and show heatmaps of the distribution over the relative depth of models in a number of layers. Layers of relative depth 0 are the layers closest to the inputs, whereas a relative depth of 1 indicates a layer at the outputs.

One interesting application of our metrics may be to use them as a data-independent criterion of layer or model “goodness”, *e.g.*, when selecting the best model for finetuning. To this end, we require a threshold for our metrics, at which a layer could be called degenerated.

While we have established when a layer can be considered random, we have not provided a good intuition of what it means when a layer has low entropy. Clearly, a layer is degenerated when all variance is distributed on one principal component, *i.e.*, VarianceEntropy($\mathbf{W}$) = 0 and there is no diversity in the kernel structure. However, in most cases, a layer is already severely defunct before that.

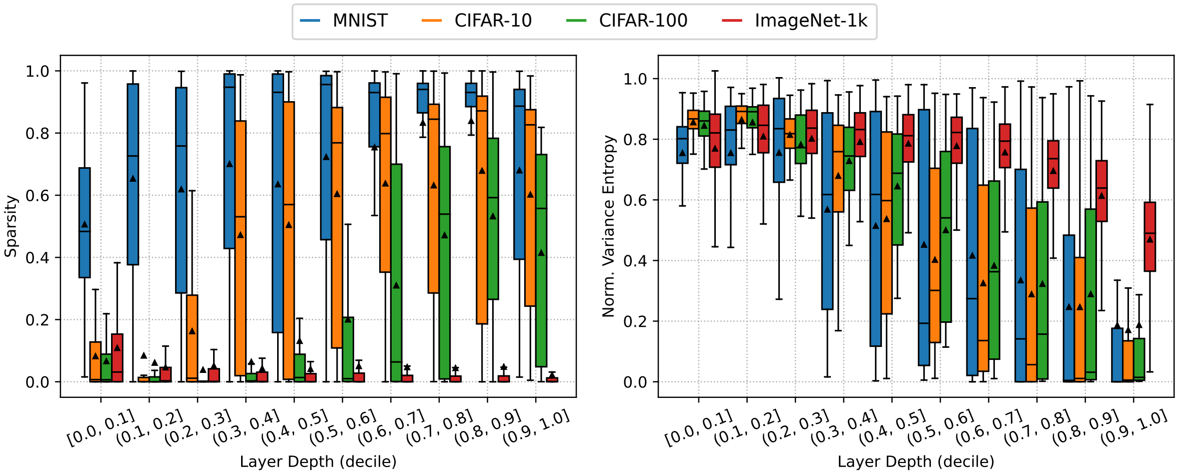
As an intuition for the threshold, we can only rely on statistics (Figure 3.4b): The average variance entropy is 0.69 over all layers and continuously decreases from an average of 0.75 to 0.5 with depth. Additionally, the minimum of the 1.5 interquartile range (IQR) also steadily decreases with depth. As a rule of thumb, we suggest considering every layer with VarianceEntropy($\mathbf{W}$) < 0.5 as degenerated due to low variance.

Equivalently, there is no intuitive way to derive a threshold for sparsity. A layer is degenerated if all kernels are sparse under $\epsilon_0(\mathbf{W}) = 0$ as this stops the feature flow, but intermediate values are hard to interpret. From a capacity perspective, we want sparsity to be as low as possible to squeeze the most performance from our model, but from a size perspective, we may prefer sparse models to decrease inference time memory consumption.

Again, relying on global statistics (Figure 3.4a), we find that the average sparsity over all layers is 0.12 and only 56.5% of the layers in our dataset hold Sparsity($\mathbf{W}$) < 0.01, and 9.9% even have a 50% sparsity. In terms of the depth of the associated convolution layer, the average sparsity varies between 9.9% and 14%, with the largest sparsity found in the last 20% of the model depth. The largest outliers of the 1.5 IQR are, however, found in the first decile.

In both cases, we find it difficult to provide a meaningful general threshold and suggest determining this value on a case-by-case basis. Our metrics can, however, be used to compare relative quality between groups of interest.

3.5.2 Overparameterization



(a) Sparsity vs. overparameterization

(b) Entropy vs. overparameterization

Figure 3.5: Development of layer filter quality with model depth under *overparameterization.* We measure the layer entropy and sparsity of image classification models trained on *MNIST* (most overparameterized), *CIFAR-10*, *CIFAR-100*, and *ImageNet-1k* (least overparameterized). We group the depth of the layers into deciles for comparability to models with different depths. Outliers are hidden for clarity.

The majority of the models that we have trained on our low-resolution datasets are heavily overparameterized for these relatively simple problems. We base this argument on the fact that we have models with different depths for most architectures and already observe near-perfect performance with the smallest variants. Therefore, it is safe to assume that larger models are overparameterized, especially since the performance only increases marginally.

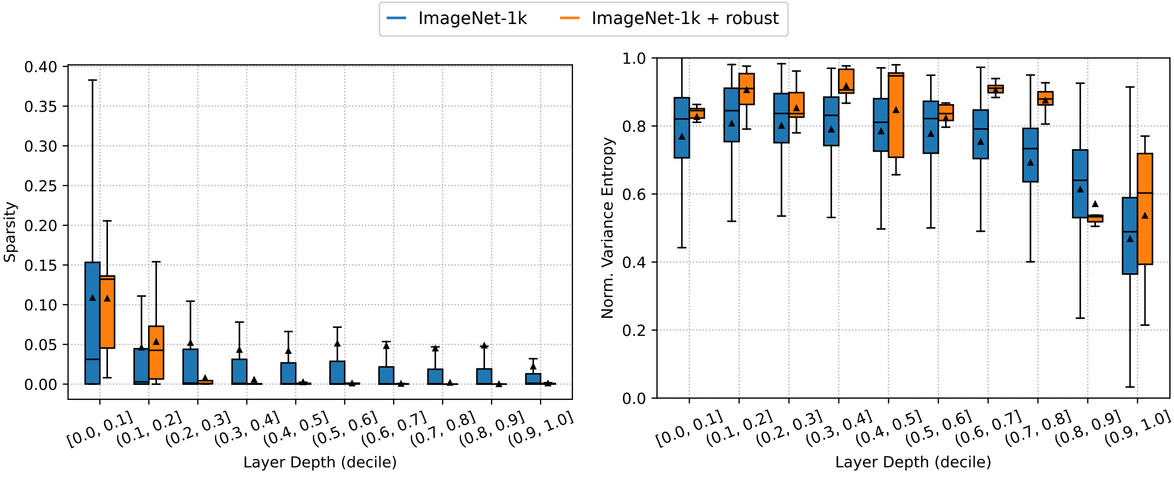
First, we analyze layer sparsity and entropy for these models trained on *MNIST*, *CIFAR-10*, *CIFAR-100*, and *ImageNet-1k* classification models found in our dataset. For all low-resolution datasets, we study models with identical network architectures trained using the same hyperparameter settings. The data for *ImageNet-1k*, on the other hand, is averaged over all respective models in our zoo.

Figure 3.5a shows a correlation between overparameterization and sparsity – models with a larger degree of overparameterization contain significantly more sparse kernels on average. Further, sparsity seems to increase with depth. In particular, we see the most sparse kernels for *MNIST*. However, *ImageNet-1k* classifiers also seem to have some kind of “natural” sparsity, even though we do not consider most of these models as overparameterized.

Entropy, on the other hand, decreases with increasing layer depth for every classifier but more rapidly in overparameterized models (Figure 3.5b). Again, the *MNIST* models degrade the fastest and overall show more degeneration.

The overparameterized models contain layers that have an entropy close to 0 towards deeper layers, which may indicate that these models are “saturated” and only produce differently scaled variants of the same filters. In line with the oversaturation, these models also have increasingly sparse filters, presumably as an effect of weight regularization (Section 2.6.1).

3.5.3 Adversarial Robustness



(a) Sparsity vs. robustness

(b) Entropy vs. robustness

Figure 3.6: Development of layer quality with model depth under *adversarial robustness*. We measure the layer entropy and sparsity of image classification models trained on ImageNet-1k with and without adversarial training. We group the depth of the layers into deciles for comparability to models with different depths. Outliers are hidden for clarity.

Our dataset also contains “robust” models from the *RobustBench* (Croce et al., 2021) leaderboard trained with *adversarial training* (Madry et al., 2018). Please see Section 2.8.2 for an introduction to adversarial robustness. When comparing robust models with non-robust models trained on *ImageNet-1k*, it becomes clear that robust models form almost no sparse filters in deeper convolution layers (Figure 3.6a), while regular models show some sparsity there. The entropy of robust models is also higher throughout depth (Figure 3.6b), indicating that robust models learn more diverse filters. We provide a more comprehensive discussion and analysis of robust models in Chapter 4.

3.6 Analysis of Filter Patterns

In the next series of experiments, we analyze only the patterns of 3 × 3 filter kernels, neglecting their magnitude. We start by analyzing the principal patterns (basis) forming the filters. Figure 3.7 shows some qualitative examples of obtained principal patterns, split by several metadata dimensions. Overall, we observe that the PCA-obtained principal patterns (basis) are often very similar and independent (ignoring variances due to inversion and the order) of the filter group we analyze, except for a few outliers. The explained variance, however, fluctuates significantly and sometimes changes the order of the basis images. Consistently, we observe substantially higher variance on the first principal components.

However, this does not necessarily correlate with the shift observed between models, which we will observe in later sections. Here, the biggest mean shift is also located in the first principal component

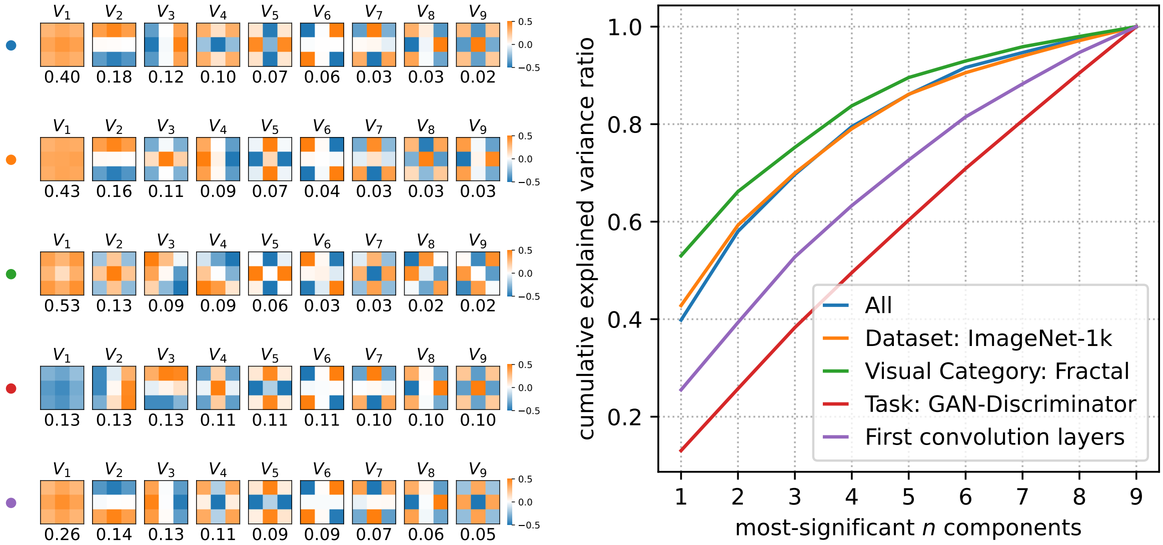


Figure 3.7: Principal patterns. Selected depiction of the principal patterns (filter basis; left) and cumulative explained variance ratio per component for filters from ● full dataset, ● models trained on *ImageNet*, ● models trained on images of *fractals*, ● GAN discriminators, ● first convolution layers (right). The values below the principal patterns denote the explained variance ratio.

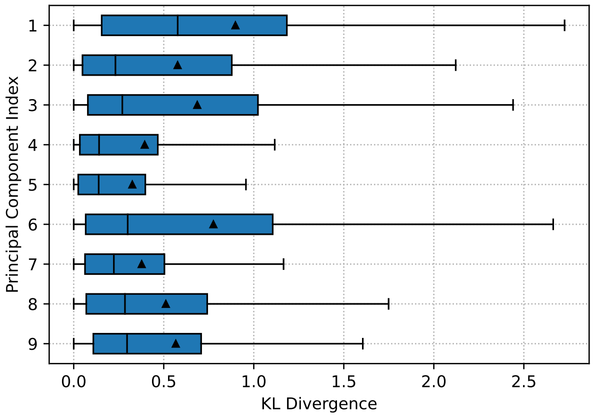


Figure 3.8: Distribution shift per axis. We measured the distribution shift D along principal components computed on all possible pairings of models. Outliers are hidden for clarity.

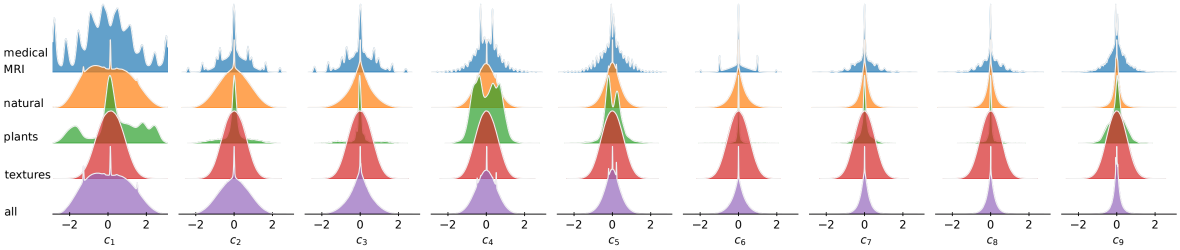


Figure 3.9: Population distributions for selected visual categories. KDEs of the coefficient distributions along every principal component for selected visual categories. Please note the changed color palette compared to Figure 3.7.

($\hat{D} = 0.90$) but is then followed by the sixth, third, and second component ($\hat{D} = 0.78, 0.69, 0.58$). The coefficients of the sixth component also contain the strongest outliers (Figure 3.8).

We can additionally visualize the distributions of pattern coefficients along every principal pattern for each group by plots of kernel density estimates (KDEs). For instance, in Figure 3.9, we depict the distributions of filters grouped by some selected visual categories compared to the distribution of coefficients for the full dataset. Noticeably, models with degenerated layers result in spiky/multi-modal KDEs (as seen in *medical MRI*).

Alternatively, the distributions can be visualized by bi-variate scatter plots between two principal pattern distributions that may reveal more details than KDEs. We roughly categorize four salient distributions that we observed (which we will call *phenotypes*) depending on their distribution characteristic in the PCA space (Figure 3.10): *sun:* distributions where both dimensions are Gaussian-like. These are to be expected coefficient distributions without significant sparsity/low diversity degeneration. Yet, this phenotype may also include non-converged filters; *spikes:* distributions suffering from a low variance degeneration resulting in local hotspots; *symbols:* at least one distribution is multi-modal, non-centered, highly sparse or otherwise non-normal (low variance degeneration); *point:* coefficients are primarily located in the center (sparsity degeneration).

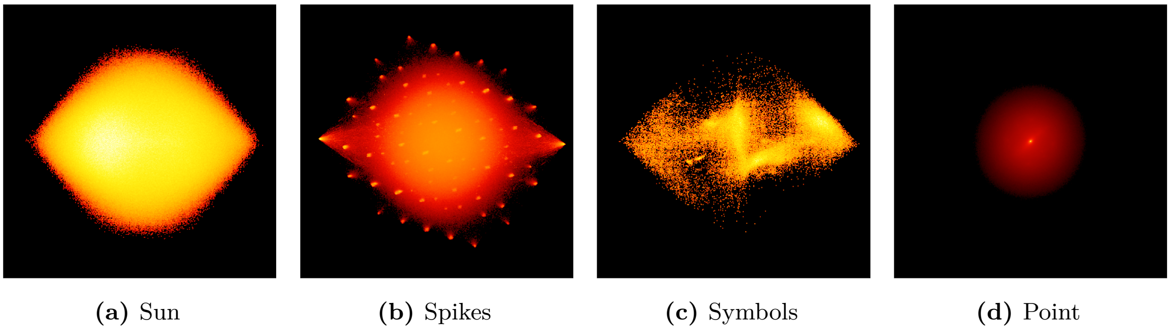

(a) Sun **(b)** Spikes **(c)** Symbols **(d)** Point

Figure 3.10: Phenotypes of coefficient distributions. We show bi-variate plots between component distributions showing the four phenotypes.

3.6.1 Formation of Filter Patterns during Training

Although our dataset only includes *trained* convolutional filters from models where we assume convergence, we want to understand how the coefficient distribution shifts during training. Therefore, we train a `ResNet-9` (Myrtle AI Team, 2019) on *CIFAR-10* and record a checkpoint every 10 training epochs beginning, including one right after the weight initialization.

Figure 3.11 shows that the coefficient distributions along all principal components are Gaussian-like distributed initially and eventually shift during training. For this specific model, distributions along major principal components retain the standard deviation during training, while less significant component distributions decrease. The initialization observation ultimately helped us remove models from our collection where we failed to load the trained parameters and is the foundation for our provided randomness metric.

Stability of emerged patterns Additionally, we want to understand how stable emerged filter patterns are, *i.e.*, would we observe other patterns if we had trained our model with a different random seed? To this end, we train low-resolution networks on *CIFAR-10* multiple times with identical hyperparameters with different random seeds and save a checkpoint of each model at the epoch with the highest validation accuracy. Then, we measure the distribution shift between all filters in the models.

We find that most models converge to highly similar coefficient distributions when retrained with different weight initialization (*e.g.*, `ResNet-9` with $D < 5.3 \cdot 10^{-4}$). However, some architectures

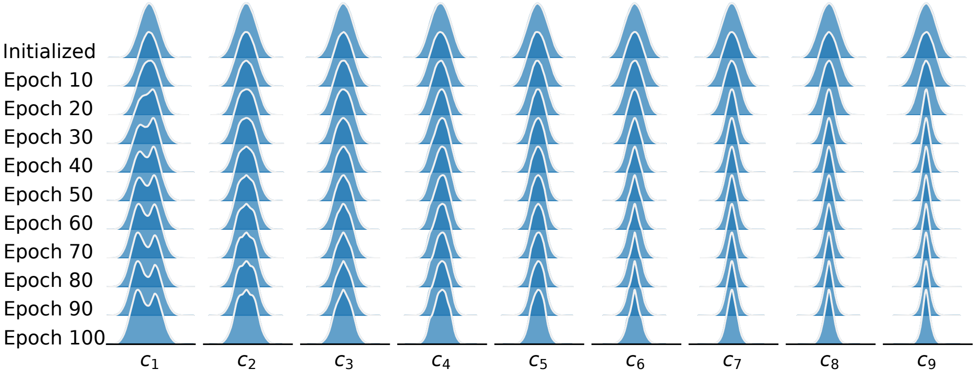


Figure 3.11: Evolution of pattern coefficient distributions in training. Distributions over all layers shown for a `ResNet-9` trained on *CIFAR-10* every 10 epochs.

such as `MobileNetv2` show higher shifts ($D < 2.6 \cdot 10^{-2}$). This number can be seen as a baseline for divergence when comparing models in later sections.

Distribution shift by precision A common "trick" to accelerate training is to reduce the precision of learned weights. Initially, we assumed that quantization might cause the *spikes* phenotype, so we decided to test what shift we obtain when training with fp16 instead of fp32 precision. Such "spiky" distributions should show high shifts compared to smooth distributions.

We train all our low-resolution models on *CIFAR-10* with the same hyperparameters and generally observe marginal shifts. Outliers with somewhat higher shifts again include `MobileNetv2`. But we have verified that this shift (and also on `ResNet-9`) does not exceed the shift we would measure by training with random seeds.

3.7 Global Distribution Shifts between Model Populations

This section investigates distribution shifts between filter groups separated by different meta-dimensions that include multiple models. We compute the shift and visualize this as heatmaps in Figure 3.12. These heatmaps show shifts between all pairings of possible *tasks, data categories*, and *layer depths (grouped by decile)*. Additionally, we show some shifts between a few architectures trained on *ImageNet*. We fix the range of the colorbar to the same values in all heatmaps for an easier comparison unless stated otherwise.

Shifts between tasks Perhaps unsurprisingly, we find that *classification*, *segmentation*, *object detection*, and *GAN-generator* distributions are quite similar since the non-*classification* models typically include a *classification* backbone. The smallest mean shift to other tasks is observed in *object detection*, *GAN-generators*, and *depth estimation* models. The least transferable distributions are *GAN-discriminators*. Their distributions barely differ along principal components and can be approximated by a Gaussian distribution. By our randomness metric, this indicates a filter distribution that is close to random initialization, implying a "confused" discriminator that cannot distinguish between real and fake samples towards the end of (successful) training. Overall, we do find that representations of tasks show fairly low shifts.

It may be surprising to see a slightly larger average shift for *classification*. This is presumably due to many degenerated layers in our collected models, which are also visible in the form of spikes when studying the KDEs. Under an ablation of distributions including only non-degenerated filters, *classifiers* showed a lower average shift due to the similarity of the aforementioned tasks.

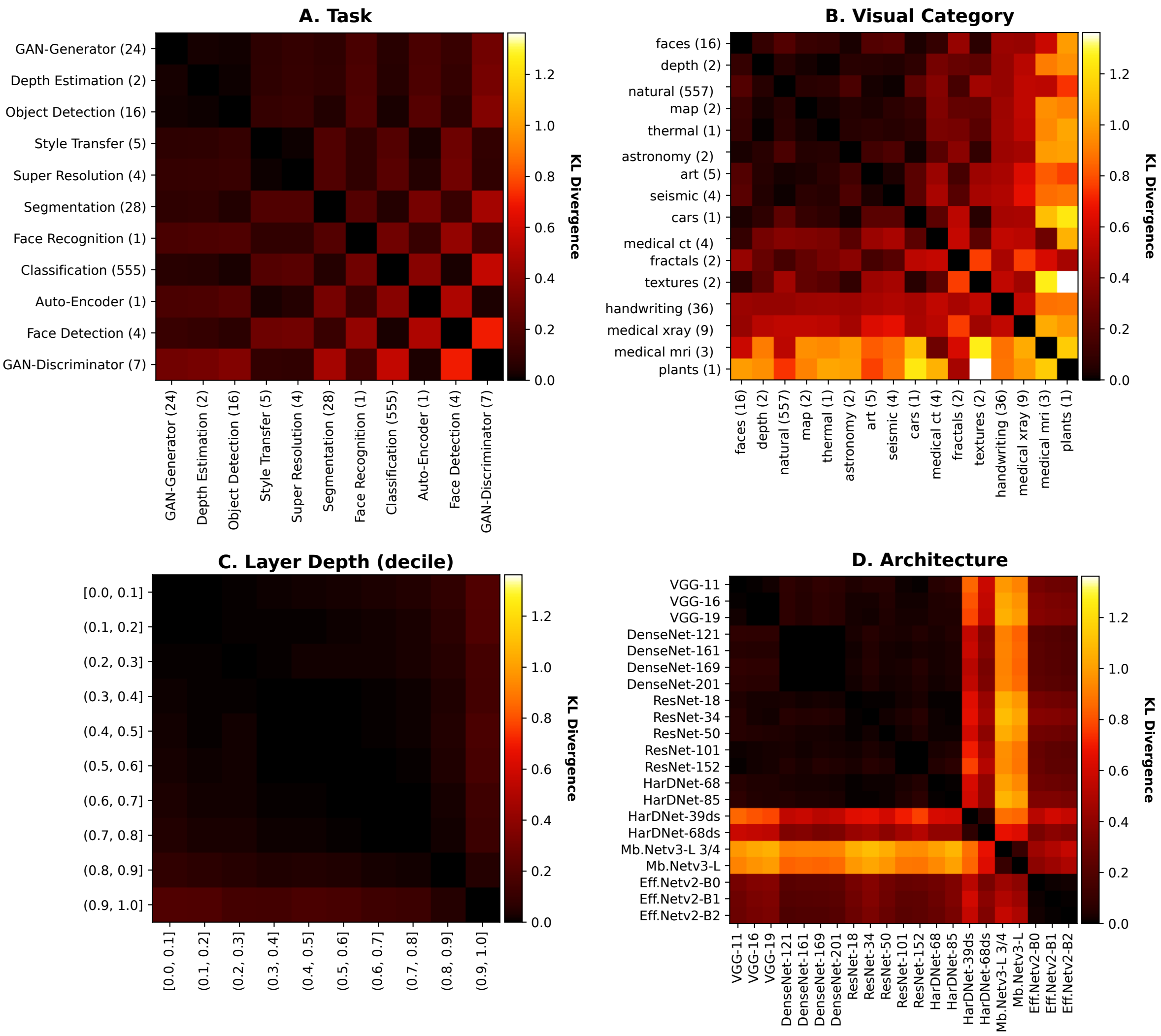


Figure 3.12: Global distribution shifts. We show population shifts between (**A**) tasks, (**B**) visual categories, (**C**) layer depths, and (**D**) architectures.

Shifts between visual categories (and training sets) We find that the distribution shift is well-balanced across most visual categories and training sets. Notable outliers include all *medical* types. They have visible spikes in the KDEs, again indicating degenerated layers. This is confirmed by the average sparsity in these models, which is extreme in the last 80% of the model depth. We provide a more detailed discussion and analysis of medical models in Section 3.9.

Another interesting, albeit less significant, outlier is the *fractal* category. It consists of models trained on *Fractal-DB*, which was proposed as a synthetic pre-training alternative to *ImageNet-1k* (Kataoka et al., 2020). For *ImageNet*-trained models, the explained variance of the coefficients tends to shrink towards the least significant principal components on our global basis. Still, this trend is not visible for this category, suggesting the common basis we use is not well suited for this category. Indeed, comparing the basis in Figure 3.7 shows us that some principal patterns are rotated compared to other models. Also notable is a remarkably high variance explained by the first principal component, suggesting a kind of “pattern collapse” where filters look relatively similar to each other and, thus, model similar transformations. Interestingly, we observe a sub-average degeneration for this category. Overall, this shows that while pretraining with *Fractal-DB* improves accuracy on *ImageNet*, it does not lead to identical representations.

Shifts in other categories can usually be explained by imbalances in our model zoo. For example, we only have one model trained on *plants* data, the *handwriting* models consist exclusively of overparameterized networks that suffer from layer degeneration, and *textures* consists of only one *GAN-discriminator* which will naturally show high randomness, as discussed before.

Shifts by layer depth The shift between layers of various depth deciles increases with the difference in depth, with distributions in the last decile of depth forming the most distinct interval and outdistancing the second-to-last and first decile that follow next. However, the distribution shift between layers of different deciles is very low overall. We attribute the observed shift in the deepest layers mostly to degeneration, as discussed in Section 3.5.1. This suggests that in filter weight space, representations are not fundamentally differently distributed throughout depth.

Shifts by architecture When we fix the training data (*ImageNet*) and task (*Classification*), we observe an interesting phenomenon between models: the "family" of the architecture (*e.g.*, `VGG`) seems to control the pattern distribution, yet the scale of the architecture only has a minor influence (*e.g.*, `VGG-13` vs. `VGG-19`). Overall, this means that the intra-family shift is low while the inter-family shift is high(er). We additionally visualize the intra-family similarity in Figure 3.13, showing the pairwise divergence between classification models with a `ResNet-50`-like architecture. The two outliers consist of models that show a high amount of sparsity. Note that in general, the intra-family similarity does no longer hold if the architecture is changed significantly during scaling, *e.g.*, `ResNet-34` based on Basic-Blocks vs. `ResNet-101` based on Bottleneck-Blocks (although in this specific analysis the Bottleneck-based `ResNet-50` is more similar to `ResNet-34`).

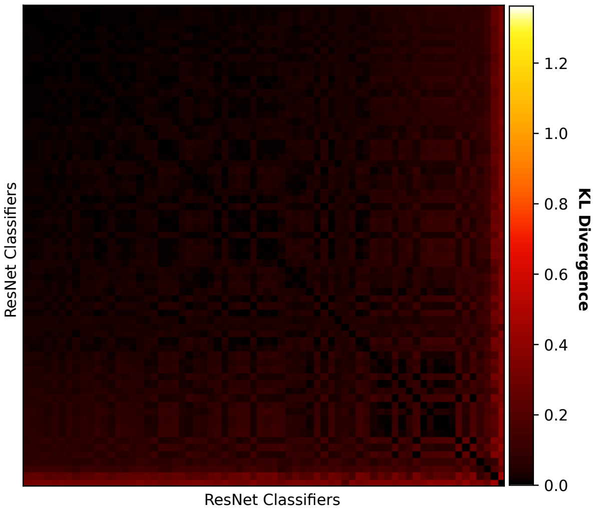
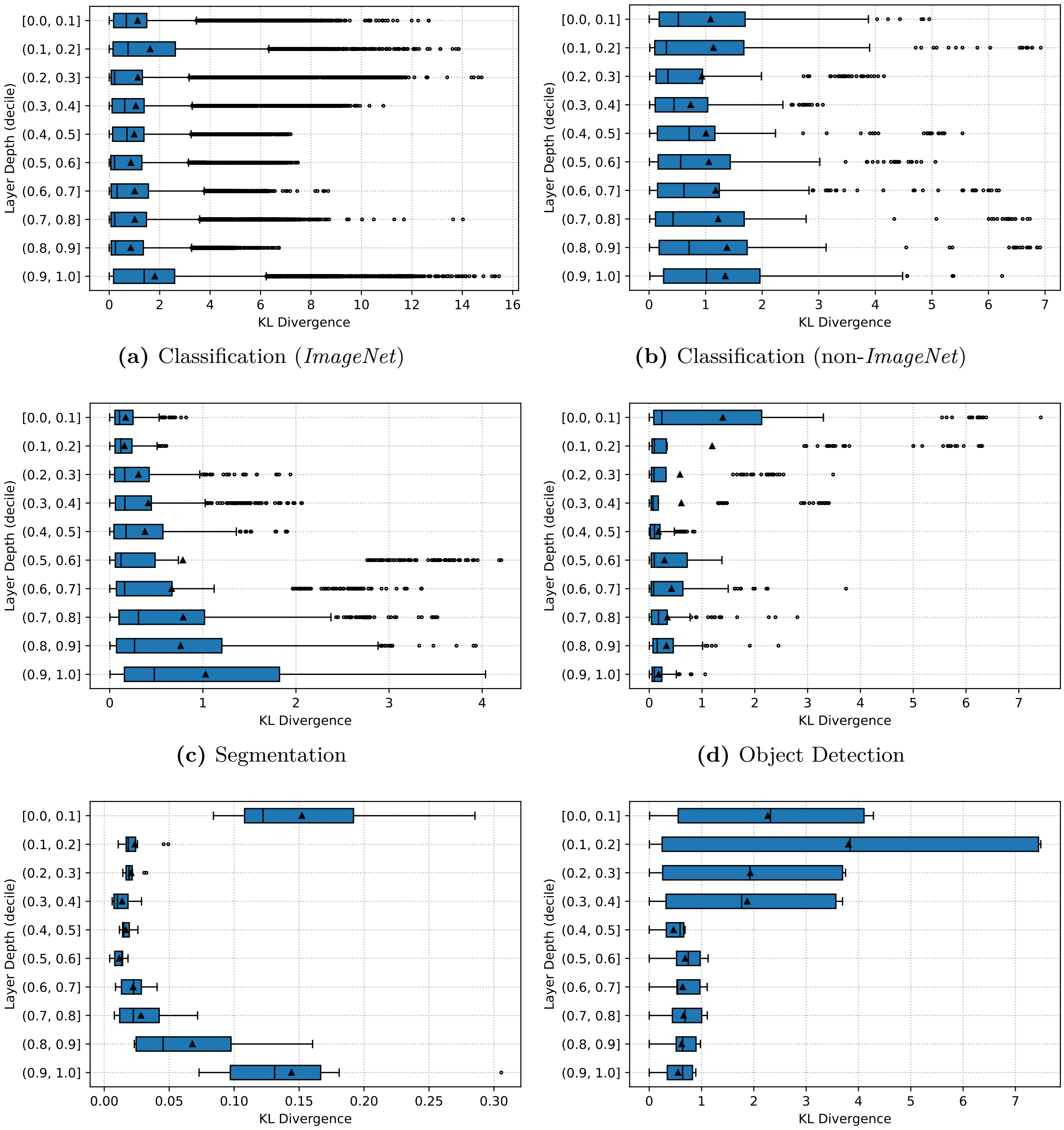


Figure 3.13: Distribution shifts in the `ResNet` family. Model-to-model shift between different pairs of *ResNet-classifiers* excluding our intentionally overparameterized models. Each row/column depicts one model.

Generally, we also find that models based on regular convolutions (*i.e.*, `VGG` (Simonyan & Zisserman, 2015), `ResNet` (He et al., 2016), `DenseNet` (Huang et al., 2017), `HarDNet` (Chao et al., 2019)) seem to be more similar in their patterns, than models based on depthwise-separable convolutions (as introduced in Section 2.3.2) such as, `HarDNet-ds` (Chao et al., 2019), `MobileNetv3` (Howard et al., 2019), `EfficientNetv2` (Tan & Le, 2021). The latter models form highly unique distributions compared to other depthwise-separable models.



(a) Classification (*ImageNet*)

(b) Classification (non-*ImageNet*)

(c) Segmentation

(d) Object Detection

(e) Style Transfer

(f) Face Detection

Figure 3.14: Model-to-model distribution shifts by task. We exclude robust models and our intentionally overparameterized models. Please mind the different X-axis scale.

3.8 Local Model-to-Model Distribution Shifts

While we have analyzed the model population in the last section to understand global patterns across selected meta-axes, we now turn our analysis to model-to-model comparisons, providing more actionable insights.

We perform our analysis by studying the distribution shift between models along (relative) depth. This shift exemplifies the uniqueness of formed filters per layer and will show us where models start to diverge in their learned representations. Based on the general assumptions about deep learning, our expectation is that early layers capture common patterns but start to specialize in deeper layers (Zeiler & Fergus, 2013; Yosinski et al., 2014), which should be visible in their learned weights.

However, our results shown in Figure 3.14 suggest that this is not always true and the spatial distribution of the shifted layers strongly depends on the training task. *Segmentation* models clearly follow the expected trend, while object detection models primarily seem to diverge in the first decile of depth. This counter-intuitive divergence along depth is also visible in *face detection* models, where divergence is particularly pronounced in the first four deciles.

For image classification, we separate *ImageNet*-trained models from others due to the significantly larger number of the former. We observe that the shift within *ImageNet* models is fairly static along depth. For non-*ImageNet* models, we find that the shift appears to increase with depth, but the shift within the first two deciles is also significant and stronger than in some of the deeper layers.

Finally, we also observe a very atypical depthwise distribution of shift for the *style transfer* models. It is important to note that all models for this task are based on the same encoder-decoder architecture (Gatys et al., 2016). The shift is particularly strong in the earliest and deepest layers, while intermediate layers seem to form more transferable pattern distributions. Assuming that the encoder and decoder occupy the same amount of parameters, the shift across the decoder is higher.

Taken together, our observations cast shadows on the general recommendation to only finetune deep layers under transfer learning – depending on the task, the strongest shifts might be located in early layers or uniformly distributed throughout the model.

3.9 A Closer Look at Medical Imaging Models

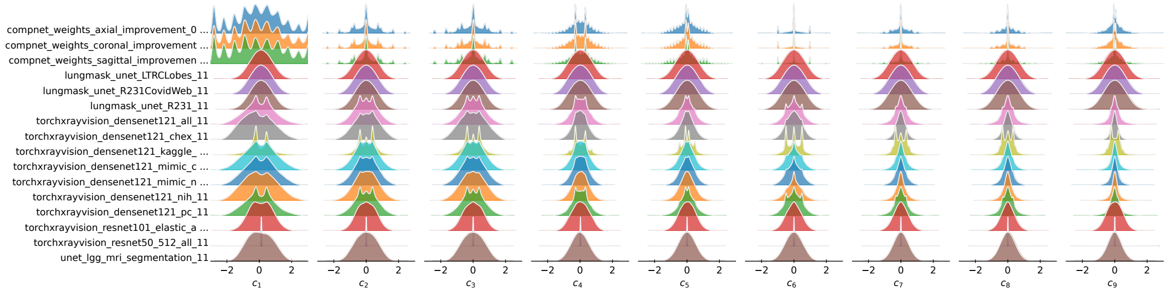


Figure 3.15: KDEs of medical imaging models. We plot KDEs for every principal component for all medical imaging models.

In Section 3.7, we have seen that medical imaging models form one of the most salient shifts by data, as this domain showed distinct "spiky" coefficient distributions (see KDE plots for all filters in each model in Figure 3.15).

This section focuses on studying these models in detail and understanding the possible causes of the shifts. Our model zoo contains the following models: *CompNet* (Dey & Hong, 2018) contains three customized `CompNet` architectures trained for brain segmentation on the *OASIS* (MRI) dataset (Marcus et al., 2007); *LungMask* (Hofmanninger et al., 2020) contains three `U-Net` trained for lung segmentation on the *LTRC* and some proprietary CT datasets; *TorchXRayVision* (Cohen et al., 2022) contains seven DensetNet-121 and one ResNet-50 classifier trained to detect various pathologies from various datasets. Additionally, the framework includes one ResNet-101 auto-encoder trained on a large aggregated dataset consisting of the previously mentioned datasets; Lastly, one individual U-Net (Buda et al., 2019) is included that is also trained to perform brain segmentation on a public *Kaggle MRI* dataset (Shih et al., 2019).

3.9.1 The Curious Case of CompNet

First, we observe that the salient "spiky" distributions are only seen in models of CompNet (on all axes), while all other models form less salient distributions. In our analysis, this model family is highly over-represented in medical classes due to their relatively large number of present convolution filter kernels.

The CompNet architecture can be split into multiple encoder-decoder models: source inputs are fed into an encoder that feeds into two parallel decoder branches. One of those constructs a binary brain segmentation mask, and the other computes a complementary output for the non-brain part of the image. Finally, the outputs are aggregated and further processed by an auto-encoder that learns a latent-space representation (Dey & Hong, 2018).

Upon closer analysis, we observe that some layers learn filters with (almost) binary weight distributions (Figure 3.16), explaining the coefficient spikes in the KDEs. In contrast, other layers learn a less conspicuous set of filters. We assume that the binary filters are learned by the final auto-encoder as indicated by the presence in later convolution stages. Yet, we cannot verify our hypothesis due to the highly entangled implementation of the source model. Still, it seems worthwhile to understand the cause of binary filters in future work, as it is possible to learn them from a significantly lower dimensional and discrete search space, which should, in theory, accelerate learning and require fewer data samples. Interestingly, we find fewer such binary filter layers in the model trained on the axial reconstruction plane.

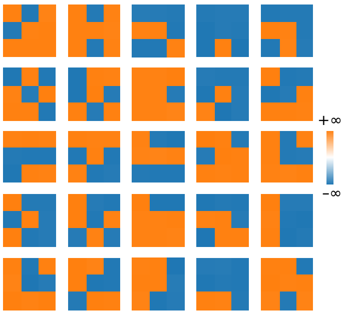


Figure 3.16: Binary filters. The filter kernels weights found in CompNet are approximately bi-modal.

3.9.2 Distribution Shifts between Medical Models

Next, complementary to our previous findings, we observe that models based on the same architecture (*e.g.*, DenseNet-121) learn quite similar distributions when compared to each other but differ substantially when compared to other architectures: all CompNets learn similar coefficient distributions although trained on different reconstruction planes; Albeit trained in different regimes, all *LungMask* models also form similar distributions to the standalone U-Net. Still, some distribution shifts can be measured, in particular in the DenseNet-121 models from *TorchXRayVision*, which were trained on different datasets. The model trained on the *Kaggle RSNA Pneumonia Challenge* dataset seems to overrepresent specific coefficients (and therefore contains clusters of specific filter patterns) by a large amount, while the models trained on other datasets such as *MIMIC-CXR* (Johnson et al., 2019), *NIH* (Wang et al., 2017; Majkowska et al., 2020), and *CheXpert* (Irvin et al., 2019) tend to learn coefficient distributions that are smoother and more similar to the distribution of the model that was trained on all datasets.

The most salient coefficient distribution is seen in the ResNet-50, which was trained on 512 × 512 px images instead of 256 × 256 px. Spikes dominate the KDEs at 0 for all axes, which indicates a highly sparse model. Indeed, by applying a structured pruning on the kernels, we find that only approx. 1% of filters (N=12900) are non-sparse. Due to a lack of access to the datasets, we cannot verify the integrity of this model and hypothesize that this model may exploit a shortcut. However, should this model's performance be on par with the other models, this opens the question of whether a) smaller and, therefore, easier-to-train models would not be more suitable for this specific problem or b) higher

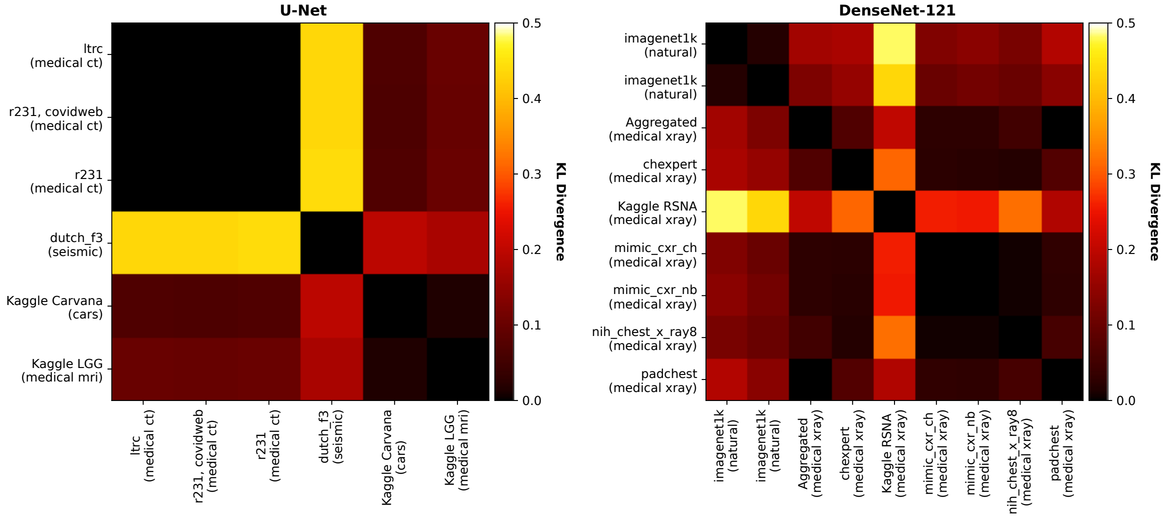


Figure 3.17: Model-to-model-shifts for medical imaging models. Heatmaps showing the pair-wise distribution shift for `U-Net` (left) and `DenseNet` (right) trained on different datasets (including non-medical for comparison). Low values/dark colors denote low shifts.

resolution images may be more suitable as they may contain more salient features for classification than their down-sampled counter-parts.

3.9.3 Distribution Shifts between Medical Imaging and other Domains

Finally, we compare the medical `U-Net` segmentation and `DenseNet-121` classification models to similar models from *CNN Filter DB* trained on other image domains (*cars, seismic data,* and *natural images*) to understand whether medical models learn different convolution filters. We compute the KL in a pair-wise manner between all model combinations and display the results in the form of heatmaps (Figure 3.17). We observe that `U-Net` learns strikingly different representations on *seismic* data, but only minor differences when trained to segment *cars*. `DenseNet-121` also learns slightly different representation when trained on *ImageNet-1k* (Russakovsky et al., 2015). Yet, it is worth noting that this is a significantly larger dataset with 1.3 M samples and 1,000 categories that occupy more of the model capacity. Finally, the most noticeable shift is seen in the model trained on the *Kaggle* dataset, although it originates from the same image domain.

In all cases, it appears that the shifts between filters of medical domains and others are insignificant and may even vanish when compared to more models.

3.9.4 Medical Imaging Models do not Learn special Filters

Taken together, the observed distribution outliers in Section 3.7 are directly caused by specific branches in architectures, such as `CompNet` (Dey & Hong, 2018). After all, it turns out that medical imaging models do not learn fundamentally different filter distributions than models of other image domains. This means that pre-training with diverse image data originating from arbitrary domains should accelerate the training of medical models (and others). Different levels of overparameterization may explain the remaining shifts to other domains and bear potential for future work, which may study the filter properties through our proposed filter metrics.

Additionally, we have observed that some layers learn highly specific clusters of filters. Others seem to develop extreme levels of sparsity. In those cases, it would be reasonable to reduce the search space during training and, in turn, accelerate training and/or reduce the amount of necessary training data.

Given the scarcity of medical data and its acquisition cost, exploring this direction in future work seems worthwhile.

3.10 Conclusions

Our results support our initial hypothesis that the distributions of trained convolutional filters are a suitable and easy-to-access proxy for investigating distribution shifts in the context of transfer learning and robustness.

One finding in our large-scale study is the presence of large amounts of degenerated filters in large, well-performing networks – resulting in the phenotypes *points*, *spikes*, and *symbols*. We postulate that their existence is a symptom in line with the *Lottery Ticket Hypothesis* (Frankle & Carbin, 2019), which suggests that neural networks benefit from overparameterization to increase the odds of finding a "good" subnetwork but not actually model the decision rule. Degenerated filters may be these "losing" subnetworks. Yet, more generally, this finding indicates that the models do not fully utilize their capacity, which we will further confirm through a different metric in Chapter 6.

Based on our metrics, we conclude that ideal models should have relatively high entropy throughout all layers (suggesting many diverse transformations) and almost no sparse filters. Models that show an increasing or generally high sparsity with depth are most likely overparameterized and could be pruned, which would benefit inference and training speed. It is also likely that, at least theoretically, it should be possible to learn smaller models reaching similar performance levels Frankle & Carbin (2019). Similarly, it is likely that we can compress layers with low entropy into more efficient counterparts. However, this would require a more sophisticated technique than zeroing out weights, as commonly done in pruning.

We can also identify randomly initialized but not yet trained models based on an overall high level of entropy throughout all layers and virtually no sparsity. This can also be an effect of the training task, *e.g.*, we have seen that *GAN-Discriminators* collapse back to their random state.

Another striking finding is the observation of very low global shifts in filter structure between different meta-groups: (1) shifts inside a family of architectures are very low; (2) shifts are mostly independent of the target image distribution and task; (3) also we observe rather small shifts between convolution layers of different depths with the highest shifts in the first and last layers.

Overall, the analysis of over 1.4 billion learned convolutional filters in the provided dataset gives a strong indication that the common practice of pre-training CNNs is indeed a sufficient approach if the chosen model is not heavily overparameterized. Moreover, our *CNN-Filter-DB* is a rich source for further research into transfer learning, robustness, pruning, and similar lines of research.

Limitations Our model zoo and, thus, data is biased by the large amount *classification* models and/or *natural* datasets such as *ImageNet-1k*. Further, some splits will over-represent specific dimensions *e.g.*, tasks may include exclusive visual categories and vice versa. Also, as previously shown, many of the collected models show a large amount of degenerated layers that impact the distributions. This also biases measurements of the distribution shifts. We performed an ablation study by removing filters extracted from degenerated layers but could not find a clear correlation between degeneration and distribution shifts, presumably due to a lack of justified thresholds.

Chapter 4

Convolution Filters under Adversarial Regularization

Deep learning models poorly generalize under distribution shifts of the input data (see Section 2.7). It is possible to adversarially cause a distribution shift by optimizing small, barely perceivable perturbations to the input data that force models to make wrong predictions with high confidence (see Section 2.8.2). The most common defense mechanism is regularization through adversarial training (Madry et al., 2018), which injects worst-case perturbations back into training to strengthen the decision boundaries and reduce overfitting to individual data points. In this context, we aim to understand how this form of training changes learned representations and perform an investigation of convolution filters based on the methodology introduced in Chapter 3. Specifically, we compare filters of adversarially-regularized models with normal ones.

This chapter is based on "Adversarial Robustness Through the Lens of Convolutional Filters", presented at the CVPR 2022 Workshop on The Art of Robustness: Devil and Angel in Adversarial Machine Learning (in-proceedings) (Gavrikov & Keuper, 2022a). As the first author, Paul Gavrikov collected and trained models, performed the experiments, and created the plots. The analyses were written under supervision and with the input of Janis Keuper.

Code: `https://github.com/paulgavrikov/cvpr22w_RobustnessThroughTheLens`

4.1 Introduction

Convolutional Neural Networks (CNNs) have been successfully applied to solve many different computer vision problems. As the state-of-the-art has been consequently pushed, research was mostly devoted to improving the performance (quality of predictions, inference speed, and others). However, recently, it has been shown that these models are sensitive to distribution shifts in image data. Even small, for humans almost imperceptible, perturbations applied to input images can force the networks to make high-confidence, false predictions on samples that would otherwise have been classified correctly (Szegedy et al., 2014b; Biggio et al., 2013).

Normal training of off-the-shelf architectures typically results in zero robustness against perturbed samples at test time. This raises the question of whether such highly vulnerable deep learning models should be used in safety-critical applications (Ma et al., 2021; Finlayson et al., 2019; Deng et al., 2020). Consequently, researchers have devoted their work to studying the sensitivity to distribution shifts, *e.g.*, by finding and understanding adversarial inputs (Goodfellow et al., 2015; Carlini & Wagner, 2016; Akhtar & Mian, 2018), and building defenses to those (Papernot et al., 2015; Goodfellow et al., 2015; Madry et al., 2018; Shafahi et al., 2019). A more detailed introduction to adversarial attacks and defenses can be found in Section 2.8.2.

While most explanatory methods study the distribution shifts in the input data and/or activations, we propose to extend our methodology introduced in Chapter 3 and evaluate differences in learned parameters, specifically convolutional filters and, therefore, round out previous findings through a different perspective. More precisely, we investigate shifts in the dominantly used 3 × 3 filters in CNN linear classification models trained on *CIFAR-10/100* (Krizhevsky et al., 2009) and *ImageNet-1k* (Russakovsky et al., 2015) datasets that were trained to withstand L^∞-bound adversarial attacks in comparison to normally trained models. Please find an introduction to linear classification in Section 2.5.1 and details about these datasets in Section 2.5.4.

We summarize our contributions and findings as follows:

- We collect 71 public robust models with 13 different architectures trained on 3 image datasets. These models contain a total of 615,863,744 filters with a size of 3 × 3. Additionally, the architectures used in robust models are trained from scratch without any robustness regularization.
- We show an in-depth empirical comparison of learned 3 × 3 convolution filters between robust and normal models. The resulting filter dataset is made available publicly.
- Our analysis shows that differences in filter structure increase with layer depth but significantly explode towards the end of the model, with a dominant outlier also showing in the primary convolution layer.
- We visualize the primary layer of L^∞-robust models and their activations and observe a large presence of thresholding filters that can remove perturbations from regions of interest.
- We discover that robust models appear to form more diverse, less sparse, and more orthogonal convolution filters. Ultimately, this finding is a necessary but not sufficient criterion for adversarial robustness.

4.2 Background

We refer the reader to Section 2.8.2 for an introduction to adversarial robustness, including a discussion of adversarial training.

4.3 Methodology

Filter quality We apply the methodology introduced in Section 3.4 and compute *sparsity* (Equation 3.4) and *variance entropy* (Equation 3.6). However, for this study, we compute the variance entropy over non-sparse filters to better decouple sparsity from pattern diversity. Recall that a variance entropy value of 0 indicates the homogeneity of present filters, as all of them can be reconstructed from one single principal pattern. In contrast, the maximum 0.954 (for 3 × 3 kernels) indicates a uniformly spread variance across all basis vectors, as found in random, non-initialized layers. Values close to both limits indicate a degeneration.

We additionally include a measurement of **orthogonality**, as it is a desirable property in convolutional weights (Brock et al., 2017, 2019), helping with gradient propagation and being coupled with the diversity of generated feature maps. For computational reasons, we measure the orthogonality between entire filterbanks (*i.e.*, stacks of filters) instead of individual filters. Formally, given a convolution layer weight $\mathbf{W} \in \mathbb{R}^{c_\text{out} \times c_\text{in} \times k \times k}$ with c_in input-channels, c_out output-channels, with a $k \times k$ kernel size (here: $k = 3$), we reshape the tensor into a matrix

$$\mathbf{W} \in \mathbb{R}^{c_\text{out} \times c_\text{in} \times k \times k} \to \boldsymbol{W} \in \mathbb{R}^{c_\text{out} \times (c_\text{in} \cdot k \cdot k)}. \tag{4.1}$$

Each filter in $\boldsymbol{W}$ is normalized to unit length. Then, we obtain the orthogonality by

$$1 - \frac{\|\boldsymbol{W}\boldsymbol{W}^\top - \boldsymbol{I}\|_1}{c_\text{out} \cdot (c_\text{out} - 1)}. \tag{4.2}$$

An orthogonality value of 1 stipulates the orthogonality of all filterbanks in a layer, whereas 0 indicates parallel filterbanks that produce perhaps differently scaled but otherwise identical feature maps.

Measuring distribution shifts We employ the same KL-based methodology introduced in Section 3.4, measuring the divergence between principal pattern distributions (Equation 3.9).

4.3.1 Model Zoos

Robust vs. normal models To compare filters of robust to normal models, we create a model zoo consisting of 71 robust model checkpoints (Rebuffi et al., 2021b; Huang et al., 2022; Zhang et al., 2020, 2019a; Hendrycks et al., 2019; Zhang et al., 2021b; Chen & Lee, 2021; Andriushchenko & Flammarion, 2020; Cui et al., 2021; Rice et al., 2020; Dai et al., 2021; Gowal et al., 2020; Sitawarin et al., 2021; Chen et al., 2021a; Zhang et al., 2019b; Wu et al., 2020; Wong et al., 2020; Huang et al., 2020; Carmon et al., 2022; Pang et al., 2020b; Gowal et al., 2021; Sehwag et al., 2020; Sridhar et al., 2021; Chen et al., 2020b; Sehwag et al., 2021; Addepalli et al., 2021; Ding et al., 2020; Rade & Moosavi-Dezfooli, 2021; Wang et al., 2020b; Engstrom et al., 2019) based on various network architectures from the *L^∞-RobustBench* leaderboard (Croce et al., 2021) for *CIFAR-10/100* (Krizhevsky et al., 2009), and *ImageNet-1k* (Russakovsky et al., 2015). We select benchmark entries on *RobustBench* that are bounded by a budget ϵ with $\epsilon = 8/255$ on *CIFAR-10/100*, and $\epsilon = 4/255$ on *ImageNet-1k* under L^∞-norm, respectively.

Additionally, we train a separate model for each utilized architecture *without* applying any robustness regularization and without using any external data, even if the robust counterpart relied on such data. We use the following hyperparameters: *CIFAR-10/100* data is zero-padded by 4 px along each dimension and then transformed using a 32×32 px random crops and random horizontal flips. We use an initial learning rate of 1e-8, a weight decay of 1e-2, a batch size of 256, and a Nesterov momentum of 0.9. Further, we use an SGD optimizer and decrease the learning rate every 30 epochs by a factor of 0.1 for a total training time of 125 epochs. The loss is determined using Categorical Cross Entropy.

Our training *ImageNet-1k* architectures with these parameters resulted in rather poor performance, and we, thus, replaced these models with pretrained *ImageNet-1k* models included from *Pytorch Image Models (timm)* (Wightman, 2019).

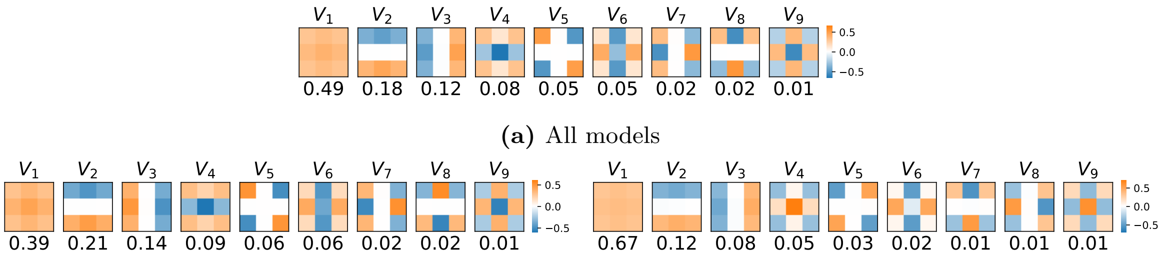
An overview of all models and their performance on normal and adversarial test data can be found in Appendix A.1. In Table 4.1, we show a small summary of key indicators.

Table 4.1: Model zoo performance overview. Comparison between average (*mean±std*) performance and parameter size on all evaluated datasets. "Clean Acc." refers to the regular validation accuracy, while "Robust Acc." refers to the robust accuracy as measured by *RobustBench*.

		Normal	**Robust**	
Dataset	**3×3 Filters** [M]	Clean Acc. [%]	Clean Acc. [%]	Robust Acc. [%]
CIFAR-10	9.1 ± 10.5	92.2 ± 4.2	86.9 ± 2.6	56.7 ± 5.9
CIFAR-100	9.3 ± 10.6	72.7 ± 8.5	62.2 ± 3.9	29.0 ± 3.9
ImageNet-1k	2.0 ± 1.7	78.5 ± 4.9	60.7 ± 6.3	30.8 ± 5.6

Models under increasing training budget Complementary to an absolute comparison between robust and normal models, we want to study how the adversarial budget ϵ influences the above observations by treating our observations as a function of ϵ. To this end, we collect `ResNet-18`, `ResNet-50` (He et al., 2016), and `WideResNet-50x2` (Zagoruyko & Komodakis, 2016) models trained on *ImageNet-1k* (Russakovsky et al., 2015) with adversarial training under increasing adversarial budget from Salman et al. (2020). Additionally, this allows us to compare models trained with L^∞-norm (ϵ from 0 to 8/255) to L^2-norm (ϵ from 0 to 5).

4.4 Analysis of Filters



(a) All models

(b) Robust models

(c) Normal models

Figure 4.1: Principal patterns. Filter basis and (cumulative) explained variance ratio per component (below) for filters from (a) all models, (b) robust (adversarially-trained) models, (c) normal models. Basis vectors are sorted by decreasing variance.

4.4.1 Principal Patterns

In the first step, we investigate the basis (principal patterns) forming the obtained filters. We, therefore, separate the filters extracted from all models into three filter sets: all filters, only filters from robust models, and only filters from normal models. Then, we apply a PCA to each set individually.

We observe that the basis vectors obtained from all three sets do not significantly differ (Figure 4.1). Changes only include minor fluctuations (note that basis vectors can be inverted, which is equivalent to inverting the coefficients). However, while 67% of the normal filter variance can be reconstructed from the first basis vector alone, robust models show a more uniform distribution of the variance, suggesting that these models form more structurally diverse filters.

4.4.2 Filter Pattern Distributions

In this section, we aim to understand the differences in the filter patterns. For this, we transform all collected filters to the common basis obtained in Section 4.4.1 and measure shifts between coefficients separated by dataset and regularization.

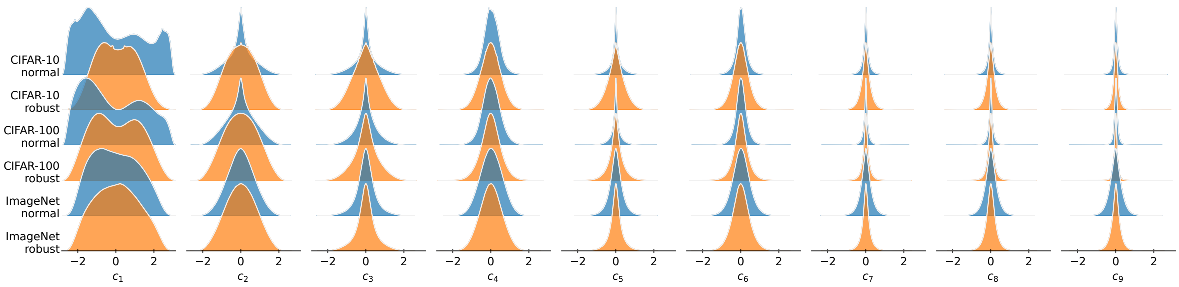


Figure 4.2: Distribution shifts between normal and robust models by training dataset. Coefficient distribution along every basis of ● normal models (blue) vs. ● robust (adversarially-trained) on different datasets. Shifts between robust and normal models appear to decrease with dataset complexity.

Shift by dataset The coefficient distributions (Figure 4.2) show clear shifts between robust and normal models, but this shift decreases with the increasing complexity of the dataset. We obtain a weighted KL-divergence of 0.55, 0.16, and 0.01 for *CIFAR-10/100*, and *ImageNet-1k* respectively. Interestingly, we also see a reduced shift for *CIFAR-100* compared to *CIFAR-10*, although it has the same amount of total training samples (but a different amount of samples per class). This suggests that more complex datasets lead to smaller distribution shifts between robust and normal models, with an emphasis on the fact that complexity does not only refer to the number of training samples.

It is worth noting that robust models achieve a significantly worse clean accuracy than their counterparts, and this performance gap increases with dataset complexity (Table 4.1). On average, robust accuracy is an additional 30% worse for all studied datasets. Further, *ImageNet-1k* models are trained and evaluated with a different ϵ, which may hide their true (non)-robustness.

The studied *ImageNet-1k* models, on average, only employ 2M 3×3 filters (plus a negligible amount of larger filters in the first layer), while the models on the arguably simpler datasets employ 9M on average. It is, therefore, likely that *CIFAR-10/100* populations show an increased effect of degeneration due to overparameterization (Section 3.5.2), which is not that extreme in *ImageNet-1k* models due to their smaller architectures.

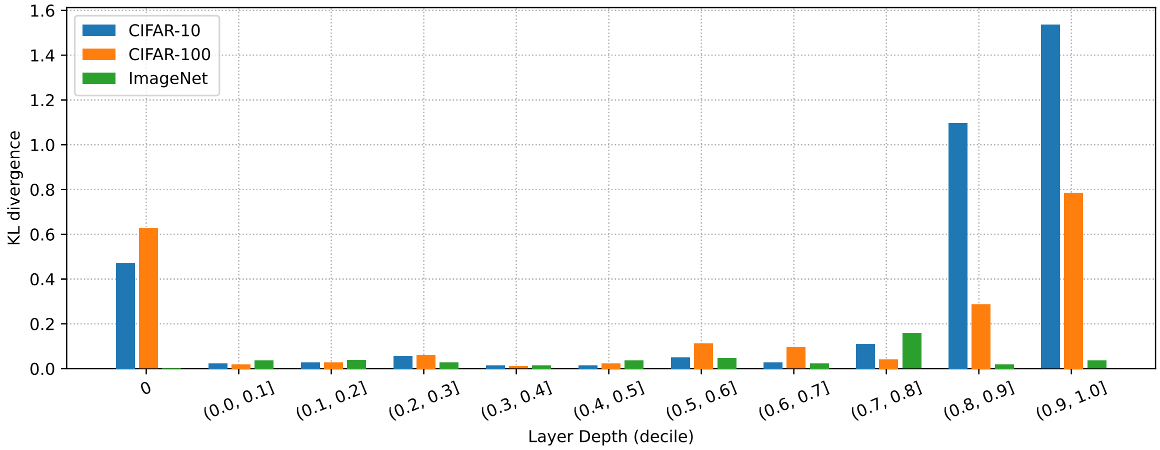
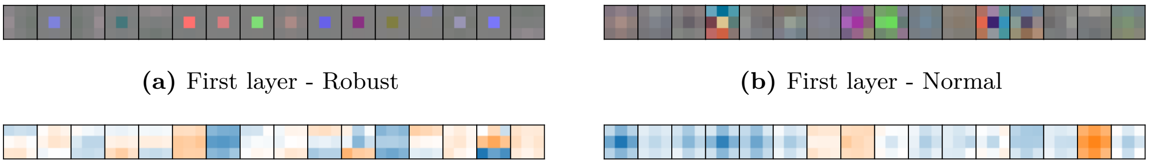


Figure 4.3: Divergence by depth. Divergence in filter patterns between learned 3×3 of robust and normal models by depth decile. The first convolutional layer is displayed separately (not shown for *ImageNet-1k*, as these models use larger kernels). The most significant shifts (large KL values) appear in the primary convolution layer and deeper stages.



(a) First layer - Robust

(b) First layer - Normal

(c) Last layer - Robust

(d) Last layer - Normal

Figure 4.4: Filter visualization. Visualization of convolution filters from both the first stage and last layer of a `WideResNet-34-10` trained with adversarial and normal training on *CIFAR-10*. (a) and (b) show filters from the first stage, while (c) and (d) show filter *kernels* from the last layer.

Shift by layer depth Following the previous observation, we investigate the most significant shifts in filter coefficients and measure the divergence at various stages of depth. To compare models with different depths, we group filter coefficients in deciles of their relative depth in the model. The obtained shifts (Figure 4.3) seem to increase with convolution depth and peak in the last 20% of the depth for *CIFAR-10/100*. For *ImageNet-1k*, the peak shift is measured in the 8th decile, whereas the shift in later stages is minimal. Aside from the shifts in later stages, for all datasets, there is a relatively low shift throughout the depth with the most salient outlier being seen in the very first convolution layer.[1] This outlier is indeed limited to the first layer, adding filters from the secondary layers vanishes the shift. Once again, the maximum shift appears to decrease with dataset complexity.

We visualize the first and last convolution layers To better understand the cause of the observed distribution shifts. The primary convolution stage (Figures 4.4a and 4.4b) shows a striking difference: Normal models show an expected (Zeiler & Fergus, 2013; Yosinski et al., 2014) diverse set of various filters including Gabor filters and color blobs, yet, almost all robust models collapse to a single filter pattern where only one weight is non-null (which is typically in the filter center) – we call these "thresholding" filters and will discuss the function in Section 4.4.3.

For the deepest convolution layers (Figures 4.4c and 4.4d), we observe the opposite: normal filters show a clear lack of diversity, and mostly remind of Gaussian blur filters, while adversarially-trained filters appear to be more diverse in patterns and are more likely to perform complex transformations. Contrary to the distinct primary layer, this observation is visible across multiple deeper layers.

4.4.3 The First Layer of Robust Models

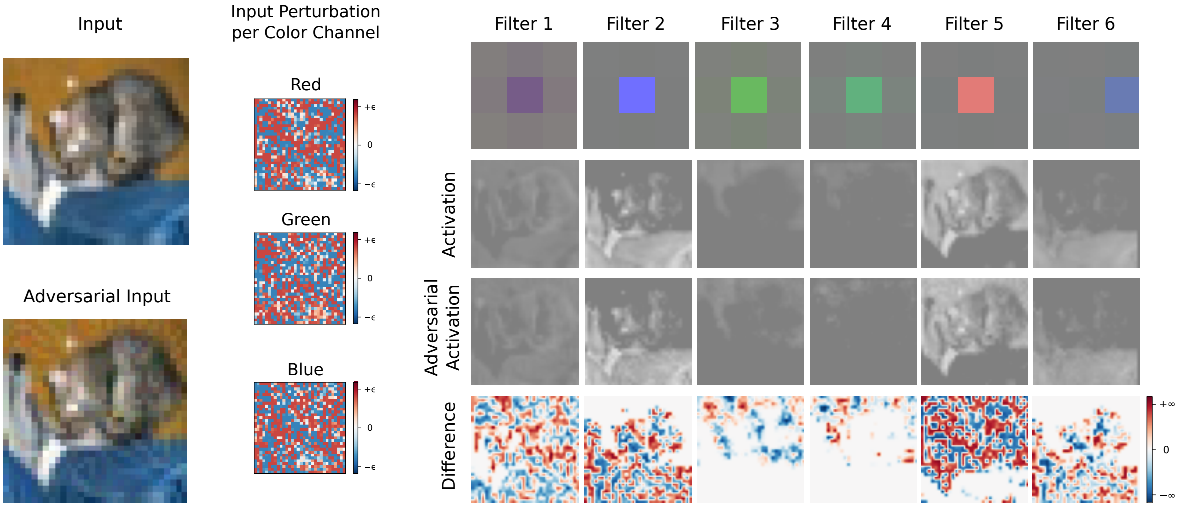


Figure 4.5: Thresholding filters in L^∞-robust models. The thresholding filters clip away certain regions of the input and effectively remove perturbations contained within these (white areas) – other regions still contain perturbations (blue and red areas).

Our analysis of the first layer of robust models reveals a prevalence of "thresholding" filters, characterized by a single non-zero weight.[2] These filters effectively compute a linear combination of the input channels, analogous to a 1×1 convolution if the weight is centered or a shifted linear combination if the weight is off-center. Notably, while typical filters in first layers operate across multiple input channels (Yosinski et al., 2014), we find that these thresholding filters generally focus on a single color channel. When combined with the commonly used ReLU activations (and their derivatives, see Section 2.2), these filters can be understood to perform a thresholding operation on the selected input channel.

Recall that L^∞ attacks modify each pixel within the range $[-\epsilon, +\epsilon]$. The filter implements an affine transformation, which may push some pixels into the negative range. The consecutive activation will remove these negative parts of the feature map (the threshold) – this includes perturbations located in there. Of course, this comes at the potential cost of discarding relevant scene information, as the global

[1] The primary convolution stages of *ImageNet-1k*-models use larger kernel sizes and are therefore not included here, but we show and discuss them in Section 4.4.5.

[2] This is idealized. In practice, the other weights may not be exactly zero, but there will be one pixel that accounts for the majority of magnitude in the kernel.

threshold is similarly applied to all pixels per channel, thus leading to either insufficient suppression of adversarial perturbations or excessive removal of genuine signal.

We show the effect of thresholding filters in Figure 4.5, where we compute the activations for a normal and adversarial input sample on `ResNet-18` (Addepalli et al., 2021) after being processed by a thresholding filter, including normalization and activation by a ReLU function. For some regions in the activations, the difference between normal and adversarial inputs is zero, showing that these filters have successfully removed perturbations in some regions. While this may remove entire objects from the scene, it often leaves their shape (*e.g.*, Filter 4 removes the cat and leg). In Chapter 7, we will show that adversarially-robust models tend to be more shape-biased, and these filters may be partially responsible for that.

It is worth noting that this mechanism is highly specific to the L^∞ norm, where the perturbation will be (mostly) spatially uniformly distributed. As we will later demonstrate in Section 4.4.5, training with the L^2 norm does indeed not result in the learning of such filters.

4.4.4 Filter Quality

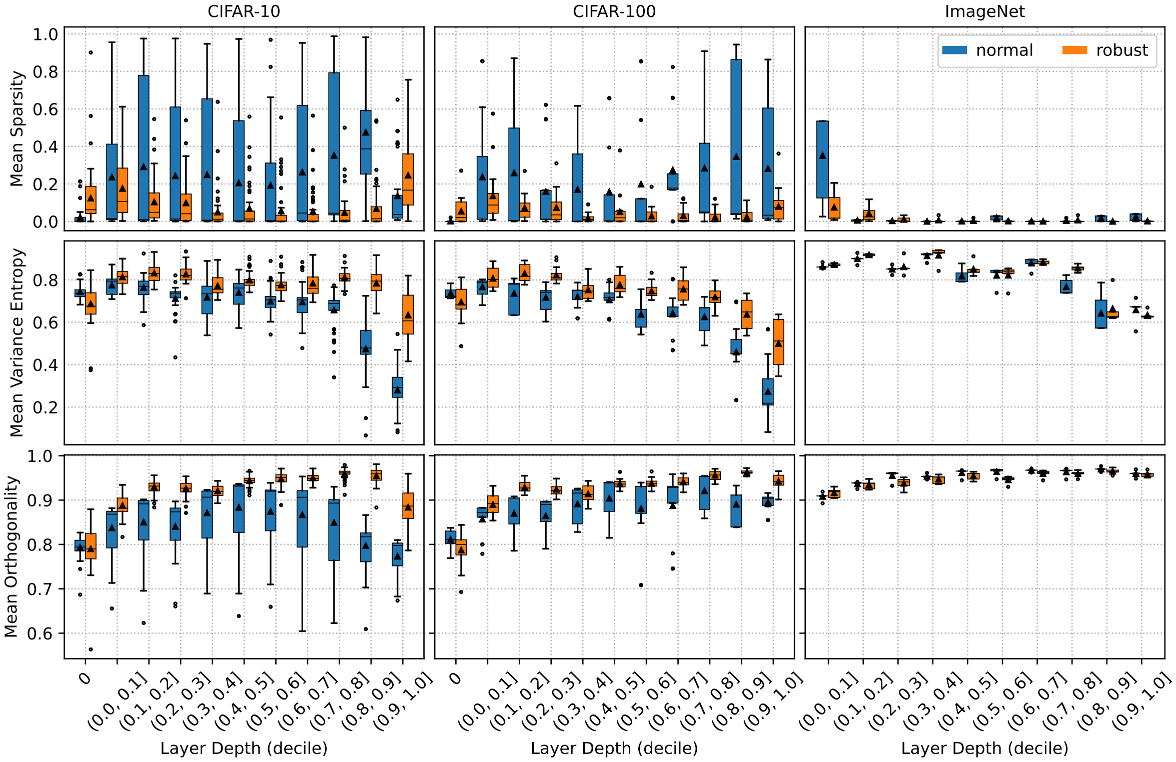


Figure 4.6: Filter quality by depth. Distribution of filter quality comparison by depth measured via *sparsity* (top), *variance entropy* (center), and *orthogonality* (bottom) between normal and adversarial-training for *CIFAR-10* (left), *CIFAR-100* (center), *ImageNet-1k* (right) datasets.

While the earlier analysis focused on distribution shifts in filter patterns, this section focuses on the related quality aspect of filters. In particular, we measure the amount of contributing filters through *sparsity* (Equation 3.4); the diversity of filters through *variance entropy* (Equation 3.6); and the expected redundancy of filterbanks through *orthogonality* (Equation 4.2). Similarly to the findings in patterns, we observe fewer differences in quality with dataset complexity (Figure 4.6), but also a general increase in quality for both robust and normal models. The results on *ImageNet-1k* are less conclusive due to a near-optimal baseline and a low sample size.

Sparsity We observe a very high span of sparsity across all layers for normal models that decreases with dataset complexity. Robust training significantly further minimizes sparsity and its span across all depths. Notable outliers include the primary stages, as well as the deepest convolution layers for *CIFAR-10*. Generally, sparsity seems to be lower in the middle stages.

Variance entropy The average variance entropy is relatively constant throughout the model but decreases with deeper layers. The entropy of robust models starts to decrease later and less significantly, but the difference between them diminishes with dataset complexity. Compared to *CIFAR-10*, robust *CIFAR-100* models show a lower entropy in deeper layers, while there is no clear difference between normal models. *ImageNet-1k* models show a higher entropy across all depths.

Orthogonality Robust models show an almost monotonic increase in orthogonality with depth, except for the last decile, whereas normal models eventually begin to decrease in orthogonality. Again, the differences diminish with dataset complexity and the span in obtained measurements of non-robust models is crucially increased.

4.4.5 The Role of Adversarial Attack Strength (Budget)

In the preceding analyses, we compared the convolution filters of normal and robust models of our main model zoo. Now we will change and treat our findings as a function of ϵ, and additionally test L^2-norm based regularization on our second model zoo.

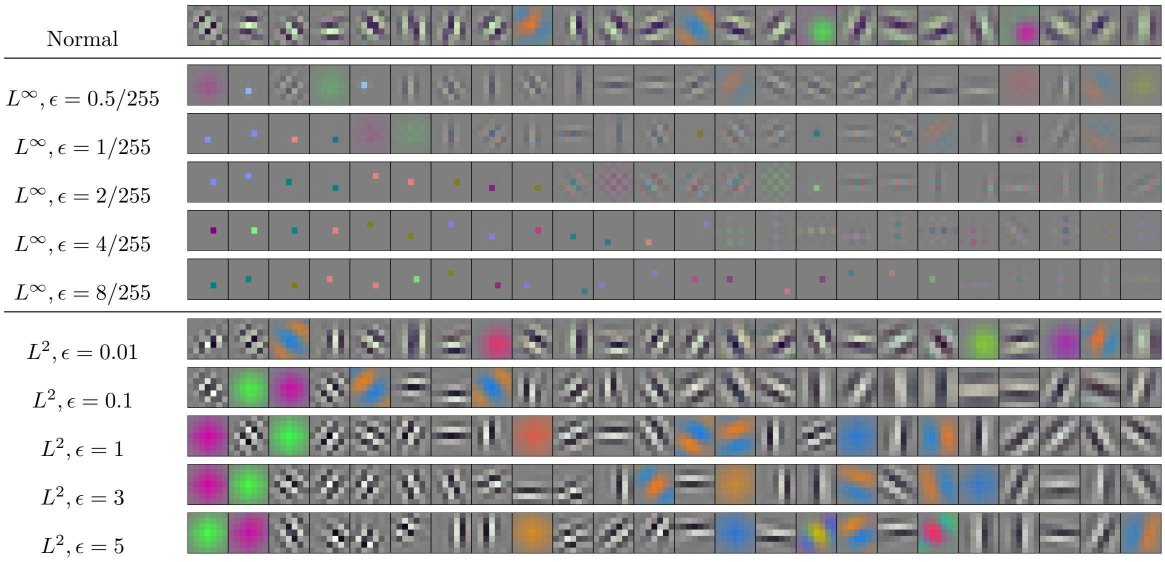

Normal
$L^\infty, \epsilon = 0.5/255$
$L^\infty, \epsilon = 1/255$
$L^\infty, \epsilon = 2/255$
$L^\infty, \epsilon = 4/255$
$L^\infty, \epsilon = 8/255$
$L^2, \epsilon = 0.01$
$L^2, \epsilon = 0.1$
$L^2, \epsilon = 1$
$L^2, \epsilon = 3$
$L^2, \epsilon = 5$

Figure 4.7: First convolutional layer filters in `ResNet-50` models trained under increasing adversarial attack budgets (ϵ) and norms. Each row corresponds to a specific attack strength ϵ, highlighting the 24 filters with the highest magnitude in the layer (based on their L^2-norm). Models were trained using adversarial training on *ImageNet-1k*.

First layer We start by analyzing the evolution of the first layer of `ResNet-50` under increasing attack strength in L^∞ training, shown in Figure 4.7. Unlike in the previous sections, we now study larger kernels, as the *ImageNet* models use a 7×7 convolution in the first stem. This further serves as an opportunity to understand if our previous findings scale to larger kernel size. We find many similarities of 7×7 filters in robust models in comparison to the ones we have observed in models with 3×3– Robust models still appear to form thresholding filters independent of the kernel size. Increasing the adversarial attack strength in training interpolates smoothly between the excepted "natural" filters and these thresholding filters. However, for the large kernels, these can be off-center in any direction more often, resulting in shifted feature maps.

Furthermore, at the largest budget ($\epsilon = 8/255$), we still notice some other filter patterns that remind us of dilated kernels. We assume that these are residues from former color blobs, and we suspect that training with even larger ϵ would fully eliminate these patterns in favor of thresholding filters. For example, the reader may trace the evolution of the green color blob ($\epsilon = 0$) in Figure 4.7. The filter first decreases in the magnitude of the green channel ($\epsilon = 0.5/255$ or $1/255$), then becomes grid-like ($\epsilon = 2/255$) before "dilating" ($\epsilon = 4/255$), and eventually becoming a thresholding filter at $\epsilon = 8/255$ (or significantly losing magnitude). A similar effect can be observed for the purple color blob.

When comparing the first layer of L^∞ to L^2 models, we find a strong difference. L^2 models do not form thresholding filters but instead strengthen the patterns that normal training yields, which can be seen through similar brightness. Overall, we observe that the magnitude distribution of the filters becomes more uniform under increasing L^2-training, while the opposite holds for L^∞.

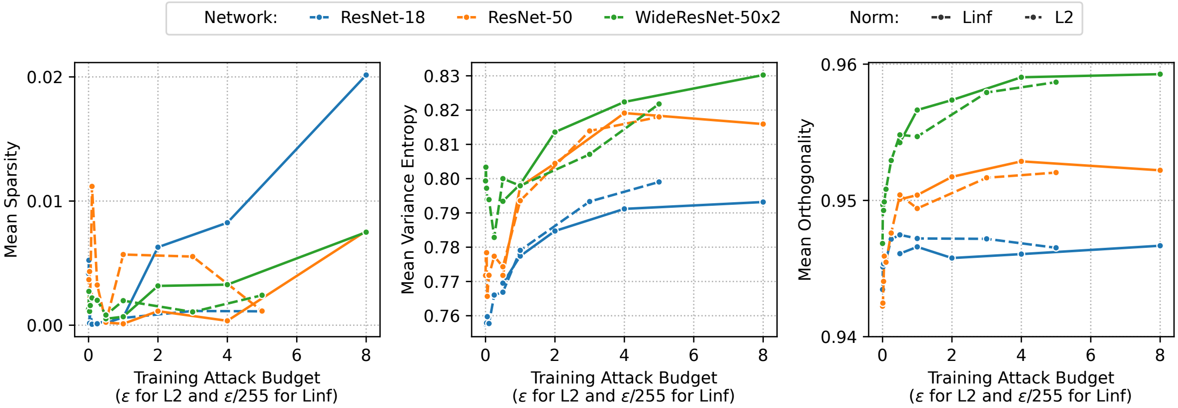


Figure 4.8: Average convolutional layer quality models trained under increasing adversarial attack budgets (ϵ) on *ImageNet*. We measure *sparsity*, *variance entropy*, and *orthogonality* for all layers with 3×3 kernels and report the average for each architecture and value for ϵ. We distinguish between L^∞- and L^2-bounded training.

Quality Next we apply our quality metrics to the model zoo and visualize the results in Figure 4.8. For comparison, we report the quality metrics as averages over all 3×3 convolution layers. Our trends show that on all architectures, all three metrics increase with ϵ. Orthogonality appears to quickly saturate, and generally, there are numerically smaller differences between robust and normal models. Larger models show the highest orthogonality, which may simply be due to the larger number of filters per layer. Variance entropy, on the other hand, does not appear to be saturated and is numerically more pronounced in robust models. Once again, larger models show higher levels of variance entropy, *i.e.*, their filters are more diverse. Surprisingly, sparsity appears to increase as well under L^∞-training – especially in the smaller `ResNet-18` model.

Unlike our findings regarding the first layer, there seems to be no significant difference between L^∞- and L^2-bounded training for filter quality beyond the first layer – *i.e.*, we obtain similar trends for both, except for sparsity, which does not reveal a clear trend. However, it is worth noting that in all cases, the average sparsity is very low compared to the measurements we have obtained on *CIFAR-10/100*, remaining at 2% in the worst-case.

4.5 Is Filter Pattern Diversity Sufficient for Robustness?

In the previous section, we have shown that there is a correlation between robustness and filter quality, specifically diversity, as measured through variance entropy. It remains to be shown if this correlation is a sufficient criterion for robustness – *i.e.*, will models become robust if we increase the diversity of filters without other forms of adversarial regularization? In light of this question, we propose a kernel pattern diversity regularization during training. Then, we measure the correlation between the diversity in filter patterns and the robust accuracy of the resulting models.

We construct the penalty term as follows:

$$\Omega_{\mathrm{VE}}(\boldsymbol{\theta}) = \mathbb{E}_l \left[\frac{\mathrm{VarianceEntropy}\left(\mathbf{W}^{(l)}\right)}{T_H\left(c_{\mathrm{out}}^{(l)} c_{\mathrm{in}}^{(l)}\right)} \right], \tag{4.3}$$

where $\mathbf{W}^{(l)}$ is the weight of the l-th convolution layer with a kernel size of 3×3, VarianceEntropy is the variance entropy defined in Equation 3.6, T_H is the randomness threshold defined in Equation 3.7, and $c_{\mathrm{out}}^{(l)} c_{\mathrm{in}}^{(l)}$ is the number of kernels in the l-th layer.

Exemplarily, we test this regularization on a `ResNet-20-64` (He et al., 2016) with an increased width of 64 (to accommodate the additional complexity) trained on *CIFAR-10*. Input samples are zero-padded by 4 px along each dimension and then transformed using 32 × 32 px random crops and random horizontal flips. We use an initial learning rate of 1e-2, a weight decay of 1e-2, a batch size of 256, and a Nesterov momentum of 0.9. Further, we use an SGD optimizer and a cosine annealing schedule (Loshchilov & Hutter, 2017) for a total training time of 75 epochs. The final loss $J(x, y; \boldsymbol{\theta}) = \ell(f(x; \boldsymbol{\theta}), y) + \lambda \cdot \Omega_{\mathrm{VE}}(\boldsymbol{\theta})$ uses Categorical Cross Entropy ℓ plus a scaled penalty term with label smoothing of 0.1. We vary the regularization scale λ in the $[-100, 100]$ range to model both penalization of diversity and conformity.

We correlate the average variance entropy over all layers in the resulting models against accuracy under *AutoAttack* at $\epsilon = 1/255, L^{\infty}$. Our ϵ is chosen smaller than the regular value of 8/255 in *RobustBench* because we expect that a simple regularization may not offer the same kind of robustness as explicit training against the threat model would.

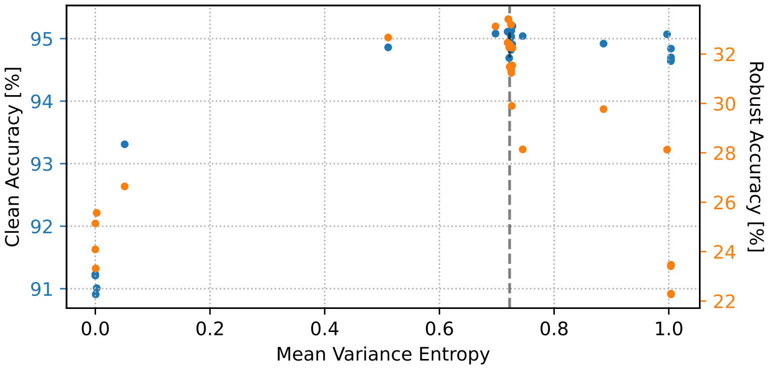


Figure 4.9: Variance entropy vs. clean and robust accuracy in variance entropy regularized models. We train `ResNet-20-64` models on *CIFAR-10* under different levels of kernel pattern regularization.

Surprisingly, our results in Figure 4.9 show that even at both extremes of regularization, models do learn an useful representation and achieve a highly non-trivial accuracy. When penalizing filter conformity, we reach an average variance entropy of 1.00, suggesting random filter patterns. The accuracy is still reaching 94.7% on the clean data and 24.1% against the adversarially perturbed data. When penalizing filter diversity, we achieve a mean variance entropy of just 4e-7, suggesting that almost all kernel patterns have collapsed to one pattern per layer. The clean test accuracy at 91.2% (23.5%) is quite impaired compared to the baseline, but still an impressive performance given the extremely limited representational capability.

We obtain our most robust models at $\lambda = -0.001$, improving the robust accuracy by 1.9% against the baseline. However, this result is quite low, given that we are testing against a low ϵ, and our results are just obtained from a single run, which can be noisy on low-resolution datasets (Picard, 2023). The average variance entropy is marginally different from the baseline. Overall, we do see that robust accuracy is best around the baseline diversity, towards both extremes of filter diversity it decreases – normal accuracy, on the other hand, does only seem to deteriorate significantly in the direction of filter pattern collapse, and stays relatively stable with increasing randomness.

Taken together, our results show that filter pattern diversity, as captured through our variance entropy metric, is only a necessary but not sufficient criterion to achieve adversarial robustness.

4.6 Conclusion

We have shown that adversarially-regularized models appear to learn a particularly more diverse, less redundant, and less sparse set of convolution filters than their non-regularized variants do. We assume that the increase in quality is a response to the additional training strain, as the more challenging adversarial problem occupies more of the available model capacity that would otherwise be degenerated. Indeed, we will obtain similar findings through the *layer criticality* metric in Chapter 6.

Frankle & Carbin (2019) presented the *Lottery Ticket Hypothesis*, claiming that neural networks (under normal training) form various redundant subnetworks that each increase the odds of finding a solution. Once a solution is found, the “losing” subnetworks can be removed without any significant impacts on accuracy. Adversarial samples activate channels of the feature extractor more uniformly and with larger magnitudes than normal ones (Bai et al., 2021). A straightforward solution to increase robustness is suppressing channels (Bai et al., 2021) or enhancing subnetworks (Guo et al., 2022a) leading to these channels in order to boost robustness. We hypothesize that these findings correlate with filter quality and, as in Chapter 3, degenerated filters in normal models may “losing” subnetworks that are activated by adversarial attacks. This is backed by our observations that filter quality seems to be a necessary criterion to achieve robustness. Specifically, robust models increase in filter diversity, suggesting overall stronger subnetworks,

However, although the filter quality of normally trained *ImageNet-1k* models is exceptionally high, their robustness is not. Taken together with our pattern diversity regularization experiments, this further confirms that filter quality alone is not a sufficient criterion to establish robustness.

Chapter 5

Filter Frequency Regularization

Regularization through adversarial training (Madry et al., 2018) improves model generalization to some extent. However, it is only one ingredient towards generally more robust models and requires knowledge about the potential threat model at training time. This chapter focuses on how we can achieve *native robustness* of models – *i.e.*, we want to learn robust behavior directly from conventional training data without out-of-distribution examples. To this end, we extend our filter studies from Chapters 3 and 4 to the frequency spectrum of learned convolution filters, showing that adversarially-trained models learn more low-frequent filters and then propose a simple regularization term that mimics this behavior in standard training. We obtain models with improved generalization that are not confined to highly specialized improvements in adversarial robustness or at the cost of other forms of robustness.

This chapter is based on "Improving Native CNN Robustness with Filter Frequency Regularization", presented at TMLR, 2023 (Lukasik et al., 2023). As joint first authors, Paul Gavrikov and Jovita Lukasik developed the code base, performed the experiments, created the plots, and wrote the paper under supervision and with input from Janis Keuper and Margret Keuper. Jovita Lukasik designed and performed the ablations in Section 5.9.

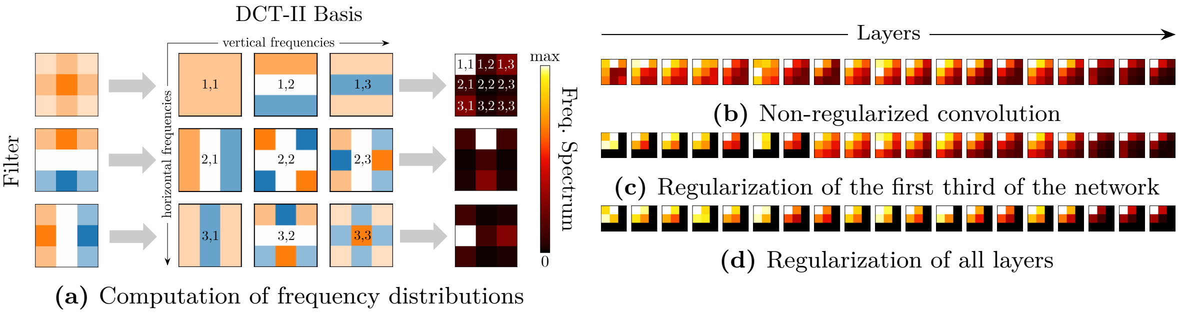
Code: `https://github.com/jovitalukasik/filter_freq_reg`

5.1 Introduction

Modern convolutional neural networks (CNNs) (He et al., 2016; Tan & Le, 2020; Liu et al., 2022) show a steady increase in performance in terms of test accuracy on a wide range of learning tasks. Yet, most models suffer from a low generalization ability, even when faced with small distribution shifts (see Section 2.7 and Section 2.8 for examples).

To improve the low generalization ability, previous work focused on aspects such as aliasing (Zhang, 2019; Zou et al., 2020; Li et al., 2020; Grabinski et al., 2022d,c), the padding operations (Gavrikov & Keuper, 2023a), the training schedule (Lopes et al., 2020; Saikia et al., 2021), analyzing the image feature spectrum (Geirhos et al., 2019; Wang et al., 2020a), or the activation of subnetworks (Bai et al., 2021; Guo et al., 2022a). In addition, introducing perturbed images into the training data, known as adversarial training (AT) (Madry et al., 2018), can alleviate low generalization to some extent. The latter is a form of regularization (see Section 2.6.1) and remains one of the most effective methods to obtain adversarial robust models.

However, AT is not the cure-all to improve network robustness and tends to overfit on training attacks (Tramèr & Boneh, 2019; Rice et al., 2020; Yu et al., 2022a). Intuitively, the adversarial attack used during training becomes an in-distribution sample of the model, while its robustness to new out-of-distribution samples (*e.g.*, a different adversarial attack) is hard to anticipate. Saikia et al. (2021); Kireev et al. (2022) show that AT can even increase the mean corruption error on



(a) Computation of frequency distributions

(b) Non-regularized convolution

(c) Regularization of the first third of the network

(d) Regularization of all layers

Figure 5.1: Study overview. Our proposed regularization decreases the magnitude of high-frequency DCT-II coefficients, as visualized in (b),(c),(d) for a `ResNet-20` trained on *CIFAR-10*. The magnitude of DCT-II coefficients, computed as shown in (a), represent the frequency distribution in the respective convolution layer.

ImageNet-C (Hendrycks & Dietterich, 2019). Therefore, we argue that AT can only be one ingredient towards building more robust models, while the main focus should rather be to encourage behavior that we call *native robustness*. We expect from natively robust models that they can learn robust behavior directly from the conventional training data.

Robust behavior includes, on the one hand, a certain degree of adversarial robustness without being confronted with adversarial attacks during training, *i.e.*, the model should not easily be fooled using attacks with very small perturbation budgets. Similarly, they should be robust against other perturbations such as common corruptions (Hendrycks & Dietterich, 2019) as long as the severity of the corruption is low. On the other hand, robust behavior implies a better alignment with human perception, *i.e.*, models should decide for a specific class more by the shape of an object than by its texture (Geirhos et al., 2019). Note that the expected degree of *specific* robustness can not be compared to the one obtained by techniques that specifically optimize for them, such as adversarial training. For instance, adversarial samples remain out-of-distribution samples for such natively robust models. Yet, additionally, training these natively robust models with AT should be complementary and have a further beneficial effect.

In this chapter, we propose a new perspective on improving native robustness by investigating the frequencies in the learned network filters directly – extending our studies in Chapters 3 and 4 from spatial to frequency space. Specifically, we propose to project CNN convolution filter weights into the frequency domain by applying a discrete cosine transformation (DCT-II).[1] Although the resulting formulation is, in principle, equivalent to the commonly adopted CNN formulation, it provides direct access to the learned filter frequencies. Thereby, we aim to investigate the following research questions:

(1) Which filter frequencies are predominantly learned in the layers of CNNs?

(2) Can we regularize the frequencies during the training process, such as to increase the native robustness of the learned model?

We investigate these questions in the context of image classification (as introduced in Section 2.5) – yet our approach bears the potential to be expanded to other tasks, such as object detection and segmentation. First, we analyze the learned filter frequencies of modern CNNs and observe that they tend to have a low-frequency bias in deep layers, while filters of earlier layers of the network are either uniformly distributed in frequency space or even biased towards higher frequencies. In the latter cases, the convolution thus relies on high-frequency information. On the contrary, adversarial training appears to shift the focus to low filter frequencies in early layers. To leverage this behavior, we introduce a regularization scheme (see Section 2.6.1 for other forms of regularization and an introduction), which increases the bias to low-frequencies in these early layers (see Figure 5.1 for a visualization). We

[1] An introduction to the related discrete Fourier transform can be found in Section 2.1.

evaluate the proposed decomposition and regularization on different CNNs under covariate shifts in test data (Section 2.7). Results on *CIFAR-10, CIFAR-100* (Krizhevsky et al., 2009), *SVHN* (Netzer et al., 2011), *MNIST* (LeCun et al., 1998b), *TinyImageNet* (Le & Yang, 2015), and *ImageNet-1k* (Deng et al., 2009; Russakovsky et al., 2015) show increased native robustness.

We summarize our contributions as follows:

- We expand our filter analysis of Chapters 3 and 4 to the frequency domain and observe that adversarial training results in a shift towards a low-frequency bias in the filter weights of early layers during the early phases of training (Section 5.3).
- Based on this observation, we propose a high-frequency penalization term in the weight space of convolution layers (Section 5.4) to mitigate the reliance on high-frequency information.
- Networks trained with this regularization become gradually, yet consistently, more robust against a wide array of out-of-distribution generalization tasks without reliance on AT or additional data - *i.e.*, networks increase their native robustness (Section 5.5). Additional AT is complementary and further improves the measurable adversarial robustness to a variety of attacks (Section 5.8).

5.2 Background

Adversarial robustness We refer the reader to Section 2.8.2 for an introduction to adversarial robustness, including a discussion of adversarial training.

Robustness beyond adversarial attacks Unfortunately, the existence of adversarial attacks is only a symptom of larger generalization issues of neural networks. For example, neural networks fail to generalize under various corruptions such as weather conditions, changes in lighting, noise, and blurring (Dodge & Karam, 2017; Hendrycks & Dietterich, 2019). We refer the reader to Section 2.8 for a more detailed overview of robustness and provide only the details important to this chapter here.

For fast and comparable benchmarks, the test datasets *CIFAR-10-C*, *CIFAR-100-C*, and *ImageNet-C* have been proposed (Hendrycks & Dietterich, 2019), which include 15 (+4 extra) types of "common" corruptions (CC) at increasing severity level (from 1 to 5).

Additionally, Geirhos et al. (2019) observed that CNNs are biased towards detecting textures of an image instead of the shape, which is in contrast to human vision behavior that focuses on shape information, *i.e.*, a dominant shape bias. To overcome this texture bias, they train on a stylized version of *ImageNet* to increase the shape bias of CNNs and hypothesize that this improves robustness. For fast evaluation of out-of-distribution (OOD) generalization Geirhos et al. (2021) proposed a benchmark including 17 OOD datasets, from which 12 contain image perturbations and the other 5 are single manipulations of *ImageNet* (Deng et al., 2009): *texture/shape cue-conflict*, *sketches* (Wang et al., 2019), *stylized (ImageNet)*, *edges*, and *silhouettes*.

Robustness from a frequency perspective Recent work highlighted the importance of learned frequencies for model robustness and generalization. Wang et al. (2020a) demonstrated that CNNs significantly rely on high-frequency information for their predictions. On the other hand, AT models predominantly classify based on low-frequency information. Given that texture information typically resides in higher frequency bands, this is a suitable explanation for the observations by Geirhos et al. (2019). As such, there is also a correlation between AT and a reduced texture bias (Geirhos et al., 2021) (which we will also examine in Chapter 7). Duan et al. (2021) exploit these findings by proposing an adversarial attack that drops DCT coefficients corresponding to high frequencies from inputs to fool neural networks. Yet, despite the common assumption, adversarial attacks are not always targeting high-frequencies, and the behavior depends on the dataset (Maiya et al., 2021; Abello et al., 2021; Bernhard et al., 2021; Ortiz-Jiménez et al., 2020).

Multiple works explore the desensitization of neural networks to high-frequency (HF) from various angles to avoid AT: Lopes et al. (2020) randomly add noise to image patches, Saikia et al. (2021) regularize the feature maps produced by convolution layers in a dedicated two-stream architecture, and Grabinski et al. (2022c) introduce a downsampling approach within the frequency domain that removes aliasing-related high-frequency information. In contrast, we regularize high-frequency information directly in convolution filters to improve the native robustness and OOD generalization of the model. This differs from traditional adversarial training but can also be used to improve robust accuracy in the context of adversarial training and mitigate robust overfitting.

Our regularization differs from previous works in the following way:

- **Regularization vs. band-pass filtering:** Previous methods often low-pass-filter signals, which leads to a hard smoothing of the resulting feature maps and the active deletion of information that may be necessary for predictions – especially in fine-grained classification problems. Instead, we only regularize the attenuation and, thus, effectively force the network to reweigh information without having to discard information.
- **Data-independent and explicit attenuation:** By regularizing weights, we induce an explicit causal bias in the operator. Alternatively, an attenuation of feature maps would be implicit and would highly depend on the frequency distribution of the inputs. Additionally, attenuating the filters results in a local suppression (*i.e.*, in the patch) of HF, while a (global) feature-map regularization would affect the entire scene.

Basis decomposition In our study, we will decompose filters from spatial to the frequency domain, analogous to our introduction to Fourier transform in Section 2.1. The decomposition of convolution filters is typically studied in the context of compression, *e.g.*, see Yaroslavsky (2014) for an overview. The majority of decomposition approaches convert the convolution layer weights to the frequency domain, *e.g.*, by utilizing the DCT-II-basis (Chen, 2004; Chen et al., 2016; Lo & Hang, 2019; Cheinski & Wawrzynski, 2020; Chen et al., 2022b; Ulicny et al., 2022) to prune and compress the number of frequency components. But works also exist that transform the input images directly for better performance and generalization (Xu et al., 2020; Hossain et al., 2019). In detail, the *discrete cosine transform* (DCT) (Ahmed et al., 1974) maps an input signal into a frequency domain represented by cosine basis functions. In particular, the common DCT-II variant is used in JPEG compression, where it successfully compresses natural images (Wallace, 1992). These works mainly explore the fact that data of multiple domains is not uniformly distributed in the frequency domain and is typically biased towards low frequencies (Singh & Theunissen, 2004; Ruderman, 1994). In Chapters 3 and 4, we showed the basis (principal patterns) of convolution filter kernels obtained via SVD is often highly similar and independent of the architecture, learned task, or dataset. These identified principal patterns have a striking similarity to the DCT-II basis.

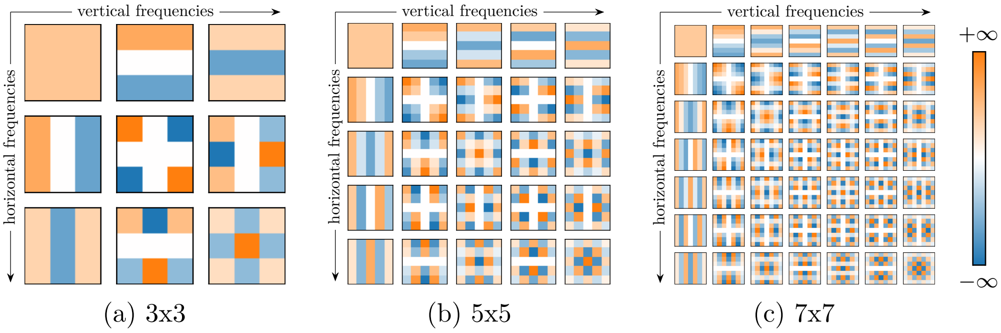
Our realization of the DCT-II basis is similar to Ulicny et al. (2022) and other previous work, however, instead of compression, we explore an orthogonal direction and study the role of individual frequencies in training and apply regularization in the frequency space to improve generalization. DCT merely serves as a tool in our study and could be replaced by any other basis with an inherent frequency hierarchy (including DFT Section 2.1).

5.3 Frequency Analysis

In this initial analysis, we transform learned convolution filters to the frequency domain (see Section 2.1 for the related Fourier transform). We implement this by changing the basis of convolution weights to DCT-II, revealing the coefficients and, therefore, frequency information. Formally, we define this as follows. Let $\mathbf{V} \in \mathbb{R}^{k \times k \times k \times k}$ denote the $k \times k$-DCT-II basis. Then every basis vector $\mathbf{V}_{i,j}$ with horizontal frequency j and vertical frequency i is defined as:

$$\mathbf{V}_{i,j,m,n} = \cos\left[\frac{\pi i}{k}\left(m + \frac{1}{2}\right)\right]\cos\left[\frac{\pi j}{k}\left(n + \frac{1}{2}\right)\right] \text{ for } i, j, m, n \in \{1, \ldots, k\}. \tag{5.1}$$

Every basis vector is additionally normalized to its L^1 length: $\mathbf{V}_{i,j} = \mathbf{V}_{i,j}/\|\mathbf{V}_{i,j}\|_1$. We show the DCT-II basis vectors for different kernel sizes k in Figure 5.2. Following the basis change, we visualize the average magnitude of coefficients in every convolution layer by heat maps (as shown in Figure 5.1). Having the frequency information at hand, we can directly analyze its distribution in common CNNs.



(a) 3x3 (b) 5x5 (c) 7x7

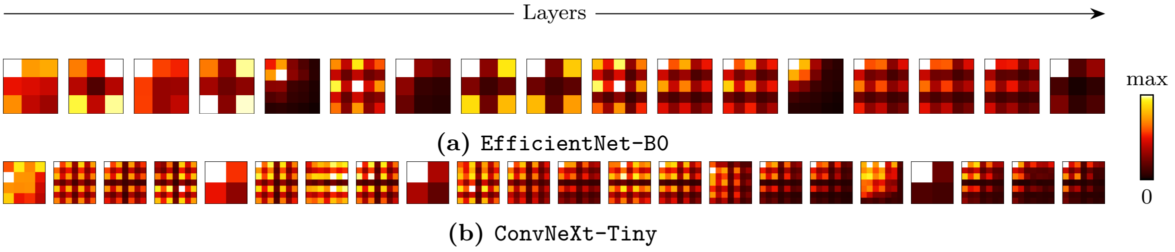
Figure 5.2: The full DCT-II basis for different resolutions.

In principle, DCT-II could be replaced by any other frequency base, such as a discrete Fourier or sine transform. We have chosen DCT-II, as this base seems to best match the principal patterns identified in Chapters 3 and 4.

5.3.1 Analysis of Learned Convolution Filters

We start by analyzing two modern networks trained on *ImageNet* without any robustness optimization techniques: `EfficientNet-B0` (Tan & Le, 2020) and `ConvNeXt-Tiny` (Liu et al., 2022). Instead of analyzing the spatial patterns as done in Chapters 3 and 4, we now focus on their representations in frequency space. Our visualizations in Figure 5.3 show that these CNNs do not always learn a uniform frequency spectrum utilization throughout the network. Earlier layers show a more uniform distribution of magnitude or are biased towards higher frequencies. However, deeper convolution layers instead reveal a salient bias towards low frequencies. Some layers even appear to discard a majority of high-frequency information.

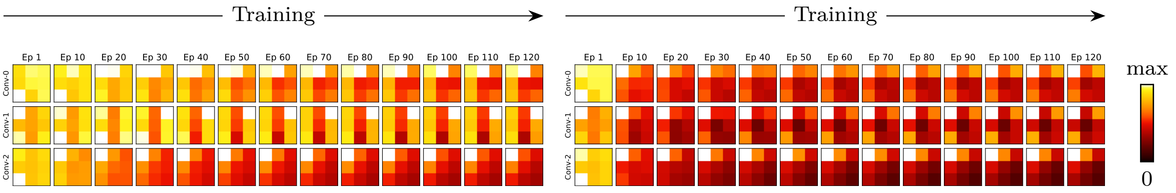
In addition, we are interested in how adversarial training affects the frequency utilization in convolution filters. As shown from various angles in (Wang et al., 2020a; Geirhos et al., 2019; Saikia et al., 2021), robust models shift their bias to low-frequencies, as this reduces the possibility of overfitting on high-frequencies and therefore provides better generalization abilities. Thus, we expect that these results transfer to the frequency utilization in weight space to some extent. Indeed, Wang et al. (2020a) stated that the very first convolution layer of AT CNNs learns smoother filters, which equals to filters



(a) `EfficientNet-B0`

(b) `ConvNeXt-Tiny`

Figure 5.3: Frequency distribution by layer. We show coefficient heatmaps for trained (a) `EfficientNet-B0` (containing 3×3 and 5×5 kernels) and (b) `ConvNeXt-Tiny` (containing 2×2 [downsampling layers], 5×5 [stem] and 7×7 kernels).



(a) Normal training

(b) Adversarial training

Figure 5.4: Frequency distribution in training per layer. Evolution of the frequency distributions in the first three convolution layers of an `EfficientNet-B0` in comparison between (a) normal and (b) adversarial training with *CIFAR-10*.

that are less reliant on high-frequency information than the equivalents in normally trained models. However, their frequency analysis was limited to the first initial layer, while we aim to provide a holistic analysis over the entire network. This is also backed by our previous observations showing that frequency utilization varies by depth. Further, their results do not appear to be representative of modern models that are trained under L^∞-norm. Such models predominantly learn thresholding filters (Madry et al., 2018) independent of architecture and dataset (Section 4.4.3) that do not resemble "common" first layers, as shown by Yosinski et al. (2014). As such, they are hardly smooth.

Exemplarily, we proceed by comparing an adversarially-trained `EfficientNet-B0` with its regularly-trained counterpart. We observe that adversarial training leads to a characteristically different distribution of learned frequencies during training (Figure 5.4). Especially in the first layers, the network learns predominantly from low frequencies, which enables the network to preserve the global image content rather than overfitting on high-frequency details such as texture. Interestingly, the adversarially-trained model learns this behavior in the early training stages and faster than under normal training conditions (Rahaman et al., 2019). Deeper layers, on the other hand, show no salient differences.

Based on these findings, we propose a transformation approach of convolution weights into the frequency domain to interact with frequency information. Secondly, based on the latter finding, we propose a high-frequency regularization to further enforce the low-frequency bias in the first network layers and thus increase the native robustness.

5.4 Filter Frequency Regularization

Recall our definition of a convolution layer in Equation 2.17. In the following, we propose a simple representation in the frequency space by replacing the convolution weight $\mathbf{W} \in \mathbb{R}^{c_{\text{out}} \times c_{\text{in}} \times k \times k}$ with a combination of learned coefficients on the DCT-II basis. In this work, we limit ourselves to kernels with $k \geq 3$. We realize this by two common implementations seen in related literature (Ulicny et al., 2022).

5.4.1 Weight Decomposition (WD)

Our first approach *Weight Decomposition (WD)*, shown in Figure 5.5, decomposes the weight in a convolution layer into learnable coefficients $\mathbf{C} \in \mathbb{R}^{c_{\text{out}} \times c_{\text{in}} \times k \times k}$ and the basis $\mathbf{V} \in \mathbb{R}^{k \times k \times k \times k}$ defined in Equation 5.1: $\mathbf{W} = \mathbf{C} \cdot \mathbf{V}$.

Then, the convolution can be rewritten as:

$$\mathbf{Y}_j = \sum_i (\mathbf{C}_{j,i} \cdot \mathbf{V}) * \mathbf{X}_i = \sum_{i,m,n} (\mathbf{C}_{j,i,m,n} \cdot \mathbf{V}_{m,n}) * \mathbf{X}_i \text{ for } j \in \{1, \dots, c_{\text{out}}\}. \tag{5.2}$$

This adds one additional tensor multiplication per layer and increases the parameters to be kept in memory by k^4 (per layer or once if shared). However, these additional parameters are constant and do

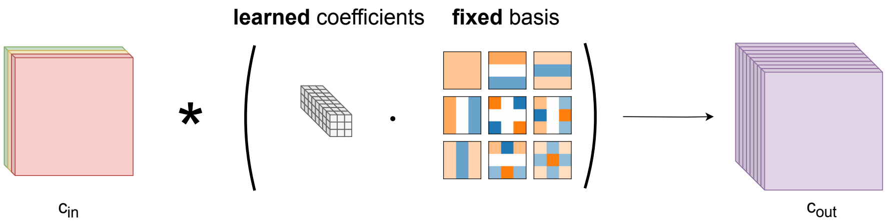


Figure 5.5: Flow of the *weight decomposition* implementation. Instead of learning the weight directly, we learn coefficients of a DCT-II basis and construct the weight via a linear combination.

not need to be learned.

5.4.2 Signal Decomposition (SD)

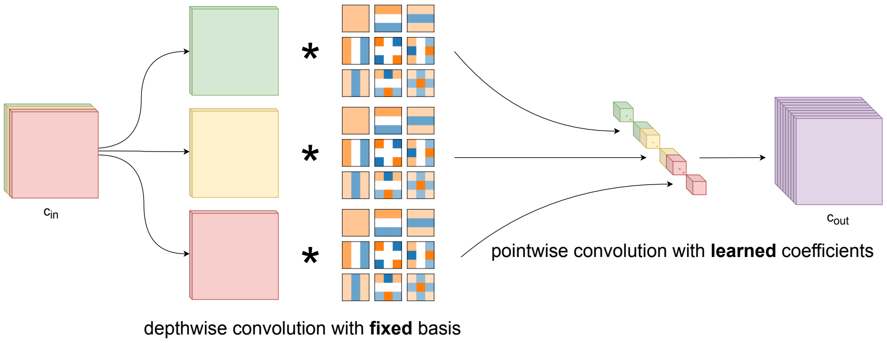

c_in
*
*
*
c_out
pointwise convolution with **learned** coefficients
depthwise convolution with **fixed** basis

Figure 5.6: Flow of the *signal decomposition* implementation. Each input channel is convolved with all basis vectors in a depthwise convolution layer. The outputs are then aggregated by a pointwise (1×1) convolution.

Alternatively, our second approach *Signal Decomposition (SD)*, shown in Section 5.4.2, does not replace the convolution weight $\mathbf{W}$ directly but performs a depthwise convolution of all combinations of inputs and the fixed basis vectors which is then aggregated by a learnable pointwise (1×1) convolution.

$$\mathbf{Y}_j = \sum_{i,m,n} \mathbf{C}_{j,i,m,n} \cdot (\mathbf{V}_{m,n} * \mathbf{X}_i) \text{ for } j \in \{1, \dots, c_{\text{out}}\}. \tag{5.3}$$

This increases the parameter number by a factor of $c_{\text{in}} \cdot k^2$ to be kept in memory. Again, the number of learnable parameters is not increased. Also, note that the associativity property of convolution reveals the equivalence of both formulations in the forward pass:

$$\mathbf{Y}_j = \sum_{i,m,n} \mathbf{C}_{j,i,m,n} \cdot (\mathbf{V}_{m,n} * \mathbf{X}_i) = \sum_{i,m,n} (\mathbf{C}_{j,i,m,n} \cdot \mathbf{V}_{m,n}) * \mathbf{X}_i \text{ for } j \in \{1, \dots, c_{\text{out}}\}. \tag{5.4}$$

However, the modifications may converge to different solutions due to different learning dynamics. In both approaches, the initial coefficient weights are sampled from a uniform distribution with an adjusted scale as per He et al. (2015). For the *weight decomposition* approach, we use $c_{\text{in}} \cdot k^2$ as fan information. The basis vectors are initialized as defined in Section 5.3 without any further adjustments.

5.4.3 Frequency Coefficient Regularization

As we have seen in Section 5.3.1, neural networks are biased towards low-frequency information, while early layers also introduce more magnitude on high frequencies. However, adversarial training increases the low-frequency bias already in the early training stages, resulting in an overall low-frequency dominance after convergence in the first layers. To make use of this finding and increase the robustness of CNNs directly without adversarial training, we propose to regularize the DCT-II coefficients and explore the frequency shift and performance. We hypothesize that this regularization mimics a "good" detection mechanic of adversarially-trained models and, thus, improves generalization and, specifically, robustness. Effectively, it can be seen as a weight penalty in frequency space (Section 2.6.1).

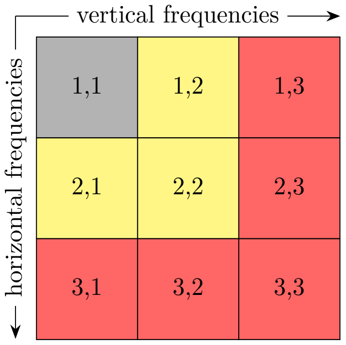


Figure 5.7: Regularization area. Shown on the coefficients of an individual 3×3 filter kernel. Colors match Equation 5.5.

$$\mathrm{SliceNorm}(\mathbf{C}, r) = \sqrt{\sum_{i,j} \left(\left(\sum_{n=1}^{r} \mathbf{C}_{j,i,n,r}^2 \right) + \left(\sum_{m=1}^{r} \mathbf{C}_{j,i,r,m}^2 \right) - \mathbf{C}_{j,i,r,r}^2 \right)}$$

$$\mathcal{R}(\mathbf{C}) = \overbrace{\left(\sum_{r=\lceil k/2 \rceil + 1}^{k} r \cdot \mathrm{SliceNorm}(\mathbf{C}, r) \right)}^{\bullet} + \rho_{\text{diff}} \cdot \max(\underbrace{\mathrm{SliceNorm}(\mathbf{C}, 2)}_{\bullet} - \underbrace{\mathrm{SliceNorm}(\mathbf{C}, 1)}_{\bullet}, 0). \tag{5.5}$$

Our proposed regularization (Equation 5.5 and Figure 5.7) penalizes the highest frequencies and additionally forces the first coefficient to have a higher magnitude than coefficients of subsequent frequencies. Such behavior – that the weight coefficients decay with their corresponding frequencies – can also be observed in the adversarially robust weights in Figure 5.7. The occurrence of this latter constraint is determined by the binary hyperparameter ρ_{diff}, with $\rho_{\text{diff}} = 1$ throughout the chapter, if not stated otherwise. Let coefs$(\boldsymbol{\theta}, h)$ denote a function that returns the set of convolution coefficient weights of the learnable parameters $\boldsymbol{\theta}$ in the first $1/h$ section of the network depth. To enforce the dominance of low frequencies in early layers, we set $h = 3$ as our default value. We train the network with the following modifications to the objective:

$$\min_{\boldsymbol{\theta}} \ell \left(f \left(\boldsymbol{x}; \boldsymbol{\theta} \right), y \right) + \lambda \sum_{\mathbf{C} \in \mathrm{coefs}(\boldsymbol{\theta}, h)} \mathcal{R}(\mathbf{C}). \tag{5.6}$$

Where ℓ is the original objective. An exemplary visualization of the learned coefficients under regularization is given in Figure 5.1 for $h \in \{1, 3\}$.

5.5 Experiments

In the following, we compare different architectures with regular convolutions and both decomposition variants (WD/SD) at varying frequency regularization (+ Reg.) (Equation 5.5). For each combination, we report results on clean accuracy and robustness on multiple datasets.

Models and datasets We evaluate low-resolution datasets such as *CIFAR-10/100* (Krizhevsky et al., 2009), *MNIST* (LeCun et al., 1998b), *SVHN* (Netzer et al., 2011), and *TinyImageNet* (Le & Yang, 2015) on `ResNet-20` (as introduced for *CIFAR* in He et al. (2016)), `ResNet-9` - a regular and larger `ResNet` with optimization for *CIFAR* and a reduced number of layers (Myrtle AI Team, 2019), and an `EfficientNet-B0` (Tan & Le, 2020) where we remove striding from the stem convolution. For *ImageNet* (Deng et al., 2009), we evaluate `EfficientNet-B0` (Tan & Le, 2020) and `ConvNeXt-Tiny` (Liu et al., 2022). We test $h \in \{1, 3\}$ and $\lambda \in \{0.01, 0.05, 0.1\}$ and report results for the best performance over the mean of 5 runs except for *ImageNet* where we only report a single run.

Note that we have selected models with different kernel sizes - *e.g.*, after the stem, `ResNets` use $k = 3$, `EfficientNets-B0` mix $k = 3$ and $k = 5$, and `ConvNeXts` $k = 7$ (and $k = 2$ downsampling layers). The variance in kernel size allows us to demonstrate the transferability of our proposed regularization beyond the common $k = 3$ kernels.

Robustness evaluation To understand the effect on robustness and generalization of our proposed decomposition and regularization approaches, we run the standard AutoAttack test suite (AA) (Croce & Hein, 2020b) and additional FGSM-, and PGD-attacks at $\epsilon = 1/255$ ($\epsilon = 16/255$ for *MNIST*) under the L^∞-norm. We use *Foolbox* (Rauber et al., 2017) to run both FGSM and PGD at the default setting (*e.g.*, 40 steps for PGD). We do not include AA results for *ImageNet*, as these models barely withstand any attacks and measure robust accuracies of 0% even at this small ϵ without adversarial training. Further, we evaluate the robustness of common corruptions of *CIFAR-10* and *ImageNet* models on the respective corrupted datasets (Hendrycks & Dietterich, 2019). In addition, we are interested in the behavior of the methods towards texture/shape bias (Geirhos et al., 2019), and OOD generalization tests (Geirhos et al., 2021). Hence, we evaluate our *ImageNet* models on 5 of these OOD datasets: *texture/shape cue-conflict*, *ImageNet-Sketch*, *Stylized-ImageNet*, and edge-/silhouette-transformations of *ImageNet* using the implementation of Geirhos et al. (2021).

5.5.1 Hyperparameters

We train models for all low-resolution datasets with the same hyperparameters (except augmentations) but use a different set of hyperparameters for *ImageNet*.

Low-resolution: CIFAR-10/100, MNIST, SVHN, TinyImageNet Models are trained for 120 epochs. For both `ResNets`, we use an SGD optimizer (with Nesterov momentum of 0.9) with an initial learning rate of 1e-2 that we downscale by 0.1 every 30 epochs and a weight decay of 1e-2. For `EfficientNet-B0`, we use an AdamW (Loshchilov & Hutter, 2019) optimizer with an initial learning rate of 1e-4 that follows a cosine annealing schedule and a weight decay of 5e-2. In all cases, we use a batch size of 256 and categorical cross entropy as a loss function with our regularization. We analyze models with weights learned after the last gradient update.

We use the following augmentations for datasets:

- **CIFAR-10/100:** Training images are zero-padded by 4 px along each dimension, apply random horizontal flips, and proceed with 32 × 32 px random crops. Test images are not modified.
- **TinyImageNet:** Training images are obtained using randomly resized 56 × 56 px crops. Test images are 56 × 56 px center crops.
- **MNIST:** Train and test images are upscaled to 32 × 32 px.
- **SVHN:** Train and test images are not modified.

For all datasets, samples are normalized by the channel mean and standard deviation.

ImageNet We train all *ImageNet* models with the default hyperparameters and augmentations for `ConvNeXt-Tiny` (Liu et al., 2022). In particular, we train 300 epochs with an effective batch size of 4096. For `EfficientNet-B0`, we reduce the batch size to 1024 due to memory constraints. Again, we evaluate model parameters learned after the last gradient update.

5.6 Results

In this section, we discuss the results of our regularization when trained on multiple low-resolution datasets (Section 5.6.1), including a comparison to prior work, and finally on *ImageNet* (Section 5.6.2).

Table 5.1: Frequency regularization experiments on *CIFAR-10*. Results are presented for ResNet-20, ResNet-9, and EfficientNet-B0, with comparisons to prior work. We report the mean clean accuracy, robust accuracy against adversarial attacks (FGSM, PGD-40, and AutoAttack) for L^{∞} norm with $\epsilon = 1/255$, and the mean corruption accuracy on *CIFAR-10-C*. All results are averaged over 5 runs. **Best**, Second Best (only for single method evaluations).

	Variant	**Clean**	**Adversarial Acc. (↑)**			**Corruption**
		Acc. (↑)	**FGSM**	**PGD-40**	**AA**	**Acc. (↑)**
`ResNet-20`	CNN	91.29	50.49	30.92	10.78	67.96
	FLC (Grabinski et al., 2022c)	**91.52**	52.49	30.25	8.48	68.75
	PaGA (Lopes et al., 2020)	91.29	50.36	31.50	11.38	67.73
	Blur Pooling (Zhang, 2019)	89.89	41.61	29.43	15.58	66.73
	Adaptive Blur Pooling (Zou et al., 2020)	89.48	41.94	32.09	18.22	67.17
	Wavelet Pooling (Li et al., 2020)	89.89	41.17	27.88	13.78	67.40
	WD	91.04	48.40	30.37	10.72	66.92
	SD	91.36	50.83	32.98	11.97	67.48
	WD + Reg.	89.86	50.85	41.81	26.79	74.04
	SD + Reg.	90.54	**53.12**	**44.42**	**29.14**	**74.14**
	WD + Reg + FLC	89.65	52.21	42.02	25.39	75.28
	WD + Reg + PaGA	89.84	50.09	40.92	25.79	73.85
	SD + Reg + FLC	90.37	54.89	45.13	28.11	74.41
	SD + Reg + PaGA	90.32	52.47	42.98	27.30	73.19
`ResNet-9`	CNN	94.29	59.58	53.04	37.49	73.38
	FLC (Grabinski et al., 2022c)	94.24	59.64	53.47	38.65	73.81
	PaGA (Lopes et al., 2020)	**94.33**	59.12	52.62	37.50	73.72
	WD	93.73	55.51	49.84	35.23	72.87
	SD	93.97	55.73	50.29	36.00	73.48
	WD + Reg.	93.18	59.25	56.08	43.62	76.41
	SD + Reg.	93.09	**59.87**	**56.89**	**44.80**	**77.72**
	WD + Reg + FLC	93.20	59.64	56.39	44.24	76.78
	WD + Reg + PaGA	92.15	56.48	52.64	39.78	78.14
	SD + Reg + FLC	93.43	60.80	58.18	46.08	77.60
	SD + Reg + PaGA	93.14	59.86	57.24	45.06	77.54
`EfficientNet-B0`	CNN	90.38	53.55	54.05	45.51	68.09
	FLC (Grabinski et al., 2022c)	89.68	51.92	53.09	45.37	69.72
	PaGA (Lopes et al., 2020)	**90.72**	54.18	54.97	46.64	69.31
	WD	90.51	49.87	49.97	40.76	67.10
	SD	90.44	51.04	51.77	43.39	66.65
	WD + Reg.	88.97	**57.91**	59.60	53.30	**72.14**
	SD + Reg.	89.18	57.83	**59.68**	**53.50**	71.87
	WD + Reg + FLC	87.67	52.68	54.66	48.14	71.17
	WD + Reg + PaGA	88.49	55.15	56.70	50.05	74.16
	SD + Reg + FLC	87.90	54.15	56.38	50.31	71.10
	SD + Reg + PaGA	89.66	56.72	58.56	51.87	72.39

5.6.1 Low-Resolution Datasets

We start our discussion of results on models trained on *CIFAR-10*. Based on the shown results in Table 5.1) we observe that replacing regular convolution layers with either decomposition variant (SD or WD) but not applying regularization has a rather insignificant impact on the clean and robust accuracy (for adversarial attacks and *CIFAR-10C*). This is to be expected, as the formulations are equivalent, and the remaining differences can be attributed to changes in the gradient flow. However, once we apply the regularization, we see clear improvements in robustness towards all threat models, with a small trade-off on clean accuracy. We can also observe that SD slightly outperforms WD on almost all tested architectures. Hence, it may be tempting to only proceed with SD. However, the additional channels necessary to implement SD account for more parameters, a large memory overhead, and slower inference and training performance. For instance, we see a 4.4x slower forward pass and 18% more total parameters on `ResNet-20`, while WD has a minimal overhead, both in parameters and throughput (Table 5.2).

Targeting adversarial robustness, we see the largest gains on models that initially performed worst

Table 5.2: Overhead benchmark. We apply various frequency attenuation techniques to a `ResNet-20` and evaluate the overhead for a batch size of 512 on an NVIDIA A100 GPU.

Variant	**Total Params** (↓)	**Learnable Params** (↓)	**Throughput (k img/sec)** (↑)	**Batch Update (ms)** (↓)
CNN	272.5k	272.5k	128.0	24.7
FLC (Grabinski et al., 2022c)	272.5k	272.5k	82.7	30.2
Blur Pooling (Zhang, 2019)	272.5k	272.5k	113.0	26.5
Adaptive Blur Pooling (Zou et al., 2020)	272.7k	272.7k	34.4	44.0
Wavelet Pooling (Li et al., 2020)	272.5k	272.5k	103.0	26.5
WD (+ Reg.)	274.0k	272.5k	124.3	24.9
SD (+ Reg.)	323.3k	272.5k	28.0	51.9

(+18.36% AutoAttack (AA) increase on `ResNet-20`). Out of all our tested models, `EfficientNet-B0` is the most robust, both before and after regularization.

All results were acquired with tuned regularization hyperparameters, and we often obtain the best results for *CIFAR-10* when regularizing only the first third of the network ($h = 3$). Other tested regularization parameters result in a slightly worse robust accuracy. Noticeably, even the worst hyperparameter combination for `ResNet-20` (WD, $\lambda = 0.01, h = 1, \rho_{\text{diff}} = 0$) still achieves a 14.22% higher AA accuracy than the unregularized baseline.

For common corruptions (CC) (the last column in Table 5.1), we analyze the mean accuracy over all corruptions and severities, as well as individual results for corruptions at the highest severity level (not shown for brevity). We observe that regularized models become significantly more robust against corruptions having predominantly high-frequency (HF) perturbations (*cf*., Yin et al. (2019) for spectrums) such as *pixelate* and *defocus/glass/gaussian blur*. Perhaps less surprisingly, regularized models become less sensitive to increased *JPEG compression*, as they rely on (quantized) DCT-II coefficients. Regularized performance remains largely unchanged for corruptions with larger variance in the frequency spectrum. We see a slight degradation of performance in low-frequency (LF) corruptions such as *brightness*, *saturation*, *contrast*, and *impulse noise*. However, the accuracy drop is relatively low, considering the evaluation at the highest severity level. On average, our regularization increases the robustness to common corruptions.

Comparison to other methods Lastly, we compare our method to other HF regularization methods in prior work. Specifically, we compare to *FrequencyLowCut Pooling (FLC)* (Grabinski et al., 2022c), *Patch Gaussian Augmentation (PaGA)* (Lopes et al., 2020), and, on `ResNet-20`, also to *Blur Pooling* (Zhang, 2019), *Adaptive Blur Pooling* (Zou et al., 2020), and *Wavelet Pooling* (Li et al., 2020) (Table 5.1). Regarding AA and CC performance, we observe that our method consistently outperforms any other approach in standalone comparisons with a small degradation of clean validation accuracy. Additionally, our regularization remains compatible with other methods and can improve their robustness significantly, as we show for *FLC* and *PaGA*. Overall, we sometimes get the highest levels of (specialized) robustness in combination with another method.

Other low-resolution datasets Although several works reported a shift in the frequency band of adversarial attacks depending on the dataset (Maiya et al., 2021; Abello et al., 2021; Bernhard et al., 2021; Ortiz-Jiménez et al., 2020), we consequently see an improvement in the results shown in Table 5.3, due to our frequency regularization on multiple datasets. Arguably, we see smaller improvements for *SVHN/TinyImageNet* – which are also the datasets that show more LF perturbations than HF. Contrary to our *CIFAR-10* results, WD outperforms SD on all datasets except *SVHN*.

5.6.2 ImageNet

Next, we aim to explore how our regularization performs on the common *ImageNet* dataset (Deng et al., 2009). In particular, more OOD tests exist for this dataset, allowing us to study aspects

Table 5.3: Regularization on other low-resolution datasets. We report the performance of various networks on multiple datasets (*CIFAR-100, SVHN, TinyImageNet, MNIST*) similarly to our experiments on *CIFAR-10* (except that we use $\epsilon = 16/255$ for *MNIST*).

		Variant	Clean Acc. (↑)	Adversarial Acc. (↑)		
				FGSM	PGD-40	AA
CIFAR-100	ResNet-20	CNN	**60.41**	14.36	5.45	1.17
		WD	58.90	12.84	4.68	1.01
		SD	60.34	13.87	5.18	1.13
		WD + Reg.	56.65	16.85	**12.73**	**5.59**
		SD + Reg.	58.19	**17.20**	12.24	5.11
	ResNet-9	CNN	75.80	30.41	24.12	10.13
		WD	75.52	29.97	23.68	10.80
		SD	**76.06**	30.73	24.65	11.48
		WD + Reg.	74.75	33.66	30.31	17.62
		SD + Reg.	75.27	**34.04**	**30.95**	**18.26**
	EfficientNet-B0	CNN	62.09	26.33	24.99	18.76
		WD	**63.49**	23.55	18.13	9.91
		SD	62.85	25.11	20.73	13.21
		WD + Reg.	59.21	27.44	27.73	22.93
		SD + Reg.	60.49	**29.63**	**30.24**	**25.42**
SVHN	ResNet-20	CNN	96.31	83.84	79.94	69.81
		WD	**96.35**	83.52	80.01	71.25
		SD	96.34	84.07	80.64	71.74
		WD + Reg.	96.28	84.11	81.21	**73.27**
		SD + Reg.	96.34	**84.17**	**81.23**	73.03
	ResNet-9	CNN	**95.92**	**83.02**	82.48	77.32
		WD	95.59	81.55	81.06	75.61
		SD	95.78	82.46	82.08	77.06
		WD + Reg.	95.69	82.08	81.70	76.81
		SD + Reg.	95.83	82.90	**82.71**	**78.14**
TinyImageNet	ResNet-20	CNN	**25.61**	9.93	9.37	4.30
		WD	24.50	9.27	8.97	4.30
		SD	25.05	9.60	9.15	4.45
		WD + Reg.	24.56	10.13	10.07	5.23
		SD + Reg.	24.92	**10.58**	**10.37**	**5.26**
	ResNet-9	CNN	**53.20**	17.79	16.76	9.57
		WD	52.08	17.11	16.19	9.40
		SD	52.12	16.85	15.88	9.15
		WD + Reg.	51.25	18.10	17.34	10.23
		SD + Reg.	51.22	**18.26**	**17.40**	**10.39**
MNIST	ResNet-20	CNN	99.68	89.74	45.37	8.92
		WD	99.69	90.45	47.06	10.22
		SD	99.65	**91.23**	54.08	16.80
		WD + Reg.	99.69	90.70	**55.84**	**25.92**
		SD + Reg.	99.69	88.98	50.02	21.71

Table 5.4: Regularization on *ImageNet*. We report results for `EfficientNet-B0` and `ConvNeXt-Tiny` on clean data, FGSM, PGD-40 ($L^\infty, \epsilon = 1/255$), *ImageNet-C*, and out-of-distribution generalization tests. All regularization hyperparameters are $\lambda = 0.05$ and $h = 3$.

	Variant	Clean Val. Acc. (↑)		Adversarial Attacks Acc. (↑)		Corruption Error (↓)	Cue-Conflict (↑)	Sketch (↑)		Stylized (↑)		Edge (↑)	Silhouette (↑)
		Top 1	Top 5	FGSM	PGD-40	ImageNet-C		Top 1	Top 5	Top 1	Top 5	Top 1	Top 1
EfficientNet-B0	CNN	75.44	92.86	16.89	2.32	54.54	23.52	65.25	84.62	52.25	79.00	35.00	51.25
	WD	**75.80**	**92.94**	14.86	2.03	53.99	22.58	66.12	86.25	48.25	78.88	**40.62**	55.00
	SD	75.62	92.82	15.05	1.41	52.85	23.67	**66.38**	84.50	**52.50**	78.50	34.38	58.13
	WD + Reg.	75.44	92.15	18.45	4.43	52.03	**29.38**	**66.38**	86.88	47.62	**79.75**	36.25	58.13
	SD + Reg.	74.42	92.19	**18.70**	**5.33**	**51.12**	25.78	64.75	**87.88**	49.12	77.12	32.50	**58.75**
ConvNeXt-Tiny	CNN	**81.32**	95.53	35.53	3.93	**41.92**	24.84	71.50	88.00	56.00	78.38	**48.12**	62.50
	WD	81.11	**95.55**	35.30	2.97	42.98	25.31	73.12	**89.62**	52.00	77.62	47.50	58.75
	WD + Reg.	79.25	94.38	**35.69**	**4.22**	44.31	**32.27**	**73.75**	88.12	**58.00**	**82.38**	38.75	**65.62**

outside adversarial robustness and robustness against common corruptions. Similar to our results on other datasets, we see an improvement in adversarial robustness at slight (1-2%) degradation of clean performance (Table 5.4). While we see an improvement in CC performance on `EfficientNet`, we see an equal decrease for `ConvNeXt`. This may be due to the larger kernels (7×7) that `ConvNeXt` utilizes and may, thus, require other hyperparameters. Importantly, we see a significant improvement of the *cue-conflict* in both cases - which is also reflected in the increased accuracy of *silhouette* (LF) and the decrease in performance of *edge* (HF). This indicates that our regularization favorably shifts models toward a shape bias (Geirhos et al., 2019).

5.7 Post-Regularization Network Behavior

The following section explores how our regularization affects the model's decision-making process. We employ attribution map visualizations to understand differences in salient input regions (see Section 2.9.3 for an introduction to attributions) and analyze the distribution shift in activations and generated perturbations of adversarial attacks.

5.7.1 Attribution Shifts

We start by visualizing and interpreting attribution maps via *SmoothGrad* (Smilkov et al., 2017). We study the attributions of some selected ImageNet validation samples which maximize the gradient for the true label. We generate attributions from a `ConvNeXt-Tiny` with the unmodified architecture and with a regularized WD layer equivalent to Table 5.4. We assert that both models correctly predict the top-1 label for all samples. Our results in Figure 5.8 show that our proposed regularization (WD + Reg.) shifts attributions from edges and local shortcuts to a more global focus on the respective objects.

5.7.2 Distribution Shifts in Activations

Further, we aim to understand the implications of our regularization on the activations directly after convolution layers. We compare a regularized `ResNet-20` (SD + Reg.) against a CNN baseline trained on *CIFAR-10*, and analyze the magnitude shift in the DCT-II coefficients of the feature maps (Figure 5.9) of a clean validation batch. Our regularization causes a clear shift towards lower frequencies in regularized layers. Interestingly, in the stem layer, we also see large shifts from entirely vertical or horizontal frequencies to more balanced ones. In non-regularized (deeper) layers, we observe a slight shift towards higher frequencies.

5.7.3 Distribution Shifts in Adversarial Perturbations

In Figure 5.10, we additionally show the attack spectrum of an FGSM attack with $\epsilon = 1/255, L^{\infty}$ on a `ResNet-20`, `ResNet-9`, and `EfficientNet-B0` trained on *CIFAR-10*. We generate attack spectra for the models for the non-robust baseline and regularized WD/SD layers. Regular convolutions are fairly uniformly attacked throughout the spectrum, with a strong peak in the highest frequencies (top-left). Under regularization, we observe a shift of attacks to lower frequencies, which indicates a successful defense against high-frequency perturbations. We observe differences in the spectrum depending on the architecture. The implementation of our approaches, on the other hand, shows marginal differences.

5.8 Regularization under Adversarial Training

So far, we investigated the effect of our proposed regularization for native robustness under normal training. To complete this, we aim to explore the role of our regularization in AT. We train our models against FGSM-adversaries on *CIFAR-10* ($L^{\infty}, \epsilon = 8/255$) (Table 5.5). We use early stopping based on PGD-40 test performance to avoid robust overfitting. We observe, that our regularization

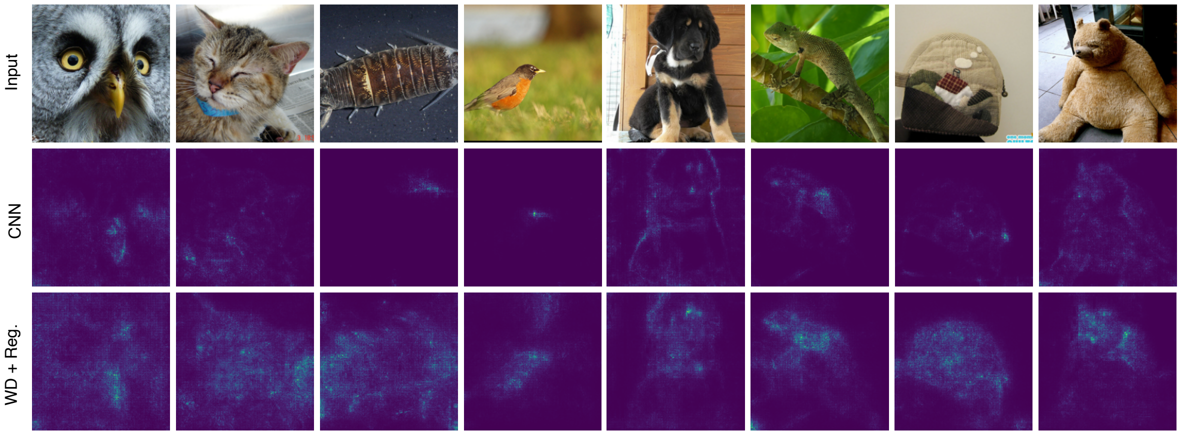


Figure 5.8: Changes in attributions. We show *SmoothGrad* attributions (Smilkov et al., 2017) of `ConvNeXt-Tiny` without changes to convolution layers (middle), and regularized WD (bottom). The regularization appears to shift attributions from edges and local shortcuts to a more global utilization of the object.

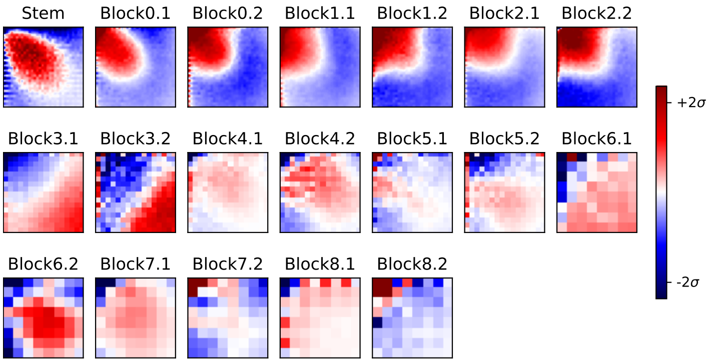


Figure 5.9: Frequency shifts in activations. Regularized layers (top row) of a `ResNet-20` show a shift towards LF after regularization, compared to non-regularized layers in rows two and three. Frequency increases from the top-left outwards.

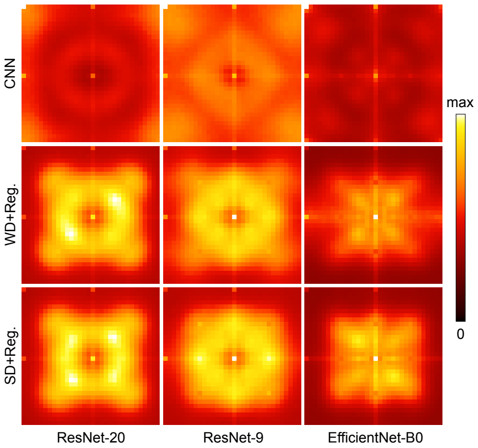


Figure 5.10: Frequency shifts in adversarial perturbations. We show the mean absolute error in the FFT spectrum of FGSM-attacks ($\epsilon = 1/255, L^{\infty}$) on *CIFAR-10* models with normal and both regularized convolution modifications. Frequency increases from the center outwards.

Table 5.5: FGSM-Adversarial Training on *CIFAR-10*. We train networks under L^∞, $\epsilon = 8/255$ attacks and report the mean over 5 runs for the FGSM train and validation accuracy of the epoch of the best PGD-40 validation accuracy as well as the AutoAttack accuracy. We also report the corresponding mean accuracy on *CIFAR-10-C* and report the difference to the clean-trained evaluations.

	Variant	Clean	FGSM (↑)	Adversarial Acc. (↑)		Corruption Acc.	
		Val. Acc. (↑)	Train Acc.	PGD-40	AA	Mean (↑)	Δ (AT-Normal) (↑)
`ResNet-20`	CNN	**73.73**	**50.39**	46.14	36.09	**66.99**	-0.97
	WD + Reg.	71.69	48.46	45.38	35.64	65.48	-8.56
	SD + Reg.	73.01	49.93	**46.34**	**36.47**	66.73	-7.41
`ResNet-9`	CNN	81.70	60.27	**52.77**	0.00	74.06	0.68
	WD + Reg.	81.56	61.66	51.52	39.97	74.47	-1.94
	SD + Reg.	**82.56**	**63.39**	51.80	**40.14**	**75.40**	-2.32
`EfficientNet-B0`	CNN	63.00	42.34	42.49	34.04	57.35	-10.74
	WD + Reg.	68.50	45.87	45.13	36.56	62.66	-9.48
	SD + Reg.	**68.89**	**46.76**	**45.57**	**36.76**	**63.07**	-8.8

Table 5.6: PGD-Adversarial Training on *ImageNet*. We train `ResNet-50` under L^∞, $\epsilon = 4/255$ attacks and report the train and validation accuracy under PGD attacks, validation accuracy under AutoAttack, corruption error, and cue-conflict. Results are from one run.

Variant	Clean	PGD (↑)	Adversarial Acc. (↑)		Corruption	Cue-
	Val Acc. (↑)	Train Acc.	PGD	AA	Error (↓)	Conflict (↑)
CNN	56.85	33.88	36.04	22.33	78.65	38.83
WD + Reg.	**58.09**	**36.00**	**37.09**	**24.32**	78.36	**39.38**
SD + Reg.	57.61	35.86	35.85	22.25	**77.33**	38.59

has a beneficial effect on the out-of-distribution attacks (*i.e.*, AA) and all runs show an increased performance after regularization. We furthermore observe that our regularization appears to mitigate *robust overfitting* of training attacks (similarly to Grabinski et al. (2022c)) on `ResNet-9`: without regularization the AT-trained CNN achieves high FGSM train accuracy and high PGD-40 validation accuracy but fails to generalize to other attacks (AA) and stagnates at 0%. All runs with regularization show comparable or even better accuracy than the best non-regularized models. However, similarly to the observations by Saikia et al. (2021), we generally see a significant decrease in CC accuracy due to AT. Again, this demonstrates that AT is not the cure-all to improve network robustness, and there is a need for other approaches, such as our proposed frequency regularization.

Additionally, we extend our experiments to AT on *ImageNet* (Table 5.6). Here, we switch to single-step PGD training with the common $\epsilon = 4/255$ and train the ResNet-50 architecture. Again, we report PGD, AA, and CC performance, but this time also the cue-conflict score. Our regularized WD architecture outperforms the baseline in all metrics: adversarial robustness, corruption error, and cue-conflict. This demonstrates that our method can mitigate some of the overfitting aspects of adversarial training while improving OOD generalization performance. We do not see inconsistent changes in robustness for SD, but we achieve the overall lowest corruption error and increase the clean accuracy compared to the baseline. We attribute the lack of robustness increases to suboptimal hyperparameters and expect similar gains when appropriately tuned.

5.9 Ablation of the Regularization Term

In order to further support the regularization approach, we perform an ablation of our regularization term. First, to motivate the regularization of high frequencies, we regularize *inverse* to our original approach: instead of regularizing high frequencies, we regularize the low frequencies. For $k = 3$ we implement this penalty by:

$$\mathcal{R}_{\text{inverse}}(\mathbf{C}) = \text{SliceNorm}(\mathbf{C}, 1) \cdot 2 + \left(\left[\min_{\substack{n,m \\ (n=3 \text{ or } m=3)}} \sqrt{\sum_{i,j} \mathbf{C}^2_{j,i,m,n}} \right] - \text{SliceNorm}(\mathbf{C}, 2) \right). \tag{5.7}$$

Second, we observe that some layers predominantly learn non-zero coefficients in the first row and column. To further investigate this observation, we regularize the lower quadrant, *i.e.*, leaving the coefficients in the first column and first row ($c_{i,1}, c_{1,i},$ for $i \in \{1, \ldots, k\}$) unregulated.

$$\mathcal{R}_{\text{quadrant}}(\mathbf{C}) = \|\mathbf{C}_{:,:,2:,2:}\|_2 \cdot 2. \tag{5.8}$$

This way, we do not regularize all coefficients corresponding to high frequencies. As a result, *quadrant* suppresses high frequencies on diagonal structures but allows vertical/horizontal high frequencies, while our original regularization enforces regularization independent of the orientation.

We present the results in Table 5.7 on `ResNet-20`. The *inverse* approach does not increase robustness; quite contrary, it even decreases the accuracy under adversarial attacks and *CIFAR-10-C*, which strengthens the formulation of our original regularization. The *quadrant* regularization increases robustness but performs subpar compared to our original approach.

Table 5.7: Ablation of different regularization types. We train `ResNet-20` on *CIFAR-10* under different regularization schemes and evaluate against adversarial attacks (FGSM, PGD-40, AutoAttack) and *CIFAR-10-C*. We report the mean accuracy in % over 5 runs.

Variant	**Clean**	**Adversarial Acc. (↑)**			**Corruption**
	Acc. (↑)	**FGSM**	**PGD-40**	**AA**	**Acc. (↑)**
CNN	91.29	50.49	30.92	10.78	67.96
WD	91.04	48.40	30.37	10.72	66.92
SD	**91.36**	50.83	32.98	11.97	67.48
WD + Reg. ($\lambda = 0.01, h = 1$)	88.91	48.93	40.50	25.94	73.97
WD + Reg. ($\lambda = 0.01, h = 3$)	90.02	50.38	41.34	25.86	73.43
WD + Reg. ($\lambda = 0.05, h = 1$)	89.05	49.06	40.83	26.24	73.76
WD + Reg. ($\lambda = 0.05, h = 3$)	89.86	50.85	41.81	26.79	74.04
WD + Reg. ($\lambda = 0.1, h = 1$)	88.91	49.16	40.95	26.39	73.84
WD + Reg. ($\lambda = 0.1, h = 3$)	89.85	50.65	41.57	26.33	73.85
SD + Reg. ($\lambda = 0.01, h = 1$)	89.48	50.59	42.61	28.22	73.92
SD + Reg. ($\lambda = 0.01, h = 3$)	90.54	53.12	**44.42**	**29.14**	74.14
SD + Reg. ($\lambda = 0.05, h = 1$)	89.20	49.59	41.05	26.28	74.24
SD + Reg. ($\lambda = 0.05, h = 3$)	90.34	53.07	44.04	28.62	73.98
SD + Reg. ($\lambda = 0.1, h = 1$)	89.29	50.06	41.90	27.58	**74.29**
SD + Reg. ($\lambda = 0.1, h = 3$)	90.28	52.39	43.29	27.75	74.07
WD + Reg. ($\lambda = 0.01, h = 3$, inverse)	90.45	45.59	23.85	6.22	65.41
WD + Reg. ($\lambda = 0.05, h = 3$, inverse)	90.47	43.95	20.39	3.94	64.49
WD + Reg. ($\lambda = 0.1, h = 3$, inverse)	90.62	42.56	18.69	3.29	64.57
WD + Reg. ($\lambda = 0.01, h = 3$, quadrant)	90.50	51.76	41.03	24.37	71.02
WD + Reg. ($\lambda = 0.05, h = 3$, quadrant)	90.08	51.33	40.20	24.08	71.55
WD + Reg. ($\lambda = 0.1, h = 3$, quadrant)	90.10	51.14	40.81	24.79	71.62
SD + Reg. ($\lambda = 0.01, h = 3$, inverse)	90.69	47.23	25.26	6.36	65.92
SD + Reg. ($\lambda = 0.05, h = 3$, inverse)	90.86	45.94	21.81	4.36	65.04
SD + Reg. ($\lambda = 0.1, h = 3$, inverse)	90.56	44.10	20.17	3.66	64.05
SD + Reg. ($\lambda = 0.01, h = 3$, quadrant)	90.67	**53.95**	43.94	27.23	72.29
SD + Reg. ($\lambda = 0.05, h = 3$, quadrant)	90.49	52.98	42.31	25.65	72.22
SD + Reg. ($\lambda = 0.1, h = 3$, quadrant)	90.54	53.02	42.06	25.77	72.10

5.10 Conclusion

Extending our spatial filter analysis from Chapter 3 to the frequency domain (as in Chapter 4) reveals another key difference between standard and adversarially-trained (robust) models: robust models exhibit increased magnitudes in their low-frequency filter components compared to normally trained ones. While our earlier attempt in Section 4.5 to replicate the filter diversity of robust models in the spatial domain via regularization proved unsuccessful, the frequency-based regularization proposed in this chapter has yielded promising results. We have shown a first step towards improving the native robustness of CNNs to multiple distribution shifts such as adversarial attacks, corruptions, and shape-biased datasets. Additionally, our method remains compatible and beneficial for adversarial training.

In particular, our regularization decreases the sensitivity to high-frequency perturbations and if covariate shifts affect these bands. Albeit our results do not approach SOTA levels, we emphasize that we improve robustness on a wide range of tests. In contrast, SOTA methods like AT often overfit to one specific type of robustness, such as adversarial attacks, and often even impair performance on other tests compared to normal baselines. Additionally, our method does not rely on OOD examples but intrinsically strengthens the model. Our approach has been shown to generalize to different networks with various kernel sizes that were trained on different datasets and different measures of robustness. We have also shown that our method can be used in combination with other approaches such as *PaGA* (Lopes et al., 2020), *FLC* (Grabinski et al., 2022c), and even AT (Madry et al., 2018) to improve robust performance further. In combination with AT, our approach shows promise to mitigate *robust overfitting* (Rice et al., 2020).

Limitations We observed that on some architectures, switching to WD/SD introduces a significant drop in accuracy (without regularization). Although the forward pass of both methods is mathematically equivalent to baselines, the backward pass is not. For instance, weight updates on linear combinations of decomposed convolution filters and feature maps are in different backward pass stages and under different quantization conditions due to limited bit precision. While we observe that our regularization generally improves a multitude of robustness aspects, the regularized counterparts may underperform CNN baselines due to the initial impairment due to the architecture change.

Chapter 6

Beyond Filter Analysis: Layer Criticality

In our filter analysis in Chapters 3 to 5, we compared distribution shifts between all convolution layers; however, do all layers matter equally? Prior research suggests that this is not true Veit et al. (2016); Zhang et al. (2022). In this section, we introduce a method to estimate the importance of a specific layer (**criticality**) as a proxy for the overall complexity or length of the learned decision rule, and then apply it to study how different forms of training regularization change the location and length of the decision rule that image classifiers learn.

This chapter is based on "How Do Training Methods Influence the Utilization of Vision Models?", presented at the NeurIPS Workshop on Interpretable AI: Past, Present and Future, 2024 (Gavrikov et al., 2024a). As the first author, Paul Gavrikov developed the code base and methodology, performed the experiments, created the plots, and wrote the paper with input from Shashank Agnihotri, Janis Keuper, and Margret Keuper.

Code: `https://github.com/paulgavrikov/layer_criticality`

6.1 Introduction

A famous neurology myth often misattributed to Albert Einstein states that humans only use 10% of the neural connections in their brains (Radford, 1999). While modern research assumes that humans use all neural connections (Boyd, 2008; Herculano-Houzel, 2009) – the same cannot be said about artificial neural networks. Quite the contrary, it is well known that trained neural networks do not utilize their entire capacity. This becomes evident through the lens of *parameter pruning* (LeCun et al., 1989b; Hassibi et al., 1993), where (large numbers) of neurons can be removed after training without affecting performance, or, alternatively, the *distillation* of large into equivalent smaller networks (Hinton et al., 2015). Alas, the learned decision rule only occupies a fraction of the neural network, and the remaining neurons seem to be wasted.

Veit et al. (2016); Zhang et al. (2022) showed that this seems to affect layers disproportionally. The learnable parameters[1] of some layers are *critical* to the decision rule and replacing them with any other values than the learned ones (significantly) affects accuracy. In contrast, the performance is barely affected when the parameters in *auxiliary* layers are randomized. For instance, entire residual blocks of `ResNets` (He et al., 2016) trained on *ImageNet* (Deng et al., 2009) can be randomized without hurting accuracy. Affected layers seem to be dictated by training data size or, more generally, the complexity of the training function in addition to the architecture.

[1] Throughout the rest of this chapter, the term *parameters* will refer to learnable parameters.

This finding has the potential to influence our analytical results in Chapters 3 to 5 as the pattern distributions of auxiliary layers may distort our analyses. However, it may also serve as a better metric to capture the weight-space – Chatterji et al. (2020) have even extended these findings to generalization by showing that the average module criticality correlates with performance - *i.e.*, more complex networks seem to generalize better.

In light of this, we revisit the prior findings of Zhang et al. (2022); Chatterji et al. (2020) on image classification models, which were obtained under rather clean conditions, such as the absence of weight decay regularization (Krogh & Hertz, 1991) or batch-normalization layers (Ioffe & Szegedy, 2015) during training, and an overall simple training pipeline. These conditions do not reflect practical training pipelines well. Thus, we raise the question: *how does the training method affect layer criticality?* To this end, we study criticality on a large model zoo of image classification models where the training data and architecture are fixed, but the training pipeline, including regularization, is modified.

We summarize our contributions as follows:

- We propose a modification of the criticality metric introduced by (Zhang et al., 2022) that is tailored to capture the predictive behavior instead of performance drops.
- We apply this metric on a large model zoo of image classifiers, all of which were trained on the same *ImageNet-1k* data (Deng et al., 2009; Russakovsky et al., 2015) and are based on the exact same `ResNet-50` (He et al., 2016) network architecture but varying training methods. This allows us to study the connection between criticality and training methods without confounders.
- We find that training methods leave clear patterns about criticality. For instance, adversarial training (Madry et al., 2018) significantly increases the average criticality, while most improved training recipes that increase accuracy tend to decrease the average criticality. We do find that criticality is affected by distribution shifts, but there appears to not be a conclusive correlation between criticality and generalization, nor our previously established filters metrics in Chapters 3 and 4.

6.2 Methodology

In the following subsections, we define how we quantify the criticality of a layer and the model zoo that we will use for our analysis.

6.2.1 Testing Layers for Criticality

Our study aims to assess the contribution or *criticality* of individual layers to a neural network's decision rule. To gauge a layer's importance, we replace its parameters with random values (*randomization*). If the network's decisions remain largely unchanged after randomization, it suggests that the learned parameters contribute little beyond noise.

This methodology largely follows Zhang et al. (2022), who reset the parameters of individual layers to values drawn from the original initialization. However, while Zhang et al. (2022) measured *criticality* of a layer by the change in accuracy due to the randomization, we measure the angle between the probability vectors resulting from the randomization. Specifically, we apply a Softmax (Equation 2.24) function to the network logits to obtain (pseudo-)probabilities, measure the cosine distance between those before and after randomization, and aggregate the measurements into a single scalar by averaging over all samples. The effect of each layer randomization on this measured distance is what we define as the *criticality* of a layer.

Formally, given a model $f(\boldsymbol{x};\boldsymbol{\theta})$ parameterized by $\boldsymbol{\theta}$ that returns probabilities (*i.e.*, applies the Softmax function to its logits), and a dataset $\mathbb{X}$, the criticality κ_l of the l-th layer is given by

$$\kappa_l(\boldsymbol{\theta}) = \frac{1}{|\mathbb{X}|} \sum_{\boldsymbol{x} \in \mathbb{X}} \left[1 - \frac{f\left(\boldsymbol{x}; \boldsymbol{\theta}\right) \cdot f\left(\boldsymbol{x}; \boldsymbol{\theta}_{\text{rand}}^{(l)}\right)}{\| f\left(\boldsymbol{x}; \boldsymbol{\theta}\right) \|_2 \| f\left(\boldsymbol{x}; \boldsymbol{\theta}_{\text{rand}}^{(l)}\right) \|_2} \right], \tag{6.1}$$

where $\boldsymbol{\theta}_{\text{rand}}^{(l)}$ denotes the parameters with randomized weights for the l-th layer.

There are multiple ways to randomize weights with different implications: (1) weights can be *rewound* to their exact initial values before training – however, this “pattern” already carries an important signal of the final weights (Frankle & Carbin, 2019) and is not what we want in this context; (2) draw new parameters from a distribution normalized for gradient flow (Glorot & Bengio, 2010; He et al., 2015)) – this would randomize the weights to our needs but may be incompatible with other layers, *e.g.*, if the initial parameters were initialized from a different distribution; (3) draw parameters from the original distribution. The latter approach is more difficult to implement, but truly randomizes weights and allows the best signal flow by matching the initialization of other layers. We, thus, opt for this approach.

We call a layer *auxiliary* if the decision is insignificantly affected by the reset ($\kappa_l(\boldsymbol{\theta}) = 0$) and *critical* ($\kappa_l(\boldsymbol{\theta}) = 1$) if the distance between decisions changes significantly. Realistically, the criticality for most layers does not lie on the extremes of this spectrum, but anywhere in between. Due to a significant variance (standard deviation of up to 45% on a few layers in specific models) in criticality on some layers, we repeat experiments with different random seeds and report the mean over three trials. For computational reasons, we evaluate layers on a subset of 10,000 random images from the *ImageNet ILSVRC-2012 challenge* validation set (Russakovsky et al., 2015).

In summary, our methodology is derived from Zhang et al. (2022) but shows key differences. First, they analyzed residual blocks as a whole – in contrast, we more meticulously randomize individual layers, which include different convolution and fully connected layers.[2] Furthermore, our methodological change can be evaluated in an unsupervised manner and, more importantly, is also sensitive to changes in the probability distribution including variations in errors (we refer the reader to (Geirhos et al., 2020b) for a discussion on why this is important). As such, it provides a more holistic measurement of consistency in the decision before and after randomization, well beyond correct predictions.

6.2.2 Model Zoo

We use the model zoo of Chapter 8 with a few replacements and select $N = 50$ different `ResNet-50` image classification models (He et al., 2016) that all utilize the exact same network architecture but were trained in different manners. A list of all models can be found in Appendix A.2. We separate our model zoo into these color-coded categories:

- **baseline**: a model following the original training (He et al., 2016).
- **augmentation**: various data regularization techniques (Hendrycks et al., 2020, 2022, 2021a; Jaini et al., 2024; Müller et al., 2023; Modas et al., 2022; Li et al., 2021; Erichson et al., 2022; Cubuk et al., 2019, 2020; Lim et al., 2019; Geirhos et al., 2019).
- **adversarial training (AT)**: training against a projected gradient descent (PGD) adversary (Madry et al., 2018; Salman et al., 2020) – which, technically, is also a form of augmentation, but we observed significantly different behavior of these models and treat them separately.
- **self-supervised learning (SSL)**: unsupervised training on *ImageNet-1k* (Chen et al., 2020a, 2021b; Caron et al., 2021, 2020) with supervised finetuning of the classification-head
- **improved training recipes**: new recipes in *Pytorch Image Models (timm)* (Wightman, 2019; Wightman et al., 2021) and *PyTorch* (Paszke et al., 2019; Vryniotis, 2023) combining multiple hyperparameter optimizations into supervised training.

[2] With the exception of batch-normalization layers (Ioffe & Szegedy, 2015) to avoid signal propagation issues

6.3 Critical Layers in ImageNet-trained ResNet-50

Due to the wide use and availability of pre-trained models, all our results are obtained on `ResNet-50` (He et al., 2016), which is introduced in Section 2.3.4. Recall that this architecture consists of a stem (denoted by *[0.*]*), 4 stages (denoted by *[1-4.*]*), a pooling layer, and a fully connected classification head (denoted by *[Head]*). Each stage consists of several *residual* Bottleneck-Blocks which include learnable 1 × 1 convolutions (*conv1*, *conv3*), 3 × 3 convolutions (*conv2*), as well as batch-normalization layers (Ioffe & Szegedy, 2015). The first residual block in each stage is special, as it downsamples by a strided convolution, thus, adding a learnable layer on the skip connection (*downsample*). All models in our study were trained on *ImageNet-1k* (Deng et al., 2009; Russakovsky et al., 2015). Please see Section 2.5.4 for details about this dataset.

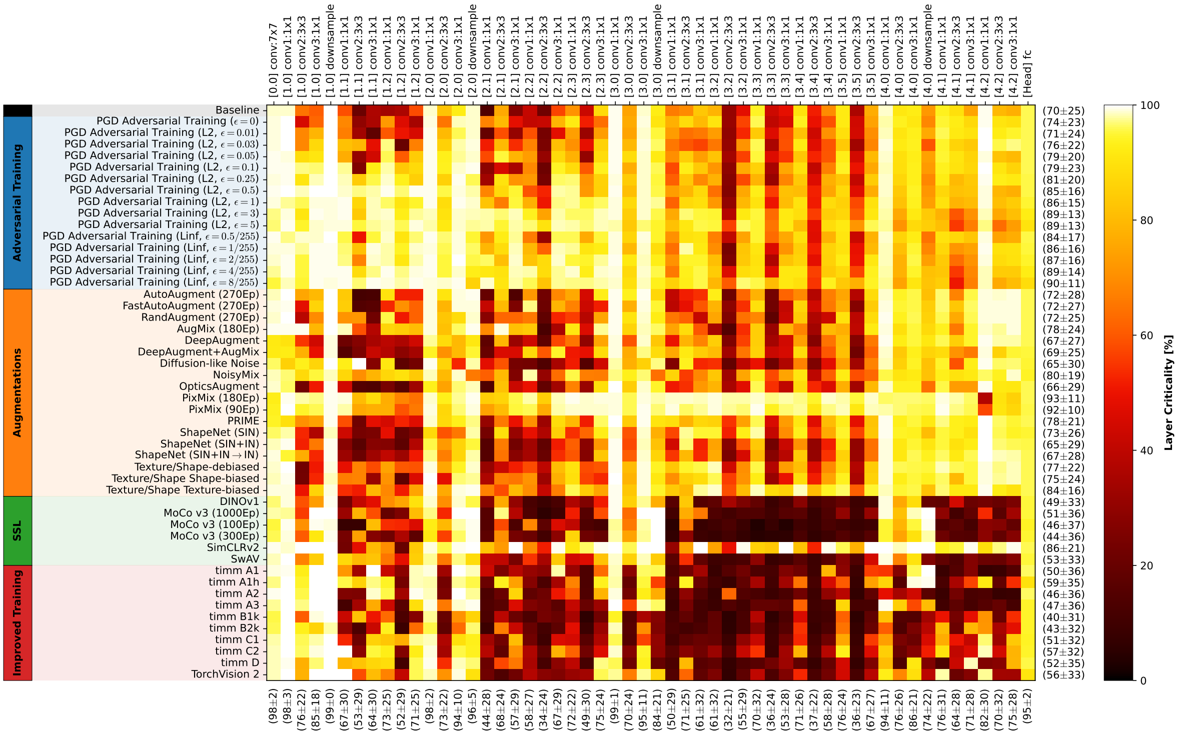


Figure 6.1: Training methods determine what layers become critical. We measure the criticality of ($N = 50$) different `ResNet-50` models that all utilize the same exact network architecture and training data (*ImageNet-1k*) but differ in their training methods. Darker spots denote layers that are *critical*, *i.e.*, resetting them results in significantly different predictions and decreased performance. Brighter spots are *auxiliary*, *i.e.*, resetting these layers does not significantly affect the model. We denote the average (mean±std) layer criticality for both, a model across layers on the right, for a layer across models on the bottom.

General observations The results in Figure 6.1 clearly show that the training method influences which layers become critical – despite that all models were trained on the same training set (some with more extreme forms of data augmentation utilizing a negligible amount of extra data).

In contrast to previous findings (Zhang et al., 2022; Chatterji et al., 2020), we observe that **no layer is always auxiliary** across training methods. For instance, we observe an average criticality of just 36% for a spatial convolution layer (*[3.5] conv2*). Yet, if we randomize the same layer in a *PixMix* (Hendrycks et al., 2022) model, we observe a strong criticality of 95%. On the contrary, we do find layers that are always critical. As expected, these include the initial stem convolution (*[0.0] conv*) and the classification head (*fc*). Beyond, we find that most first convolution layers in each stage

(*[*.0] conv1*) are critical – yet the number of outliers increases with depth. Similarly, we find that the downsampling convolution (*[*.0] downsample*) in each stage is often critical. In stage 1, this layer is critical for all models, but again, the criticality of deeper downsampling convolutions depends on the training strategy used for the model. Lastly, akin to Gavrikov & Keuper (2023b), we find that pointwise convolution layers tend to be more critical than spatial convolution layers (except for the stem). For all other layers, criticality depends on the training method. In the following paragraphs, we analyze specific categories of training methods.

Adversarial Training (AT) This training technique intends to increase the robustness of neural networks by training on adversarially perturbed training samples (Madry et al., 2018). To avoid perturbations that cause a shift in semantic meaning, perturbations are often constrained by an attack perturbation budget ϵ for some L^p norm. We consider AT using a PGD attack (Madry et al., 2018), which optimizes the perturbations over several iterations (here: three).

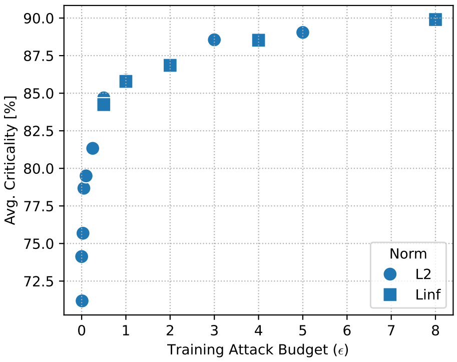


Figure 6.2: Adversarial training increases the average criticality proportional to the training attack budget ϵ. Please note, that reported ϵ values for L^∞ norms are short for $\epsilon/255$ (but not for the L^2 norm). Each marker represents one model.

We find that AT increases the criticality proportional to the attack budget ϵ during training. To make this more tangible, we average the criticality over all layers and show the results in Figure 6.2. We do not observe differences between training that utilizes L^2 or L^∞ norms for attacks. Our findings suggest that neural networks utilize more of their capacity under increasing training attack strength. This augments previous findings that showed similar insights through accuracy improvements with larger networks (Madry et al., 2018) or richer representations in convolution filters that we made in Chapters 3 and 4. AT primarily increases the criticality in the layers of the first and second stages and slightly in the third stage. The criticality of layers in the fourth stage is barely affected but rather decreases compared to the baseline.

Augmentations Compared to AT, the influence of different augmentation strategies seems weaker. We do find that many augmentations tend to increase the average criticality, *i.e.*, they do occupy more of the network capacity – but most changes are rather small. Affected layers seem to fluctuate by method, but we find that all augmentation methods consistently increase the criticality of some of the deepest layers (*[4.0] downsample, [4.1] conv2, [4.2] conv2/3*). Like the organization of the visual cortex (Hubel & Wiesel, 1959, 1979; Olshausen & Field, 1997), deeper layers in neural networks are associated with activations of more complex features. For images, these tend to correspond to shapes as opposed to texture information that is captured by early layers (LeCun et al., 1989a; Yosinski et al., 2014; Zeiler & Fergus, 2013). Indeed, we will show that our tested augmentations increase in their shape responses in Chapter 8. Thus, a reasonable hypothesis is that the increased criticality in deeper layers correlates with stronger shape representations.

Strong outliers to our observations are the *PixMix* models (Hendrycks et al., 2022). These models have the highest average criticality in our model zoo without a single auxiliary layer. The augmentation technique has been shown to improve multiple safety dimensions beyond test accuracy and combined with the findings of Chatterji et al. (2020) it may indeed suggest that a higher degree of criticality correlates with "better" neural networks.

Self-Supervised Learning (SSL) SSL has been shown to produce rich representations for many downstream tasks (see Oquab et al. (2024) for a recent example) as the granularity and implicit biases of annotations do not confine it. Interestingly, we find that *DINO* (Caron et al., 2021), *MoCo v3* (Chen et al., 2021b), and *SwAV* (Caron et al., 2020) severely differ in their criticality measurements from the supervised models we discussed before. These SSL models have a large presence of auxiliary layers in the last three stages but (slight) increase in criticality of early layers. This suggests that SSL learns shorter decision rules and, thus, seeks to strengthen early operations. However, we again find an outlier: *SimCLRv2* (Chen et al., 2020a) has no auxiliary layers and shows a somewhat similar distribution of criticality to *PixMix* (Hendrycks et al., 2022).

Improved Training Recipes The standard 90 epoch training with simple augmentations (Krizhevsky et al., 2012) was shown to be suboptimal for many models, including `ResNets` (Wightman et al., 2021). Modern training recipes utilize a significantly more complex set of training tweaks – often in combination with longer training schedules (Wightman et al., 2021; Vryniotis, 2023). Similar to our findings on SSL, these improved training methods appear to shift model decisions to early operations while relaxing deeper operations. Most notably, this increases the criticality of the first and second residual blocks. These models show great improvements in generalization on many datasets, but at the same time, they even further prioritize texture information (Gavrikov & Keuper, 2024). Ultimately, this may suggest that these techniques better fit *ImageNet-1k*-like problems but may not provide improvements for representations beyond.

6.3.1 Criticality under Distribution Shifts

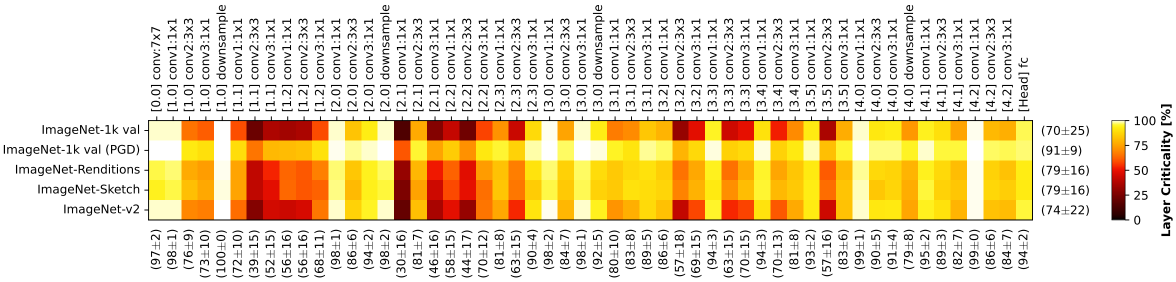


Figure 6.3: Criticality under different test data. We measure criticality of a baseline `ResNet-50` against the *ImageNet* validation set, *ImageNet* validation set under PGD attack, *ImageNet-v2*, and *ImageNet-Sketch*.

It is well known that different parts of a neural network are responsible for different kinds of features (Zeiler & Fergus, 2013). We may, thus, assume that a covariate shift in the test data also results in a different utilization of the layers. To assess if layer criticality is indeed influenced by test data, we repeat the layer criticality evaluation of the baseline model (He et al., 2016) on *ImageNet-v2* (Recht et al., 2019), *ImageNet-Sketch* (Wang et al., 2019), *ImageNet-Renditions* (Hendrycks et al., 2021a), and a PGD-attacked (Madry et al., 2018) ($L^\infty, \epsilon = 1/255, \alpha = 0.01/0.3, t = 40$) *ImageNet* validation set. We then compare these results to the previous measurements on the *ImageNet* validation set to determine if there are any significant differences.

Our findings, presented in Figure 6.3, highlight shifts in layer criticality under distribution shifts. Specifically, the *average* criticality increases across all datasets and appears correlated with the extent of the covariate shift relative to the training data. For the relatively similar *ImageNet-v2* (Recht et al., 2019), we observe only a 4% increase in criticality. In contrast, datasets such as *ImageNet-Rendition* (Hendrycks et al., 2021a) and *ImageNet-Sketch* (Wang et al., 2019), which are substantially more distinct from the training data, exhibit a 9% increase. Notably, adversarial attacks designed to maximize covariate shifts lead to the most pronounced increase in criticality (21%).

While the average criticality shifts, the layers that remain critical or become non-critical do not fundamentally change. This effect is particularly pronounced under adversarial attacks, resulting in nearly uniformly critical layers. We anticipate that analyzing the criticality of models employing dynamic feature routing, such as mixture-of-experts (MoE) (Jacobs et al., 1991), could reveal changes in the distribution of criticality.

6.3.2 Criticality and Generalization

Next, we attempt to connect criticality with generalization (*i.e.*, we sanity-check the claims of Chatterji et al. (2020)). We cannot fully replicate the methodology of Chatterji et al. (2020) which correlates the difference between training and test loss against average criticality, as we do not have access to the training loss (or accuracy) and measuring it faithfully (including all forms of regularization) would be computationally infeasible for our large model zoo. Instead, we retreat to a simpler proxy, by correlating accuracy on the *ImageNet-1k* validation set against the average criticality.

Our results in Figure 6.4 are quite sobering. Ignoring the category labels, we observe a moderate Spearman's $R = -0.46$, but when we remove adversarially-trained models, the correlation is faint (Spearman's $R = -0.17$). Thus, there is a low likelihood of a causal connection between *ImageNet-1k* accuracy (and generalization as a whole) and layer criticality.

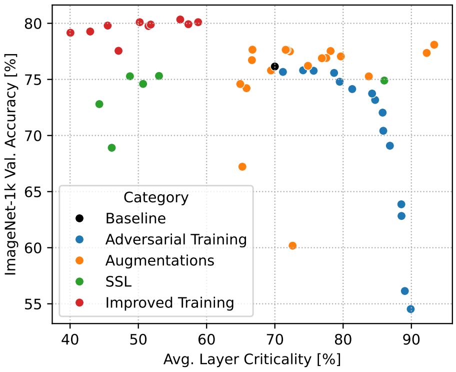


Figure 6.4: Correlation between average network criticality and performance on *ImageNet-1k*. Each marker represents one model.

6.3.3 Criticality and Filter Quality

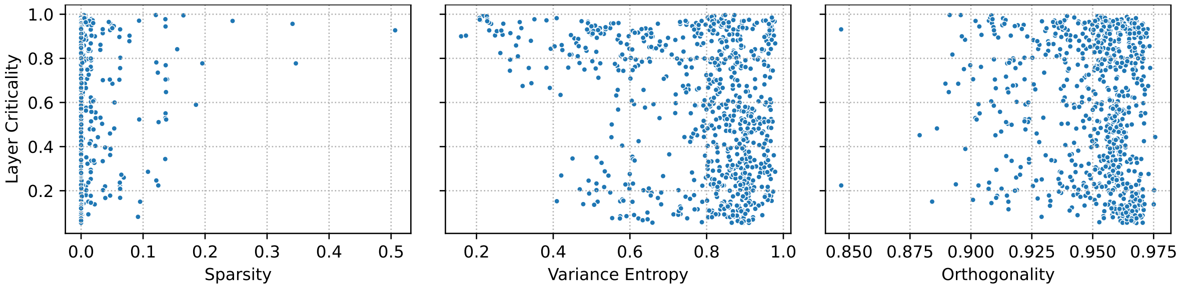


Figure 6.5: Correlation between the filter quality and criticality of the respective layers. We correlate sparsity, variance entropy (normalized by number of kernels), and orthogonality of 3 × 3 convolution layers in all our `ResNet-50` models with the criticality of the respective layer. Each marker represents one layer.

In this section, we aim to identify the weight-space properties of critical layers by leveraging our filter quality metrics from Chapters 3 and 4. Since these metrics are specifically designed for convolutional layers with 3 × 3 kernels, we focus our analysis on such layers. For these layers, we measure three key attributes next to criticality: *sparsity* (Equation 3.4), *variance entropy* (Equation 3.6), and *orthogonality* (Equation 4.2).

Our findings in Figure 6.5 reveal no clear linear relationship between criticality and the quality metrics. However, a constrained mutual dependence can be observed. For example, layers with higher sparsity tend to exhibit greater criticality, although criticality cannot be reliably inferred for non-sparse layers. This observation aligns with our intuition: sparse kernels effectively mask out input features, and increased sparsity leads to more distinct masking patterns.

Similarly, layers with lower entropy appear to be more critical, whereas high-entropy layers do not allow us to estimate criticality. This, too, reflects the increased individuality of low-entropy layers compared to their high-entropy counterparts.

Surprisingly, even high-entropy layers can be critical despite the fact that these should be most similar to random initializations. It seems that they still may capture subtle patterns or successive layers may have highly specialized in their patterns – we show an extreme example of this in (Gavrikov & Keuper, 2023b), where we train pointwise convolutions but leave all spatial convolution layers frozen. This conditional dependency between layers is captured by layer criticality but not by our filter quality metrics, which treat all layers independently.

For orthogonality, we do not find any meaningful relationship.

6.4 Summary

Our study offers an additional angle to our survey of learned representations beyond filter patterns (Chapters 3 to 5) by quantifying the criticality of learned parameters. In this chapter, we specifically focused on the common impact of training methods over model populations on layer criticality and extended previous findings about the complexity of learned decision rules in image classification models Our results show that the training method leaves distinct patterns in the decision rule. In the studied `ResNet-50` models, this distinctiveness prevails even under changes in test data. However, overall, the criticality of layers seems to be correlated with the magnitude of covariate shifts in the test data.

Given the large number of auxiliary layers in some of the studied models, we can assume that the studied representations contain a lot of noise, potentially masking actual differences in critical parts of the network. Unfortunately, on the one hand, we cannot generally identify critical layers by our data-free filter quality metrics introduced in Chapters 3 and 4 (only for very specific cases), making it hard to exclude auxiliary layers from representation studies without having to compute criticality. However, criticality requires at least one forward pass per layer and is conditionally dependent on the parameters. For instance, if criticality were used to greedily prune all auxiliary layers, the complexity would be in $\mathcal{O}(L^2 * F)$, where L is the number of layers to be tested, and F the number of forward passes per layer.

On the other hand, there seems not to be a conclusive link between criticality and generalization either, to just use criticality as a metric – at least as quantified in our study. One pitfall may be that assessing the generalization of neural networks through benchmarks is, of course, tricky, as models may specialize in specific settings at a cost to others. Capturing all of these nuances in static test datasets is often an unrealistic journey. For example, models may excel in accuracy on the "clean" *ImageNet-1k* data, but drop in performance if the same samples are corrupted (Hendrycks & Dietterich, 2019). In that sense, the decision rules' complexity may still be valuable, as it offers a better (relative) assessment of generalization by focusing on the inner mechanics of models as opposed to benchmark results. Yet, it remains to be seen how the observed correlations of Chatterji et al. (2020) (and our lack of correlations) hold on larger model zoos and wider notions of generalization beyond clean accuracy on the *ImageNet-1k* validation set.

Further, even if criticality and generalization are orthogonal to each other, the complexity of the decision rule may still be relevant and offer a better explanation of phenomena that were linked to (adversarial) robustness before. For instance, these include more human-likeness as we will show in Chapter 7, better calibration (Grabinski et al., 2022a), and transferability (Salman et al., 2020) under adversarial training.

Part II
Visual Perception Biases

Our second viewpoint on generalization in this thesis is the model perception, specifically, biases for or against visual features that diverge between human observers and models as introduced in Section 2.9. Such biases have also often been proposed as explanations of why models fail to generalize (Geirhos et al., 2019; Wang et al., 2020a; Subramanian et al., 2023). We refer the reader to Section 2.9.4, where we discussed a few of them.

Bridging our research from the first part, we start with an investigation of the connection between adversarial training and alignment with the visual perception of human observers (Chapter 7). Then, we continue with an investigation of the correlation between biases in *ImageNet*-models and multiple forms of generalization (Chapter 8). Finally, we study visual biases in the current generation of deep learning models: *large vision-language models*. We show that the perception, including biases in these models, can be intuitively controlled at inference time via simple prompts, offering a powerful and cost-effective alternative to regularization in training (Chapter 9).

Chapter 7

Adversarial Training and Alignment to Human Vision

Having analyzed the learned filter space, particularly for robust, adversarially-regularized models in Part I, we now investigate how adversarial training influences alignment to human vision in *ImageNet* models.

This chapter is based on "An Extended Study of Human-Like Behavior Under Adversarial Training", presented at the CVPR Workshop on Adversarial Machine Learning on Computer Vision: Art of Robustness, 2023 (in-proceedings) (Gavrikov et al., 2023). As the first author, Paul Gavrikov collected the models, developed the code base, performed the experiments, created the plots, and wrote the paper with input from Janis Keuper and Margret Keuper.

Code: `https://github.com/paulgavrikov/adversarial_training_vs_humans`

7.1 Introduction

Models trained on *ImageNet* (Deng et al., 2009; Russakovsky et al., 2015) have demonstrated significant misalignment with human vision in several ways. These include poor generalization under distribution shifts (Geirhos et al., 2018), inconsistent predictions compared to humans, including in their errors (Geirhos et al., 2020b), and inherent biases to other features such as a *texture bias* (Geirhos et al., 2019).

This raises a key question: *are current models progressing towards better alignment with human vision?* To explore this, Geirhos et al. (2021) conducted a large-scale analysis of state-of-the-art models, focusing on their alignment with human vision. Their findings revealed that robust models improve in some aspects; for instance, adversarial training (AT) shifts the texture bias in CNNs toward shape-based decisions (also shown by Zhang & Zhu (2019); Chen et al. (2022a)). However, their analysis was limited to `ResNet-50` (He et al., 2016) trained on *ImageNet* using AT against an L^2-bounded adversary.

We argue that adversarial robustness is also a form of alignment to human vision and revisit and extend their analysis to the more common L^∞ setting for AT. Additionally, we include a wider range of CNN architectures, such as `ResNet-18` and `WideResNet-50-2` (Zagoruyko & Komodakis, 2016), alongside transformer-based models (`XCiT-S/M/L12`) (Ali et al., 2021), which lack distinct inductive biases compared to CNNs, as well as hybrid models (`ConvMixer-768-32`) (Trockman & Kolter, 2023). We evaluate these models under standard training and AT with varying norms and budgets, assessing their alignment with human vision. Taken together, our methodology allows us to understand if the selection of architecture and form of robustness biased the previously reported results.

Furthermore, we investigate adversarially-trained networks' generalization under covariate shift from a frequency perspective (see Section 2.1 for an introduction to Fourier transform). Specifically, we analyze the frequency spectra of out-of-distribution (OOD) image datasets to address two questions:

(1) Why does adversarial training cause accuracy decay on some datasets?

(2) How does AT influence the shape bias?

Taken together, our study aims to provide another angle to our representation analysis of adversarially robust models in Part I through the lens of alignment to human vision.

We summarize our contributions as follows:

- We conduct a multi-faceted comparison of human visual alignment under adversarial training, augmenting our weight analysis in Part I, particularly in Chapter 4. In this study, we investigate the impact of network scale, the L^p-norm used in adversarial training (AT), and the model's inductive bias on this alignment.
- Our analysis of image frequency power distributions provides a novel explanation for the degradation of out-of-distribution (OOD) performance under adversarial training. This framework also transfers to the observed shift from texture bias to shape bias induced by adversarial training.

7.2 Background

We refer the reader to Section 2.8.2 for an introduction to adversarial robustness, including a discussion of adversarial training. Additionally, we have exclusively studied CNNs (Section 2.3) so far, and this is the first chapter to include results on transformer-based models. We introduce Vision Transformers (ViTs) in Section 2.4, including important concepts for this section, such as "patchification".

7.2.1 Measuring Human Alignment in Vision

Geirhos et al. (2021) propose to measure the similarity or alignment to human vision through three axes: (1) out-of-distribution (OOD) generalization, (2) feature biases (specifically the texture/shape bias (Geirhos et al., 2019)), and (3) consistency in predictions (correct and wrong) with humans (Geirhos et al., 2020b).

To measure OOD generalization, they propose benchmarking against a set of 12 *ImageNet* modification datasets (Geirhos et al., 2018) at various intensities/conditions. At first glance, this may sound familiar to *ImageNet-C* (Hendrycks & Dietterich, 2019), but benchmarks a different set of "corruptions": *(the absence of) colour*, *contrast (changes)*, *eidolon I/II/III* (Koenderink et al., 2017), *false-colour*, *high/low-pass (frequency filtering)*, *phase-scrambling*, *power-equalisation*, *rotation*, *uniform-noise*. Additionally, they propose to benchmark against a set of five datasets related to the identification of shape/texture-bias (Geirhos et al., 2019): *stylized*, *edge*, *silhouette*, *texture/shape cue-conflict*, and *sketch* (the latter provided by (Wang et al., 2019)). All datasets contain samples that belong to 16 *ImageNet* (super-)classes and are therefore classifiable by *ImageNet* models. For all datasets, the authors include a human baseline obtained in lab settings over 4–10 annotators that can be used to select "reasonable" levels of corruption. The *texture/shape cue-conflict* dataset is of particular interest, as neural networks are not only prone to overfit but – at least in the vision domain – they also tend to be biased to texture features in images rather than shapes, which does not align with human vision (Geirhos et al., 2019). For example, an image of an elephant with an overlaid lion texture will most likely result in a prediction as "lion." At the same time, most humans would predict "elephant" as the true label when given a choice between both. It is worth noting that *ImageNet* can be largely accurately classified solely based on texture (Brendel & Bethge, 2019). We show examples of all datasets in Figure 7.1.

As an additional metric to quantifying performance through accuracy, Geirhos et al. (2020b) propose to evaluate the agreement in predictions. In particular, they analyze false predictions (*error consistency*) as well as the intersection rate of predictions between correct prediction of humans and models (*observed consistency*).

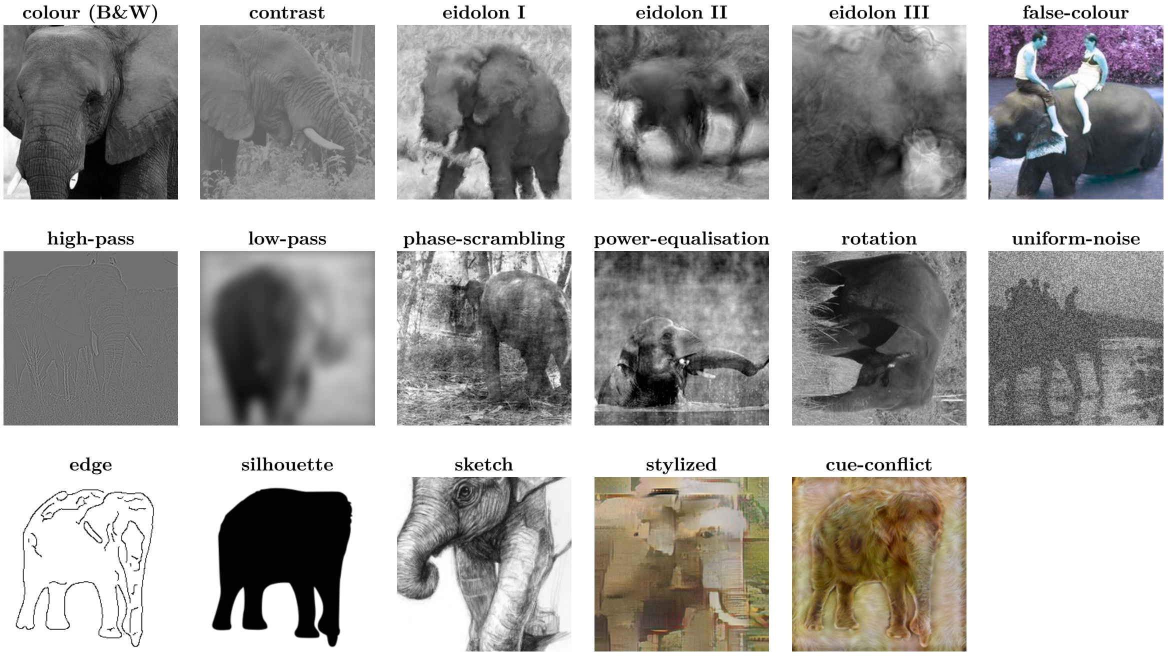


Figure 7.1: Test samples. We show one sample for the “elephant” class of each test dataset. **Image Source:** (Deng et al., 2009; Geirhos et al., 2019, 2018, 2021; Wang et al., 2019)

The authors maintain a leaderboard of the most “human-like” models, which is currently dominated by transformer-based architectures such as `ViT` (Dosovitskiy et al., 2021; Zhai et al., 2022a) and `CLIP` (Radford et al., 2021), or large convolutional neural networks (Yalniz et al., 2019; Kolesnikov et al., 2020). A common denominator is that they are all being pre-trained on massive datasets.

7.3 Methodology

To study the likeliness to human-like behavior of adversarially-trained models, we use publicly available checkpoints and perform an analysis according to the setting proposed in (Geirhos et al., 2021).

Model zoo We analyze pre-trained `ResNet-18, ResNet-50, WideResNet-50-2` models trained against L^{∞}-bound adversaries with $\epsilon \in \{0.5/255, 1/255, 2/255, 4/255, 8/255\}$, and against L^2-bound adversaries with $\epsilon \in \{0.01, 0.03, 0.05, 0.1, 0.25, 0.5, 1, 3, 5\}$, and clean baselines (all provided by (Salman et al., 2020)). Further, we analyze `XCiT-S/M/L12` transformer models trained against L^{∞}-bound adversaries with $\epsilon = 4/255$ provided by (Debenedetti et al., 2023) and a clean `XCiT-S.`[1] baseline provided in *Pytorch Image Models (timm)* (Wightman, 2019) Lastly, to better understand the differences between CNNs and transformers, we also analyze a clean `ConvMixer-768-32` checkpoint, again obtained from *timm*, serving as “middle ground”. All models were trained on *ImageNet-1k* without any additional pre-training.

Reported metrics For all models, we measure the accuracy of all datasets by reporting the mean overall conditions in the dataset where average human performance was above 20% accuracy. Lastly, we determine the *observed* and *error consistency* against human annotators as a mean over all datasets and conditions.

Comparing AT under different L^p-norms We want to compare L^2-AT models to models that are trained using L^{∞}-AT. Comparing these two training types is not straightforward due to the

[1] Clean pre-trained `XCiT-M/L12` with the same configuration were not available.

different types of perturbations they cause (as discussed in Section 2.8.2). As the L^2-norm penalizes the Euclidean distance, perturbations can locally be more severe than under L^∞. Yet, if the perturbation magnitude increases, the area of perturbations has to decrease under L^2-norm. At the same time, attacks under the L^∞-norm can add perturbations to the entire image without constraints except for the magnitude. Thus, there are multiple options for choosing comparable budgets between the two norms. We choose a straightforward way and select budgets for both norms that approximately result in the same clean accuracy:

ϵ under L^2	0.1	1	3	5
ϵ under L^∞	0.5/255	1/255	4/255	8/255

As we have more checkpoints for L^2-AT training, we only use a subset of those when comparing L^∞ with L^2 models, but we use all checkpoints for other analyses.

Power spectrum of datasets In the last part of our analysis, we impose a frequency perspective on OOD performance and shape bias under AT. To back this analysis, we plot the power distribution in Fourier space (see Section 2.1) for each OOD dataset and clean *ImageNet* validation samples belonging to the same classes. Then, we compare each OOD distribution to the clean distribution to understand where shifts in the frequency distribution are located. We obtain the frequency distribution plots as introduced in (Durall et al., 2020): we compute the log-scaled FFT power spectrum and compute the radial integral under increasing frequency, resulting in a frequency power distribution (Figure 7.2). For comparability, we scale the resulting distributions by their area under the curve.

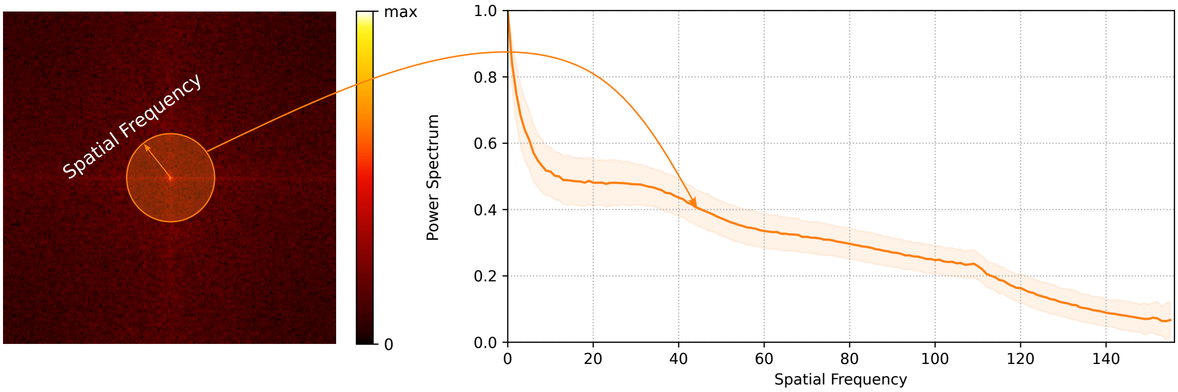


Figure 7.2: Generation of power spectrum plots. Each frequency measurement in the spectrum plot corresponds to the integral over the centered FFT power spectrum (frequency increases from the center to outer edges) up to that particular frequency.

7.4 Results

This section presents an analysis of alignment under adversarial training (AT), particularly the potential influence of network scale, adversarial attack norm, and inductive bias on correlation alignment.

7.4.1 Scale: Width and Depth of L2-bound CNNs

First, we want to evaluate the effect of L^2-training on CNN architectures: `WideResNet-50-2`, `ResNet-50` and `ResNet-18` (Figure 7.3, top row), which allows investigating the effect of L^2-AT depending on model depth and width.

We observe that increasing both width and depth improves clean performance and performance on all OOD datasets, as well as the observed and error consistency. As such, `WideResNet-50-2` performed best on clean performance, OOD mean, and observed/error consistency. It is also worth noting that switching from `ResNet-50` to `WideResNet-50-2` has a smaller impact on performance than switching from `ResNet-18` to `ResNet-50`. Also, we observe that in some cases `ResNet-18` shows opposite trends with respect to training budget than 50-layer deep `ResNets`, *e.g.*, for *power-equalisation*. Still, `ResNet-18` performs best on the *edge* dataset for large training budgets (not shown due to space limitations). Overall, this suggests that increasing the model scale of L^2-bound adversarially-trained models correlates with increased alignment.

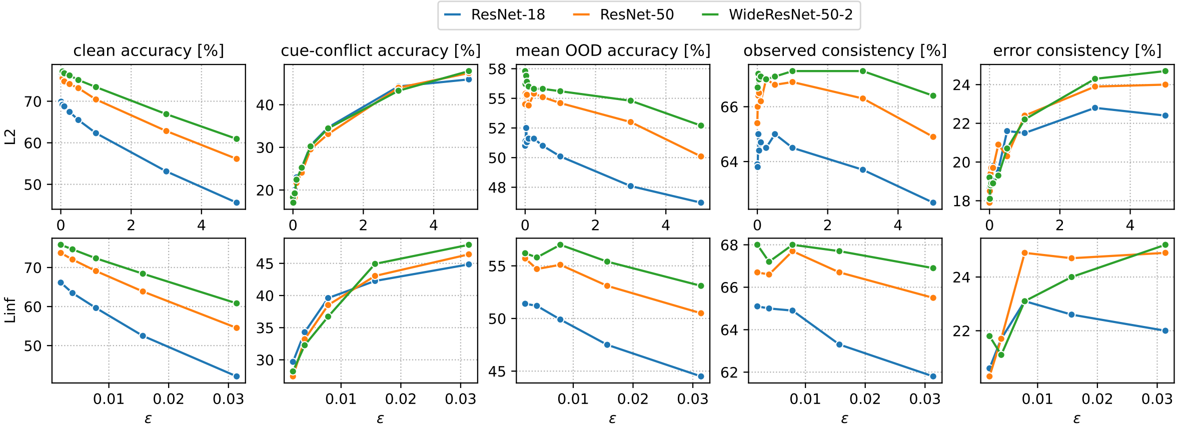


Figure 7.3: Scale comparison. Performance of L^2 vs. L^∞-AT-trained `ResNet-18`, `ResNet-50`, `WideResNet-50-2` on clean data, texture/shape-bias cue-conflict datasets, the average mean of all OOD datasets, and observed/error consistency compared to humans under increased training attack budget ϵ.

7.4.2 Norm: L2- vs. Linf-bound AT

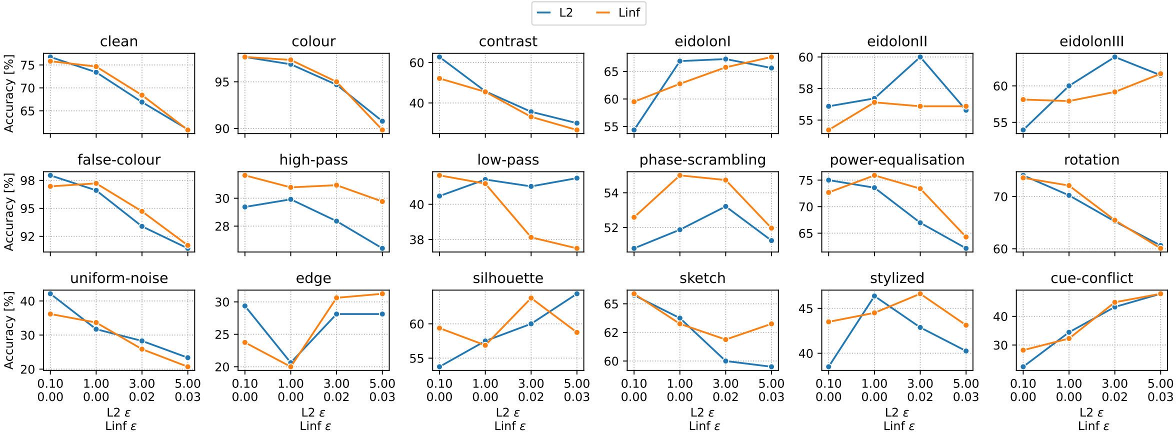


Figure 7.4: Norm comparison. Comparison of performance on OOD datasets between robust `WideResNet-50-2` trained against L^2- (upper ϵ values) and L^∞-bound (lower ϵ values) adversaries under increasing training budget ϵ. ϵ are selected in a way that clean accuracy approximately matches between the norms.

Next, we compare how L^2-AT relates to L^∞-AT concerning human-like reasoning. Exemplarily, we analyze the trend under comparable budgets of a `WideResNet-50-2` (Figure 7.4). We observe that on some datasets, there is barely any perceivable difference between norms as the budget increases (*colour, contrast, false-colour, uniform-noise, rotation, cue-conflict*), but there are cases where one norm or the other clearly performs better. L^∞-robust models seem to be more robust against *high-pass*, *phase-scrambling*, and *power-equalisation.* On the other hand, L^2-robust models appear to perform better on *low-pass* and *eidelonII/III.* Lastly, there are some inconclusive settings where one performs better depending on the budget (*silhouette, eidolon I, stylized*). Besides *cue-conflict*, none of the OOD categories clearly benefit from AT for `WideResNet-50-2`. These observations only partly transfer to other CNN architectures in Table 7.1. In general, Figure 7.3 (bottom) shows a similar trend for L^2 and L^∞-AT, and all results support the finding that the *cue-conflict* score increases consistently under both types of AT, *i.e.*, the behavior becomes more human-like towards shape bias in both cases. Therefore, we conclude that the more commonly used L^∞-AT is equally effective in inducing human-like behavior in CNNs, with respect to *cue-conflict*, and consistency.

7.4.3 Inductive Bias: AT on CNNs vs. Transformers

Table 7.1: Detailed comparison of accuracy and consistency. For robust models, we only report $\epsilon = 3$ (L^2) and $\epsilon = 4/255$ (L^∞) for brevity. Models without adversarial training are highlighted in gray. All results are in [%], and **bold** values indicate the best performance amongst all models per column.

Model	Clean	Out-of-distribution Accuracy																		Consistency	
		colour	contrast	eidolon I	eidolon II	eidolon III	false colour	high pass	low pass	phase scr.	power equal.	rotat.	uniform noise	edge	silh.	sketch	styliz.	cue conflict	**Mean**	correct	error
`R18`	69.79	95.47	71.88	47.50	51.88	49.38	93.39	32.66	37.73	48.21	61.25	68.36	34.22	18.12	41.88	59.00	36.00	19.61	51.00	63.90	18.60
`R18` (L^2)	53.12	86.25	27.50	60.25	49.53	51.46	85.27	24.14	35.39	47.50	51.96	55.23	20.78	27.50	61.25	51.12	39.50	44.30	47.60	63.70	22.80
`R18` (L^∞)	52.49	84.69	23.62	61.12	50.94	51.67	83.57	25.86	35.55	47.05	56.07	55.23	18.83	26.88	56.88	50.88	40.62	42.27	47.50	63.30	22.60
`R50`	75.80	97.19	83.62	49.12	52.66	51.04	95.62	33.67	38.98	49.11	70.71	73.91	37.97	23.75	48.12	61.25	34.38	17.42	54.50	65.40	17.90
`R50` (L^2)	62.83	92.81	32.12	66.12	56.41	62.71	90.71	26.17	40.31	53.84	63.57	63.75	26.09	25.62	60.62	59.38	41.75	43.98	53.00	66.30	23.90
`R50` (L^∞)	63.86	91.25	29.25	64.25	54.37	57.50	91.07	30.70	38.52	53.39	68.04	64.06	26.25	25.62	58.75	60.50	43.25	43.05	53.10	66.70	24.70
`WRN50-2`	76.97	98.28	82.38	51.00	54.69	54.17	97.23	34.92	40.62	50.98	75.18	75.39	42.27	28.75	56.88	64.12	36.50	18.28	57.30	67.20	19.20
`WRN50-2` (L^2)	66.90	94.69	35.62	67.25	60.00	63.96	93.39	28.36	41.02	53.21	66.96	65.23	28.28	28.12	60.00	60.00	42.88	43.28	54.80	67.30	24.30
`WRN50-2` (L^∞)	68.41	95.00	33.12	65.75	56.09	59.17	94.73	30.94	38.12	54.73	73.39	65.47	25.86	30.63	63.75	61.88	46.62	44.92	55.40	67.70	24.00
`ConvMixer`	80.16	**99.22**	98.00	50.62	56.72	56.25	98.04	39.77	43.91	56.43	86.25	80.23	**56.02**	26.88	64.38	70.75	44.50	22.73	63.30	69.50	19.50
`XCiT-S`	**81.97**	98.91	**98.88**	55.12	59.38	64.17	**98.75**	**69.84**	**46.72**	**62.14**	**91.07**	**81.41**	55.62	**37.50**	61.88	71.12	**57.75**	25.55	**68.90**	70.90	19.50
`XCiT-S` (L^∞)	72.34	96.88	47.62	66.50	58.91	61.04	96.88	36.95	39.77	56.61	82.14	70.70	40.47	31.87	63.75	70.75	48.75	46.80	60.60	70.00	24.10
`XCiT-M` (L^∞)	74.04	97.34	48.25	66.88	60.16	62.29	96.96	36.80	39.06	57.59	81.43	70.86	41.17	26.25	66.88	71.00	52.62	47.27	60.90	70.40	**24.80**
`XCiT-L` (L^∞)	73.76	98.12	47.38	**69.38**	**60.62**	**64.58**	98.66	41.95	41.72	58.21	84.11	70.62	42.27	35.62	**69.38**	**74.00**	54.12	**48.83**	63.40	**71.10**	22.70
Humans	-	88.67	66.09	60.75	58.28	63.91	88.82	46.43	56.09	55.11	75.89	**84.51**	55.37	**87.12**	**75.31**	**91.62**	47.12	**77.55**	-	-	-

Finally, we expand our analysis to transformer architectures – specifically `XCiT` – for which we only report clean and L^∞-training performance.

On clean training, even the smallest transformer (`XCiT-S12`), which has a comparable number of parameters to `ResNet-50`, performs significantly better than the largest CNN. Contrary to CNNs, it is also able to surpass human performance on *eidolon II/III*, *high-pass* (with an impressive improvement of approx. 35% above CNNs), *phase-scrambling, power-equalisation, uniform-noise*, and *stylized.* Under AT, we largely see the same shift as with CNNs with one exception: while AT improves *stylized* performance of CNNs, it decreases it on transformers. Still, transformers achieve higher accuracies than humans in this category. Of all studied models, the adversarially-trained `XCiT-L12` performs best on *eidolon I-III*, *silhouette, sketch, and cue-conflict.* However, it is also worth noting that it contains 50% more parameters compared to the largest CNN we analyze. In general, we can not conclude that more parameters are always better as we, *e.g.*, see some reduction in error consistency from robust `XCiT-S/M12` to `XCiT-L12`. Further, the clean `ConvMixer`, which contains no self-attention (Equation 2.22) but patch-embeddings (see Section 2.4), shows also an increased *cue-conflict.* Generally, there is a trade-off between CNNs and transformers in almost all studied datasets. We hypothesize that patch embeddings may naturally be slightly shifted toward shape bias compared to traditional CNN processing. This is because they allow convolutions to model long-range relationships through operations on multiple patches, which are necessary for shape understanding, whereas, under normal processing, the convolution can only access the receptive field.

7.4.4 Is Adversarial Training a good Option for Alignment?

While AT does improve the shape bias significantly and shifts the internal decision process toward human-like shape bias behavior, it also decreases OOD performance across many datasets. Most notably, it causes a significant drop in robustness to changes in *contrast*, *rotation*, and *uniform noise* compared to clean training. Interestingly, it also always reduces *high-pass* performance. In the case of XCiT-models, this performance is slightly worse than for humans after AT, although the clean model significantly outperformed humans (by approx. 23%). Generally, we often see that super-human performance against specific corruptions deteriorates after adversarial training. Overall, we see the largest OOD drops after in AT in XCiT, while the ResNets show only minor impairments.

Based on these findings, AT alone is insufficient to shift models toward human-like reasoning in all aspects.

7.5 A Frequency Perspective on Adversarial Training

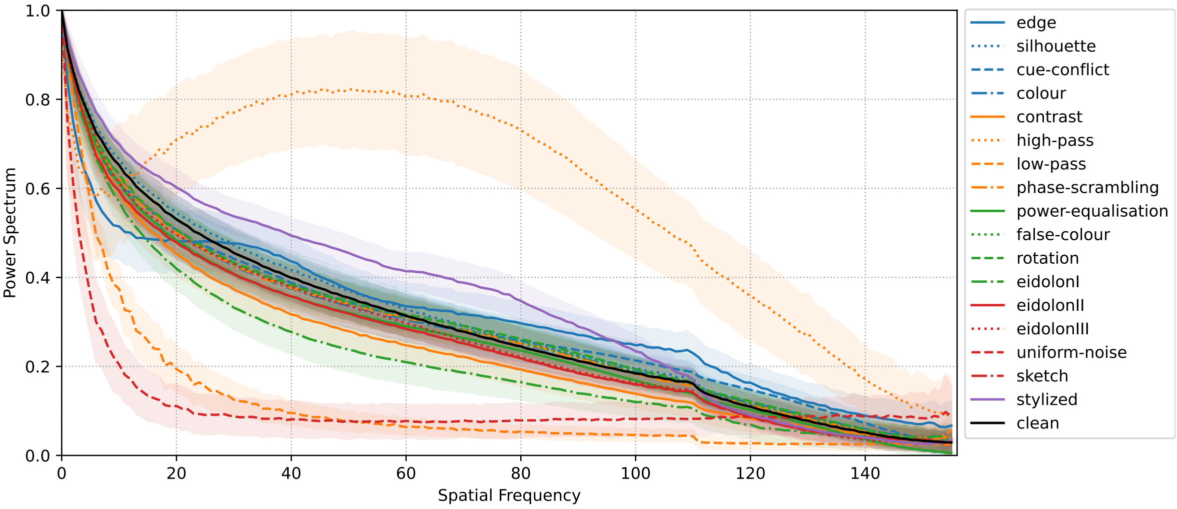


Figure 7.5: Dataset power spectra. Average spectral power distribution of the utilized OOD datasets in comparison to comparable ImageNet validation samples (clean). Frequency increases along the X-axis. Shaded areas denote std.

In Figure 7.5, we plot for all considered OOD image categories their frequency power spectra, radially integrated as described in Figure 7.2, and compare the frequency spectra to the spectrum of the clean training images. This comparison shows that some OOD categories deviate heavily from the natural image distribution in terms of their spectra. This is obviously true for *high-pass* and *low-pass* images as well as for *uniform-noise* and *edge*, where the differences are particularly strong in the high-frequency regime, but it is also visible for *contrast*, *rotation*, and *power equalization*, or *phase scrambling*, with significant differences in the lowest frequencies. Although adversarial attacks might slightly alter the frequency spectrum of attacked images, they are ϵ bounded and will, therefore, not significantly change the frequency distribution over all samples. Thus, it would be natural that AT (*i.e.*, adding more training samples with a spectral distribution similar to one of clean images) would harm the transferability of models to such out-of-distribution categories. In fact, Table 7.1 shows exactly this trend: both types of AT cause a consistent decay in classification accuracy for the OOD categories *high-pass*, *low-pass*, *uniform-noise*, *contrast*, *rotation* and *power equalization*. When the differences are in the low-frequency regime, as for *contrast*, the decay seems to be particularly strong. This observation supports the findings by (Saikia et al., 2021) that AT can harm robustness to other corruption types and provides an initial explanation: Adding more training samples from the original spectral distribution can harm the generalization to other diverging spectral distributions.

Figure 7.5 also shows that some OOD categories have power spectra that are quite similar to the spectra of the original data (and thus of adversarial examples). For these categories, *e.g.*, *eidolon I*, *eidolon II*, *eidolon II*, *false colour* or *cue-conflict*, AT does not lead to a decay in accuracy but can even lead to improvement in some cases. In the following, we will discuss in which cases we might expect this improvement.

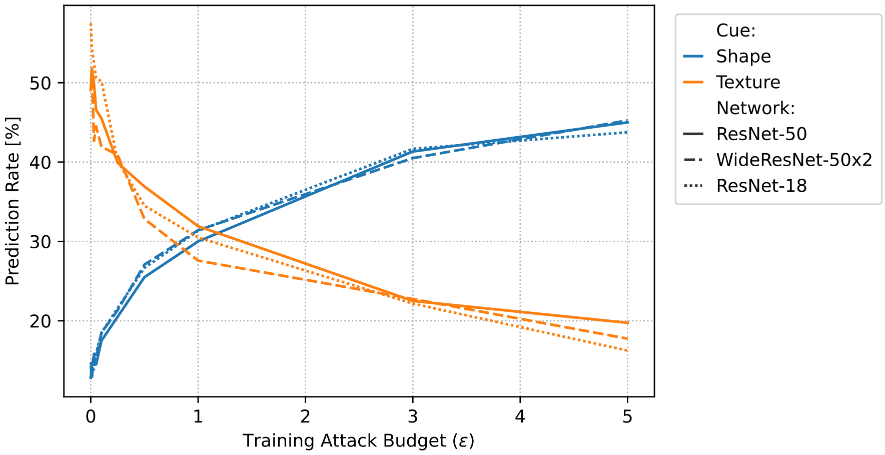


Figure 7.6: Evolution of shape or texture predictions on *cue-conflict* under increasing adversarial training budget (ϵ). Here, we show different `ResNet` models under L^2-training, but we observe similar findings with L^∞. Stronger training attacks result in fewer predictions based on textures and increase predictions based on shapes at approximately the same rate.

From the above observation, we see that the OOD data should share some important properties with the clean data to benefit from AT, *i.e.*, the frequency distribution should not differ too much. At the same time, it has been argued that convolutional neural networks tend to decide based on texture information (Geirhos et al., 2019), which is local and rather mid to high-frequency. Thus, adversarial examples can attack such models by slightly altering the image in these frequency bands. While this may vary by dataset (Maiya et al., 2021; Abello et al., 2021; Bernhard et al., 2021), at least some high-frequency is always present as, *e.g.*, adversarial attacks can be detected in the frequency spectrum (Harder et al., 2021).

To compensate for these attacks, robust models desensitize to high-frequency and instead shift their decisions towards global cues that involve low-frequency information, which can typically be observed in FFT-spectra of perturbations after AT (*cf*. (Grabinski et al., 2022d), Fig. 8). The desensitization of high-frequencies during training also results in more robust models, as shown from various perspectives such as injecting noise patches to inputs (Lopes et al., 2020), blurring feature-maps (Sinha et al., 2020), splitting and regularizing frequency information (Saikia et al., 2021), or low-pass filtering intermediate feature-maps (Grabinski et al., 2022c) during training. There seem to be sufficient indicators to reasonably assume that shifting the decisions toward low-frequency information by removing the focus from high-frequency is at least a necessary ingredient of robustness. Clearly, AT encourages this shift, which can also be seen in weights of convolution filters of robust models (see Chapter 5).

Likewise, the texture/shape bias can also be analyzed from a frequency perspective. Textures contain high-frequency information, while shapes can not be represented without low-frequency bands. As non-robust neural networks naturally prefer high-frequency information for predictions, they also tend to reason based on textures. Under AT, models rely less on local high-frequency information and prioritize the lower-frequent information that encodes global structures such as shapes. This effect can be well seen in the *cue-conflict* performance, where images contain both types of information in the images, and models can choose which information to prioritize. From an information perspective alone, both choices would be acceptable. We want to emphasize that AT does not break the processing of one cue but actually reprioritizes its features. This can be well seen when studying the prediction rate of shape or texture classes (Figure 7.6). With increasing ϵ, the number of texture predictions decreases, but the rate of shape predictions increases at an approximately similar rate.

Ultimately, this perspective does not explain all findings and other mechanics may influence the decision process. For example, *stylized* performance improves under AT for CNNs while the accuracy of transformers, starting at a higher level, is decreasing. It can just provide an intuition of why the model decisions learn to shift towards a more global shape bias – given that the overall spectral distribution remains very similar to the original training data distribution in the *cue-conflict* category.

7.6 Summary

In Part I of this thesis, we have studied the representations of adversarially robust models in weight space, showing salient differences in filter patterns (Chapters 3 to 5) and the utilization of capacity Chapter 6. In this chapter, we have now analyzed these models from a different angle: alignment to human vision.

We have refined the previous experiments of Geirhos et al. (2021) with a closer view of alignment with human vision. Overall, our findings show that adversarial training increases some aspects of alignment, specifically the texture/shape bias (Geirhos et al., 2019) and error consistency (Geirhos et al., 2020b). Generally, we find that robust transformers appear to be more aligned than CNNs, mostly as they perform better on OOD datasets and increasingly predict based on shape cues. However, AT also misaligns model vision in other aspects, which can be seen in degradation against some corruptions that do not seem to affect humans or models trained without AT (*e.g.*, *contrast*).

Finally, we propose an explanation of why AT enforces shape bias from a frequency perspective: AT causes the model to shift its decision from local high-frequency information to global shape information, which better resembles the behavior of humans. However, doing so seems to hurt generalization against OOD datasets where the spectral distribution significantly diverges from the training data.

Chapter 8

Are Visual Biases Correlated with ImageNet Generalization?

The current understanding of generalization in neural networks is limited. However, in computer vision, specific biases distinguishing models from human vision have been proposed as a potential cause of a lack of robustness or, more broadly, generalization. Consequently, various methods have been developed to improve (or align) these biases during training in an effort to improve generalization (including our frequency regularization in Chapter 5). In this chapter, we provide a critical evaluation of this line of work by correlating biases with generalization across a large model zoo, carefully controlling for confounding factors.

This chapter is based on "Can Biases in ImageNet Models Explain Generalization?", presented at CVPR, 2024 (Gavrikov & Keuper, 2024). As the first author, Paul Gavrikov collected the models, developed the code base, performed the experiments, created the plots, and wrote the paper with input from Janis Keuper.

Code: `https://github.com/paulgavrikov/biases_vs_generalization`

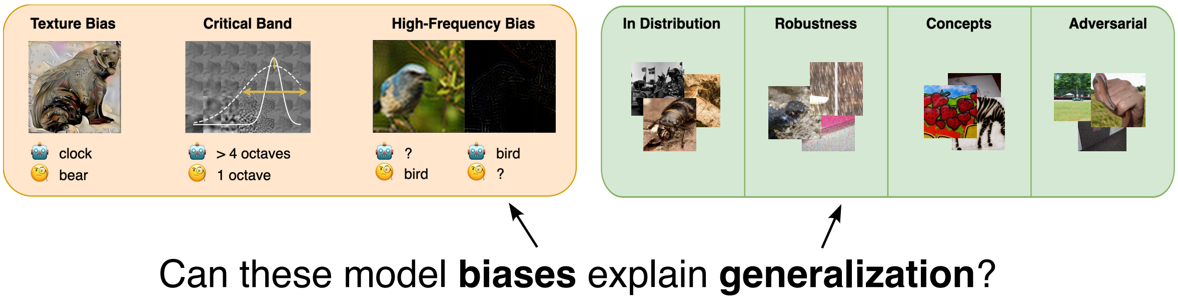


Figure 8.1: Study overview. We study the influence of three selected biases that separate models 🤖 from humans 🧐 on the generalization of *ImageNet* models. Our study suggests that no single bias correlates with generalization in a holistic sense. Specifically, we measure the texture/shape bias (Geirhos et al., 2019), critical band (Subramanian et al., 2023), and low/high-frequency biases (Wang et al., 2020a) on 48 models and correlate these biases against generalization that we measure on several benchmarks belonging to four categories: in distribution, robustness, conceptual changes, and adversarial robustness.

8.1 Introduction

Artificial neural networks have achieved outstanding performance in various tasks, particularly excelling in vision tasks like image classification – commonly benchmarked on *ImageNet* (Deng et al., 2009). While accuracy on *ImageNet* has dramatically improved in the last 15 years (Krizhevsky et al., 2012; He et al., 2016; Dosovitskiy et al., 2021), with models matching or even surpassing human performance, a critical issue remains: these models often fail to generalize beyond the specific data distribution they were trained on (see Section 2.7 for a taxonomy of distribution shifts). This fragility becomes evident when faced with covariate shifts, such as changes in weather conditions (Taori et al., 2020), digital artifacts (Hendrycks & Dietterich, 2019), or even replacing photos objects with artistic renditions (Hendrycks et al., 2021a). On an abstract level, this can be unified by the vulnerability to adversarial attacks (Szegedy et al., 2014b; Biggio et al., 2013), where carefully crafted image perturbations can fool models while remaining imperceptible to humans. This raises serious concerns about deploying such models in safety-critical applications where human lives are at stake (Finlayson et al., 2019; Cao et al., 2019).

Researchers seeking explanations for this lack of generalization have identified various biases that differentiate model and human perception (see Section 2.9 for a detailed definition and an overview). Effectively, these biases cause classifiers to latch on spurious features that may be sufficient to classify samples at train time but do not transfer under distribution shifts (Section 2.9.1).

Three recent examples that have received significant attention are:

- **texture/shape bias** (Geirhos et al., 2019): models rely heavily on texture for object recognition, whereas humans prioritize shape information
- **critical band bias** (Subramanian et al., 2023): models utilize other critical bands than humans for object recognition
- **high-frequency bias** (Wang et al., 2020a): models exhibit a strong dependence on high-frequency information and perform poorly on low-pass filtered data

Following the intuition that aligning these biases with human perception might improve generalization, several studies have explored regularization techniques during training (Li et al., 2021; Lopes et al., 2020; Saikia et al., 2021), including our proposed frequency regularization in Chapter 5. However, the fundamental question (visualized in Figure 8.1) remains: *how well do these biases actually correlate with the ability to generalize?* Or, in other words, how promising is it to just optimize the bias to improve generalization? Although we have shown that enhancing adversarial robustness aligns certain biases – such as the texture/shape bias discussed in Chapter 7 – it remains unclear whether the reverse is true or if this effect extends beyond adversarial robustness and the texture/shape bias.

In this study, we take a step back from developing new methods and instead analyze these biases in *ImageNet* classification models, investigating their connection to generalization performance under various aspects. A cornerstone of this analysis is that we explore changes *in training* (*e.g.*, regularization like augmentation, contrastive learning, and others), *while keeping the model architecture constant* to isolate the effect of inductive biases, as well as fixing the training data to ensure that improvements are not caused by the memorization of additional data (similarly as in Chapter 6). We assess generalization by measuring performance on different datasets, modeling in-distribution data, corrupted data, conceptual changes, and under adversarial attacks.

This investigation aims to shed light on the true impact of these biases and inform the development of more robust and generalizable models.

We summarize our contributions as follows:

- We systematically evaluate 48 models trained on *ImageNet* using diverse techniques, assessing their performance across multiple generalization benchmarks, including in- and out-of-distribution tests for robustness, concept generalization, and adversarial robustness. This comprehensive evaluation allows us to analyze correlations between these generalization benchmarks and inherent biases in the models, such as texture/shape bias, frequency biases, and characteristics of the critical band.

- **Key findings:** (1) No single tested bias fully explains generalization; some even exhibit negative correlations with human perception. (2) Many previously reported correlations contain numerous outliers. (3) Several biases show no correlation with generalization in our evaluation. (4) We identify a surprising positive correlation between a high-frequency bias and generalization.

8.2 Background

In this section, we will introduce the three biases more closely and outline our testing method (following Section 2.9.3) for each one.

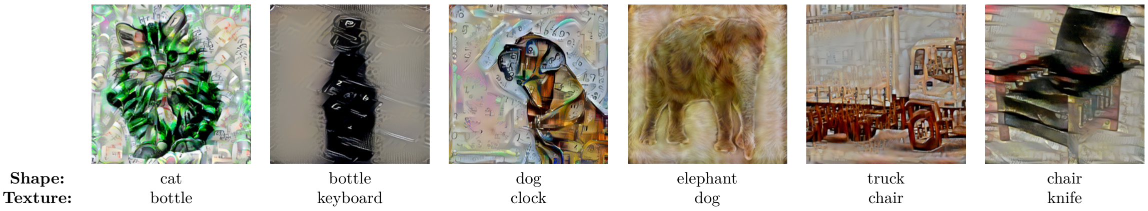


Figure 8.2: Examples from texture/shape cue-conflict. Images are constructed from synthetically combining two conflicting shape and texture cues and, thus, contain two labels per sample: one for *shape* and one for *texture*. **Image Source:** (Geirhos et al., 2019)

8.2.1 Texture/Shape Bias

Geirhos et al. (2019) discovered that CNNs primarily rely on texture cues to identify objects (in *ImageNet*). This is in stark contrast to the strong shape-biased decision observed in human perception (Landau et al., 1988). They argue that improving the shape bias may be linked to improved robustness and show that (pre-)training on a dataset without discriminative texture information indeed improves robustness on some benchmarks, including common corruptions (Hendrycks & Dietterich, 2019). Following this observation, multiple works have attempted to understand the origins of this bias (Hermann et al., 2020; Islam et al., 2021), improve shape bias in training (Li et al., 2021; Shi et al., 2020; Lukasik et al., 2023), or study representations beyond *ImageNet*-CNNs (Naseer et al., 2021; Jaini et al., 2024; Gavrikov et al., 2024b; Dehghani et al., 2023; Geirhos et al., 2021). Prior works and our study in Chapter 7 have also shown a correlation between adversarial training (AT) (Madry et al., 2018; Zhang & Zhu, 2019; Chen et al., 2022a) and shape bias (Geirhos et al., 2021; Gavrikov et al., 2023). However, an improved shape bias seems not to be the cure-all for generalization: (Mummadi et al., 2021) designed an augmentation technique that improves shape bias but not robustness to common corruptions (Hendrycks & Dietterich, 2019) and, thus, shows a counter-example to reject a causal connection at least for this aspect of generalization. Similarly, our frequency regularization in Chapter 5 increased the shape bias but not always the generalization of *ImageNet* models. In our study, we test the shape bias beyond common corruptions against multiple benchmarks to derive a more holistic understanding of this bias in the context of generalization.

Our testing We measure the texture/shape bias on the *texture/shape cue-conflict* dataset (Geirhos et al., 2019) – corresponding to the behavioral cue-conflict testing methodology in Section 2.9.3. Exemplary samples are shown in Figure 8.2. This dataset consists of *ImageNet* samples where shape and texture cues belong to one of 16 different (*conflicting*) *ImageNet* super-classes (*e.g.*, a cat (shape) and a glass (texture)). We report the *shape bias*, which is defined by the ratio of shape predictions compared to all correct decisions on the dataset – with a value of 0/1 indicating a texture/shape bias, respectively. Humans achieve a strong average shape bias of 0.96, whereas most models are severely texture-biased.

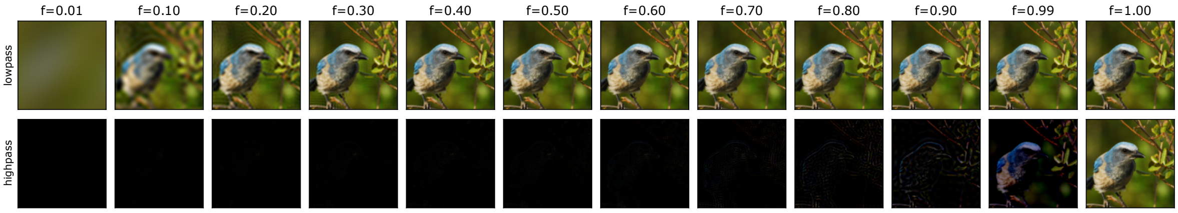


Figure 8.3: Band-pass filtering of images. Visualization of low-pass (top) and high-pass (bottom) filtered data at cutoff frequency f on one *ImageNet* sample.

8.2.2 Frequency Bias

Natural signals, such as images (Ruderman, 1994) and sounds (Singh & Theunissen, 2004), concentrate most information in low-frequency bands. As such, it seems to be reasonable to draw predictive cues from this band. Contrary, Wang et al. (2020a) demonstrated that image classification models reach high accuracies on high-pass filtered images that mostly resemble noise and should not carry discriminative information. Low-pass filtered images easily recognizable by humans, on the other hand, are poorly recognized by the same models. While this seems to be more extreme on low-resolution images, higher-resolution models such as *ImageNet* classifiers also unreasonably heavily rely on cues from high-frequency (HF) bands (Abello et al., 2021). (Wang et al., 2020a) also link this *spectral* bias to adversarial robustness (Szegedy et al., 2014b; Biggio et al., 2013) and show that adversarial training (AT) (Madry et al., 2018) results in a reversed bias. Indeed, perturbations generated by adversarial attacks on *ImageNet* seem to particularly target HF bands (Yin et al., 2019). A similar connection to non-adversarial robustness can be made, as most common corruptions (Hendrycks & Dietterich, 2019) primarily target HF bands (Yin et al., 2019). In a similar fashion, we showed that our HF regularization in Chapter 5 increases robustness to adversarial attacks and common corruptions. Altogether, this bias has received attention in the robustness community, where some hope to improve robustness by stronger low-frequency biases (Lopes et al., 2020; Saikia et al., 2021). We seek to understand how frequency biases, in particular from low/high-frequency cues, more broadly affect generalization.

Our testing To understand the frequency bias, we measure the performance of band-pass-filtered *ImageNet* samples, similar to (Wang et al., 2020a). This corresponds to the behavioral feature ablation methodology in Section 2.9.3. We differentiate between low and high-frequency biases by filtering the test data at different cutoff frequencies. For instance, a low-pass filter with a cutoff of 30% will contain the lowest 30% of the Fourier spectrum. Formally, let $\mathcal{F}$ denote the Fast Fourier Transformation (as introduced in Section 2.1; including shifting the zero-frequency component to the center) and $\mathcal{F}^{-1}$ the inverse operation, then we obtain the frequency filtered sample $\boldsymbol{X}' \in \mathbb{R}^{c\times w\times h}$ from an input sample $\boldsymbol{X} \in \mathbb{R}^{c\times w\times h}$ as follows:

$$\boldsymbol{X}' = \mathcal{F}^{-1}\left(\mathcal{F}\left(\boldsymbol{X}\right) \odot \boldsymbol{M}_f\right) \tag{8.1}$$

where $\boldsymbol{M}_f \in \mathbb{R}^{w\times h}$ denotes the frequency mask (filter) in the Fourier space, parameterized by the cutoff frequency f implemented as follows:

```python
h, w = X.shape[-2:]
cy, cx = h // 2, w // 2
ry = int(cutoff_freq * cy)
rx = int(cutoff_freq * cx)
if low_pass:
    mask = torch.zeros_like(X)
    mask[:, cy-ry:cy+ry, cx-rx:cx+rx] = 1
else:
    mask = torch.ones_like(X)
    mask[:, cy-ry:cy+ry, cx-rx:cx+rx] = 0
```

Listing 8.1: Frequency Mask Computation

An example of the resulting samples for both low- and high-pass filtering under different cutoff frequencies can be found in Figure 8.3.

To compensate for different test performances of the models on the clean data, we report the *ratio* of the accuracy on the filtered data to the clean accuracy. In some cases, this may result in ratios > 1, *i.e.*, the accuracy on filtered data may exceed the original validation accuracy. As low/high-frequency bands are not precisely defined, we average the performance observed at 10, 20, 30, 40, and 50% cutoffs along the low-/high-pass filter tests to report the *low/high-frequency bias*, respectively.

8.2.3 Critical Band

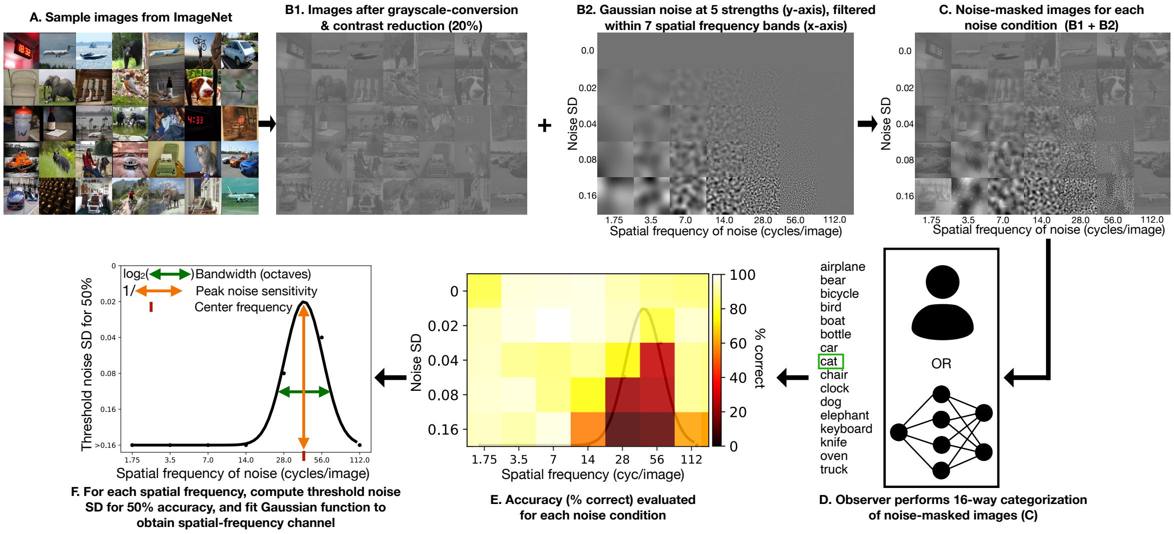

A. Sample images from ImageNet
B1. Images after grayscale-conversion & contrast reduction (20%)
+
B2. Gaussian noise at 5 strengths (y-axis), filtered within 7 spatial frequency bands (x-axis)
Noise SD
0.0
0.02
0.04
0.08
0.16
1.75
3.5
7.0
14.0
28.0
56.0
112.0
Spatial frequency of noise (cycles/image)
C. Noise-masked images for each noise condition (B1 + B2)
Noise SD
0.0
0.02
0.04
0.08
0.16
1.75
3.5
7.0
14.0
28.0
56.0
112.0
Spatial frequency of noise (cycles/image)
airplane
bear
bicycle
bird
boat
bottle
car
cat
chair
clock
dog
elephant
keyboard
knife
oven
truck
OR
D. Observer performs 16-way categorization of noise-masked images (C)
Noise SD
0
0.02
0.04
0.08
0.16
1.75
3.5
7
14
28
56
112
Spatial frequency (cyc/image)
100
80
60
40
20
0
% correct
E. Accuracy (% correct) evaluated for each noise condition
log2( ) Bandwidth (octaves)
1/ Peak noise sensitivity
Center frequency
Threshold noise SD for 50%
0
0.02
0.04
0.08
0.16
>0.16
1.75
3.5
7.0
14.0
28.0
56.0
112.0
Spatial frequency of noise (cycles/image)
F. For each spatial frequency, compute threshold noise SD for 50% accuracy, and fit Gaussian function to obtain spatial-frequency channel

Figure 8.4: Overview of the critical band testing methodology. (**A**) Samples from *ImageNet*, (**B1**) are grayscaled and contrast reduced, and (**B2+C**) then turned into stimuli testing inserted Gaussian noise at different spatial frequencies and standard deviations. (**D**) These images are shown to humans and models to obtain an (**E**) accuracy heatmap. (**F**) This heatmap is thresholded to fit a 2D Gaussian, characterizing the critical band. **Image Source:** (Subramanian et al., 2023)

Recently, Subramanian et al. (2023) have found another bias that separates human from model vision in the *critical band*. This band defines what spatial-frequency channels are used to detect objects. On *ImageNet*, humans use a one-octave wide channel. Yet, models measure significantly wider channels, making them more sensitive to noise perturbations in frequency bands that do not affect human vision. The study showed a strong correlation between the parameters of the critical band and shape bias, as well as the adversarial robustness of adversarially-trained networks. Surprisingly, adversarial training (AT) seems to further increase the bandwidth and, thus, increases misalignment – albeit that it typically improves the texture/shape bias as we have shown in Chapter 7. We propose a few optimizations to the underlying test process, and as for the previous biases, we aim to understand possible connections to generalization beyond adversarial robustness/training.

Our testing Following the methodology of Subramanian et al. (2023) shown in Figure 8.4, we measure the critical band by insertions of Gaussian noise to contrast-reduced and grayscaled *ImageNet* samples with varying strength and different frequencies. This can also be understood as a behavioral feature ablation – however, the feature is gradually replaced by noise and not directly ablated (Section 2.9.3). Conditions where the accuracy falls below a predefined threshold mark the critical band. The fitting of a 2D Gaussian obtains the parameters of the band over the performance results of a grid search for noise strength and frequency pairs, characterized by the *bandwidth (BW)*, *center frequency (CF)*, and *peak-noise sensitivity (PNS)*.

A caveat of the original test is that the critical band is determined by conditions where the accuracy (on a 16-class *ImageNet* subset) drops below 50%. However, adversarially-trained models perform poorly on contrast-reduced images, as we have shown in Chapter 7. To compensate for this, we *normalize* the accuracy on the stimuli by a baseline accuracy on the contrast-reduced images before applying noise. Furthermore, the original test only uses an average of just 30 *ImageNet* samples for each condition. Instead, we use all 50,000 *ImageNet* validation samples. Finally, the original test measures the accuracy against 16 super-classes for each subset. This was mainly done to keep the study feasible for human subjects. Since we do not compare to human trials, we measure the top-1 accuracy against all 1,000 classes. Taken together, this results in a more computationally expensive but also more accurate measurement. We will show an ablation of these changes in Section 8.4.3.

8.2.4 Limitations of Previous Studies

Previous studies have already correlated (some of) these biases with (some aspects of) generalization (Wang et al., 2020a; Subramanian et al., 2023; Geirhos et al., 2021, 2019). In this section, we discuss the limitations of these studies and show why their findings may not transfer to a more holistic view of generalization.

Inductive biases of architectures were not ablated The architecture plays a significant role in robustness (Tang et al., 2021) and, thus, can also be expected to affect a more holistic view of generalization. As such, previously discovered correlations may be spurious concerning the bias itself and just an effect of the architecture. To suppress this potential confounder, our model zoo is limited to `ResNet-50` models without any modifications to the original architecture as defined in (He et al., 2016). As a matter of fact, we observe that many correlations no longer hold when we fix the architecture.

Findings on subsets do not generalize It is well known that correlation does not imply causation, and thus, it may not be surprising that we find that many observed correlations no longer hold for our study. Common causes include (1) benchmarking only a specific type of training, *e.g.*, adversarial training. Yet, correlations for adversarial training may be differently strong or even reversed compared to other forms of training (for example, see Figure 8.5); and/or (2) using only one benchmark, *e.g.*, adversarial attacks. Performance on many *ImageNet*-benchmarks is moderate to strongly correlated (Taori et al., 2020; Miller et al., 2021; Mania & Sra, 2021), which we also confirm in our study (Figure 8.12b) but clear outliers exist. For instance, adversarial robustness (here measured by PGD (Kurakin et al., 2017; Madry et al., 2018)) is not correlated to most other benchmarks (see Figure 8.12b) and generally might be divergent to other forms of robustness (Liu et al., 2023a; Tsipras et al., 2019; Raghunathan et al., 2019; Stutz et al., 2019). Our study focuses on generalization beyond but also including adversarial robustness and is performed on various models including adversarially-trained ones.

Inconsistent data transformations Comparing benchmarks without standardized transformation pipelines can lead to misleading and incomparable results. We found that some authors produced results using inconsistent transformation pipelines during testing, which resulted in non-comparable results. For instance, the works by *Hendrycks et al.* (Hendrycks & Dietterich, 2019; Hendrycks et al., 2020, 2022) upsample the 224 px *ImageNet-C* samples to 256 px, and then center-crop to 224 px. This "Inception-like" preprocessing (Szegedy et al., 2016a) is common on *ImageNet* but clearly oversamples the already processed samples. On the other hand, `PRIME` (Modas et al., 2022) does not resize or crop loaded test samples, which is conceptually correct but leads to different results. In our study, we use identical test transformations for all models and datasets to avoid discrepancies.

System noise Additionally, even system noise can lead to significant errors in evaluations *e.g.*, due to low-level implementation details in data loaders such as JPEG decompression or subsampling techniques (Wang et al., 2021). These issues advocate for benchmarking under comparable conditions to obtain comparable results. Therefore, we produce all benchmarks on the same system, using the same environment.

8.3 Methodology

In our study, we measure models' ability to generalize and correlate this to the biases that separate human and model vision. The following paragraphs introduce our methodology in more detail.

8.3.1 Measuring Generalization

To benchmark generalization, we loosely follow the suggestions of prior work (Hendrycks et al., 2021a) and rely on the accuracy of multiple semantically different benchmarks to faithfully assess multiple facets. Our benchmarks can be separated into **four categories**: *in-distribution (ID)* datasets with considerable statistical similarity to the *ImageNet* training set, *robustness* datasets that apply various corruptions to *ImageNet* samples, datasets challenging models to recognize objects based on *concepts* (*e.g.* sketches), and *adversarial attacks* to test robustness adaptively. For all benchmarks, we evaluate the top-1 accuracy using a resolution of 224^2 px and *consistent* transformation pipelines on the *same system* to avoid the risk of falsifications due to system noise (Wang et al., 2021). We report the average over the individual benchmarks for each of the four categories.

In distribution (ID)

- The *ImageNet (IN)* (Deng et al., 2009) validation set from the ILSVRC2012 challenge is the de facto standard *ImageNet* test dataset, containing 50,000 images with 1,000 different classes.
- *ImageNet v2 (IN-v2)* (Recht et al., 2019) is a newer 10,000 images test set that was sampled a decade later following the methodology of the original curation routine.
- *ImageNet-ReaL (IN-ReaL)* (Beyer et al., 2020) is a re-annotated version of the original *ImageNet* validation set. It assigns multiple labels per image and contains multiple corrections of the original annotations.

Robustness

- *ImageNet-C (IN-C)* (Hendrycks & Dietterich, 2019) is a dataset consisting of 19 synthetic corruptions applied to the original *ImageNet* validation set under increasing severity. The original protocol suggests averaging the error over all corruptions and severities normalized by the error of `AlexNet` (Krizhevsky et al., 2012) on the same. On the contrary, we simply report the mean accuracy over all 19 corruptions and severity levels.[1]
- *ImageNet-C (IN-$\overline{C}$)* (Mintun et al., 2021) extends *IN-C* by adding 10 new corruptions which were chosen to be perceptually dissimilar but conceptually similar to the corruptions in *IN-C*. We report the mean accuracy akin to *IN-C*.
- *ImageNet-A (IN-A)* (Hendrycks et al., 2021b) contains 7,500 additional images belonging to 200 *ImageNet* classes that are naturally hard to classify for *ImageNet* models and are, thus, posing natural adversarial examples.

Concepts

- *ImageNet-Renditions (IN-R)* (Hendrycks et al., 2021a) is a dataset of 30,000 images of 200 different *ImageNet* classes in different styles, such as cartoons, paintings, toys, etc.
- *ImageNet-Sketch (IN-S)* (Wang et al., 2019) is a dataset of over 50,000 hand-drawn sketches belonging to all *ImageNet* classes. Semantically, it can be seen as a subset of *IN-R*.
- *Stylized ImageNet (SIN)* (Geirhos et al., 2019) contains *ImageNet* validation images that have been stylized using different artistic filters to destroy texture information. We use the official 16-class subset given in (Geirhos et al., 2021).

[1] The results remain comparable by a simple linear transformation.

Adversarial robustness

- *Projected Gradient Descent (PGD)* (Madry et al., 2018) We apply a PGD attack on the *ImageNet validation set* to adaptively evaluate robustness. Contrary to the previous benchmarks, this benchmark is *adaptive* as it can continuously generate weaknesses for a given model. In theory, a model might overfit a static dataset due to its limited number of test samples; however, an (ideal) adaptive benchmark remains unaffected by this limitation. As models trained without adversarial training (Madry et al., 2018) are highly susceptible to such attacks, we attack with a reduced budget of $\epsilon = 0.5/255$ under L^∞ norm, using 40-steps with step size $\alpha = 2/255$.

Data transformation pipeline We use the same data processing pipeline for all models to ensure a fair comparison. We resize the smaller edge of the inputs to 256 px and the other edge with the same ratio using bilinear interpolation, then center-crop to 224 × 224 px. Channel-wise normalization matches the individual normalization during the training of each model – typically, this is the mean and std over all samples of the *ImageNet* dataset.

The samples in *IN*-$\overline{C}$/C are preprocessed by default, and we skip the "Inception-like" (Szegedy et al., 2016a) resizing and cropping steps. We want to point out that some prior works also apply the above transformations to these datasets. This is questionable because it results in undersampling and, thus, a loss of details and inconsistent evaluation compared to the clean *ImageNet* dataset and approaches using "correct" preprocessing.

8.3.2 Models

We conduct our main analysis on models based on the `ResNet-50` architecture (He et al., 2016) trained on *ImageNet-1k* (Deng et al., 2009; Russakovsky et al., 2015). Fixing the architecture is crucial for our study, as each architecture has its own inductive biases that may influence results. Thus, to disentangle the inductive bias from generalization, we ensure that all models are based on an identical architecture. This also means avoiding even minor changes like different activation functions, as they can drastically affect the performance and introduce confounders to our analysis (Xie et al., 2021). Limiting the study to `ResNet-50` has additional benefits: the architecture is highly popular and comes without bells and whistles. This status also ensures that a variety of techniques have been evaluated for this architecture, specifically. The architecture also represents architectures used in real-life applications, where training from scratch is necessary, but computational resources are limited. Further, we ensured no model was trained with external data to avoid target leakage on our benchmarks. Most models were also not specifically optimized for the tested biases, which provides an opportunity to understand if and to what extent (aligned) biases arise through changes in training.

To represent changes in generalization, we use 48 models that differ *in training.* We loosely sort these models into seven categories that we color-code. Throughout the study, we use different marker symbols to differentiate individual models. Marker sizes of adversarially-trained models correlate with the attack budget ϵ during training.

Our tested models belong to the following categories:

- **baseline**: a model following the original training (He et al., 2016)
- **augmentation**: techniques (Hendrycks et al., 2020, 2022, 2021a; Jaini et al., 2024; Müller et al., 2023; Modas et al., 2022; Li et al., 2021; Erichson et al., 2022)
- **stylized**: (pre-)training on *Stylized ImageNet (SIN)*, which can be seen as an extreme form of augmentation that breaks correlations between textures and class labels (`ShapeNet`) (Geirhos et al., 2021)
- **adversarial training (AT)**: training against a PGD adversary (Madry et al., 2018) with increasing attack budget ϵ under L^2 and L^∞-norm (Salman et al., 2020). Technically, this too could be seen as augmentation, but we will show significant differences for this group of models

- **self-supervised learning (SSL)**: unsupervised learning through contrastive learning (Chen et al., 2021b; Caron et al., 2021, 2020; Chen et al., 2020a) with supervised finetuning of the classification-head
- **freezing**: a model trained with all spatial convolution filters weight frozen to the random initialization (Gavrikov & Keuper, 2023b).
- **improved training recipes**: new recipes in *Pytorch Image Models (timm)* (Wightman, 2019) based on the findings of Wightman et al. (2021) and *PyTorch* (Vryniotis, 2023; Paszke et al., 2019);

A full model legend including all benchmark results and bias measurements can be found in Appendix A.3.

8.3.3 Measuring Correlations

We measure correlations via the Spearman's rank-order correlation coefficient ρ. Due to the low sample size ($N \ll 500$), we also assess the statistical significance by obtaining a p-value from a two-sided permutation test. p is given by the ratio of generated ρ larger than the baseline ρ on all samples. We consider results as significant if $p < 0.05$.

8.3.4 Implementation Details

All evaluations were performed with *Python* 3.10.12, *PyTorch* 2.0.1, *CUDA* 11.7, and *cuDNN* 8500 on *NVIDIA A100-SXM4* GPUs.

8.4 Are Vision Biases Correlated with Generalization?

In this section, we apply the previously introduced methodology to each bias individually to understand if and to what extent aligning these biases can help with generalization. Figure 8.5 shows an overview of the *statistical* correlation for all tested biases. Additionally, we inspect scatter plots (Figures 8.6 to 8.8) to detect non-monotonic correlations missed by Spearman's ρ and pay attention to whether outliers exist.

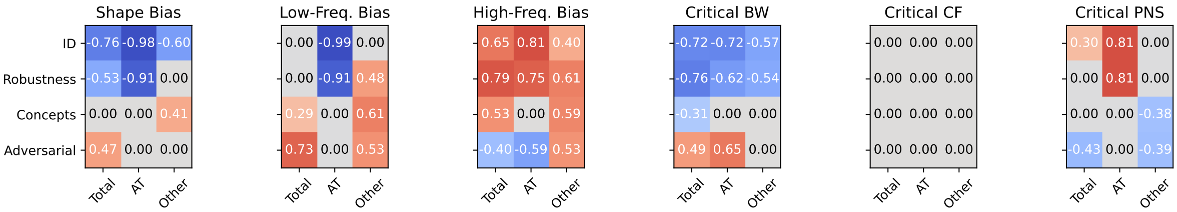


Figure 8.5: Biases often only correlate with specific aspects of generalization or model groups. We measure Spearman's ρ correlation on all models (*Total*) and separately on adversarially-trained (*AT*), and all *other* models, as there is often a different trend. Non-significant correlations with $p \geq 0.05$ are set to 0. Please note that ρ does not capture non-monotonic relations.

8.4.1 Shape Bias

For the shape bias, we observe a strong *negative* correlation to ID tests (Figure 8.6) – suggesting that increasing misalignment with human vision increases ID accuracy. While it has been shown before that *ImageNet*-trained CNNs are strongly texture-biased (Geirhos et al., 2019) and *ImageNet* can be well separated by texture alone (Brendel & Bethge, 2019), the monotonic decrease in performance with increasing shape bias without significant outliers is quite surprising. While this does not prove causality, it raises the question of whether strongly shape-biased models can achieve strong performance on *ImageNet* without relying on extra data.

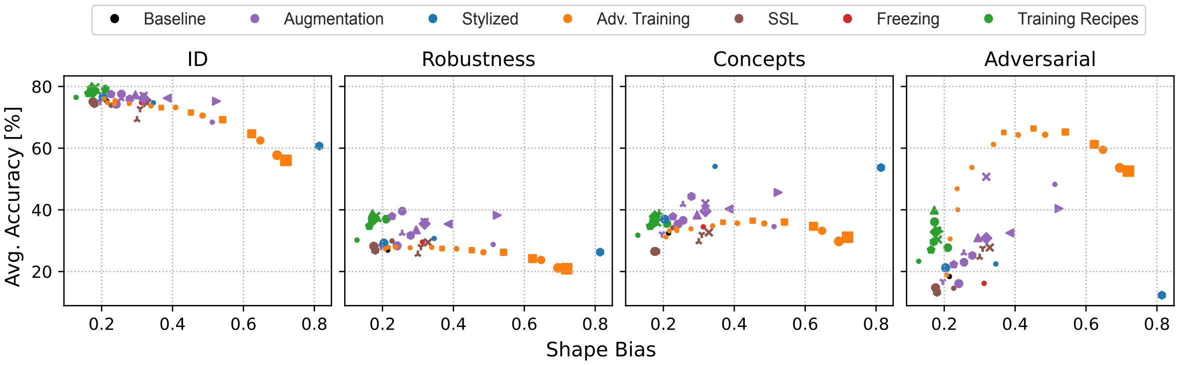


Figure 8.6: Shape bias vs. generalization. A value of 0/1 indicates a texture/shape bias, respectively. The full marker legend is shown in Appendix A.3.

Similarly, we only note significant improvements in *IN-A* on strongly *texture*-biased models. Usually, poor performance on *IN-A* is linked to spurious cues (Hendrycks et al., 2021b), and one may assume that these are related to textures (similar to adversarial attacks on fragile features (Ilyas et al., 2019)). However, it seems like these cues are more shape-related, and a certain degree of texture bias is necessary to overcome them. But neither does a texture bias guarantee the best performance, as the strongest texture-biased model (`timm (A3)` (Wightman et al., 2021)) with a shape bias of 0.13 still performs 10.4% worse on *IN-A* than the best model (`torchvision (v2)`) with a (higher) shape bias of 0.17.

In other aspects of generalization, we often obtain unrelated trends between adversarially-trained (AT) and other models. For AT models, we observe a curvilinear (inverted "U") relation (which is not captured by Spearman's ρ) to all aspects of generalization, which also seems to correlate with the attack budget during training (pay attention to the marker size). For concepts and adversarial robustness, we observe the best performance by AT models that balance shape and texture bias (≈ 0.5 for $\epsilon = 1/255$ (L^∞), $\epsilon = 2/255$ (L^∞), and $\epsilon = 1$ (L^2)). There are no notable differences in the trends between models trained on L^2 and L^∞-norm attacks. Non-AT models only show a (moderate) correlation to concepts.

Our results extend previous findings showing that shape bias is not causally correlated with *IN-C* performance (Mummadi et al., 2021) to a much broader view of generalization and contrast the claims about improved robustness in (Geirhos et al., 2019). For adversarial robustness, shape bias may define a ceiling where a balanced representation of shapes and textures works best. However, the model set of top performers exclusively consists of adversarially-trained models, which may lead to spurious correlations. Shape bias may also improve recognition of concepts – which is intuitive, as these samples do not contain the same textures as *ImageNet* samples.

Summary In all the models, the only strong and holistic pattern we see is that the shape bias is inversely related to ID performance. This means that as misalignment with human vision increases, ID accuracy increases.

8.4.2 Frequency Bias

It is well known that adversarial training results in models with improved low-frequency at cost in high-frequency performance (Wang et al., 2020a; Yin et al., 2019; Geirhos et al., 2021; Gavrikov et al., 2023). Yet, the frequency bias of other methods has been less studied, especially in correlation to other aspects of generalization beyond adversarial robustness.

Low-frequency bias To our surprise, we find no meaningful correlation between a low-frequency bias and most aspects of generalization when considering all models (Figure 8.7) – except for adversarial

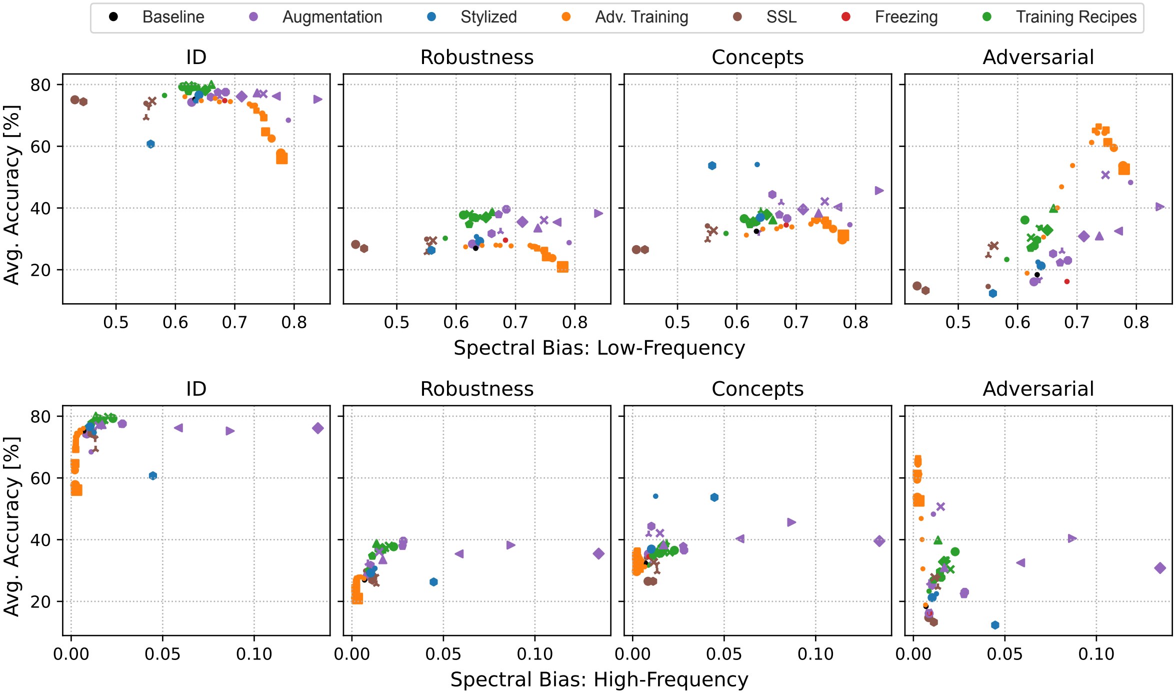


Figure 8.7: Frequency biases vs. generalization. We correlate low-frequency (top) and high-frequency bias (bottom). A value of 0 indicates no recognition of objects on the band pass, and 1 indicates no difference from the original performance. The full marker legend is shown in Appendix A.3.

robustness, where a stronger low-frequency bias improves adversarial robustness. This is intuitive, as attacks prefer high-frequency bands on *ImageNet* (Yin et al., 2019). Improved reliance on cues from non-affected bands is, therefore, a good defense strategy.

When separating AT from other models, we again observe divergent trends. On AT, this bias shows a curvilinear correlation to all aspects of generalization (not captured by Spearman's ρ in Figure 8.5). For concepts and adversarial robustness, some low-frequency bias helps but eventually starts hurting generalization. For ID and non-adversarial robustness, increasing low-frequency bias always seems to reduce generalization. On non-AT models, it shows a moderate monotonic correlation with all categories except ID. However, there is often a high variance in performance.

Summary Improving the focus to cues from low-frequency bands improves adversarial robustness but hardly offers any guarantees beyond that independent of the training method. For adversarial training, too much low-frequency bias is not desirable.

High-frequency bias A high-frequency bias, on the other hand, shows surprisingly moderate to strong correlations to all aspects of generalization (Figure 8.5). For adversarial robustness, it is negatively correlated but positive for all others. Upon manual inspection of the scatter plots (Figure 8.7), we find that this correlation is slightly deceptive. There is a strong monotonic increase in generalization as models start to classify some cues in high-frequency, but after some threshold, the gains stagnate or even fall. A common outlier is `ShapeNet` (Geirhos et al., 2019), which shows some ability to recognize high-frequency cues despite being trained on data without discriminative texture cues that typically reside in these high-frequency bands. For adversarial robustness, we completely reject a connection due to the variance for the lowest measurements – but it seems like only models without any high-frequency bias can reach peak adversarial robustness. Unsurprisingly, AT models show next to no high-frequency bias as they are desensitized to this frequency band.

Summary Overall, a slight capability to detect high-frequency cues seems to improve generalization, except for adversarial robustness. In this context, it also seems like more high-frequency bias is better than no bias. For adversarial robustness, even the slightest increase in high-frequency bias decreases performance rapidly.

8.4.3 Critical Band

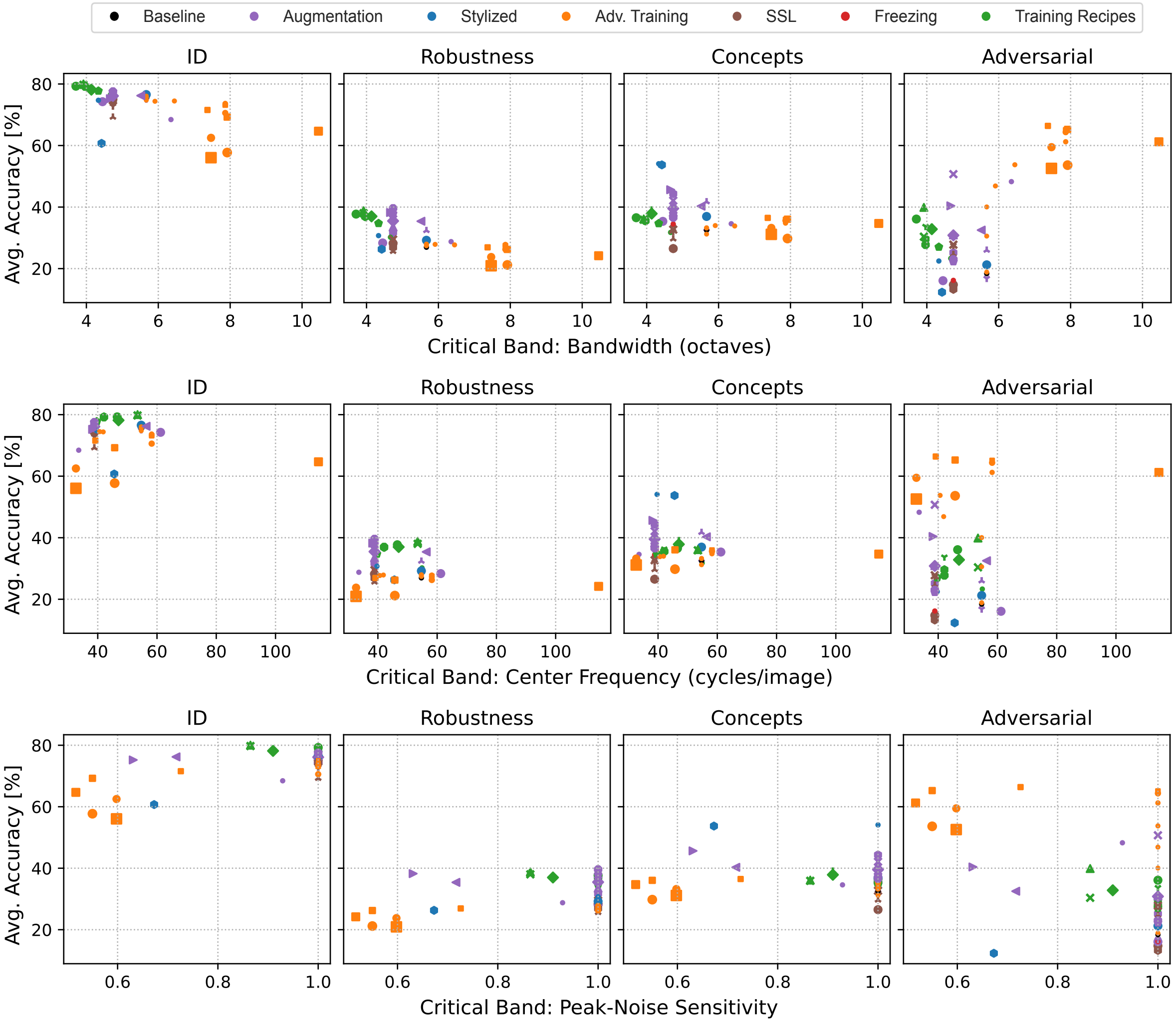


Figure 8.8: Critical band vs. generalization. We correlate three properties: the bandwidth (top), center frequency (center), and peak-noise sensitivity (bottom). The full marker legend is shown in Appendix A.3.

The critical band parameterized by its bandwidth, center frequency, and peak-noise sensitivity has been recently shown to correlate with adversarial robustness (and shape bias) (Subramanian et al., 2023). Akin to what we have observed on other biases, results show different trends between adversarially-trained (AT) and other models. For AT, a higher bandwidth of the critical band correlated positively with adversarial robustness, but the correlation was negative for other models. We repeat the measurements using a more careful collection of models and benchmarks. Additionally, we tweak the original test by using normalization, more samples, and more classes. Our normalization allows us to include models trained under L^∞-norm and larger attack budgets (ϵ) that couldn't be tested with the original methodology. It also allows us to scale the evaluation to all 1,000 *ImageNet* classes. In both cases, non-normalized measurements will break the detection of the critical band, as all regions would be classified as critical. Overall, our measurements are more accurate but may not be directly comparable to the original study.

Bandwidth The bandwidth not only moderately correlates with adversarial robustness (similarly to the findings in (Subramanian et al., 2023)) but also strongly correlates with ID and robustness (Figure 8.5). However, there is an inverted trend, and the correlation to adversarial robustness is negative. For generalization to concepts, we only see a weak positive correlation. Generally, we observe many outliers in the scatter plots (Figure 8.8), and often, the best performance is not achieved by the models with the lowest bandwidth. Further, some trends only become noticeable or accelerated by adversarially-trained models, which accumulate for models with the lowest bandwidth. It remains to be discovered whether low-critical-bandwidth non-AT `ResNet-50` exist and how they perform on generalization tests.

Center frequency In contrast to the original study, we find *no correlation* between the center frequency and any of our benchmarks (Figure 8.8). Even when we single out adversarially-trained models, we still observe no statistically significant correlation.

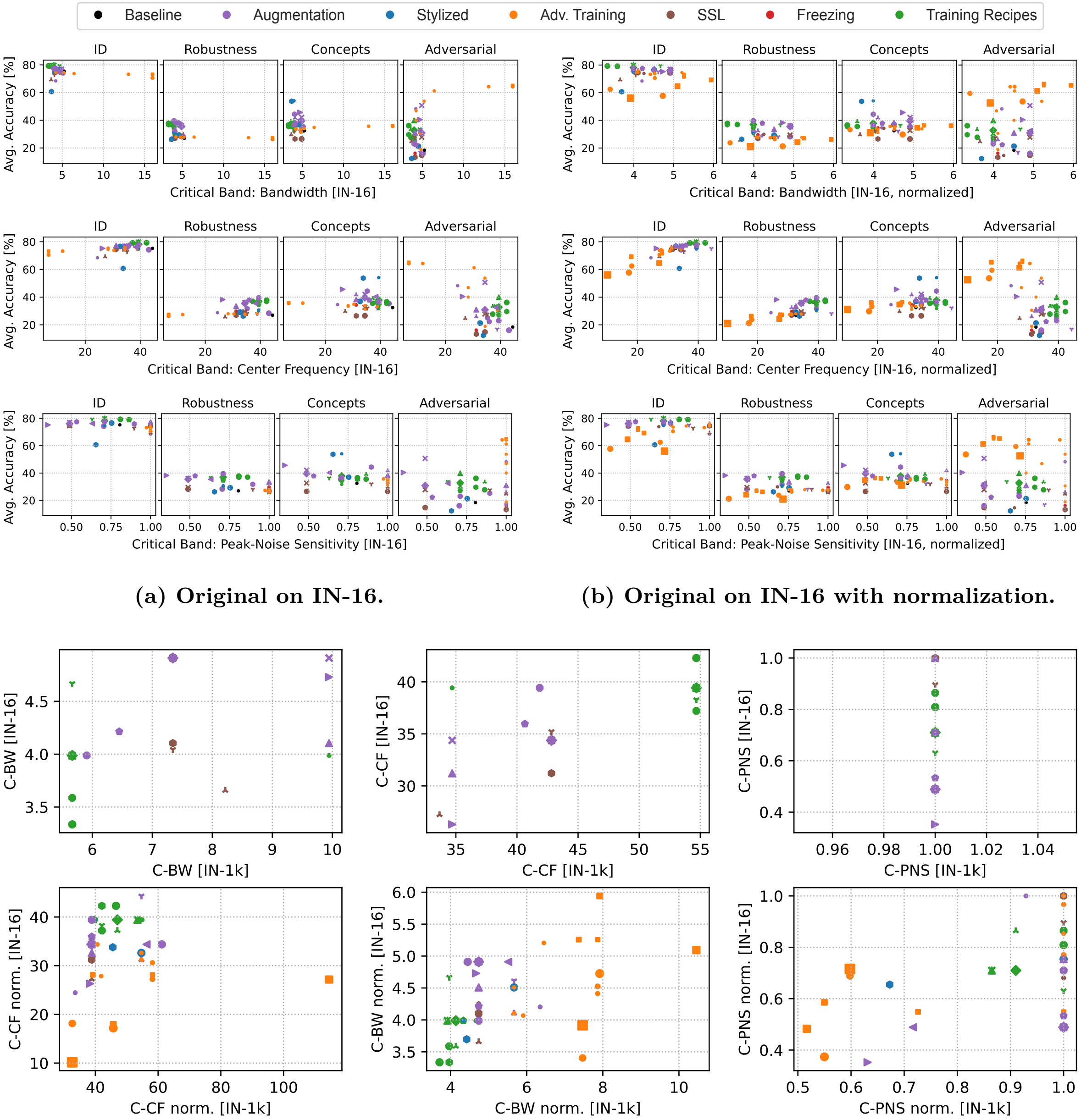
Peak-noise sensitivity We measure a slight statistical correlation among both adversarial training (AT) and non-AT models (Figure 8.5). However, the significant concentration (Figure 8.8) around the value of 1 indicates an issue in the test and suggests that this metric might not effectively capture trends in extensive evaluations such as ours. As for the rest, AT models don't exhibit a distinct clustering pattern yet display a notably strong positive correlation with both ID and non-adversarial robustness.

Summary For the most part, the critical band seems to poorly explain generalization. However, a lower bandwidth seems to be necessary for generalization except for adversarial robustness, where the opposite holds.

Ablation of our Changes to the Methodology

For completeness, we have also evaluated our model zoo with the original method, and the results are shown in Figure 8.9. In Figure 8.9a we replicate the evaluation of Subramanian et al. (2023), in Figure 8.9b we additionally apply accuracy normalization as outlined in Section 8.2.3. Under the original condition, we again see no reasonable correlation for any bias except when limiting the study to AT models. For the normalized study, we also see no correlations between any bias for all models, except for ID and robustness, which show some correlation to the center frequency. However, the correlation is mostly determined by the tail of AT models. Removing these models would break the correlation and, thus, make a causal connection highly unlikely.

In Figure 8.9c, we show the scatter plots between *IN-1k* and *IN-16* obtained results and find no correlation indicating that the results obtained by our method deviate from the original test. While our modifications may not be perfect either (*e.g.*, both our and Subramanian et al. (2023) arbitrarily pick 50% as threshold) our modifications are theoretically better grounded and, thus, introduce an improved measurement of the critical band for models. Overall, all tested methodologies fail to show meaningful and holistic correlations.



(a) Original on IN-16.

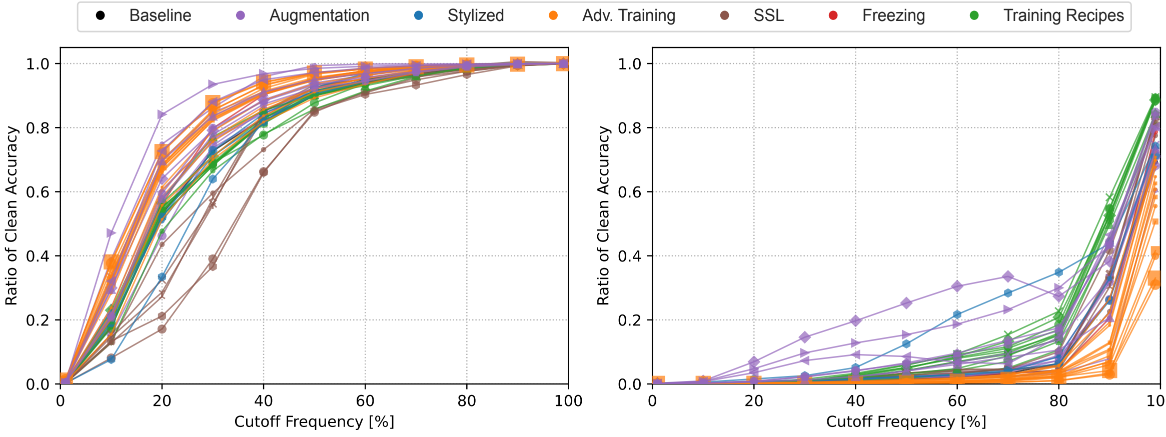
(b) Original on IN-16 with normalization.

(c) Critical band evaluation on IN-1k and IN-16 in comparison. We compare original (top) and normalized (bottom) evaluations.

Figure 8.9: Methodological ablation for the critical band. Measurement of the critical band following the original methodology of Subramanian et al. (2023). (a) original test; (b) original test with normalized accuracy; (c) Comparison between results on *ImageNet (IN-1k)* and the 16-super-class subset *(IN-16)*. Models with unreasonable measurements (C-BW ≥ 100) were removed. The full marker legend is shown in Appendix A.3.

8.5 Observations on Low-/High-Pass-Filtered Images

In this section, we want to provide a few high-level findings split by training method from our low/high-pass data test based on the visualization in Figure 8.10.



(a) Low-pass **(b)** High-pass

Figure 8.10: Frequency band-pass test. We show the relative accuracy to the accuracy on the original samples using (a) low-pass and (b) high-pass filters with increasing cutoff frequency (from 1% to 99% of lowest/highest frequency components to, respectively). The distance to the original image decreases with increasing cutoff. The full marker legend is shown in Appendix A.3.

SSL Firstly, we find that contrastive (self-supervised) learning models underperformed the baseline in low-frequency recognition but performed on par for high-frequency recognition. We cannot prove this to be indicative of a shortcoming of contrastive learning itself, as we primarily benchmark older techniques that perform worse than supervised learning because newer methods are almost exclusively designed for `ViTs`. Still, this may deserve some attention in future works.

Augmentations We also find that some augmentation techniques lead to an unreasonably strong high-frequency recognition rate (the purple outliers in Figure 8.10). This frequency band contains limited information and is almost imperceptible to humans without normalization, *e.g.*, see Figure 8.11) and yet, some models seem to be able to classify a non-negligible amount of samples. This may be related to frequency shortcuts (Wang et al., 2023b) and, in fact, be a sign of overfitting.

On average, we also see that augmentation techniques improve low-frequency bias, and from all our tested models, the strongest performance there is achieved by an augmentation model (`DeepAugment + AugMix` (Hendrycks et al., 2021a)).

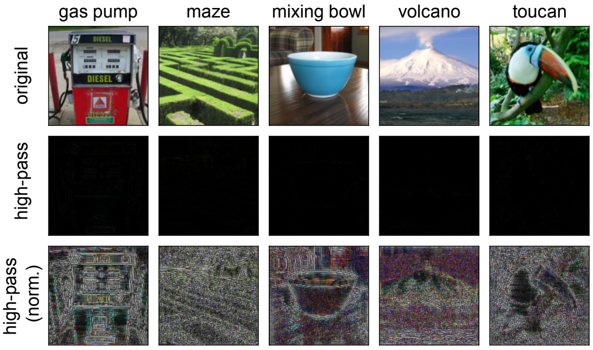


Figure 8.11: Examples of high-pass (30%) filtered *ImageNet* samples. High-pass filtered samples (center) are correctly classified by some models but are unlikely to be recognized correctly by humans. Only *std* normalized samples (bottom) reveal some similarity to the original samples (top).

Improved Training Recipes Newer training recipes seem to mostly improve high-frequency detection rates without significant changes to the low-frequency rates.

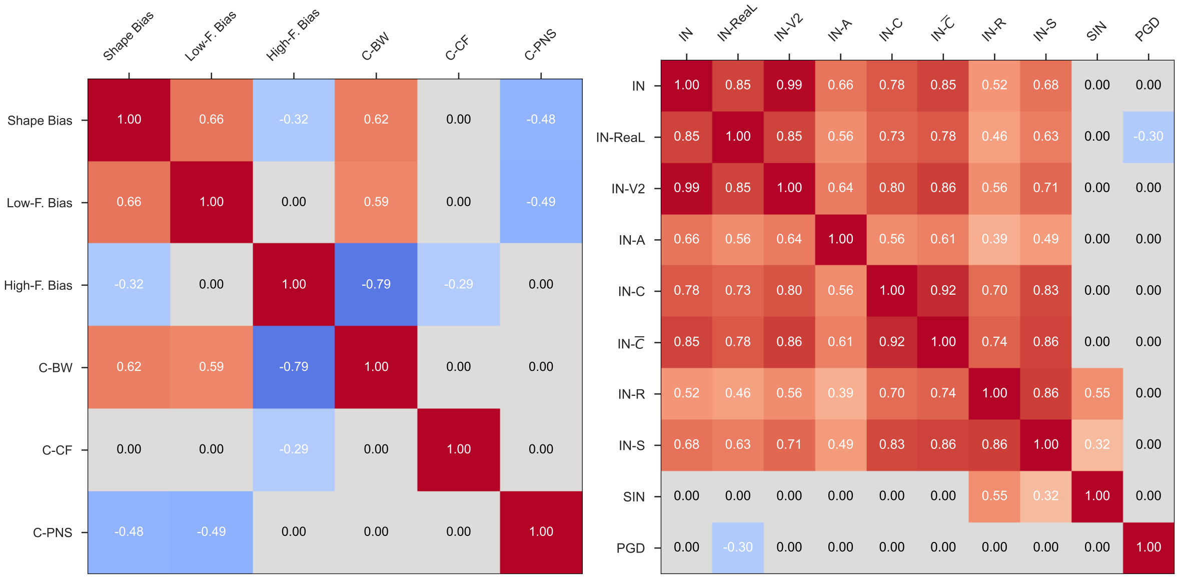
Stylized *SIN*-only training reduces low-frequency detection rates but significantly raises performance in mid-bands and sometimes even outperforms augmentation on some specific cutoffs.

Adversarial Training Similar to our findings in Chapter 5, we find that adversarially-trained models improve in their low-frequency detection at a severe cost in high-frequency prediction rates.

Notably, all models make significant improvements on the lowest 1% of the spectrum, with training recipes showing the least and AT the largest leaps. Gains from additional high-frequency information saturate much earlier for almost all models.

8.6 Other Correlations

We also notice a couple of other correlations that may not directly fit into our storyline but are related to the overall theme of this thesis.



(a) Bias-to-bias

(b) Benchmark-to-benchmark

Figure 8.12: Correlations in our study. We measure correlations between (a) biases and (b) benchmarks through Spearman's ρ. Non-significant correlations with $p \geq 0.05$ are set to 0.

Improved low-frequency performance does not guarantee a shape bias A shape bias causes a model to prefer global over local information, as textures are mostly local information found in higher frequency bands. Indeed, we find a moderately strong positive correlation between low-frequency and shape bias ($\rho = 0.66, p = 0.0002$) in Figure 8.12a, but also clear outliers in both directions: `ShapeNet` (trained on *SIN*) (Geirhos et al., 2019) achieves a high shape bias of 0.81 but only a low-frequency bias of 0.56; the baseline (He et al., 2016) achieves a low-frequency bias of 0.63 but only a shape bias of 0.21.

Scaling data does not guarantee a shape bias Previous works have observed a trend where upscaling (pre-)training data results in higher shape bias (and better generalization) (Geirhos et al., 2021; Dehghani et al., 2023). However, when we extend our analysis to a `ResNet-50` trained with self-supervised learning on *YFCC-100M* (Thomee et al., 2016) containing 100 million additional samples (Yalniz et al., 2019), we notice that it obtains the same 0.21 shape bias as the baseline despite generally better performance. This shows that the shape bias is not simply determined by the dataset scale alone.

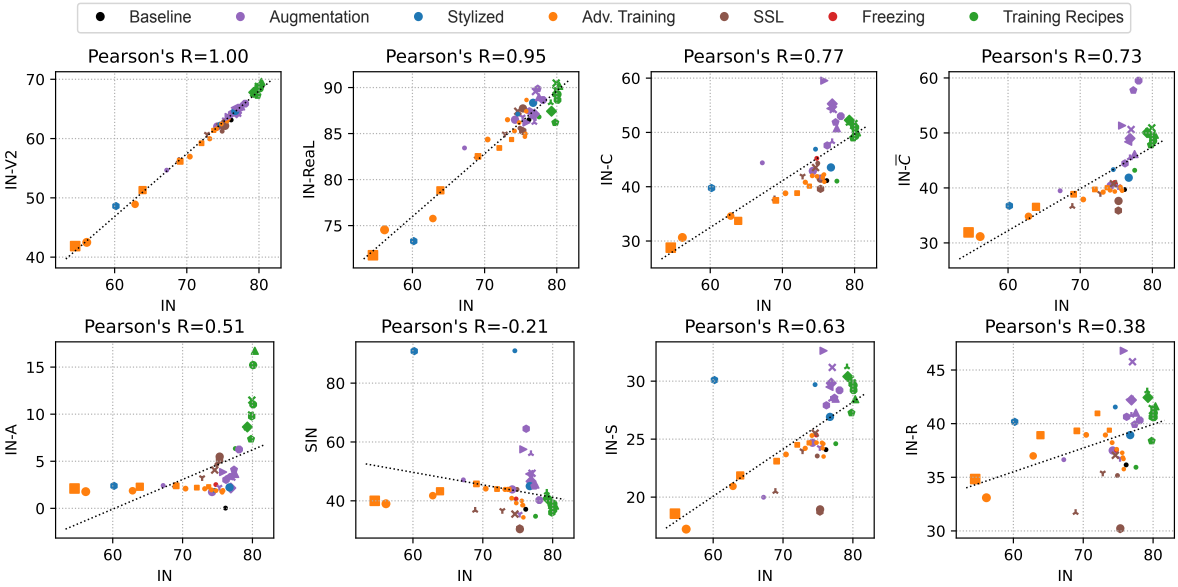


Figure 8.13: ID vs. OOD accuracy *not* on the line. We notice that on our model zoo, there is often no perfect linear correlation between ID and OOD accuracy (Miller et al., 2021). The dotted line is a linear regression on the accuracies, and we report Pearson's R. The full marker legend is shown in Appendix A.3.

ID vs. OOD accuracy *not* "on the line" Since we do observe a non-linear correlation between *ImageNet (ID)* performance and performance on many datasets, we rarely see that the accuracy "is on the line" in Figure 8.13, *i.e.* perfectly linearly correlated as claimed in (Miller et al., 2021). The observation holds on *IN-V2* and *IN-ReaL*, which aligns with the findings of Recht et al. (2019); Beyer et al. (2020), but we already see many underspecified areas where there is no correlation on other datasets, which have also been reported in other studies (Andreassen et al., 2021; Wenzel et al., 2022; Teney et al., 2023). The clear outliers to "accuracy on the line" are *SIN* and *IN-R*, where correlations are very weak or even statistically insignificant.

***IN-$\overline{C}$* is a good proxy for non-adversarial robustness** While OOD accuracy is not always on the line, we notice that *ImageNet* accuracy strongly correlates with the *average* ($\overline{R} = 0.79$) of the other benchmarks (excluding *SIN* and PGD) (Figure 8.12b). However, we notice that *IN-$\overline{C}$* shows an even stronger correlation with the remaining benchmarks ($\overline{R} = 0.83$). Even its weakest correlation (*IN-A*) is still moderately strong. Yet, just like *ImageNet*, it does not correlate with PGD and *SIN*. Future studies may prefer this benchmark to study robustness if they are restricted in their benchmarking budget, as *IN-$\overline{C}$* is a faster benchmark than *IN-C* due to fewer corruption classes.

8.7 Discussion and Conclusion

We have identified numerous shortcomings in previous studies showing correlations between shape bias, frequency biases, and the critical band to generalization. Our holistic and more carefully designed study shows that while correlations between *some* of these biases and *some* benchmarks exist, most of them fail to correlate to a broader view of generalization. Thus, an isolated measure of bias cannot be used to estimate generalization. Moreover, even if correlations exist, they might not be symmetric. For instance, we have seen that increasing adversarial robustness through adversarial training increases the shape bias in Chapter 7, but increasing the shape bias does improve adversarial robustness.

This brings us to the central question of this thesis: *Is it useful to directly optimize for these biases?* Or, *should we use these biases as priors?* After all, we have seen a case where improving one bias seems to have improved broad generalization in Chapter 5. Ideally, a prior would *monotonically* and *causally* correlate with wide-spectrum generalization. However, for the tested biases, we have seen many outliers, non-monotonic trends, and/or only correlations with some aspects of generalization. It is important to understand that, in this case, the utility of the bias concerning wide-spectrum generalization is significantly lowered, but it does not necessarily mean that the bias itself is not useful. Instead, it only tells us that such a bias alone is not sufficient to improve generalization but might still be helpful in combination with other aspects or helpful to optimize generalization for a specific and perhaps known distribution shift. Beyond that, there might be other valid reasons (*e.g.*, interpretability or alignment) to optimize these biases.

Additionally, we have seen that most biases show strong correlations on adversarially-trained models (*e.g.*, shape bias, low-frequency bias), but not on the entire model zoo. On the one hand, this means that the bias is not useful as a prior independent of the regularization, *i.e.*, it does not correlate if we discard this metadata. On the other hand, this might also be a limitation of our study, as our AT models are extremely homogenous compared to the other models. All the models were trained with the same hyperparameters, except for the adversary parameters (see Salman et al. (2020)). While we cannot reject causal correlations based on our model zoo, we cannot prove them either and look forward to future studies that study adversarial training more closely – also extending beyond L^p robustness.

Finally, we have seen some counter-intuitive trends where models misaligned with human perception perform better – *e.g.*, a stronger texture bias increases ID performance, and a high-frequency bias seems to improve generalization (except for adversarial robustness) despite containing next to no discriminative cues (see Figure 8.11 for examples). For the latter, it is worth noting that even the strongest high-frequency biased model only detects an average of 9% of the samples based on this band alone. As such, we cannot claim that a “high-frequency bias is all you need,” but drawing some predictive cues from this band seems to be necessary. In general, we want to issue a word of caution, as we are concerned that the reason for improvements we have seen might be due to dataset biases like frequency shortcuts (Wang et al., 2023b) and not an intrinsically better representation. A counter-argument to that theory is that we have seen correlations on many semantically different datasets, and it seems unlikely that they all contain the same shortcuts. Yet, we are curious to see if future work can derive a better understanding of generalization and design better benchmarks.

In summary, it seems that the generalization in neural networks – even when fixing the architecture – is too complex to be explained by a single bias. Yet, certain biases may be necessary (*e.g.*, a low bandwidth of the critical band, or a bit of high-frequency bias) to achieve generalization. It remains to be shown if a single bias or a combination of them exists that can explain generalization better.

Limitations This study explores biases in the *ImageNet* classification problem. While this dataset and problem are representative of a significant portion of computer vision research, our (and previous) findings may be limited in their transferability to other problems. For example, while there is a theoretical grounding for shape bias in object recognition, it is not intuitively clear if this applies to other classification tasks, such as medical recognition tasks. We ask researchers to exercise caution when extrapolating findings from specific contexts to broader applications and encourage a rigorous evaluation of model performance on their specific problem – in particular, in safety-critical domains where human lives are at stake.

Chapter 9

Talking Models Into Seeing the World Differently

In our last chapter, we want to deep-dive into biases beyond discriminative *ImageNet* classifiers. Unlike such traditional unimodal classifiers, large vision-language models (VLMs) offer an intuitive way to interact with visual content through language prompting by combining a large language model (LLM) with a vision encoder. However, both the LLM and the vision encoder come with their own set of cue preferences and shortcuts. While these have been rigorously studied in uni-modal models, a timely question is how these (potentially misaligned) vision biases behave under multi-modal fusion in VLMs and how we can control them, perhaps even through language – literally, talking models into seeing the world differently.

This chapter is based on "Can We Talk Models Into Seeing the World Differently?", accepted at ICLR 2025 (Gavrikov et al., 2025). As the first author, Paul Gavrikov collected the models, developed the code base, performed the experiments, created the plots, and wrote the paper with input from all authors. The teaser figure Figure 9.1 was created by Jovita Lukasik and refined by Steffen Jung, who also implemented access to `GPT-4V` and helped to polish plotting code.

Code: `https://github.com/paulgavrikov/vlm_shapebias`

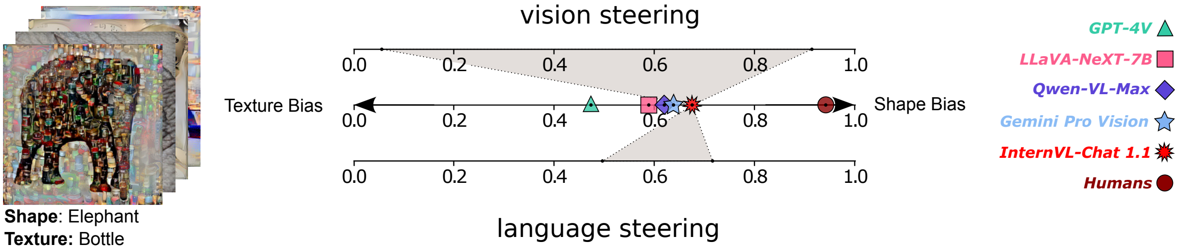


Figure 9.1: Language can be used to steer visual cue preferences (biases) in vision-language models (VLMs). Here we illustrate the (visual) texture/shape bias (Geirhos et al., 2019) of some exemplary VLMs and highlight the steerability of `InternVL-Chat 1.1` (Chen et al., 2024) through the processing of vision and language inputs (prompts).

9.1 Introduction

As the old adage goes, all models are wrong, but some are useful (Box, 1976). Similarly, recent machine learning models have proven to be very useful in practice, although we know their decisions to be impacted by specific biases, such as cue preferences misaligned with human perception and shortcuts

(Geirhos et al., 2020a). Some of these cue biases are particularly misaligned in traditional, uni-modal models and often reveal fundamental differences in the decision rule compared to humans (Geirhos et al., 2019; Subramanian et al., 2023; Wang et al., 2020a; Buolamwini & Gebru, 2018; Raji & Buolamwini, 2019). However, the current generation of deep learning models is increasingly multi-modal, for example, by fusion of large language models (LLMs) with modality-specific encoders (OpenAI, 2023; Alayrac et al., 2022; Huang et al., 2023). On the one hand, this approach allows for an exciting array of applications that can be defined at inference via prompts in natural language. On the other hand, the once well-studied biases are now combined in multi-modal fusion, leaving open questions on how and if the specific biases interact.

Specifically for vision-language models (VLMs), we are therefore asking if we can *talk* models into *seeing* the world differently – *i.e.*, to what extent does the LLM-based multi-modal fusion change the cues predominantly used for image classification and, furthermore, can we utilize natural language prompts to override the inductive biases of vision encoders. If language is indeed able to influence a vision-only bias, this may offer the possibility of aligning model behavior (with human behavior) using intuitive language prompting.

In general, determining biases in the visual cues used by a model to make a particular prediction is difficult (as outlined in Section 2.9.3). While we assume that there is a multitude of cue biases learned in vision models, only a few of them are harmful or misaligned.[1] An example of a benign bias would be the "foreground" bias, *i.e.*, models mostly classify images by their foreground objects. Similarly, objects in the image center are usually perceived to be more important than objects in the periphery. These biases follow human intuition and have, therefore, not been causing much controversial discussion.

This is in contrast to the texture vs. shape bias (Geirhos et al., 2019) – one of the best-studied cue biases in object recognition models (Hermann et al., 2020; Shi et al., 2020; Islam et al., 2021; Benarous et al., 2023; Naseer et al., 2021; Subramanian et al., 2023). It states that humans predominantly recognize objects in images by their shape (96% shape over texture decisions), whereas vision models strongly prioritize texture cues and discount the object's shape often. Machine perception is, thus, at odds with human intuition, even if the model accuracy is high.

We are the first to provide a large-scale study for VLMs for which we investigate the texture/shape bias in object recognition. Our investigation shows that the texture bias is, by default, far less pronounced in VLMs than in most previously studied vision-only models. As shown in Figure 9.1, VLMs decide by shape more often than by texture – albeit not matching the human shape bias (96%). Further, we find that by using biased instructions, we can steer the model output to some extent in both directions toward a texture or shape bias. This demonstrates that visual biases in multi-modal models can be influenced by language, opening up an exciting new possibility of aligning model outputs using language prompts without the need for retraining.

We summarize our contributions as follows:

- We show that VLMs preserve the bias of their vision-encoders only to some extent, yielding decisions that are more shape-biased than pure vision models, albeit not reaching human levels of the shape bias (Section 9.4).
- We show that the vision encoder generates a biased representation that contains texture and shape cues. Ultimately, the language modality (the LLM) tends to suppress either of the cues, such that objects are recognized purely on the grounds of either shape or texture (Section 9.4.2).
- We find that VLMs offer a unique opportunity to steer visual biases through language alone, stemming from the multi-modal fusion. For instance, we can steer the shape bias as low as 49% and as high as 72% through prompting alone without significantly affecting the accuracy (Section 9.5.2).

[1] We are explicitly focusing on biases in terms of low-level vision cues such as *shape* versus *texture* or *high-frequency* versus *low-frequency*. High-level biases that may have a societal impact are, therefore, explicitly excluded from this investigation. Please see Section 9.2 for a discussion.

9.2 Background

Sparked by the success of vision-language pretraining (Radford et al., 2021; Jia et al., 2021; Zhai et al., 2022b; Sun et al., 2023b), where features extracted from image-text paired data are aligned in a joint embedding space, recent VLMs added language modeling during training (Li et al., 2022; Yu et al., 2022b; Li et al., 2023b), enabling models to reason about images. Subsequently, finetuning these VLMs on instruction-following data (Alayrac et al., 2022; Liu et al., 2023c; Luo et al., 2023; Dai et al., 2023; Huang et al., 2023), such as reinforcement learning from human feedback (RLHF) (Ouyang et al., 2022), enables users to prompt these models, easing their usability for humans. Resulting models are commercialized (OpenAI, 2023; Gemini Team, 2023; Qwen Team, 2024) or open-sourced (Liu et al., 2023c; Dai et al., 2023; Chen et al., 2024), and consequently become accessible to a wide range of users.

This success of vision-language models calls for improving our understanding of the visual cues leveraged by VLMs and the degree to which these can be affected through language. In particular, the fact that vision models leverage cues different from the ones intuitively used by humans has been widely discussed. In our study, we focus on the texture vs. shape bias as a particularly well-studied example in vision-only models. Humans primarily rely on shape information to recognize objects. This is in contrast to standard *ImageNet*-trained vision models, such as Convolutional Neural Networks (CNNs), are biased towards texture to make their classification decisions (Geirhos et al., 2019). Nonetheless, shape information can still be present in layers/latent space of the model before the classifier (Hermann et al., 2020; Islam et al., 2021). Prior research has shown that the texture bias of CNNs can be reduced in training (Geirhos et al., 2019; Lukasik et al., 2023; Li et al., 2021; Hermann et al., 2020; Geirhos et al., 2021; Gavrikov et al., 2023; Jaini et al., 2024). But the network architecture has a high influence, and vision-only `Vision Transformers (ViTs)` (Dosovitskiy et al., 2021) were shown to be more shape-biased by default (Naseer et al., 2021), more human-like (Tuli et al., 2021), scalable by data size (Zhai et al., 2022a), and can be explicitly designed to separate shape and texture in their token space (Naseer et al., 2021). Jointly embedding vision and language in these networks (but not CNNs), through `CLIP` (Radford et al., 2021), further increases their shape bias in zero-shot classification (Geirhos et al., 2021). Yet, these models still do not reach human levels. The only known models to achieve such levels are image-generative classifiers (Jaini et al., 2024), which also combine vision and language in a different manner.

Measuring the texture/shape bias A cornerstone of our analysis is the measurement of the texture/shape bias in (LLM-based) VLMs when performing tasks that are based on object recognition. In the following, we summarize how this bias is measured for vision-only models, which forms the basis for our study.

Like most studies on the shape-texture bias in vision models, we use the texture-shape cue-conflict classification problem (or simply *cue-conflict*) (Geirhos et al., 2019) consisting of 1,280 samples with *conflicting* shape and texture cues synthetically generated via a style transfer model (Gatys et al., 2016) from *ImageNet* (Deng et al., 2009) samples (see Figure 9.1 for examples). The shape and texture classes belong to 16 super-classes of *ImageNet* (airplane, bear, bicycle, bird, boat, bottle, car, cat, chair, clock, dog, elephant, keyboard, knife, oven, truck). Following (Geirhos et al., 2019), we have excluded 80 images from the dataset where texture and shape cues belong to the same class. From an information perspective alone, predicting either cue label (or both) would be correct. However, humans tend to prioritize the shape cue for predictions, which is in stark contrast to most models (Geirhos et al., 2019).

Using the shape or texture cue label as the correct label allows us to measure the *shape* and *texture accuracy*, respectively. Based on these measurements, we measure the *cue accuracy* as the ratio of predictions that contain either the shape or texture label (as opposed to a misclassification):

$$Cue\ Accuracy = Shape\ Accuracy + Texture\ Accuracy \tag{9.1}$$

Throughout this chapter, we will refer to this as the *accuracy*. We use the definition of *shape bias*

(Geirhos et al., 2019), which is the ratio of shape decisions to all correct decisions (a specific case of the bias measurement defined in Equation 2.41):

$$\textit{Shape Bias} = \textit{Shape Accuracy}/\textit{Cue Accuracy} \tag{9.2}$$

While we primarily focus on measuring the shape bias in this study, the accuracy provides an important signal of the model's quality and robustness and for the comparability of results.

Other biases in deep neural networks We want to emphasize that we deliberately exclude high-level, societal biases from our considerations. Our study merely considers biases in the sense of low-level feature-based cues preferences.

High-level vision biases have been widely investigated, such as single-demographic effects (race and gender) for face recognition tasks (Buolamwini & Gebru, 2018; Raji & Buolamwini, 2019). For language models, several works focus on investigating societal biases, such as gender and race (Barikeri et al., 2021; Lauscher et al., 2021) and ways of debiasing them (Lauscher et al., 2021; Meade et al., 2022; Guo et al., 2022b), or explicitly forcing them (Haller et al., 2023). A recent study also found that LLMs are biased towards high-value over likely options (Sivaprasad et al., 2024). Another study focused on encoded moral beliefs (Scherrer et al., 2023). LLMs can also pick up human traits - one study found that adding "take a deep breath" to prompts improves performance (Yang et al., 2024). Of course, some of the uni-modal biases also apply to VLMs (*e.g.*, (Yang et al., 2024)), but a few works have also explicitly focused on biases in VLMs. For example, neurons of `CLIP` (Radford et al., 2021) were studied in (Goh et al., 2021), revealing that some neurons respond to the same concept regardless of its presentation, which is a potential reason for the high generalizability. On the other hand, this enables attacks by rendering text on images (*typographic attacks*). Additionally, several works demonstrated that VLMs fail to count objects (Radford et al., 2021; Liu et al., 2021a; Thrush et al., 2022) and generally struggle in structured tasks (Zhai et al., 2022b).

9.3 Measuring Cue Biases in VLMs

Given a dataset such as proposed in (Geirhos et al., 2019) for shapes and textures, we propose to measure the cue bias of VLMs through behavioral cue-conflicted testing (Section 2.9.3) in two tasks: *visual question answering (VQA)* (Antol et al., 2015), where we seek to obtain a zero-shot classification (Radford et al., 2021) of the object, and *image captioning* (Vinyals et al., 2015) where we look for an accurate but brief description of objects in the image. For both tasks, we evaluate single-round answering with no shared conversation history between conversations. To reduce the VLM behavior to a classification task, we follow the methodology outlined in Section 2.5.3 and explain the nuances in the following subsections.

9.3.1 VQA Classification

Following the questioning style in `LLaVA` (Li et al., 2023a), we ask the model to select the best option from an alphabetic enumeration of all class labels in the style `"A. airplane"`. The prompt for VQA Classification is:

```
"{VQA_INSTRUCTION}
A. airplane
...
Answer with the option's letter from the given choices directly."
```

with a default setting `VQA_INSTRUCTION="Which option best describes the image?"`.

For a simpler response extraction and confidence evaluation, we end the prompt by instructing the model to answer with only the letter corresponding to the correct answer. Compared to captioning, this is similar to the discrimination in *ImageNet* (Deng et al., 2009) image classifiers (Krizhevsky et al., 2012) in the sense that it only allows the model to respond with a single class and does not provide an option to not answer - if models follow the instruction.

Response extraction Despite instructing the models to only respond with an option letter, we observe multiple response styles: option letter + label (`"H. cat."`), just the label (`"cat."`), long explanation containing the option letter and/or label (`"The image features a black and white image of a cat."`). In all cases, punctuation and capitalization may be different (`"H.", "H", "h)"`). The first two response styles are easily correctable by simple post-processing (we prioritize the option letter in case of a conflicting option letter and label), and in some cases, explanations can be corrected as well if the response includes the option letter. However, we avoid heavy post-processing and consider individual answers wrong if they are not recoverable. In most cases, the ratio of these is negligible.

Active Steering through Prompts

The above setting allows testing of the inherent cue bias of a given VLM. Yet, the multi-modal nature of VLMs paves the way to not only test for a given bias but also to *actively steer* the model towards using particular types of visual cues. Note that models are not explicitly trained to perform well under this task, and it is unclear how flexible they are in basing a particular decision on one or another type of cue (for example, on texture or shape).

To explore the flexibility of model predictions for given types of visual cues, we conduct experiments comparing performance under a default neutral prompt and biased instructions.

In the simplest case, we can test the cue bias steering through hand-crafted prompting, where a model is asked to identify the class using a particular visual cue (*e.g.*, `"Identify the primary shape in the image."`). More generally, we set `VQA_INSTRUCTION` to `"Identify the primary {BIASED_TERM} in the image."`, with `BIASED_TERM` being `"shape"` and `BIASED_TERM` being `"texture"`, for shape-, and texture-biased prompts, respectively.

Automated prompt engineering To further enhance the steering signal provided by the language prompt, we further evaluate *automatically crafted prompts*. This is achieved by employing an LLM as optimizer (Yang et al., 2024) to continuously generate new prompts in natural language targeting to maximize either shape or texture bias in a feedback loop. We provide the LLM feedback about the achieved accuracy and shape bias. Additionally, we opt for greedy token sampling in the VLM to reduce noise in the feedback loop.

We utilize `Mixtral-8x7B-Instruct-v0.1` (Jiang et al., 2024), an open-source SOTA model at the time of writing, that performed better than the Nous Hermes LLM we use for response extraction in Section 9.3.2.

We instruct the model to provide a prompt in a new line starting with `"PROMPT: "` that we then extract and automatically evaluate. Afterward, we return the results to the LLM and ask it to generate the next prompt. We have experimented with multiple prompts, but ultimately, our approaches can be loosely divided into prompts that try to *maximize* or *minimize* a given bias without significantly affecting accuracy. Besides linguistic tweaks, we experimented with the following techniques:

1. **Offering rewards:** We offered tips to the LLM to encourage it to generate more and better results.[2] However, `Mixtral` seems to be finetuned to refuse such attempts.

2. **Adding in-context examples**: We added an example (in language) of what it means to be shape or texture-biased in classification. This often seemed to bias the model to generate prompts that contain the example, too.

3. **Summarizing previous attempts**: We encouraged the LLM to summarize previous attempts before generating the next prompt, hoping to keep the most important aspects in context. The LLM did not always follow this suggestion.

[2] `https://twitter.com/voooooogel/status/1730726744314069190` [Online; accessed 6. Mar. 2024]

4. **Returning the extracted prompt:** The LLM sometimes did not start the prompt with the requested prefix or misplaced it. We mitigated this by including the extracted prompt in our responses.

5. **Encouragements in response:** Initially, we only returned the accuracy and shape bias but found that the LLM sometimes abruptly quits the search. Thus, we included encouragements in the form of questions like `"What is your next prompt?"`. This seemed to improve conversations in terms of length but could not entirely prevent the LLM from quitting.

6. **Simple but creative prompts:** When just instructed the LLM to generate prompts, we noticed that it would sometimes collapse to verbose prompts where it would attempt to rephrase terms by synonyms. Inspired by regularization terms in optimization, we ask the model to keep its prompt simple and creative to avoid minor tweaking in favor of more radical changes.

In all cases, we append a mock conversation (*i.e.*, both roles are written by us) to the history containing the neutral prompt and the respective shape bias/accuracy. An example conversation is shown in Table 9.1. The first message is the final iteration integrating all of the above techniques.

It is worth noting that optimization with LLMs is a highly exciting but also very active research field where best practices have not yet emerged. For example, we have noticed that our instruction sometimes caused the LLM to refuse to continue when it deliberately determined that the search was exhausted or caused the LLM to maximize shape bias despite the instruction to minimize it. Overall, this is not an issue for our study, as we are merely interested in understanding if quantitatively more texture/shape-biased prompts exist.

9.3.2 Image Captioning

In this task, we are instructing models to generate brief descriptions (`"Describe the image. Keep your response short."`). We specifically request the model to provide a short response to encourage it to single out the most crucial aspects of the image according to its judgment. Additionally, this has the benefit of faster inference.

Response extraction As the responses are open-ended, we rely on zero-shot classifications of the generated description to marginalize the most descriptive class. To this end, we embed the generated descriptions and all (raw) class labels using `ember-v1` (LLMRails, 2024) and predict the class with the smallest cosine distance (similar to zero-shot classification in `CLIP` (Radford et al., 2021)). However, the generated caption may refer to multiple class labels (or none). As an additional signal, we perform a more granular analysis using an additional LLM (`Nous-Hermes-2-Mixtral-8x7B-DPO` (Teknium et al., 2024)) by instructing the model to extract all mentioned classes (similar to Yan et al. (2021)). This allows us to understand if the model detects both cues (but the embedding model enforces a specific prediction) and to quantify how often the model response is too generic to detect any class.

Specifically, we utilize the following prompt:

```
"Your task is to extract all objects that are described in the given message. Only
↪  answer with all letters from the given choices that apply. If none apply, reply
↪  with X. Do not explain. These are the possible objects:
A. airplane
...
Message: {Generated Image Caption}"
```

We post-process the output by splitting the generated string into a list. We found this prompt by manually testing some examples and picking the best-performing one. For example, we experimented with other options to denote generic responses like “-”. However, we found that this increases hallucinations, presumably as “-” is often used to begin bullet points and, thus, causes the model to continue generation.

Table 9.1: Example conversation for the LLM-guided prompt search. * denotes our mock conversation.

User*	Your task is to help me design an ideal image classification prompt for a vision/language model to MINIMIZE the shape bias without significantly hurting accuracy. The provided photos show natural objects modified to contain shape and texture of conflicting object classes. E.g., it could be a photo of an elephant (shape) with dog fur (texture). The model should classify the image as 'dog' based on the texture of the object and ignore the shape. The model's accuracy is the percentage of correctly classified images. The shape bias is the ratio of how often the model classified based on shape over texture. You can test your prompt by outputting a single new line starting with 'PROMPT: '. Do not list options - the system will provide them automatically. Try to keep the prompt as short and simple as possible but be creative. It might be reasonable to summarize insights of previous attempts and to outline your goals before responding with a new prompt, but make sure that only the prompt starts with 'PROMPT:'. In response to the prompt you will be told the accuracy and shape bias. Then you will refine the prompt and we will continue until I say stop. Let's go!
Assistant*	PROMPT: Which option best describes the image?
User*	Prompt: [...], Accuracy: 82.58 %, Shape Bias: 59.43 %. What is your next prompt?
Assistant	To minimize the shape bias while maintaining accuracy, let's encourage the model to focus more on textures than shapes. Here's my next prompt: PROMPT: Based on the surface characteristics and patterns, which category does the image most resemble?
User	Prompt: [...], Accuracy: 78.25 %, Shape Bias: 50.9 %. Can you improve this?
	. . .

9.4 Cue Biases in VLMs: An Analysis of the Texture/Shape Bias

We base our core analysis on the *cue-conflict* dataset evaluation of the texture versus shape bias (Geirhos et al., 2019). Traditional image classification models like `ResNet-50` trained on *ImageNet* have been shown to severely prioritize texture cues (only 22% shape bias), starkly contrasting the strong shape bias of humans (96%). As of now, results on multi-modal LLM-based models are missing. Therefore, we start our experimental evaluation by measuring the shape bias for the VQA and Image Captioning tasks, using a collection of diverse VLMs reflecting the multitude of research directions. These models include connections of common pretrained `CLIP` encoders and LLMs (Liu et al., 2023c,b, 2024; Dai et al., 2023; Bai et al., 2023b; Hong et al., 2023; Wang et al., 2023c; Sun et al., 2023a), mixture-of-expert LLMs (Lin et al., 2024), optimized architectures for resource-constrained systems (Kim et al., 2023), finetuning with RLHF (Sun et al., 2023c; OpenAI, 2023), or massive vision encoders (Chen et al., 2024). Additionally, we survey commercial, closed-source models like `Gemini Pro Vision 1.0` (Gemini Team, 2023), `GPT-4V (Preview)` (OpenAI, 2023), and `Qwen-VL Plus/Max` (Qwen Team, 2024) where access is limited to APIs and few details are known. For a brief description of these models, please refer to Table A.5 in Appendix A.4.

9.4.1 Key Results

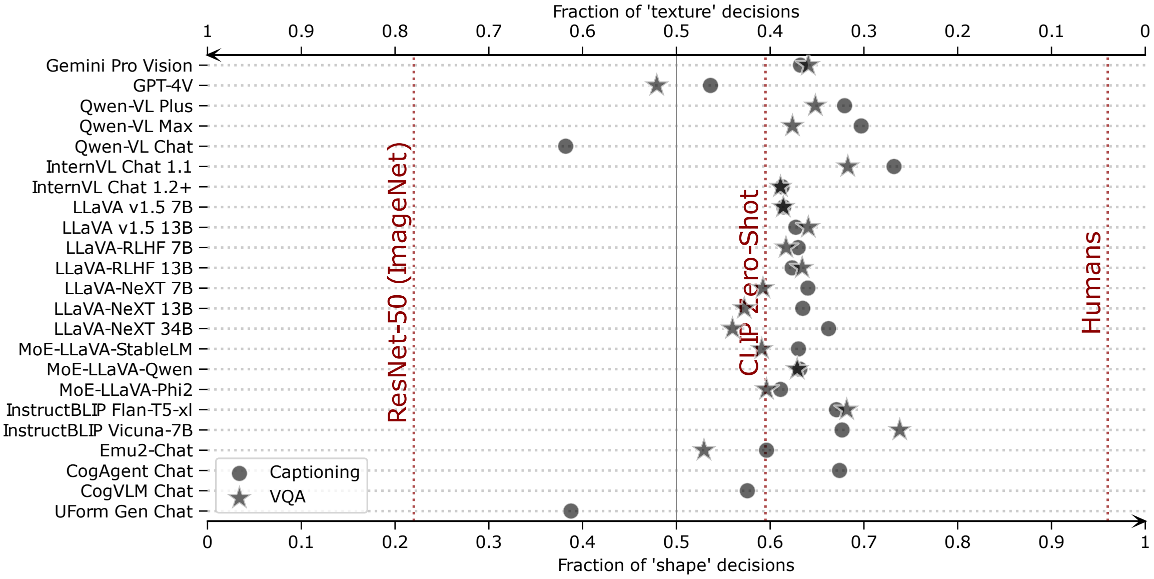


Figure 9.2: Most VLMs prioritize shapes over texture cues. We measure the shape bias on the *texture/shape cue-conflict* dataset (Geirhos et al., 2019). For reference, we also provide measurements on an *ImageNet*-trained `ResNet-50` (He et al., 2016), zero-shot classification with `CLIP ViT-L/14` (Radford et al., 2021), and a human average (over 10 subjects (Geirhos et al., 2019)). The results in table format are shown in Table A.6.

The results in Figure 9.2 paint a fairly uniform picture across different models and on two different tasks (a full table of results is shown in Table A.6 in Appendix A.4). Overall, the shape bias of VLMs is still significantly lower than that of humans (96%), but higher than in typical image-only discriminative classifiers (*e.g.*, 22% for an *ImageNet*-trained `ResNet-50` (Geirhos et al., 2019)). Additionally, **for most models, the shape bias is higher than the ca. 60% shape bias of `CLIP ViT-L/14`** (Radford et al., 2021) - an interesting result given that this model is a common vision encoder used in many of our tested models. `GPT-4V` (OpenAI, 2023) is an unexpected outlier both in terms of accuracy and in terms of texture bias. We further discuss this particularity in Section 9.7.

The task only marginally affects the shape bias. Despite conceptually different tasks, *i.e.*, the discriminative VQA task and the one open-ended captioning task, we do not observe fundamental shifts in the utilized information cue. We were able to report shape bias under the image captioning task for all models. However, a few models did not follow the VQA instructions and are, thus, not reported in Figure 9.2. Most of these models displayed a pronounced texture bias, which might hint towards a correlation between underfitting and texture bias, but to answer this question conclusively, we would need more samples.

On average, the shape bias is slightly higher for the image captioning task than for VQA (on average 63.9% versus 61.6% for those models that could be evaluated on both tasks). However, this comes at some cost in accuracy (on average 71.0% versus 78.9%). This decrease in accuracy is due to generic captions that do not refer to any class (see Table A.6 for details). For VQA the range is from 52.9 - 73.8% and 54.1 - 73.2% for captioning – yet outliers with a significantly lower (38.2%) shape bias in captioning exist, and for the individual models, the cue bias strongly depends on the considered task (refer to the gap between circles and stars for several of the models in Figure 9.2). Exceptions seem to be, for example, the `Gemini Pro Vision 1.0` model, several of the `LLaVA` models, and the `InternVL-Chat 1.2+` model for which the considered task barely influences the cue bias.

Which models are the most shape-biased? The strongest shape bias is observed in `InstructBLIP Vicuna-7B` (Dai et al., 2023) for VQA, but the model generally shows a lower accuracy compared to other models. A more accurate model is `InternVL-Chat 1.1` (Chen et al., 2024) which ranks second place for VQA but first for captioning.

Does LLM scale matter? LLM capacity does not seem to correlate with shape bias and unpredictably skews the shape bias by a few percent in each way, as can be seen in `Qwen-VL`, `LLaVA v1.5/NeXT/RLHF`, or `InternVL`. Similarly, the overall largest models do not have the highest shape bias. However, following the overall general trend, in our experiments, we also found that scale usually improves accuracy.

Does RLHF align shape bias? RLHF-tuned VLMs were still rare at the creation of this survey, and we only have three model samples. On both `LLaVA-RLHF` (Sun et al., 2023c) models, we see no changes in comparison to the default `LLaVA` models. `GPT-4V` (OpenAI, 2023) (though it is unclear if vision was also RLHF trained) shows one of the lowest shape biases in our study, but we do not know how the base model ranks. Overall, it is hard to derive a conclusive answer, but it seems that at least RLHF does *not necessarily guarantee* an alignment of visual preferences.

9.4.2 Mechanistic Analysis

We have observed that the shape bias in VLMs differs from that of `CLIP (ViT-L/14)`, the vision encoding model used in most of the tested models. This prompts an inquiry into the VLM's decision process. Specifically, it raises the question of whether language can affect a purely visual like the texture/shape bias. In this section, we want to specifically look at the vision encoder and its representation, having only access to the visual input and the LLM combining both modalities.

Vision Encoding

Most VLMs combine a frozen `CLIP` vision tower with an LLM via some projector (Dai et al., 2023; Liu et al., 2023c; Alayrac et al., 2022). Hypothetically, the LLM could learn to perform zero-shot classification using their encoders akin to a function call whenever the prompt requires some form of classification and then simply forward the result. In such a case, the VLM would also inherit the shape bias from the encoder. To gain more insights, we ablate the encoder using zero-shot classification from the full model. We derive the encoder's predictions by calculating the cosine similarity between the encoded class labels[3] and the input sample, selecting the label with the highest similarity to the

[3] We explore various prompt templates (and ensembles) in Section 9.6.2, yielding consistent results.

Table 9.2: Comparison between VLMs and their encoders. We show the relative difference between a VLM and its encoder in shape bias and accuracy when evaluated on *texture/shape cue-conflict* tasks, along with error consistency (Geirhos et al., 2020b). VLM performances are assessed using VQA, while the vision encoders are evaluated using zero-shot classification. Statistically significant changes in shape bias ($p < 0.05$ in a two-sided t-test) are denoted by an * next to the value.

		VLM - Encoder [%]		**Error Con-**
VLM	**Vision Encoder**	Accuracy	Shape Bias	**sistency** [%]
`LLaVA v1.5 13B` (Liu et al., 2023b)	❄ `CLIP ViT-L/14@336px` (Radford et al., 2021)	−3.50	+4.2*	73.5
`LLaVA v1.5 7B` (Liu et al., 2023b)		−3.00	+1.5	76.8
`LLaVA-NeXT 34B` (Liu et al., 2024)		−9.92	−3.9*	67.1
`LLaVA-NeXT 13B` (Liu et al., 2024)		−0.33	−2.7	70.9
`LLaVA-NeXT 7B` (Liu et al., 2024)		−1.08	−0.2	67.5
`MoE-LLaVA v1.5 Phi2 x4` (Lin et al., 2024)		−1.42	−0.3	73.4
`MoE-LLaVA v1.5 Qwen x4` (Lin et al., 2024)		−24.25	+3.0	51.4
`MoE-LLaVA v1.5 StableLM x4` (Lin et al., 2024)		−3.67	−0.8	73.1
`InstructBLIP FLAN-T5-XL` (Dai et al., 2023)	❄ `EVA-01-CLIP ViT-g/14@224px` (Sun et al., 2023b)	−6.83	+1.8	73.7
`InstructBLIP Vicuna-7B` (Dai et al., 2023)		−14.42	+7.4*	78.7
`Emu2-Chat` (Sun et al., 2023a)	❄ `EVA-02-CLIP-E/14+@448px` (Sun et al., 2023b)	−11.08	−9.5*	61.0

image (as described in Section 2.5.2). Specifically, we measure the difference in accuracy and shape bias between the encoders of the VLM and the VLM itself, as well as their *error consistency* (Geirhos et al., 2020b).

Error consistency is a metric to assess whether two observers (*e.g.*, a human and a model) systematically make errors on the same images. If that is the case, this suggests a deeper underlying similarity compared to simply reaching similar overall accuracies since those could be reached with very different strategies. The metric is based on Cohen's κ (Cohen, 1960) and computes how frequently errors made by two observers overlap (up to perfect overlap) while correcting for the overlap expected by chance. Cohen's κ is within $[-1, 1]$, with a value of 0 indicating chance-level consistency, positive values indicating systematic agreement, and negative values indicating systematic disagreement. For our measurement, we treat all predictions other than the shape label that are not related to the shape cues as errors. We show results for the VQA task in Figure 9.3.

The vision encoder provides a *flexible* representation Our comparisons in Table 9.2 show that VLMs' decisions differ from their isolated encoders. Even on the rather simple 16-way cue-conflict problem, all VLMs decrease in accuracy compared to their encoders in zero-shot classification. This may be expected as the increase in supported tasks in VLMs comes at a cost in specialization but is already a sign of changes. Further, we note that the error consistency to their respective encoders only matches up to 78.7%. This leaves at least 20% room for decisions that differ from the encoder, proving that the LLM and the text prompt further influence the shape bias despite being a vision-only bias. This deviation can only be possible if the generated vision tokens are flexible to some degree by containing information belonging to both cues. This is further confirmed by the measurement of shape bias, which shows a −9.5% to +7.4% difference in **both** directions.

LLM Processing of Vision Tokens

In the VQA task, we force the model to predict a single class – yet the previous section has shown that the vision tokens generated by the encoder contain information from both cues. To better understand the processing, we evaluate the VQA prediction confidences as follows.

All answer options in our VQA prompts correspond to a single character and, thus, a token. Well-behaving models, where the response consistently starts with the option letter (and nothing else), allow us to gather insights into the prediction process. For these models, the logits of each token correspond to the logits of option letters. By applying a Softmax function (Equation 2.24) over the token logits belonging to answer keys, we can analyze the confidence in each option (we select the top 16 tokens, ignore all invalid tokens, and set the probability of missing answer tokens to 0). This allows us to better study the sampling behavior and make the following observation:

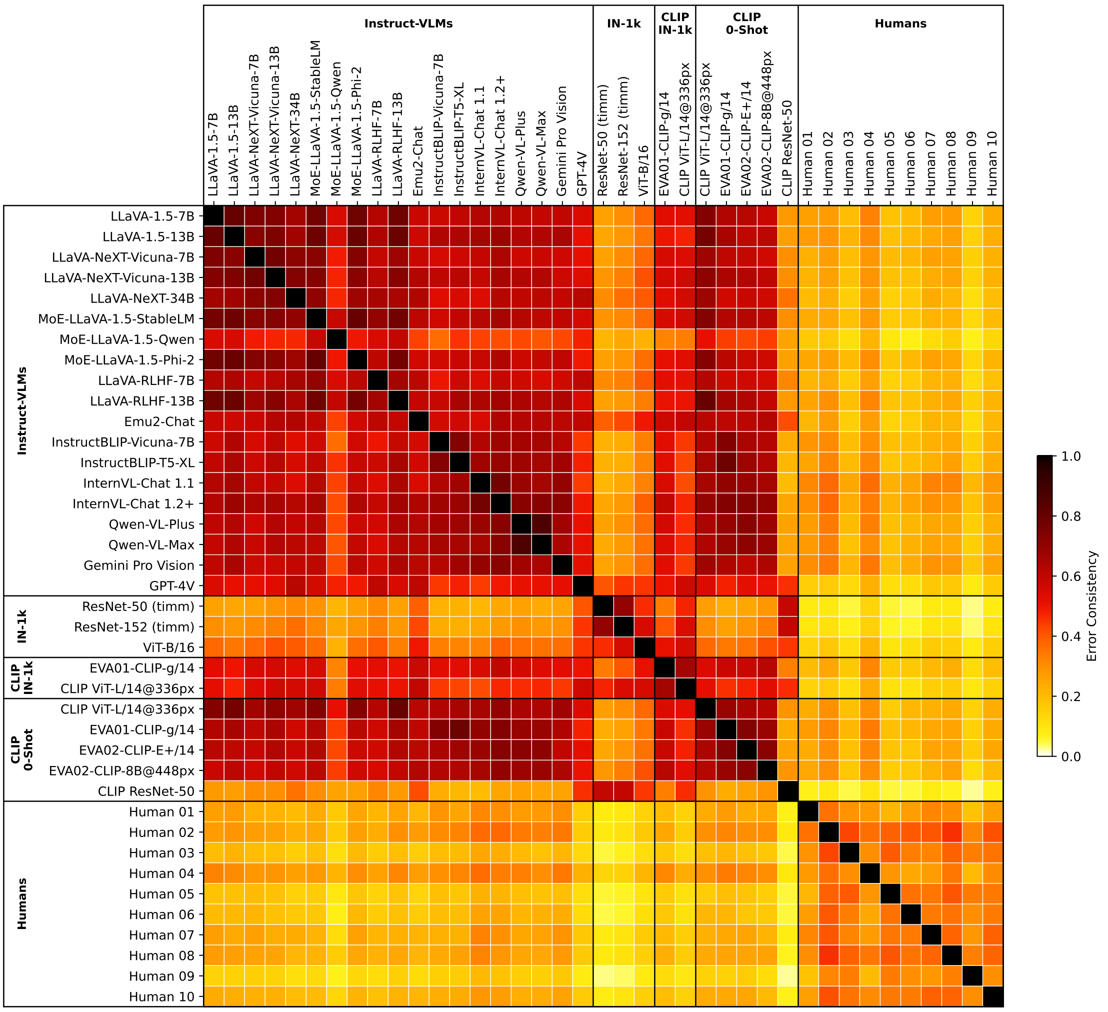
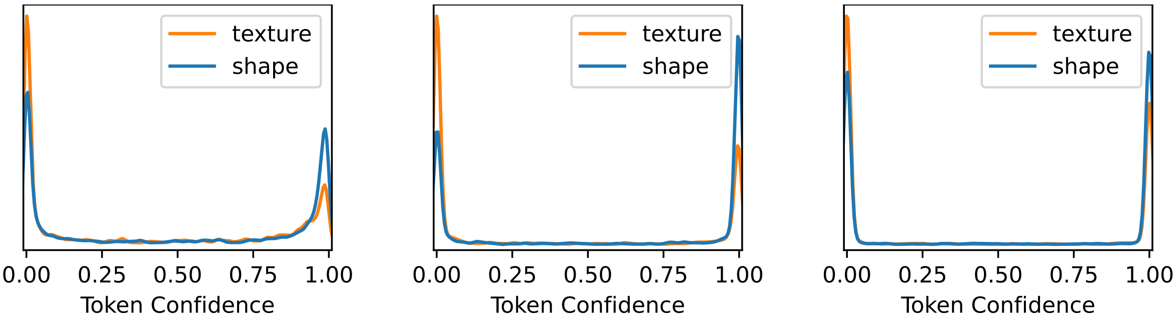


Figure 9.3: Error consistency between *texture/shape cue-conflict* observers. We measure the pair-wise error consistency (Geirhos et al., 2020b) between all observers. For this analysis, an error is any answer that does not belong to the shape class. Shown responses belong to LLM-based VLMs (under the VQA task), other selected models including *ImageNet* models, (some) VLM encoders under *ImageNet*-finetuning and zero-shot classification, and ten human subjects.



(a) `LLaVA-NeXT 7B` (b) `InternVL-Chat 1.1` (c) `MoE-LLaVA-Phi2`

Figure 9.4: Confidence distribution of shape and texture tokens for all samples. All models form highly biased decisions by completely ignoring one cue. Measured on `LLaVA-NeXT 7B`, `InternVL-Chat 1.1`, and `MoE-LLaVA-Phi2` for the VQA task.

LLMs turn flexible representations into biased decisions In Figure 9.4, we visualize the confidence of the token corresponding to the shape or texture answer option. This experiment can only be performed on models where we have access to the logits, and the model consistently follows the instructions to respond only with the predicted options letter. This limits the analysis to a few models in our zoo, and we show similar results on `LLaVA-NeXT 7B` (Liu et al., 2024), `InternVL 1.1` (Chen et al., 2024) and `MoE-LLaVA-Phi2` (Lin et al., 2024).

To our surprise, we find that confidence in both options is almost binary. Analogously, when we only focus on correct answers, we observe that the model is highly confident in its responses. As the model places such high confidence in the selected cue, this suggests that information from the alternative cue is effectively disregarded during LLM processing. This conclusion is further supported by the observation that the second, significantly less confident prediction token does not align with the conflicting cue. For example, in `LLaVA-NeXT 7B` (Liu et al., 2024), this occurs in 70.7% of cases. Only 17.7% of the top-2 pairs contain both shape and texture. Thus, while the encoder has its own inductive bias that directs the final decision, the actual biasing happens in the LLM, similar to linear classification heads in discriminative models (Islam et al., 2021).

While it is not clear if these findings generalize to all other VLMs, the overall high error consistency between all VLMs (see Figure 9.3 for a heatmap) on our task hints that our results may generalize well to other models.

9.5 Bias Steering in VLM Outputs

In the previous section, we have seen that in VLMs, visual biases are not simply inherited from the vision encoder, but the fusion with an LLM, including the text prompt, plays a crucial role. Given the somewhat flexible representation of texture/shape bias, we test in the following if we can actively talk VLMs into seeing the world differently, *i.e.*, systematically *steer* the output towards either end of the bias and go beyond the inductive bias.

Since the texture/shape bias is a vision-only bias, we first look into visual steering through image preprocessing (Section 9.5.1). Then, we explore the influence of language through prompt engineering on the visual texture/shape bias (Section 9.5.2). Furthermore, while we expect strong model steerability through image preprocessing (obviously at high costs in model accuracy), Figure 9.1 indicates significant flexibility of (some) VLMs through language, potentially offering a powerful way of shaping visual biases in a user-specified way without the need to retrain a model. Finally, we study if prompt-based bias steering can generalize beyond the texture/shape bias in Section 9.5.3.

9.5.1 Vision-based Steering of the Texture/Shape Bias

Before we turn our analysis to steerability through prompts, we show how and to which extent shape bias can be steered in vision through perturbations of the input images.

Earlier work demonstrated that *ImageNet*-models can still detect objects even if the image is split into patches and shuffled (Zhang & Zhu, 2019; Shi et al., 2020; Naseer et al., 2021). As patch size decreases, the operation is destroying more global shape information, yet retaining local texture information. We utilize this technique to significantly increase *texture bias*. Oppositely, to increase the *shape bias* we experiment with added Gaussian noise to inputs. This is loosely inspired by applying "diffusion-like noise" during training

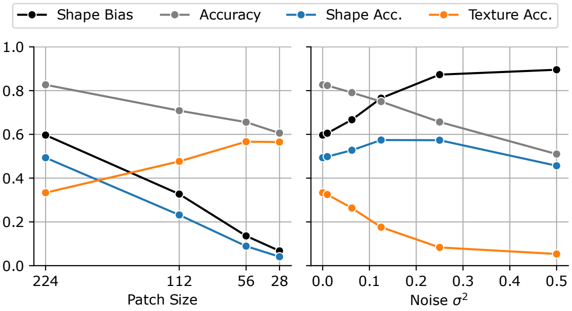


Figure 9.6: Image preprocessing can strongly steer texture/shape bias. Left: Shuffling image patches with decreasing patch size results in a strong texture bias. Right: Increasing Gaussian noise introduces a strong shape bias.

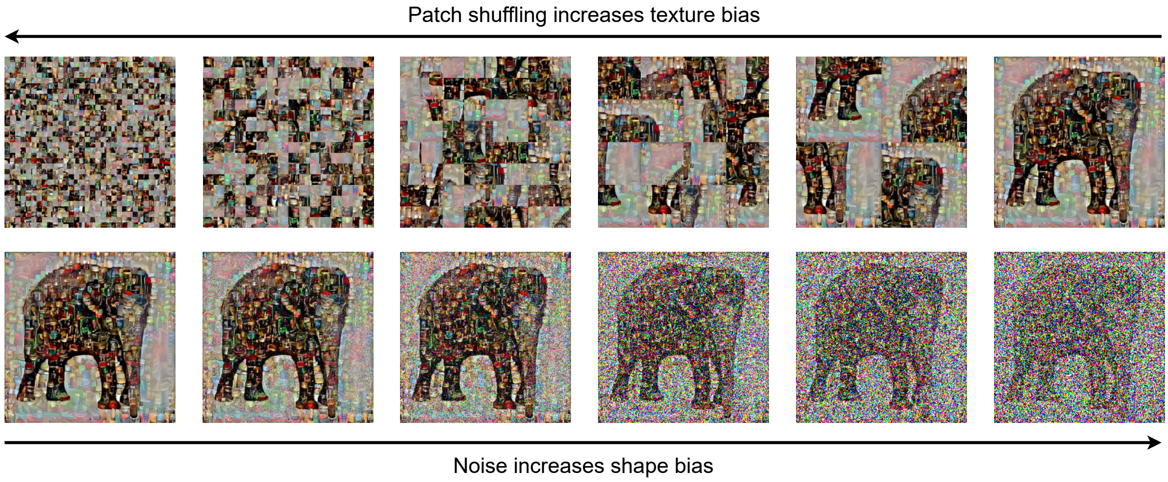

Patch shuffling increases texture bias
Noise increases shape bias

Figure 9.5: Steering in the vision modality. For one example image, we show how patch shuffling (top) increases texture bias by destroying shape information. Below, we show how adding Gaussian noise increases shape bias by destroying texture information. Please note that we show more extreme values than those used in our experiments for visualization purposes.

(and inference), which has been shown to drastically improve the shape bias of *ImageNet*-`ResNets` (Jaini et al., 2024). However, we only apply the noise during inference and use a more simplistic approach by adding $\mathcal{N}(0, \sigma^2)$ noise to all channels, consecutively clamping values to $[0, 1]$. We visualize the effects on one *cue-conflict* sample in Figure 9.5.

We show results on `LLaVA-NeXT 7B` in Figure 9.6. Adding noise results increases the VLMs shape bias up to 89.5% at $\sigma^2 = 0.5$, and patch shuffling decreases shape bias (increases texture bias) to 8.4% at 28×28 patches. In both cases, the bias is indeed *steered* up to a certain threshold: the accuracy on one cue (texture or shape accuracy) increases, whereas the accuracy on the other decreases. Beyond a specific point, the operation is destroying one cue entirely and can no longer be considered steering. Further, this form of steering comes at a cost in accuracy – yet all results are still well beyond random chance.

Inspired by these strong results, we repeat the experiments on the naturally more shape-biased and larger `InternVL-Chat 1.1` (Chen et al., 2024). In this model, we can further extend the range to 91.7% ($\sigma^2 = 0.3$) and down to 6.1% (28×28 patches) shape bias.

9.5.2 Prompt-based Steering of the Texture/Shape Bias

Our previous results suggest that VLMs learn a connected multi-modal understanding of shape and texture. This opens the question of whether visual biases in outputs can be influenced through text processing in these models. We test this hypothesis by recording texture/shape bias as a function and steering it via text through prompt engineering.

Hand-crafted prompting We start by asking VLMs to specifically identify either the "shape" or the "texture" category in a given *cue-conflict* image. As shown in Figure 9.7, prompting can steer a visual bias (*without* significantly affecting accuracy). Neutral prompts often perform similarly to shape-biased prompts, whereas texture-biased prompts deviate more significantly. This suggests that models may be more inclined to use shape by default but also have access to a certain amount of texture information, which can be accessed through biased prompting.

To better test the representation of "shape" and "texture", we additionally replace the respective terms with strong synonyms obtained from *Thesaurus.com* (Dictionary.com, 2024a,b):

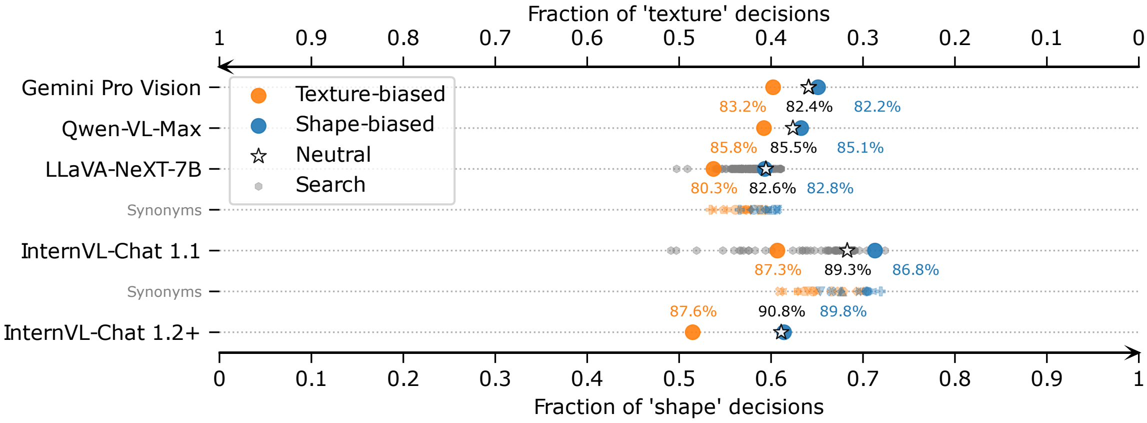


Figure 9.7: Language can steer the texture/shape bias to some extent. We test the same texture/shape-biased instructions on multiple models, showing that these can already shift some decisions (usually in favor of texture). The stated percentages refer to the achieved accuracy on *cue-conflict*. For `InternVL 1.1` and `LLaVA-NeXT 7B`, we additionally test the understanding of texture/shape by using synonyms. Furthermore, we use an LLM to automatically search for specific prompts to optimize in either direction.

- **shape:** architecture, aspect, body, configuration, contour, format, frame, model, outline, pattern, shadow, silhouette
- **texture:** balance, character, composition, consistency, fabric, feeling, make-up, nature, pattern, quality, sense, smoothness, structure, surface, taste, touch

Our biased prompt is formed by replacing `BIASED_TERM` in the biased VQA prompt with the corresponding synonym.

Then we measure shape bias on (`InternVL-Chat 1.1` (Chen et al., 2024) and `LLaVA-NeXT 7B` (Liu et al., 2024)). Synonyms of either term can steer shape bias as well to a certain degree. For "texture" synonyms, we observe more variance, as "texture" is overloaded by different meanings (*e.g.*, some synonyms like "feeling", "taste", or "touch" are unrelated to texture in vision). In contrast, "shape" is a fairly well-defined term. This demonstrates that the steering is not coincidental but leverages a learned representation.

While the effect of steering by language is systematically visible, language steering alone does not fundamentally change the reliance on the underlying cue. This effect does not appear to be a limitation of LLM capacity – the evaluation on `InternVL-Chat 1.2+` (34B vs. 13B) does not provide evidence that larger LLMs offer more steerability.

Automated prompt engineering Did our results indicate a limit on how much language/prompting can influence biases, or merely reflect that the handcrafted prompts were chosen suboptimally? To address this question, we test *automatically crafted prompts* (see Section 9.3).

The results are shown in Figure 9.7 in gray and denoted as "search". We observe that for both `LLaVA-NeXT-7B` and `InternVL-Chat 1.1`, automatically generated prompts exceed the manually crafted biased prompts in terms of their effectiveness to increase texture bias and roughly match them when it comes to increasing shape bias. For `InternVL-Chat 1.1`, the delta between both extremes is 23.3%, which can only serve as a lower bound and is likely improvable by a better design of the LLM task (or using other optimizers). In line with hand-crafted prompts, overall accuracy does not change considerably or sometimes even improves. We should also note that the optimization is done for the *texture/shape cue-conflict* test set; this is simply done as a proof of concept to show that there are prompts that can influence visual biases substantially, and not to claim a SOTA shape bias.

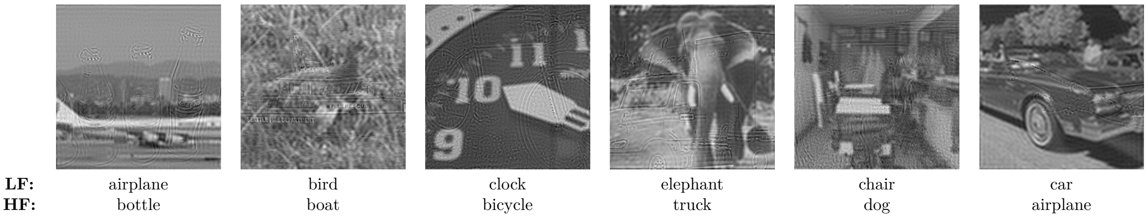


Figure 9.8: Examples from *frequency-cue-conflict*. Images are constructed from conflicting low (LF) and high-frequency (HF) cues. Zooming may be necessary to see the high-frequent cues.

Steering human vision We also briefly compare our findings to human perception experiments conducted by Geirhos et al. (2019). In their control experiments, humans were either instructed to identify the shape while ignoring texture or, conversely, to identify the texture while ignoring the shape. This "human prompt steering" worked, but only to a certain extent: When humans were tasked to ignore the shape, the human shape bias decreased from 96% (neutral instruction) only to approx. 70% shape bias (texture-biased instruction). Our tested VLMs behave somewhat similarly: their visual shape bias can be steered through prompting, but it appears hard for them to completely go against their default visual bias.

9.5.3 Prompt-based Steering of Other Biases

So far, we have analyzed the texture/shape bias. In this section, we show that steering is possible for other biases, too. To this end, we explore a bias originating in the spectral domain, specifically focusing on low-/high-**frequency cue conflicts** (Oliva et al., 2006). This bias has been shown to affect a classification model's robustness, for example, in (Wang et al., 2020a) and our study in Chapter 5. A related observation has been made in (Subramanian et al., 2023), showing that the critical frequency band of object recognition separates human perception from model vision.

Table 9.3: Prompt steering on *frequency-cue-conflict*. Statistically significant changes are marked by * (two-sided t-test with $p < 0.05$). We compare the "neutral" prompt with found prompts to maximize ("search (max)") or minimize the respective bias ("search (min)").

Model	**Prompt**	**Accuracy** [%]	**LF Bias** [%]
`InternVL-Chat 1.1`	neutral	92.92	34.5
	search (max)	90.33	38.6*
	search (min)	91.33	32.9
`LLaVA-NeXT 7B`	neutral	82.83	52.4
	search (max)	84.25	54.5
	search (min)	82.67	48.7*

To investigate the steerability of frequency biases, we present a novel dataset of stimuli following the *texture/shape cue-conflict* benchmark methodology outlined by (Geirhos et al., 2019). This dataset consists of 1,200 samples across 16 *ImageNet* super-categories. Each stimulus is created by blending the low and high-frequency components of two images with conflicting labels from a selected subset of 16 *ImageNet* classes. First, we randomly select two samples with different labels from this subset. Each image is converted to grayscale by extracting the L channel, resized to 256 pixels along the shortest edge while maintaining the original aspect ratio, and then center-cropped to 224×224 pixels. The blending process involves combining 30% of the low-frequency components from one image with 70% of the high-frequency components from the other. The resulting stimuli are saved in JPEG format at 100% quality to preserve high-frequency details and avoid compression artifacts. Examples of these stimuli are shown in Figure 9.8.

We test the same neutral prompt as for previous experiments and use our automated prompt search to either maximize or minimize the bias. The results in Table 9.3 show that, again, we can steer the bias by language. The range, of course, depends on the vision representation and language training and is not as pronounced as for texture/shape bias. Still, we find that our prompts can result in statistically significant changes in bias. On `LLaVA-NeXT 7B`, the prompts even improved accuracy alongside the bias.

Independent of the prompting, it is interesting that the smaller `LLaVA-NeXT 7B` model almost perfectly balances the conflicting cues, whereas the larger `InternVL 1.1` model is significantly biased toward HF. We hope that this dataset can pave a new avenue for future research on frequency bias.

9.6 Ablations

In this section, we aim to ablate our experiments to see if we introduced biases through our methodology.

9.6.1 VLM Prompts

VQA

In initial testing, we found that the choice of prompts affects the eventual results and has the potential to inevitably influence our study. Thus, in an effort to address this, we extensively evaluated our models with multiple different prompting techniques used in literature (Liu et al., 2023b; Dai et al., 2023) and chose the best one. Our prompt for VQA is inspired by `LLaVA`'s prompts for multiple-choice questions (LLaVA Team, 2024). In an additional experiment (Table 9.4), we ablated alternative prompts on `LLaVA-NeXT 7B` (Liu et al., 2024). We change or use an empty `VQA_INSTRUCTION` and change options to `CLIP`-style options (`"X. a photo of a {class}"`). However, we only observed a minor fluctuation in accuracy and shape bias and no significant effects. Our default prompt delivers the best accuracy and is, thus, our preferred choice.

Table 9.4: Exploration of alternative VQA prompts.

Prompt	**Shape Bias** [%]	**Accuracy** [%]
`"Which option best describes the image? [...]"` (default)	59.2	82.58
Default with `CLIP`-style options	59.5	81.92
`"Describe the object in the image: [...]"`	60.2	81.33
`"Describe the object in the image: [...]"` with `CLIP`-style options	59.4	80.17
Empty instruction (just options)	59.5	81.33

Image Captioning

Our image captioning prompt is a reformulation of the VQA prompt (`"Which option best describes the image?"` → `"Describe the image."`). In the following, we ablate if the suffix (`"Keep your response short."`) may have interfered with our results. Additionally, we tested an alternative suffix that explicitly asks for more details on `LLaVA-NeXT 7B` (Liu et al., 2024). The results for the former investigation, in Table 9.5, show that our suffix indeed did not heavily bias the results in terms of shape bias. While adding the suffix leads to an impact on accuracy, it reduces the ratio of generic descriptions (not referring to any class). It has, on average, almost 4x fewer tokens, resulting in significantly faster inference. Switching the suffix to `"Be precise."` increases shape bias but, at the same time, also increases the number of generated tokens and, worryingly, the ratio of generic responses. Overall, we find that captioning prompts are more fragile, but our chosen default prompt provides a fair balance. For all ablated prompts, we find that the shape bias is higher than in VQA.

Table 9.5: Exploration of alternative Image Captioning prompts.

Prompt	Shape Bias [%]	Accuracy [%]	Avg. Tokens	Generic Ratio [%]
`"Describe the image. Keep your response short."` (default)	64.0	65.08	55.5	39.5
`"Describe the image."`	63.6	68.25	202.9	46.8
`"Describe the image. Be precise."`	67.3	64.50	166.2	50.6

9.6.2 CLIP Prompts

In this section, we provide results for different `CLIP` models under three different prompting strategies: a computation of zero-shot centroids from 80 different prompts including usage of the class name (Radford et al., 2021), `"a photo of {class}."` which is often used as a default prompt (note the dot), and `"{class}"` (without dot). We will argue that the latter is more comparable to the VQA task of our VLMs, but, of course, VLMs may have a better representation in weights. Either way, the shape bias does not significantly deviate between the three strategies, as can be seen by shape bias (and accuracy) measurements in Table 9.6.

We also noticed that the observed scaling laws in (Geirhos et al., 2021) do not always hold for vision encoders, despite an increase in parameters from `EVA02-CLIP-E/14+ (5B)` to `EVA02-CLIP-8B`, we actually see a significant decrease in shape bias (but an improvement in accuracy). This complements our findings about shape bias in Section 8.6.

Our results also contain (rather uncommon) `ResNet-based CLIP` models. Note these are the only models, where the 80 prompts significantly improve accuracy. In terms of shape bias, `CLIP-ResNets` significantly underperforms any `ViT` or `CLIP-ViT`.

Table 9.6: Zero-shot classification on *texture/shape cue-conflict* with different `CLIP`(-like) joint embedding models.

Model	Prompt	Shape Bias [%]	Accuracy [%]
`EVA01-CLIP-g/14` (Sun et al., 2023b)	80 Prompts (Radford et al., 2021)	66.03	87.83
`EVA01-CLIP-g/14` (Sun et al., 2023b)	`"a photo of a {class}."`	66.03	87.08
`EVA01-CLIP-g/14` (Sun et al., 2023b)	`"{class}"`	66.44	86.67
`EVA02-CLIP-8B@448px` (Sun et al., 2024)	80 Prompts (Radford et al., 2021)	58.26	91.83
`EVA02-CLIP-8B@448px` (Sun et al., 2024)	`"a photo of a {class}."`	57.58	89.00
`EVA02-CLIP-8B@448px` (Sun et al., 2024)	`"{class}"`	56.60	88.33
`EVA02-CLIP-E/14+` (Sun et al., 2023b)	80 Prompts (Radford et al., 2021)	65.62	90.67
`EVA02-CLIP-E/14+` (Sun et al., 2023b)	`"a photo of a {class}."`	64.44	89.75
`EVA02-CLIP-E/14+` (Sun et al., 2023b)	`"{class}"`	62.48	86.17
`CLIP-ViT-L/14` (Radford et al., 2021)	80 Prompts (Radford et al., 2021)	60.95	84.08
`CLIP-ViT-L/14` (Radford et al., 2021)	`"a photo of a {class}."`	60.20	84.17
`CLIP-ViT-L/14` (Radford et al., 2021)	`"{class}"`	60.16	81.17
`CLIP-ViT-L/14@336px` (Radford et al., 2021)	80 Prompts (Radford et al., 2021)	61.52	86.83
`CLIP-ViT-L/14@336px` (Radford et al., 2021)	`"a photo of a {class}."`	60.56	86.42
`CLIP-ViT-L/14@336px` (Radford et al., 2021)	`"{class}"`	59.80	83.75
`CLIP-ResNet-50` (Radford et al., 2021)	80 Prompts (Radford et al., 2021)	19.70	77.83
`CLIP-ResNet-50` (Radford et al., 2021)	`"a photo of a {class}."`	20.96	72.75
`CLIP-ResNet-50` (Radford et al., 2021)	`"{class}"`	20.77	71.83
`CLIP-ResNet-101` (Radford et al., 2021)	80 Prompts (Radford et al., 2021)	25.50	74.83
`CLIP-ResNet-101` (Radford et al., 2021)	`"a photo of a {class}."`	25.23	71.00
`CLIP-ResNet-101` (Radford et al., 2021)	`"{class}"`	25.41	70.83

9.6.3 Temperature Scaling

We are also interested in determining if generation parameters can influence the behavior of shape bias. Generally, VLMs only expose a few controllable parameters, but all offer some form of stochastic sampling of tokens, often via *temperature scaling* of the token logits (see Section 2.5 and Equation 2.24). Most models default to low-temperature settings (or settle for a greedy token strategy), which is more correlated with precise answers and is reasonable for VQA. On the contrary, higher temperatures are correlated with more creative outputs and eventually token gibberish at extreme values. In general, temperature scaling also results in better-calibrated models (Guo et al., 2017).

Exemplarily, we study this on `LLaVA-NeXT 7B` for both VQA and Image Captioning. We repeat the non-greedy experiments three times for statistically meaningful results; however, we generally notice a marginal error between runs. Our results in Figure 9.9 show no significant correlation between temperature in the $[0, 1]$ range and shape bias. As expected, the accuracy (slightly) decreases with increasing temperature because less confident tokens mapping to correct predictions are now replaced by false predictions. Yet, this appears to affect texture/shape information alike. This can easily be explained by our token sampling analysis in Section 9.4. On average, texture/shape options are fairly similarly confident, and top-1 tokens have very high confidence (in the VQA setting), which the temperature scaling barely affects.

On the one hand, this finding serves as important confirmation that our comparison of VLMs at default values (picked by the original authors) is reasonable, as it does not interfere with the shape bias. On the other hand, this implies that users seeking more creative outputs can tune the temperature (and similarly other parameters that control stochastic token sampling) without changing the utilized cues for vision inputs.

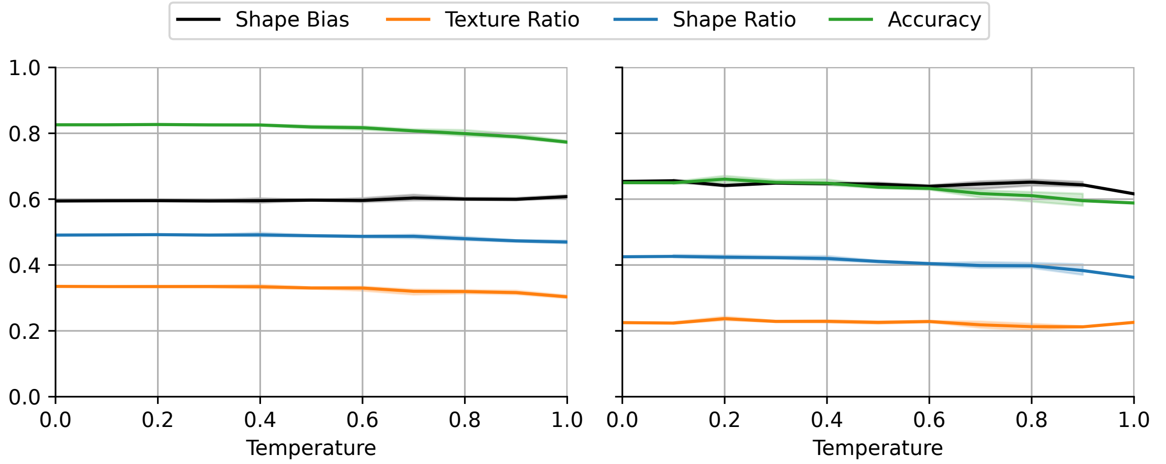


Figure 9.9: Ablation of temperature scaling. Temperature scaling has no significant effect on shape bias neither under VQA (left) nor Image Captioning (right) tasks but starts to decrease accuracy at higher levels. Experiments performed on `LLaVA-NeXT 7B` with 3 seeds (except Temperature = 0 and Temperature = 1 of Image Captioning where we use a single seed).

9.6.4 Additional Thoughts on the Image Captioning Task

Did our choice of embedding model bias the results? While we assume that most embedding models will provide similar classification performance if the description clearly mentions one class, it is unclear how the classification is biased if the description refers to multiple classes, invalid classes, or is generic. Thus, we additionally ablate results with `SFR-Embedding` (Meng et al., 2024), which at the time of writing was the overall SOTA English embedding model on the *Massive Text Embedding Benchmark (MTEB)* (Muennighoff et al., 2023). While the accuracy improved by a negligible amount, shape bias results were largely unaffected. Thus, we settled for the faster `ember-v1` models.

What happens if the description mentions multiple classes? Based on our LLM analysis, we notice that in the majority of cases (min: 79.3%, mean: 92.2%, median: 93.0%, max: 97.6%; minimum is given by `InstructBLIP Flan-T5-xl` (Dai et al., 2023)), descriptions do not refer to multiple labels. Thus, a potential bias of the embedding model is negligible for our analysis.

What happens if the description is generic? According to our LLM analysis, many generated descriptions do not refer to any object class (min: 11.4%, mean: 31.9%, median: 32.3%, max: 60.4%) - in stark contrast to VQA responses. However, we also notice that the embedding accuracy is above random choice in these cases. This suggests that the LLM may have missed objects and slightly overreported the ratio.

In the cases where the caption is indeed generic, our choice of embedding model may have biased our study. However, switching to `SFR-Embedding` (Meng et al., 2024) showed similar trends. Thus, we assume that most other SOTA embedding models would behave similarly, yet we are excited about how future embedding models will embed these cases.

Another option is to remove generic responses from the analysis. We observe that this typically increases shape bias (and accuracy) – naturally more notably in cases where the generic ratio was high (Table 9.7). *E.g.*, for the extreme case of `GPT-4V` (OpenAI, 2023), shape bias increases by 9.3% and accuracy by 40.47% (!). This preprocessing also seems to restore scaling laws to a large extent: larger models achieve higher shape bias and accuracy. One outlier to this trend is `InternVL Chat v1.2+` (Chen et al., 2024). It may be intriguing to replace the reported results in Section 9.4/Figure 9.2 with the analysis on non-generic responses, but we avoid doing so, as this would a) remove a significant portion of results; b) lead to poorly comparable results obtained on different subsets.

Table 9.7: Comparison of Image Captioning performance under ablation of responses. We compare the shape bias and accuracy for the Image Captioning task computed over all responses and only responses that an external LLM did not classify as generic.

	All responses		**Non-generic**	
Model	**Shape Bias** [%]	**Accuracy** [%]	**Shape Bias** [%]	**Accuracy** [%]
`Gemini 1.0 Pro Vision`	63.2	68.00	65.7	88.40
`GPT-4V (Preview)`	53.6	52.67	62.9	93.14
`Qwen-VL Plus`	67.9	65.50	71.9	88.56
`Qwen-VL Max`	69.7	68.50	72.1	91.52
`Qwen-VL Chat`	38.2	67.42	40.1	83.92
`InternVL Chat 1.1`	73.2	75.58	74.5	87.89
`InternVL Chat 1.2+`	61.3	82.42	62.3	88.15
`LLaVA v1.5 7B`	61.4	76.08	62.8	87.24
`LLaVA v1.5 13B`	62.7	75.58	65.1	88.24
`LLaVA-RLHF 7B`	63.0	71.83	64.7	83.80
`LLaVA-RLHF 13B`	62.3	73.25	66.3	86.08
`LLaVA-NeXT 7B`	64.0	65.08	66.9	92.48
`LLaVA-NeXT 13B`	63.5	65.25	65.3	92.95
`LLaVA-NeXT 34B`	66.2	57.50	73.6	96.39
`MoE-LLaVA-StableLM`	63.0	73.92	64.1	86.28
`MoE-LLaVA-Qwen`	63.2	75.33	64.4	88.03
`MoE-LLaVA-Phi2`	61.1	75.42	63.1	86.05
`InstructBLIP Flan-T5-xl`	67.1	81.50	68.7	89.09
`InstructBLIP Vicuna-7B`	67.7	80.67	68.4	90.27
`Emu2-Chat`	59.6	65.00	60.3	89.94
`CogAgent Chat`	67.4	60.33	70.8	97.38
`CogVLM Chat`	57.6	66.58	61.9	93.72
`UForm Gen Chat`	38.8	64.50	37.9	83.00

9.6.5 Multi-Modal Training Stages

In this last subsection, we want to ablate the potential impact of differences in multi-modal training.

`LLaVA` models are trained in two stages. During Stage 1 training, the vision encoder and LLM remain frozen, and only the parameters of the connector in between are updated. During Stage 2, the parameters of the LLM are included as well. To ablate the effect of these different training stages on our results, we repeated our experiments with a Stage 1 `LLaVA-1.5-Vicuna-7B` checkpoint in comparison to the final Stage 2 model and show the results in Table 9.8.

It is worth noting that due to the lack of proper instruction-tuning, the Stage 1 model does not follow the VQA instructions and, depending on the prompt, either generates gibberish (*e.g.*, `"The image is a collage of various items, including a bottle, a jar, a can, a spoon, a fork, a knife, [..keeps repeating..]"`) or is consistently giving a wrong prediction. However, we can assess the bias in the captioning setting. These answers are repetitive and noisy, too, but can still be discriminated by the embedding models.

We observe the following trends: instruction tuning (Stage 2) reduces the verbosity of generated descriptions (avg. tokens) and increases accuracy. The shape bias of all responses is only marginally affected (slightly decreases). The instruction-tuned model generates significantly fewer generic captions (*i.e.*, those that do not refer to any label), but it also reduces the ratio of answers referring to multiple classes and, thus, becomes more biased by forgetting one cue. Because we force the sentence embedding models to make a prediction even on generic captions, we may bias the results. To compensate for this, we repeated the analysis only on the non-generic captions (similar to Table 9.7).

For non-generic responses, the Stage 1 model performs better (higher accuracy) and achieves a higher shape bias. Taken together, this indicates that instruction tuning (at least in this specific `LLaVA` model) seems to force the model to make stronger predictions (*i.e.*, more correct predictions but also forgetting one cue) and increases texture bias.

Table 9.8: Ablation of multi-modal training stages. Comparison of `LLaVA v1.5-7B` models after Stage 1 and Stage 2 training with additional measurements on non-generic responses.

	All responses					**Non-generic**	
Model	Shape Bias [%]	Accur-acy [%]	Avg. Tokens	Single Class Ratio [%]	Generic Ratio [%]	Shape Bias [%]	Accur-acy [%]
`Stage 1`	61.8	73.25	143.5	54.1	31.4	64.5	90.32
`Stage 2`	61.4	76.08	12.1	73.8	19.2	62.8	87.24

9.7 Conclusion

We acknowledge that the broader research question – whether models can be influenced to perceive the world differently – extends beyond the scope of any single study. In our work, we focus specifically on a well-defined and understood visual bias (texture/shape bias) and investigate if and how we can steer the prediction of VLMs toward a larger texture/shape bias.

Surprisingly, we do find that through simple language prompts alone, we can affect the bias of predictions. While this form of steering was not able to fundamentally change the utilized cue for the texture/shape bias, it comes at almost no impact in accuracy and, most importantly, does not require any retraining of the model. In fact, many attempts at steering shape bias in training have yielded worse results through more expensive methods (*e.g.*, (Li et al., 2021) or even our filter frequency regularization in Chapter 5). Instead, we provide a simple and intuitive way for users to adjust the output beyond the inductive bias at runtime with minimal effort. Thus, we can indeed talk VLMs into seeing the world differently.

Fundamentally, this indicates that the LLM can significantly alter the visual tokens generated by the vision encoder, even at very low levels, such as shapes and textures. We complement the study by steering of low-/high-frequency bias, showing that language-based steerability is not limited to one specific bias. Thus, our prompt-based steering is an effective and computationally cheap method to recalibrate the perception of VLMs.

In Chapter 8 we have shown that individual biases do not seem to correlate with wide-spectrum generalization, *i.e.*, we may not hope that steering an individual bias in VLMs will significantly improve performance on vision benchmarks. However, given the cheap cost of steering, two future directions open up:

(1) The texture/shape bias correlated with performance improvements on individual datasets (Chapters 5 and 8). Thus, if the distribution shift is narrow and can be anticipated for a specific task, our steering may improve performance.

(2) Our approach bears the potential to steer multiple biases at once. This combination of steering may align the models' perception closer to an "optimal" set of biases that correlates with improved generalization.

Limitations Our study, like most studies, is not free of limitations. Even though we utilized a diverse array of VLMs, there is a possibility that different models would lead to different conclusions. We believe our results provide a fair reflection of the VLM landscape at the time of writing, but radically different VLM architectures may lead to changes in bias mechanics and consequently alter our findings.

Further, not all models have been behaving as expected in our study: Given that `GPT-4V` often achieves SOTA performance and was considered an important baseline (at the time of writing), it has surprisingly poor accuracy in both VQA and image captioning tasks compared to most other models – mostly due to refusal to answer which affected 131/1280 VQA conversations, *i.e.*, roughly 10%. This is substantially higher than the refusal rate of all other models ($< 1\%$). It is worth noting that refusal rates do not affect the shape bias measurement. `GPT-4V` is also the model with the largest amount of generic image captions (60.4%). Additionally, we acknowledge that other prompts may have led to better results. However, the result is noteworthy, as the other VLMs mostly behave well under the same prompts. Overall, prompting is a potential source of bias in our study. Different prompts could have yielded different results, and certain models might have performed differently, particularly in Visual Question Answering (VQA) tasks. While we mitigated this by utilizing simple, widely used prompts, other choices remain to be explored in future investigations. We provide a brief exploration of alternatives in Section 9.6.1.

Chapter 10

Discussion

This chapter brings the thesis to its conclusion. In Section 10.1, we summarize the key findings of the previous chapters, reflect on their impact, and discuss how this knowledge contributes to the research community. Then, in Section 10.2, we explore promising directions for future research. Finally, we provide a broader view of the topic in Section 10.3.

10.1 Summary and Conclusion

This thesis decoded mechanisms of generalization by comparing the decision rules of models with strong and weak generalization capabilities. We mostly focused on adversarially-trained models (Madry et al., 2018), which demonstrate exceptional robustness to adversarial covariate shifts by training against specific pixel-level adversarial attacks. While this form of regularization achieves impressive results, it also vastly increases training costs due to the additional backpropagations necessary to identify adversarial perturbations. Additionally, it often fails to provide robustness beyond the threat at training time. As such, identifying the resulting changes to the decision rule, specifically the "good" ones, may provide an avenue to the design of regularization techniques aiming to achieve broader generalization with cheaper means.

Due to the complexity of analyzing the decision rules of models with millions or billions of parameters, we cannot understand the encoded decision rule directly. Thus, we have attempted to understand two viewpoints and their relationship to generalization: analyzing properties of the learned parameters and examining encoded feature biases.

How can we measure and compare model representations in weight space?

Instead of analyzing and comparing individual models, we chose a population-based approach. We collect large numbers of models and study groups of models based on our central hypothesis that any patterns separating our studied dimensions will generalize better. This led to the creation of multiple model zoos tailored to our individual weight surveys (Chapter 3, Chapter 4, Chapter 6).

To compare the weight representations of convolutional neural networks (CNNs), we analyzed distribution shifts in learned convolution filters using four data-free metrics that we designed: sparsity, principal pattern variance entropy (Chapter 3), filter orthogonality (Chapter 4), and filter frequency (Chapter 5). These metrics are suitable for assessing filter quality and enable comparisons between individual models or entire model populations.

Additionally, we developed a PCA-based methodology to study distribution shifts in learned principal patterns of convolution filters (Chapter 3). Applying this to our large model zoos revealed how various factors influence filter formation and model specialization, both at global (group) and local (model-to-model) levels.

Finally, we adapted the layer criticality metric from Zhang et al. (2022) (Chapter 6) to estimate the importance of any learned layer or parameter to the decision rule. Unlike our data-free metrics, this approach requires forward passes with representative data batches but is applicable to any parameterized model, not just CNNs or discriminative classifiers.

Impact Our study sparked the development of further model zoos under more controlled conditions. For instance, Schürholt et al. (2022b) created 3,844,360 model checkpoints of various image classifiers, and Honegger et al. (2023) provided sparsified counterparts for these. Often, these works aim to generate parameters by modeling a weight-space distribution (Schürholt et al., 2022a; Schürholt, 2024).

Other works directly utilized *CNN Filter DB* (Chapter 3), like Ourghi et al. (2024), which used it to train an auto-encoder to compress filter weights.

What changes do we see between different CNN filter representations?

Our study in Chapter 3 showed that filter kernel distributions (at least those of 3 × 3 kernels) are fairly similar across model groups trained for different tasks and/or on different dataset domains unless they are severely overparameterized – in which case the convolution filters will "degenerate", usually affecting deeper layers. Overall, however, the shifts between different layers are also very low. We identify some stronger shifts between models with different network architectures. This suggests that, from a filter distribution perspective alone, any kind of pretraining should work if it is only complex enough not to cause overfitting.

When comparing individual models, we find task-specific differences where distribution shifts occur. For instance, we find that segmentation model representations increase in divergence with depth, but image classification models remain fairly static throughout. Object and face detection models show the strongest shifts in early layers. Overall, this may overhaul the common idea that the deepest layers are the most specialized and suggest that transfer learning could also benefit from the adaptation of early layers.

Impact In the medical domain, Juodelyte et al. (2023) tested our "any pretraining works" hypothesis and confirmed our findings by showing that both pre-training with a domain-specific radiology dataset and *ImageNet* converge to similar representations. Other works built on this idea, by training networks with fixed filter weights. For instance, Linse et al. (2023, 2024) find that fixing filters to our discovered principal patterns and some additional patterns matches the accuracy and sometimes even improves the generalization.

Other works refined and extended our filter analysis to spatiotemporal filters (Kobayashi & Ye, 2024) or more narrow cases like filters of depthwise-separable CNNs (Babaiee et al., 2024b). While Babaiee et al. (2024b,c) utilize clustering to detect patterns in depthwise separable convolutions, they arrive at the same conclusion that learned filters are similar across tasks, datasets (Babaiee et al., 2024a).

What changes do we see in the decision rules of robust models?

Analyzing convolutional filters in the weight space of our model zoo revealed that robust models learn more diverse, less sparse, and more orthogonal filters (Chapters 3 and 4). These differences were especially pronounced in overparameterized models, suggesting that adversarial training utilizes more of the network's capacity. This is further supported by our findings in Chapter 6, which show increased parameter utilization (across all layers) with larger adversarial budgets (ϵ). This characteristic appears unique to adversarial training, distinguishing it from other regularization methods like data augmentation.

We also consistently observed a specialized first layer in models trained with L^∞-norm adversarial training (Chapter 4), regardless of the specific adversarial regularization method, training dataset, or model architecture. These robust first layers (unlike those in L^2-norm trained models) can mitigate certain additive perturbations by applying ReLU activations to a linear combination of input channels.

While robust models are known to favor low-frequency signals, we demonstrated that this is also directly reflected in filter weight frequencies, enabling regularization without direct interaction with input samples or activations (Chapter 5).

Finally, we observed increased alignment with human perception in some aspects (Chapter 7): robust models exhibit a significantly higher shape bias and improved error consistency with human annotations. However, their average out-of-distribution (OOD) performance suffers, particularly under contrast changes or rotation, indicating that adversarial regularization overfits specific forms of generalization.

Impact Grabinski et al. (2022a)[1] used our robust vs. normal model zoo Chapter 4 to study the calibration of robust CNNs. Cianfarani et al. (2022) repeated and extended our study on a smaller collection of models via Centered Kernel Alignment (CKA) and also found that robust models increase in representational divergence to normal models with depth. Others used our findings to back arguments in their position papers (Lonnqvist et al., 2022; Mangal et al., 2023).

Which of the observed mechanisms in weight space should we replicate?

While adversarial training (in its current form) does not improve generalization and often decreases average robustness to distribution shifts as a trade-off for highly specialized robustness, we aim to incorporate beneficial aspects of the decision rules it produces selectively.

We have seen that decision rules of robust models occupy more of the network capacity. However, artificially inflating the complexity of the decision rule's complexity to match the complexity of adversarially-trained models through regularization of filter pattern diversity (Section 4.5) in normal training did not increase robustness and sometimes even diminished it. Overall, while complexity and diversity are necessary for robustness, they are not sufficient criteria.

However, replicating their preference for low-frequency features, directly observable in convolution weights (Chapter 5), offers a promising direction. This can be achieved by penalizing the learning of coefficients on a frequency-ordered basis, such as the DCT-II, rather than directly learning the weights. Our regularization across various datasets and architectures has increased *native robustness* – broad robustness without defining an adversary during training. Notably, this approach may result in weaker but more desirable biases towards low-frequency signals than the aggressive high-frequency suppression induced by adversarial training. We also observed that this regularization shifts CNN decisions away from texture and towards shape.

Impact While we observed that robust models had more diverse filters, we could not improve the robustness of CNNs via regularization of filter pattern entropy. However, Gao & Spratling (2025) managed to significantly improve robustness without adversarial training by modifying the architecture to increase filter competition.

Suresh et al. (2024) combined our findings of a special first layer and frequency attenuation and proposed architectural changes to the first layer of CNNs to improve the native robustness. Vries (2024) utilized our frequency regularization to craft shape-selective filters and studied their properties in classification tasks.

Can we improve generalization by an alignment of biases?

Robust models show improved biases in the direction of human alignment (Chapter 7), a trend also observed in large-scale models regarding biases like texture/shape bias (Dehghani et al., 2023). Several studies have proposed that mismatches between model and human visual biases explain why models

[1] Disclaimer: Paul Gavrikov is a co-author of this paper.

fail to generalize (*e.g.*, (Geirhos et al., 2019; Wang et al., 2020a; Subramanian et al., 2023)), suggesting that bias alignment might be necessary or even sufficient for generalization. This would imply that optimizing biases during training could improve generalization “for free”, *i.e.*, without requiring additional data.

However, our study (Chapter 8), examining a wider range of biases through different regularization techniques when fixing architecture and training data, found no meaningful correlation between individual biases and holistic generalization across multiple benchmarks. Instead, we sometimes even observed instances where better-aligned models generalized worse than misaligned ones, and sometimes the worst-aligned models performed best. While certain biases correlate with robustness against specific threats (*e.g.*, a shape bias improves robustness to high-frequency corruptions by inducing a low-frequency bias), we found no correlation between changes in shape bias (in any direction) and broad generalization improvement. However, we found that extreme forms of bias, such as completely ignoring high-frequency cues, deteriorate generalization.

Therefore, directly manipulating individual biases may not be a universal solution for generalization across arbitrary distribution shifts. However, it can be useful for designing robust models against known threat models. In this context, we demonstrated that biases are particularly easy to align in vision-language Models (VLMs) (Chapter 9). Unlike in discriminative models, we can simply steer VLM biases in any direction without retraining the model (or its classification head) by using natural language prompts. While bias alignment alone may not be the key to universal generalization, it represents a crucial puzzle piece in the ongoing quest for truly robust and human-like AI.

Impact Multiple works have cited our findings in the frequency domain in Chapter 8 as motivation to learn more carefully in the frequency domain, for instance through wavelets (Finder et al., 2024; Liu & Yang, 2025), or frequency priors Cao et al. (2025); Ma et al. (2024). Others directly applied our methodology – for instance, Ho et al. (2024) utilized it to benchmark `ViTs` in neural architecture search.

How do large VLMs perceive the world?

In Chapter 9, we have analyzed the visual perception of Large vision-language models (VLMs), expanding our previous work on unimodal discriminative classifiers. Through the lens of texture/shape bias, we found that the large language models (LLMs) can interact with the visual information fed from an unimodal vision encoder and significantly modify it.

For instance, most VLMs tend to be shape-biased and, in that respect, do more closely approximate human vision. However, while their tested vision encoders generally provide flexible representations capable of predicting either cue, the LLM selects one cue. It almost exclusively utilizes it for the prediction, completely ignoring the other. This complements previous works on discriminative classifiers, which have shown that the decision head in the form of the linear layer is often more biased than the feature encoder itself (Islam et al., 2021; Kirichenko et al., 2023). In our case, the decision head is the LLM.

As the LLM is conditioned by prompts, we were able to steer the visual perception of a VLM by utilizing natural language prompts. This is a unique feature of prompt-driven models, clearly separating them from discriminative classifiers, which would require retraining or finetuning to adjust their perception.

Impact Multiple works extended our shape study on VLMs to other problems where a shape bias is necessary for recognition: Hemmat et al. (2024) analyzed synthetic visual shape illusions; Wood (2024) studied the recognition of two-tone Mooney images (Mooney, 1957); Eppel (2025) studied the recognition of synthetic 3D objects under different viewpoints and changes of the object material and background. Ye et al. (2024) built the *MM-SpuBench* benchmark testing for a wider range of spurious biases in VLMs, including shapes.

Chang et al. (2024) tested if the steerability we observed for visual cues processing extends to language-specific tasks with limited success. Liang et al. (2024) repurposed parts of our methodology to study backdoor attacks in VLMs.

10.2 Future Work

Our work paves the way for several promising future research avenues, offering new directions for further exploration and development. This section outlines these opportunities and discusses their potential impact on the field.

10.2.1 Weight Space

Improvement of weight space characterizations Future work should focus on developing more comprehensive characterizations of learned representations within the weight space. While metrics like variance entropy, sparsity, orthogonality, weight frequency, and layer criticality have proven useful for comparing robust and standard models, many appear sufficient but not necessary for either robustness or generalization. A key limitation is their predominantly single-layer focus (except for layer criticality), neglecting layer connectivity. We hypothesize that metrics considering the model's topology, or at least spanning multiple layers (*e.g.*, a residual block), are crucial for a deeper understanding.

One potential avenue is exploring post-hoc linear approximations, inspired by *LIME* (Ribeiro et al., 2016), to create single-layer convolution surrogates for multi-layer blocks, allowing the application of existing metrics. However, ensuring the surrogate accurately reflects the original block's decision rule, as highlighted by Slack et al. (2020) through adversarial attacks, presents a significant challenge – surrogates themselves may overfit to capture a limited set of behavior of the underlying model.

Furthermore, it is advisable that future research prioritizes the development of *differentiable* and ideally *data-free* metrics, enabling direct training and optimization for desired representational properties across multiple layers, ultimately leading to a more holistic understanding of robust and generalizable representations.

As demonstrated in Chapter 5, regularizing the frequency content of convolutional filters can enhance native robustness. Our initial implementation involved an architectural modification, representing filter weights using a DCT-II basis instead of directly learning the weights. While this representation is theoretically equivalent in expressiveness to direct weight learning, we observed instances of decreased clean and robust accuracy even without explicit regularization. We attribute these performance drops to potential disruptions in gradient flow, possibly stemming from suboptimal initialization or the absence of intermediate normalization. Given the small size of convolutional kernels, an alternative approach is to compute the frequency decomposition on the fly just before the backward pass, eliminating the need for architectural changes. We hypothesize that such an in-loss decomposition may further improve robustness by avoiding the baseline performance degradation observed with the architectural approach.

Extension of weight studies to transformers and attention This thesis has extensively studied learned weights, particularly convolution filters, of CNNs (Chapters 3 to 6). With the rise of transformers and models like `ViTs` (Dosovitskiy et al., 2021), the focus has shifted away from CNNs, specifically for large-scale training (*cf.*, Oquab et al. (2024); Chen et al. (2024); Sun et al. (2024); Dehghani et al. (2023)). Fortunately, our existing metrics can be adapted to transformers. While most transformer parameters are one-dimensional and unsuitable for spatial analysis, we can analyze attention maps (*cf.* Equation 2.21) by relaxing the data-free condition of our metrics. By collecting activation maps for a single attention head across a batch of inputs, we could compute variance entropy to assess their diversity. Low entropy would then suggest sample-independent behavior, while high entropy would correlate with noise. This analysis would also allow us to investigate redundancy between attention heads.

While studies on Natural Language Processing (NLP) models have shown limited diversity among attention heads (Clark et al., 2019; Voita et al., 2019; Michel et al., 2019), equivalent studies for vision and multi-modal models are lacking. A key question is whether differences in attention mechanisms between robust and non-robust `ViTs` exist, which could inform regularization strategies for model training.

10.2.2 Visual Perception Biases

Methodological refinements & improvements for bias measurements Current empirical methodologies for measuring bias may be a significant source of noise in our understanding of its relationship with generalization. As discussed in Chapter 8, observed biases do not always correlate with generalization performance, even when intuitive explanations for such a connection exist. This discrepancy could stem from limitations in the measurement methods themselves and not the actual biases. For example, the original methodology for the *critical band* bias (Subramanian et al., 2023) suffers from a limited sample size, introducing considerable noise. Overall, we hypothesize that each bias quantification method operates under specific assumptions and restrictions, contributing to measurement inaccuracies. Therefore, the lack of observed empirical correlation between bias and generalization may be attributable to these noisy measurements rather than a fundamental lack of causal connection. Future research should focus on identifying these limitations and developing improved methodologies, ranging from minor adjustments to entirely new approaches (*e.g.*, the alternative approach to texture/shape bias measurement proposed by Islam et al. (2021)).

Deeper exploration of biases in multi-modal LLMs While biases in uni-modal models have been extensively studied, their investigation in multi-modal contexts, particularly in multi-modal LLMs where modality-specific tokens are fused into entangled representations, remains significantly underexplored. We have investigated the propagation of texture/shape bias (Geirhos et al., 2019) in instruction-tuned models in Chapter 9, but many other biases warrant further investigation. Beyond neuroscientific applications comparing human and artificial vision, understanding multi-modal biases could still be crucial for improving robustness and generalization in machine learning. Key open questions include: Are there optimal bias distributions that provide robustness to distribution shifts, similar to human vision? If so, which biases are critical, and what are their optimal levels? Does this optimality depend on the task, or is there a universally optimal bias configuration?

Enhancing test-time steering of model behavior As shown in Chapter 9, text prompting can influence the outputs of VLMs by manipulating inherent visual biases, offering a powerful mechanism for aligning model behavior with user preferences at inference time without retraining. However, our findings on texture/shape and low/high-frequency biases revealed limitations in the steerable range. While prompt optimization (*e.g.*, using methods like (Mirza et al., 2024)) and in-context learning (similar to (Hemmat et al., 2024)) may offer some improvements, they are ultimately constrained by the vision encoder's representations, which are not conditioned on the prompt. If the encoder produces a strongly biased representation favoring a single cue, language steering becomes ineffective.

Two potential solutions exist for future exploration beyond prompt optimization: (1) adopting early modality-fusion models like `Chameleon` (Chameleon Team, 2024), which minimize modality-specific encoding beyond tokenization; or (2) directly steering the visual encoding itself. While simple image processing can achieve the latter (as demonstrated in Section 9.5.1), we anticipate greater efficacy with more sophisticated techniques such as *visual prompting* (Bahng et al., 2022). A specifically crafted border padding (as used for adversarial robustness in (Chen et al., 2023a)) could be employed to steer the vision encoder's bias, either towards specific cues or towards equally fair and unbiased representations, which can then be further steered by language. We believe these and other techniques hold significant potential for expanding the steerability of VLM behavior and, ultimately, generalization.

10.2.3 Testing Generalization

Perhaps the most central question in AI development is whether we are creating genuine intelligence. In computer vision, this translates to asking whether models develop coherent visual concepts that generalize beyond the inner percentiles of the training distribution or simply memorize long-tail features. But how can we effectively evaluate this?

The empirical approach is to test models on large, diverse, held-out datasets representing various distribution shifts. However, what constitutes in-distribution data for one model may be out-of-distribution for another, depending on the training set. For example, recognizing an object in sketches or paintings after training only on photographs suggests some generalization, but this would be less impressive if such data were included in the training set. While we have largely avoided this issue by comparing models trained on the same data (*e.g.*, in Chapters 6 and 8), modern state-of-the-art (SOTA) models are increasingly trained on vast datasets that can vary significantly between models. This raises the question: how do we measure generalization in such cases? Without a clear understanding of a model's learned decision rule and its limitations, we are forced to rely on post-training problem detection, requiring extensive domain knowledge or large-scale validation akin to random guessing. For instance, even in a seemingly narrow task like face recognition, commercial systems from major tech companies have exhibited significant gender and ethnic biases (as noted by Buolamwini & Gebru (2018)), despite presumably passing internal tests.

Therefore, a key area for future work is developing improved methodologies for assessing generalization – perhaps, beyond classical static datasets. This includes both creating more comprehensive out-of-distribution datasets and enhancing the interpretability of learned decision rules.

10.3 A Broader View on the Topic

Deploying machine learning models in safety-critical domains where human lives are at stake, like medical diagnostics (Ma et al., 2021; Finlayson et al., 2019), and autonomous driving (Deng et al., 2020), requires an extreme level of generalization and robustness. While scaling training data can improve observed performance, relying solely on this approach presents several significant drawbacks. Beyond cost and feasibility, it hinders the democratization of AI development by creating a barrier to entry for smaller research groups and organizations lacking access to massive datasets and computational resources. Second, it remains unclear whether simply increasing data volume truly leads to genuine generalization or merely creates a closer proximity to existing benchmarks, effectively "faking" generalization by covering more of the known data distribution without necessarily understanding its underlying structure. Crucially, scaling and regularization can be orthogonal directions that should be pursued in tandem. While scale provides the raw material for learning complex decision rules, effective regularization guides the learning process, preventing overfitting and promoting generalization. However, regularization must be designed carefully as it can also be detrimental, limiting the model's capacity to discover fundamentally different and potentially superior solutions if it is too strict. Ultimately, designing regularization and achieving the necessary level of trust in safety-critical applications necessitates a stronger understanding of model mechanisms through interpretability – understanding why and how a model makes a particular prediction is crucial for validating its reliability and identifying potential failure modes, complementing the quantitative measures of robustness and generalization.

“I felt not the regret of an ending, but the foreboding of a beginning.”

— *Margaret Astrid Lindholm Ogden (Robin Hobb)*

Appendix

A.1 Model Overview for Chapter 4

We provide a table of all model pairs and their normal and robust accuracy on the corresponding test datasets in Table A.1 used in Chapter 4. We assume a (close to) 0% robust accuracy for normal models without explicitly measuring it.

Table A.1: Overview of our robust vs. normal model zoo. We show the clean and robust accuracy for both the robust (adversarially-trained) and normal models. For the datasets, we denote the usage of extra data by "extra" and synthetic data by "ddpm". The robust accuracy for normal models is always set to 0.

			Robust		**Normal**	
Paper	**Dataset**	**Architecture**	Clean Acc. [%]	Robust Acc. [%]	Clean Acc. [%]	Robust Acc. [%]
(Andriushchenko & Flammarion, 2020)	*CIFAR-10*	`PreActResNet-18`	79.84	43.93	94.51	0.0
(Carmon et al., 2022)	*CIFAR-10*	`WideResNet-28-10`	89.69	59.53	95.10	0.0
(Sehwag et al., 2020)	*CIFAR-10*	`WideResNet-28-10`	88.98	57.14	95.10	0.0
(Wang et al., 2020b)	*CIFAR-10*	`WideResNet-28-10`	87.50	56.29	95.10	0.0
(Hendrycks et al., 2019)	*CIFAR-10*	`WideResNet-28-10`	87.11	54.92	95.35	0.0
(Rice et al., 2020)	*CIFAR-10*	`WideResNet-34-20`	85.34	53.42	95.46	0.0
(Zhang et al., 2019b)	*CIFAR-10*	`WideResNet-34-10`	84.92	53.08	95.26	0.0
(Engstrom et al., 2019)	*CIFAR-10*	`ResNet-50`	87.03	49.25	94.90	0.0
(Chen et al., 2020b)	*CIFAR-10*	`ResNet-50`	86.04	51.56	86.50	0.0
(Huang et al., 2020)	*CIFAR-10*	`WideResNet-34-10`	83.48	53.34	95.26	0.0
(Pang et al., 2020b)	*CIFAR-10*	`WideResNet-34-20`	85.14	53.74	76.30	0.0
(Wong et al., 2020)	*CIFAR-10*	`PreActResNet-18`	83.34	43.21	94.25	0.0
(Ding et al., 2020)	*CIFAR-10*	`WideResNet-28-4`	84.36	41.44	94.33	0.0
(Zhang et al., 2019a)	*CIFAR-10*	`WideResNet-34-10`	87.20	44.83	95.26	0.0
(Zhang et al., 2020)	*CIFAR-10*	`WideResNet-34-10`	84.52	53.51	95.26	0.0
(Wu et al., 2020)	*CIFAR-10*	`WideResNet-28-10`	88.25	60.04	95.10	0.0
(Wu et al., 2020)	*CIFAR-10*	`WideResNet-34-10`	85.36	56.17	95.64	0.0
(Gowal et al., 2020)	*CIFAR-10*	`WideResNet-70-16`	85.29	57.20	87.91	0.0
(Gowal et al., 2020)	*CIFAR-10* + extra	`WideResNet-70-16`	91.10	65.88	87.91	0.0
(Gowal et al., 2020)	*CIFAR-10*	`WideResNet-34-20`	85.64	56.86	88.33	0.0

Continued on next page

			Robust		Normal	
Paper	Dataset	Architecture	Clean Acc. [%]	Robust Acc. [%]	Clean Acc. [%]	Robust Acc. [%]
(Gowal et al., 2020)	*CIFAR-10* + extra	`WideResNet-28-10`	89.48	62.80	88.20	0.0
(Sehwag et al., 2021)	*CIFAR-10*	`WideResNet-34-10`	85.85	59.09	95.64	0.0
(Sehwag et al., 2021)	*CIFAR-10*	`ResNet-18`	84.38	54.43	94.87	0.0
(Sitawarin et al., 2021)	*CIFAR-10*	`WideResNet-34-10`	86.84	50.72	95.26	0.0
(Chen et al., 2021a)	*CIFAR-10*	`WideResNet-34-10`	85.32	51.12	95.35	0.0
(Cui et al., 2021)	*CIFAR-10*	`WideResNet-34-20`	88.70	53.57	95.44	0.0
(Cui et al., 2021)	*CIFAR-10*	`WideResNet-34-10`	88.22	52.86	95.26	0.0
(Zhang et al., 2021b)	*CIFAR-10*	`WideResNet-28-10`	89.36	59.64	95.10	0.0
(Rebuffi et al., 2021b)	*CIFAR-10* + ddpm	`WideResNet-28-10`	87.33	60.75	88.20	0.0
(Rebuffi et al., 2021b)	*CIFAR-10* + ddpm	`WideResNet-106-16`	88.50	64.64	86.92	0.0
(Rebuffi et al., 2021b)	*CIFAR-10* + ddpm	`WideResNet-70-16`	88.54	64.25	87.91	0.0
(Rebuffi et al., 2021b)	*CIFAR-10* + extra	`WideResNet-70-16`	92.23	66.58	87.91	0.0
(Sridhar et al., 2021)	*CIFAR-10*	`WideResNet-28-10`	89.46	59.66	95.10	0.0
(Sridhar et al., 2021)	*CIFAR-10*	`WideResNet-34-15`	86.53	60.41	95.50	0.0
(Rebuffi et al., 2021b)	*CIFAR-10* + ddpm	`PreActResNet-18`	83.53	56.66	89.01	0.0
(Rade & Moosavi-Dezfooli, 2021)	*CIFAR-10* + extra	`PreActResNet-18`	89.02	57.67	89.01	0.0
(Rade & Moosavi-Dezfooli, 2021)	*CIFAR-10* + ddpm	`PreActResNet-18`	86.86	57.09	89.01	0.0
(Rade & Moosavi-Dezfooli, 2021)	*CIFAR-10* + extra	`WideResNet-34-10`	91.47	62.83	88.67	0.0
(Rade & Moosavi-Dezfooli, 2021)	*CIFAR-10* + ddpm	`WideResNet-28-10`	88.16	60.97	88.20	0.0
(Huang et al., 2022)	*CIFAR-10*	`WideResNet-34-R`	90.56	61.56	95.60	0.0
(Huang et al., 2022)	*CIFAR-10*	`WideResNet-34-R`	91.23	62.54	95.60	0.0
(Addepalli et al., 2021)	*CIFAR-10*	`ResNet-18`	80.24	51.06	94.87	0.0
(Addepalli et al., 2021)	*CIFAR-10*	`WideResNet-34-10`	85.32	58.04	95.26	0.0
(Gowal et al., 2021)	*CIFAR-10* + ddpm	`WideResNet-70-16`	88.74	66.11	87.91	0.0
(Dai et al., 2021)	*CIFAR-10*	`WideResNet-28-10-PSSiLU`	87.02	61.55	85.53	0.0
(Gowal et al., 2021)	*CIFAR-10* + ddpm	`WideResNet-28-10`	87.50	63.44	88.20	0.0
(Gowal et al., 2021)	*CIFAR-10* + ddpm	`PreActResNet-18`	87.35	58.63	89.01	0.0
(Chen & Lee, 2021)	*CIFAR-10*	`WideResNet-34-10`	85.21	56.94	95.64	0.0
(Chen & Lee, 2021)	*CIFAR-10*	`WideResNet-34-20`	86.03	57.71	95.29	0.0
(Gowal et al., 2020)	*CIFAR-100*	`WideResNet-70-16`	60.86	30.03	60.56	0.0
(Gowal et al., 2020)	*CIFAR-100* + extra	`WideResNet-70-16`	69.15	36.88	60.56	0.0
(Cui et al., 2021)	*CIFAR-100*	`WideResNet-34-20`	62.55	30.20	80.46	0.0
(Cui et al., 2021)	*CIFAR-100*	`WideResNet-34-10`	70.25	27.16	79.11	0.0
(Cui et al., 2021)	*CIFAR-100*	`WideResNet-34-10`	60.64	29.33	79.11	0.0

Continued on next page

Paper	Dataset	Architecture	Robust		Normal	
			Clean Acc. [%]	Robust Acc. [%]	Clean Acc. [%]	Robust Acc. [%]
(Chen et al., 2021a)	*CIFAR-100*	`WideResNet-34-10`	62.15	26.94	78.75	0.0
(Wu et al., 2020)	*CIFAR-100*	`WideResNet-34-10`	60.38	28.86	78.79	0.0
(Sitawarin et al., 2021)	*CIFAR-100*	`WideResNet-34-10`	62.82	24.57	79.11	0.0
(Hendrycks et al., 2019)	*CIFAR-100*	`WideResNet-28-10`	59.23	28.42	79.16	0.0
(Rice et al., 2020)	*CIFAR-100*	`PreActResNet-18`	53.83	18.95	76.18	0.0
(Rebuffi et al., 2021b)	*CIFAR-100* + ddpm	`WideResNet-70-16`	63.56	34.64	60.56	0.0
(Rebuffi et al., 2021b)	*CIFAR-100* + ddpm	`WideResNet-28-10`	62.41	32.06	61.46	0.0
(Rebuffi et al., 2021b)	*CIFAR-100*	`PreActResNet-18`	56.87	28.50	63.45	0.0
(Rade & Moosavi-Dezfooli, 2021)	*CIFAR-100* + ddpm	`PreActResNet-18`	61.50	28.88	63.45	0.0
(Addepalli et al., 2021)	*CIFAR-100*	`PreActResNet-18`	62.02	27.14	76.66	0.0
(Addepalli et al., 2021)	*CIFAR-100*	`WideResNet-34-10`	65.73	30.35	79.11	0.0
(Chen & Lee, 2021)	*CIFAR-100*	`WideResNet-34-10`	64.07	30.59	79.11	0.0
(Wong et al., 2020)	*ImageNet*	`ResNet-50`	55.62	26.24	76.13	0.0
(Engstrom et al., 2019)	*ImageNet*	`ResNet-50`	62.56	29.22	76.13	0.0
(Salman et al., 2020)	*ImageNet*	`ResNet-50`	64.02	34.96	76.13	0.0
(Salman et al., 2020)	*ImageNet*	`ResNet-18`	52.92	25.32	69.76	0.0
(Salman et al., 2020)	*ImageNet*	`WideResNet-50-2`	68.46	38.14	78.47	0.0

A.2 Model Overview for Chapter 6

We provide a table of all models and their *ImageNet* validation accuracy in Table A.2 as used in Chapter 6.

Table A.2: An overview of the utilized models (training/regularization methods) in our study.

	Model	ImageNet Accuracy [%]
	Original Baseline (He et al., 2016)	76.15
Adversarial Training (AT)	PGD-AT (ϵ=0) (Salman et al., 2020; Madry et al., 2018)	75.81
	PGD-AT (L^2, ϵ=0.01) (Salman et al., 2020; Madry et al., 2018)	75.67
	PGD-AT (L^2, ϵ=0.03) (Salman et al., 2020; Madry et al., 2018)	75.77
	PGD-AT (L^2, ϵ=0.05) (Salman et al., 2020; Madry et al., 2018)	75.58
	PGD-AT (L^2, ϵ=0.1) (Salman et al., 2020; Madry et al., 2018)	74.79
	PGD-AT (L^2, ϵ=0.25) (Salman et al., 2020; Madry et al., 2018)	74.14
	PGD-AT (L^2, ϵ=0.5) (Salman et al., 2020; Madry et al., 2018)	73.17
	PGD-AT (L^2, ϵ=1) (Salman et al., 2020; Madry et al., 2018)	70.42
	PGD-AT (L^2, ϵ=3) (Salman et al., 2020; Madry et al., 2018)	62.83
	PGD-AT (L^2, ϵ=5) (Salman et al., 2020; Madry et al., 2018)	56.14
	PGD-AT (L^∞, ϵ=0.5/255) (Salman et al., 2020; Madry et al., 2018)	73.74
	PGD-AT (L^∞, ϵ=1.0/255) (Salman et al., 2020; Madry et al., 2018)	72.04
	PGD-AT (L^∞, ϵ=2.0/255) (Salman et al., 2020; Madry et al., 2018)	69.09
	PGD-AT (L^∞, ϵ=4.0/255) (Salman et al., 2020; Madry et al., 2018)	63.87
	PGD-AT (L^∞, ϵ=8.0/255) (Salman et al., 2020; Madry et al., 2018)	54.53
Augmentations	AutoAugment (270Ep) (Cubuk et al., 2019)	77.50
	FastAutoAugment (270Ep) (Lim et al., 2019)	77.65
	RandAugment (270Ep) (Cubuk et al., 2020)	77.64
	AugMix (180Ep) (Hendrycks et al., 2020)	77.53
	DeepAugment (Hendrycks et al., 2021a)	76.65
	DeepAugment+AugMix (Hendrycks et al., 2021a)	75.80
	Diffusion-like Noise (Jaini et al., 2024)	67.22
	NoisyMix (Erichson et al., 2022)	77.05
	OpticsAugment (Müller et al., 2023)	74.22
	PRIME (Modas et al., 2022)	76.91
	PixMix (180Ep) (Hendrycks et al., 2022)	78.09
	PixMix (90Ep) (Hendrycks et al., 2022)	77.36
	ShapeNet (SIN) (Geirhos et al., 2019)	60.18
	ShapeNet (SIN+IN) (Geirhos et al., 2019)	74.59
	ShapeNet (SIN+IN → IN) (Geirhos et al., 2019)	76.72
	Texture/Shape-debiased Augmentation (Li et al., 2021)	76.89
	Texture/Shape-Shape Bias Augmentation (Li et al., 2021)	76.21
	Texture/Shape-Texture Bias Augmentation (Li et al., 2021)	75.27
SSL	DINOv1 (Caron et al., 2021)	75.28
	MoCo v3 (1000Ep) (Chen et al., 2021b)	74.60
	MoCo v3 (100Ep) (Chen et al., 2021b)	68.91
	MoCo v3 (300Ep) (Chen et al., 2021b)	72.80
	SimCLRv2 (Chen et al., 2020a)	74.90
	SwAV (Caron et al., 2020)	75.31
Improved Training	timm A1 (Wightman, 2019; Wightman et al., 2021)	80.10
	timm A1h (Wightman, 2019; Wightman et al., 2021)	80.10
	timm A2 (Wightman, 2019; Wightman et al., 2021)	79.80
	timm A3 (Wightman, 2019; Wightman et al., 2021)	77.55
	timm B1k (Wightman, 2019; Wightman et al., 2021)	79.16
	timm B2k (Wightman, 2019; Wightman et al., 2021)	79.27
	timm C1 (Wightman, 2019; Wightman et al., 2021)	79.76
	timm C2 (Wightman, 2019; Wightman et al., 2021)	79.92
	timm D (Wightman, 2019; Wightman et al., 2021)	79.89
	TorchVision 2 (Vryniotis, 2023; Paszke et al., 2019)	80.34

A.3 Model Overview for Chapter 8

We provide a table of all models and quantification of their biases in Table A.4, as well as their performance on each benchmark in Table A.3 as used in Chapter 8. These tables also contain the full symbol legend.

Table A.3: Overview of performance (accuracy) of each individual model on our generalization benchmarks.

			Top-1 Test Accuracy [%] (↑)									
			In Distribution			*Robustness*			*Concepts*			*Adv.*
		Model	*IN*	*IN-ReaL*	*IN-V2*	*IN-A*	*IN-C*	*IN-*$\overline{C}$	*IN-R*	*IN-S*	*SIN*	PGD
	•	Original Baseline (He et al., 2016)	76.15	86.50	63.14	0.03	41.12	39.70	36.16	24.09	37.12	18.39
Adversarial Training (AT)	•	PGD-AT (L^2, ϵ=0) (Salman et al., 2020; Madry et al., 2018)	75.81	88.65	63.70	1.85	40.90	39.48	35.76	23.50	34.38	18.88
	•	PGD-AT (L^2, ϵ=0.01) (Salman et al., 2020; Madry et al., 2018)	75.67	84.97	63.64	1.69	42.13	39.78	36.85	24.22	38.50	30.56
	•	PGD-AT (L^2, ϵ=0.03) (Salman et al., 2020; Madry et al., 2018)	75.77	87.42	63.33	1.92	42.25	39.72	36.71	24.60	38.50	40.05
	•	PGD-AT (L^2, ϵ=0.05) (Salman et al., 2020; Madry et al., 2018)	75.58	84.66	62.93	1.79	41.66	40.18	37.28	24.69	40.00	46.86
	•	PGD-AT (L^2, ϵ=0.1) (Salman et al., 2020; Madry et al., 2018)	74.79	86.20	62.44	1.88	41.91	39.35	37.61	24.70	39.25	53.76
	•	PGD-AT (L^2, ϵ=0.25) (Salman et al., 2020; Madry et al., 2018)	74.14	85.28	61.65	1.96	42.02	39.58	38.23	25.31	40.88	61.23
	•	PGD-AT (L^2, ϵ=0.5) (Salman et al., 2020; Madry et al., 2018)	73.17	86.50	59.97	2.08	40.82	39.23	38.94	24.21	43.88	64.30
	•	PGD-AT (L^2, ϵ=1) (Salman et al., 2020; Madry et al., 2018)	70.42	84.36	56.95	2.09	38.79	37.90	38.95	23.68	44.12	64.37
	●	PGD-AT (L^2, ϵ=3) (Salman et al., 2020; Madry et al., 2018)	62.83	75.77	48.91	1.87	34.60	34.83	36.99	20.93	41.75	59.47
	●	PGD-AT (L^2, ϵ=5) (Salman et al., 2020; Madry et al., 2018)	56.14	74.54	42.49	1.77	30.65	31.15	33.09	17.24	39.00	53.63
	▪	PGD-AT (L^∞, ϵ=0.5/255) (Salman et al., 2020; Madry et al., 2018)	73.74	84.36	61.38	2.29	40.11	40.04	39.39	24.68	43.88	65.11
	▪	PGD-AT (L^∞, ϵ=1/255) (Salman et al., 2020; Madry et al., 2018)	72.04	83.44	59.21	2.20	38.82	39.72	40.96	24.51	44.00	66.39
	▪	PGD-AT (L^∞, ϵ=2/255) (Salman et al., 2020; Madry et al., 2018)	69.09	82.52	56.15	2.39	37.49	38.85	39.33	23.10	45.75	65.25
	■	PGD-AT (L^∞, ϵ=4/255) (Salman et al., 2020; Madry et al., 2018)	63.87	78.83	51.31	2.29	33.71	36.56	38.92	21.87	43.25	61.22
	■	PGD-AT (L^∞, ϵ=8/255) (Salman et al., 2020; Madry et al., 2018)	54.53	71.78	41.86	2.11	28.78	31.91	34.84	18.57	40.00	52.57
Augmentations	▲	AugMix (180ep) (Hendrycks et al., 2020)	77.53	88.96	65.42	3.65	50.77	46.16	41.03	28.49	45.50	30.96
	◀	DeepAugment (Hendrycks et al., 2021a)	76.65	86.81	65.20	3.40	54.40	48.39	42.25	29.50	49.12	32.51
	▶	DeepAugment+AugMix (Hendrycks et al., 2021a)	75.80	86.20	63.65	3.85	59.53	51.34	46.79	32.62	57.50	40.40
	•	Noise Training (clean eval) (Jaini et al., 2024)	67.22	83.44	54.67	2.43	44.40	39.48	36.64	19.99	47.12	48.27
	×	NoisyMix (Erichson et al., 2022)	77.05	89.57	64.28	3.32	54.23	50.62	45.77	31.18	49.38	50.70
	●	OpticsAugment (Müller et al., 2023)	74.22	86.50	62.03	1.73	42.90	40.39	37.50	24.69	43.88	16.08
	♦	PRIME (Modas et al., 2022)	76.91	87.12	64.34	2.16	55.27	49.00	42.20	29.83	46.62	30.82
	⬢	PixMix (180ep) (Hendrycks et al., 2022)	78.09	88.65	65.89	6.25	52.99	59.51	40.31	29.21	40.25	23.02
	⬟	PixMix (90ep) (Hendrycks et al., 2022)	77.36	89.88	65.20	4.11	51.87	57.76	39.92	28.57	45.00	22.28
	⬣	Shape Bias Augmentation (Li et al., 2021)	76.21	87.42	64.20	3.03	47.60	44.46	40.64	27.92	64.50	25.18
	⅄	Texture Bias Augmentation (Li et al., 2021)	75.27	86.81	63.18	2.25	41.82	40.26	36.76	24.28	35.50	16.83
	Y	Texture/Shape Debiased Augmentation (Li et al., 2021)	76.89	86.20	65.04	3.39	48.28	45.47	40.77	28.42	56.00	25.99
SSL	⬣	DINO V1 (Caron et al., 2021)	75.28	85.28	62.70	5.15	39.61	35.88	30.17	18.75	30.63	13.26
	×	MoCo V3 (1000ep) (Chen et al., 2021b)	74.60	87.42	62.01	4.07	43.53	40.76	37.05	25.51	35.50	27.79
	Y	MoCo V3 (100ep) (Chen et al., 2021b)	68.91	82.52	56.28	2.43	37.75	36.62	31.71	20.48	36.75	24.61
	⅄	MoCo V3 (300ep) (Chen et al., 2021b)	72.80	84.97	60.74	3.27	41.97	39.00	35.41	24.00	36.75	27.57
	•	SimCLRv2 (Chen et al., 2020a)	74.90	85.58	61.24	4.65	44.32	40.73	35.16	23.55	43.88	14.57
	●	SwAV (Caron et al., 2020)	75.31	87.73	62.15	5.49	41.48	37.63	30.24	18.94	30.38	14.75
Freezing	•	Frozen Random Filters (Gavrikov & Keuper, 2023b)	74.76	87.12	62.47	2.52	45.22	40.98	37.52	25.36	40.62	16.18
Styl.	⬢	ShapeNet: SIN Training (Geirhos et al., 2019)	60.18	73.31	48.61	2.39	39.76	36.76	40.17	30.09	90.88	12.33
	•	ShapeNet: SIN+IN Training (Geirhos et al., 2019)	74.59	87.12	62.43	1.91	46.91	43.33	41.55	29.70	91.00	22.47
	●	ShapeNet: SIN+IN Training + FT (Geirhos et al., 2019)	76.72	88.34	64.65	2.23	43.55	41.86	38.93	26.92	45.00	21.23
Training Recipes	⬢	timm (A1) (Wightman, 2019; Wightman et al., 2021)	80.10	88.65	68.73	11.03	50.93	49.01	40.60	29.22	36.88	27.74
	●	timm (A1H) (Wightman, 2019; Wightman et al., 2021)	80.10	89.26	68.47	15.21	49.36	48.57	40.99	29.64	39.00	36.11
	⬟	timm (A2) (Wightman, 2019; Wightman et al., 2021)	79.80	86.20	67.29	7.36	48.98	47.83	38.39	27.27	38.38	27.03
	•	timm (A3) (Wightman, 2019; Wightman et al., 2021)	77.55	86.81	65.04	6.35	41.03	43.20	35.93	24.61	34.75	23.32
	×	timm (B1K) (Wightman, 2019; Wightman et al., 2021)	79.16	88.34	67.41	8.51	51.64	50.30	43.04	31.22	42.88	33.40
	⅄	timm (B2K) (Wightman, 2019; Wightman et al., 2021)	79.27	87.42	67.79	8.64	52.25	50.05	42.44	30.40	40.75	32.82
	♦	timm (C1) (Wightman, 2019; Wightman et al., 2021)	79.76	89.88	68.54	10.07	50.60	49.40	41.54	30.29	37.12	33.72
	Y	timm (C2) (Wightman, 2019; Wightman et al., 2021)	79.92	90.49	68.80	11.49	51.62	50.92	40.73	29.85	37.12	30.38
	⬣	timm (D) (Wightman, 2019; Wightman et al., 2021)	79.89	89.26	68.73	9.76	51.26	49.53	40.61	29.85	36.00	29.62
	▲	torchvision (V2) (Vryniotis, 2023; Paszke et al., 2019)	80.34	90.18	69.57	16.73	50.02	49.67	41.62	28.44	38.38	39.90

Table A.4: Overview of all bias measurements.

			Bias								
						Critical Band					
			Shape	Frequency		*(IN-1k non-normalized)*			*(IN-1k normalized)*		
		Model	Bias	LF	HF	C-BW	C-CF	C-PNS	C-BW	C-CF	C-PNS
	●	Original Baseline (He et al., 2016)	0.21	0.63	0.01	11295.35	3681.95	1.0	5.67	54.68	1.00
Adversarial Training (AT)	●	PGD-AT (L^2, ϵ=0) (Salman et al., 2020; Madry et al., 2018)	0.21	0.62	0.01	11295.35	3681.95	1.0	5.67	54.68	1.00
	●	PGD-AT (L^2, ϵ=0.01) (Salman et al., 2020; Madry et al., 2018)	0.22	0.64	0.01	11295.35	3681.95	1.0	5.67	54.68	1.00
	●	PGD-AT (L^2, ϵ=0.03) (Salman et al., 2020; Madry et al., 2018)	0.24	0.67	0.00	11295.35	3681.95	1.0	5.67	54.68	1.00
	●	PGD-AT (L^2, ϵ=0.05) (Salman et al., 2020; Madry et al., 2018)	0.24	0.67	0.00	11295.35	3681.95	1.0	5.91	41.85	1.00
	●	PGD-AT (L^2, ϵ=0.1) (Salman et al., 2020; Madry et al., 2018)	0.28	0.69	0.00	11295.35	3681.95	1.0	6.45	40.65	1.00
	●	PGD-AT (L^2, ϵ=0.25) (Salman et al., 2020; Madry et al., 2018)	0.34	0.72	0.00	11295.35	3681.95	1.0	7.86	58.20	1.00
	●	PGD-AT (L^2, ϵ=0.5) (Salman et al., 2020; Madry et al., 2018)	0.41	0.73	0.00	11295.35	3681.95	1.0	7.86	58.20	1.00
	●	PGD-AT (L^2, ϵ=1) (Salman et al., 2020; Madry et al., 2018)	0.48	0.75	0.00	11295.35	3681.95	1.0	7.86	58.20	1.00
	●	PGD-AT (L^2, ϵ=3) (Salman et al., 2020; Madry et al., 2018)	0.65	0.76	0.00	11295.35	3681.95	1.0	7.47	32.63	0.60
	●	PGD-AT (L^2, ϵ=5) (Salman et al., 2020; Madry et al., 2018)	0.69	0.78	0.00	11295.35	3681.95	1.0	7.92	45.71	0.55
	■	PGD-AT (L^∞, ϵ=0.5/255) (Salman et al., 2020; Madry et al., 2018)	0.37	0.73	0.00	11295.35	3681.95	1.0	7.86	58.20	1.00
	■	PGD-AT (L^∞, ϵ=1/255) (Salman et al., 2020; Madry et al., 2018)	0.45	0.74	0.00	11295.35	3681.95	1.0	7.37	39.19	0.73
	■	PGD-AT (L^∞, ϵ=2/255) (Salman et al., 2020; Madry et al., 2018)	0.54	0.75	0.00	11295.35	3681.95	1.0	7.92	45.71	0.55
	■	PGD-AT (L^∞, ϵ=4/255) (Salman et al., 2020; Madry et al., 2018)	0.62	0.75	0.00	11295.35	3681.95	1.0	10.46	114.50	0.52
	■	PGD-AT (L^∞, ϵ=8/255) (Salman et al., 2020; Madry et al., 2018)	0.72	0.78	0.00	11295.35	3681.95	1.0	7.47	32.63	0.60
Augmentations	▲	AugMix (180ep) (Hendrycks et al., 2020)	0.30	0.74	0.02	9.95	34.70	1.0	4.74	38.85	1.00
	◀	DeepAugment (Hendrycks et al., 2021a)	0.39	0.77	0.06	11295.35	3681.95	1.0	5.51	56.31	0.72
	▶	DeepAugment+AugMix (Hendrycks et al., 2021a)	0.52	0.84	0.09	9.95	34.70	1.0	4.67	38.28	0.63
	●	Noise Training (clean eval) (Jaini et al., 2024)	0.51	0.79	0.01	11295.35	3681.95	1.0	6.35	33.58	0.93
	×	NoisyMix (Erichson et al., 2022)	0.32	0.75	0.01	9.95	34.70	1.0	4.74	38.85	1.00
	●	OpticsAugment (Müller et al., 2023)	0.24	0.63	0.01	11295.35	3681.95	1.0	4.45	61.21	1.00
	♦	PRIME (Modas et al., 2022)	0.32	0.71	0.13	7.34	42.82	1.0	4.74	38.85	1.00
	⬢	PixMix (180ep) (Hendrycks et al., 2022)	0.26	0.68	0.03	5.91	41.85	1.0	4.74	38.85	1.00
	⬟	PixMix (90ep) (Hendrycks et al., 2022)	0.23	0.67	0.03	6.45	40.65	1.0	4.74	38.85	1.00
	⬣	Shape Bias Augmentation (Li et al., 2021)	0.28	0.66	0.01	11295.35	3681.95	1.0	4.74	38.85	1.00
	⅄	Texture Bias Augmentation (Li et al., 2021)	0.20	0.64	0.01	11295.35	3681.95	1.0	5.67	54.68	1.00
	Y	Texture/Shape Debiased Augmentation (Li et al., 2021)	0.26	0.67	0.01	11295.35	3681.95	1.0	5.67	54.68	1.00
SSL	⬣	DINO V1 (Caron et al., 2021)	0.18	0.44	0.01	7.34	42.82	1.0	4.74	38.85	1.00
	×	MoCo V3 (1000ep) (Chen et al., 2021b)	0.33	0.56	0.01	7.34	42.82	1.0	4.74	38.85	1.00
	Y	MoCo V3 (100ep) (Chen et al., 2021b)	0.30	0.55	0.01	8.22	33.66	1.0	4.74	38.85	1.00
	⅄	MoCo V3 (300ep) (Chen et al., 2021b)	0.31	0.55	0.01	7.34	42.82	1.0	4.74	38.85	1.00
	●	SimCLRv2 (Chen et al., 2020a)	0.23	0.55	0.01	11295.35	3681.95	1.0	4.74	38.85	1.00
	●	SwAV (Caron et al., 2020)	0.18	0.43	0.01	7.34	42.82	1.0	4.74	38.85	1.00
Freezing	●	Frozen Random Filters (Gavrikov & Keuper, 2023b)	0.31	0.68	0.01	11295.35	3681.95	1.0	4.74	38.85	1.00
Styl.	⬢	ShapeNet: SIN Training (Geirhos et al., 2019)	0.81	0.56	0.04	11295.35	3681.95	1.0	4.42	45.57	0.67
	●	ShapeNet: SIN+IN Training (Geirhos et al., 2019)	0.35	0.63	0.01	11295.35	3681.95	1.0	4.33	39.67	1.00
	●	ShapeNet: SIN+IN Training + FT (Geirhos et al., 2019)	0.20	0.64	0.01	11295.35	3681.95	1.0	5.67	54.68	1.00
Training Recipes	⯃	timm (A1) (Wightman, 2019; Wightman et al., 2021)	0.21	0.63	0.02	5.67	54.68	1.0	3.96	42.09	1.00
	●	timm (A1H) (Wightman, 2019; Wightman et al., 2021)	0.17	0.61	0.02	5.67	54.68	1.0	3.71	46.56	1.00
	⬟	timm (A2) (Wightman, 2019; Wightman et al., 2021)	0.16	0.62	0.01	5.67	54.68	1.0	4.33	39.67	1.00
	●	timm (A3) (Wightman, 2019; Wightman et al., 2021)	0.13	0.58	0.01	9.95	34.70	1.0	4.67	54.89	1.00
	×	timm (B1K) (Wightman, 2019; Wightman et al., 2021)	0.19	0.64	0.02	5.67	54.68	1.0	4.14	47.02	0.91
	⅄	timm (B2K) (Wightman, 2019; Wightman et al., 2021)	0.18	0.65	0.02	5.67	54.68	1.0	4.14	47.02	0.91
	♦	timm (C1) (Wightman, 2019; Wightman et al., 2021)	0.18	0.64	0.02	5.67	54.68	1.0	3.96	42.09	1.00
	Y	timm (C2) (Wightman, 2019; Wightman et al., 2021)	0.18	0.62	0.02	5.67	54.68	1.0	3.92	53.43	0.86
	⬣	timm (D) (Wightman, 2019; Wightman et al., 2021)	0.17	0.63	0.01	5.67	54.68	1.0	3.96	42.09	1.00
	▲	torchvision (V2) (Vryniotis, 2023; Paszke et al., 2019)	0.17	0.66	0.01	5.67	54.68	1.0	3.92	53.43	0.86

A.4 Model Overview for Chapter 9

We provide a brief description of utilized VLMs in Table A.5 and show detailed results for the texture/shape bias in Table A.6 that were used in Chapter 9. Specifically, Table A.6 shows the shape bias and accuracy for the VQA and Image Captioning task (see Figure 9.2 in the main chapter for a visualization). For the open-ended Image Captioning responses, we additionally provide evaluations through an LLM. These include the number of generated tokens (to measure how effective our `"Keep [...] short."` instruction is), the ratio of responses where exactly one class was detected (*single class ratio*), and the ratio of responses that do not refer to any description (*generic ratio*).

Table A.5: An overview of the utilized VLMs.

Qwen-VL-Chat (Bai et al., 2023b)	Adds vision capabilities to Qwen-7B (Bai et al., 2023a). We set a repetition penalty of 1.2 for this model.
Qwen-VL Plus/Max (Qwen Team, 2024)	AliBaba's proprietary larger variants of Qwen-VL-Chat. Access only via API.
CogAgent (Hong et al., 2023)	A special model for interaction with graphical user interfaces (GUIs) at high-resolution.
CogVLM (Wang et al., 2023c)	Adds "trainable visual expert module" in LLM layers to combine vision and language.
Emu2 (Sun et al., 2023a)	The 37B model claims "strong multi-modal in-context learning abilities".
InstructBLIP (Dai et al., 2023)	Connects frozen vision encoders and LLMs through a trainable Q-Former. Uses Vicuna or FLAN-T5 as LLMs.
LLaVA v1.5 (Liu et al., 2023b)	Improvements of LLaVA with modifications on the image encoder, the projector, and task-specific data. Uses Vicuna-7/13B as LLM.
LLaVA-NeXT (Liu et al., 2024)	Successor of LLaVA v1.5 supporting higher resolutions through patching, and using better SFT training data for training, claiming *"improved reasoning, OCR, and world knowledge"* (Liu et al., 2024). The 34B version switches from Vicuna-7/13B to Nous Hermes 2 Yi 34B.
MoE-LLaVA v1.5 (Lin et al., 2024)	Variants of LLaVA v1.5 employing 4 sparsely activated Mixture-of-Experts (MoE), and smaller LLMs (Qwen, Phi-2, StableLM).
LLaVA-RLHF (Sun et al., 2023c)	Variants of LLaVA v1.5 aligned with Factually Augmented RLHF (Fact-RLHF) (Sun et al., 2023c).
UForm-Gen Chat (Kim et al., 2023)	A small (1.5B) model for VQA and image captioning finetuned for multi-modal chats.
Gemini 1.0 Pro Vision (Gemini Team, 2023)	Google's proprietary multi-modal model based on the Gemini Pro LLM. Access only via API.
InternVL Chat 1.1/1.2+ (Chen et al., 2024)	An open-source effort to provide an alternative to ViT-22B (Dehghani et al., 2023). V1.1 is based on a 6B ViT and Vicuna-13B, V1.2+ uses Nous Hermes 2 Yi 34B as LLM including additional SFT on 10x more data.
GPT-4V (Preview) (OpenAI, 2023)	OpenAI's proprietary multi-modal model based on the GPT-4 LLM. Access only via API. Often considered to be the most powerful model.

Table A.6: Detailed results of our texture/shape study. We measure shape bias and respective accuracy on the *texture/shape cue-conflict* dataset for various VLMs in VQA classification or image description tasks. For the image description task, we additionally provide the average number of tokens generated by `Vicuna`'s tokenizer and the ratio of responses that only contain a single class or are generic (do not mention any class) as judged by a separate LLM. "-" indicates models that did not follow instructions on VQA and could, thus, not be evaluated.

	VQA		**Image Captioning**				
Model	Shape Bias [%]	Accuracy [%]	Shape Bias [%]	Accuracy [%]	Avg. Tokens	Single Class Ratio [%]	Generic Ratio [%]
`Gemini 1.0 Pro Vision` (Gemini Team, 2023)	64.1	82.33	63.2	68.00	18.9	63.0	32.3
`GPT-4V (Preview)` (OpenAI, 2023)	47.9	69.75	53.6	52.67	44.8	37.2	60.4
`Qwen-VL Plus` (Qwen Team, 2024)	64.8	82.92	67.9	65.50	21.9	59.2	36.0
`Qwen-VL Max` (Qwen Team, 2024)	62.4	85.50	69.7	68.50	151.9	52.1	41.0
`Qwen-VL Chat` (Bai et al., 2023b)	-	-	38.2	67.42	27.3	59.1	33.2
`InternVL Chat 1.1` (Chen et al., 2024)	68.3	89.33	73.2	75.58	16.9	74.9	19.4
`InternVL Chat 1.2+` (Chen et al., 2024)	61.1	90.83	61.3	82.42	15.8	80.4	11.4
`LLaVA v1.5 7B` (Liu et al., 2023b)	61.4	80.75	61.4	76.08	12.1	73.8	19.2
`LLaVA v1.5 13B` (Liu et al., 2023b)	64.1	80.25	62.7	75.58	28.9	65.8	23.8
`LLaVA-RLHF 7B` (Sun et al., 2023c)	61.7	68.08	63.0	71.83	47.9	65.1	24.7
`LLaVA-RLHF 13B` (Sun et al., 2023c)	63.4	80.42	62.3	73.25	38.3	64.7	27.7
`LLaVA-NeXT 7B` (Liu et al., 2024)	59.2	82.58	64.0	65.08	20.2	55.5	39.5
`LLaVA-NeXT 13B` (Liu et al., 2024)	57.2	83.42	63.5	65.25	48.8	52.6	40.9
`LLaVA-NeXT 34B` (Liu et al., 2024)	56.0	73.83	66.2	57.50	93.4	36.2	59.1
`MoE-LLaVA-StableLM` (Lin et al., 2024)	59.1	80.08	63.0	73.92	24.1	67.4	21.6
`MoE-LLaVA-Qwen` (Lin et al., 2024)	62.9	59.50	63.2	75.33	13.3	69.4	20.7
`MoE-LLaVA-Phi2` (Lin et al., 2024)	59.6	82.33	61.1	75.42	34.9	67.0	18.6
`InstructBLIP Flan-T5-xl` (Dai et al., 2023)	68.2	79.58	67.1	81.50	116.7	57.0	22.3
`InstructBLIP Vicuna-7B` (Dai et al., 2023)	73.8	72.25	67.7	80.67	94.0	60.9	28.0
`Emu2-Chat` (Sun et al., 2023a)	52.9	75.08	59.6	65.00	13.6	63.0	34.0
`CogAgent Chat` (Hong et al., 2023)	-	-	67.4	60.33	40.1	49.6	47.7
`CogVLM Chat` (Wang et al., 2023c)	-	-	57.6	66.58	35.8	53.2	40.1
`UForm Gen Chat` (Kim et al., 2023)	-	-	38.8	64.50	30.2	59.3	33.0

List of Acronyms

AI	Artificial Intelligence
AA	AutoAttack
AT	Adversarial Training
CC	Common Corruptions
CNN	Convolutional Neural Network
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
FGSM	Fast Gradient Sign Method
FFT	Fast Fourier Transform
GAN	Generative Adversarial Networks
HF	High-Frequency
ID	In Distribution
i.i.d.	Independent and identically distributed
IN (Dataset)	ImageNet
KL	Kullback-Leibler (Divergence)
KDE	Kernel Density Estimation
LF	Low-Frequency
LLM	Large Language Model
ML	Machine Learning
MLP	Multi-Layer Perceptron
NLP	Natural Language Processing
OOD	Out of Distribution
PCA	Principal Component Analysis
PGD	Projected Gradient Descent
ReLU	Rectified Linear Unit
RGB	Red Green Blue
SGD	Stochastic Gradient Descent
SIN (Dataset)	Stylized ImageNet
SOTA	State-of-the-art
SSL	Self-Supervised Learning
SVD	Singular Value Decomposition
ViT	Vision Transformer
VLM	Vision-Language Model
VQA	Visual Question Answering

List of Figures

List of Tables

Bibliography

Abello, A. A., Hirata, R., and Wang, Z. Dissecting the High-Frequency Bias in Convolutional Neural Networks. In *Proceedings of the Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2021. 78, 86, 110, 115

Abnar, S. and Zuidema, W. Quantifying Attention Flow in Transformers. In *Proceedings of the Annual Meeting of the Association for Computational Linguistic (ACL)*, 2020. 38

Addepalli, S., Jain, S., Sriramanan, G., Khare, S., and Radhakrishnan, V. B. Towards Achieving Adversarial Robustness Beyond Perceptual Limits. In *ICML 2021 Workshop on Adversarial Machine Learning*, 2021. 67, 71, 162, 163

Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., and Kim, B. Sanity Checks for Saliency Maps. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. 38

Ahmed, N., Natarajan, T., and Rao, K. Discrete Cosine Transform. *IEEE Transactions on Computers*, 1974. 79

Akhtar, N. and Mian, A. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, 2018. 45, 65

Alain, G. and Bengio, Y. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2018. 47

Alayrac, J.-B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., Ring, R., Rutherford, E., Cabi, S., Han, T., Gong, Z., Samangooei, S., Monteiro, M., Menick, J. L., Borgeaud, S., Brock, A., Nematzadeh, A., Sharifzadeh, S., Bińkowski, M. a., Barreira, R., Vinyals, O., Zisserman, A., and Simonyan, K. Flamingo: a Visual Language Model for Few-Shot Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 131, 132, 138

Alhamoud, K., Hammoud, H. A. A. K., Alfarra, M., and Ghanem, B. Generalizability of Adversarial Robustness Under Distribution Shifts. *Transactions on Machine Learning Research (TMLR)*, 2023. 36

Ali, A., Touvron, H., Caron, M., Bojanowski, P., Douze, M., Joulin, A., Laptev, I., Neverova, N., Synnaeve, G., Verbeek, J., and Jegou, H. XCiT: Cross-Covariance Image Transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 103

Andreassen, A., Bahri, Y., Neyshabur, B., and Roelofs, R. The Evolution of Out-of-Distribution Robustness Throughout Fine-Tuning. *arXiv preprint arXiv:2106.15831*, 2021. 128

Andriushchenko, M. and Flammarion, N. Understanding and Improving Fast Adversarial Training. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 67, 161

Andriushchenko, M., Croce, F., Flammarion, N., and Hein, M. Square Attack: A Query-Efficient Black-Box Adversarial Attack via Random Search. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 34, 35

Anthropic. Introducing the next generation of Claude, 2024. [Online; accessed 30. Nov. 2024]. 1

Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Zitnick, C. L., and Parikh, D. VQA: Visual Question Answering. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015. 26, 133

Anwar, S., Hwang, K., and Sung, W. Structured Pruning of Deep Convolutional Neural Networks. *J. Emerg. Technol. Comput. Syst.*, 2017. 23

Athalye, A., Carlini, N., and Wagner, D. Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples. *arXiv preprint arXiv:1802.00420*, 2018. 36

Aygun, M., Aytar, Y., and Kemal Ekenel, H. Exploiting Convolution Filter Patterns for Transfer Learning. In *Proceedings of the International Conference on Computer Vision Workshops (ICCVW)*, 2017. 47

Babaiee, Z., Kiasari, P., Rus, D., and Grosu, R. The Master Key Filters Hypothesis: Deep Filters Are General. In *NeurIPS Workshop on Scientific Methods for Understanding Deep Learning*, 2024a. 4, 152

Babaiee, Z., Kiasari, P., Rus, D., and Grosu, R. Unveiling the Unseen: Identifiable Clusters in Trained Depthwise Convolutional Kernels. In *International Conference on Learning Representations (ICLR)*, 2024b. 4, 23, 152

Babaiee, Z., Kiasari, P., Rus, D., and Grosu, R. We Need Far Fewer Unique Filters Than We Thought. In *NeurIPS 2024 Workshop on Scientific Methods for Understanding Deep Learning*, 2024c. 4, 152

Baehrens, D., Schroeter, T., Harmeling, S., Kawanabe, M., Hansen, K., and Müller, K.-R. How to Explain Individual Classification Decisions. *Journal of Machine Learning Research*, 2010. 38

Bafghi, R. A. and Gurari, D. A New Dataset Based on Images Taken by Blind People for Testing the Robustness of Image Classification Models Trained for ImageNet Categories. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 33

Bahng, H., Jahanian, A., Sankaranarayanan, S., and Isola, P. Exploring Visual Prompts for Adapting Large-Scale Models. *arXiv preprint arXiv:2203.17274*, 2022. 156

Bai, J., Lu, F., and Zhang, K. ONNX: Open Neural Network Exchange. `https://github.com/onnx/onnx`, 2019. 48

Bai, J., Bai, S., Chu, Y., Cui, Z., Dang, K., Deng, X., Fan, Y., Ge, W., Han, Y., Huang, F., Hui, B., Ji, L., Li, M., Lin, J., Lin, R., Liu, D., Liu, G., Lu, C., Lu, K., Ma, J., Men, R., Ren, X., Ren, X., Tan, C., Tan, S., Tu, J., Wang, P., Wang, S., Wang, W., Wu, S., Xu, B., Xu, J., Yang, A., Yang, H., Yang, J., Yang, S., Yao, Y., Yu, B., Yuan, H., Yuan, Z., Zhang, J., Zhang, X., Zhang, Y., Zhang, Z., Zhou, C., Zhou, J., Zhou, X., and Zhu, T. Qwen Technical Report. *arXiv preprint arXiv:2309.16609*, 2023a. 167

Bai, J., Bai, S., Yang, S., Wang, S., Tan, S., Wang, P., Lin, J., Zhou, C., and Zhou, J. Qwen-VL: A Versatile Vision-Language Model for Understanding, Localization, Text Reading, and Beyond. *arXiv preprint arXiv:2308.12966*, 2023b. 137, 167, 168

Bai, Y., Zeng, Y., Jiang, Y., Xia, S.-T., Ma, X., and Wang, Y. Improving Adversarial Robustness via Channel-wise Activation Suppressing. In *International Conference on Learning Representations (ICLR)*, 2021. 75, 76

Banerjee, S. and Roy, A. *Linear Algebra and Matrix Analysis for Statistics.* Taylor & Francis, 2014. 49

Barikeri, S., Lauscher, A., Vulic, I., and Glavas, G. RedditBias: A Real-World Resource for Bias Evaluation and Debiasing of Conversational Language Models. In *Proceedings of the Annual Meeting of the Association for Computational Linguistic (ACL)*, 2021. 133

Bartoldson, B. R., Diffenderfer, J., Parasyris, K., and Kailkhura, B. Adversarial Robustness Limits via Scaling-Law and Human-Alignment Studies. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2024. 4

Benarous, E., Anagnostidis, S., Biggio, L., and Hofmann, T. Harnessing Synthetic Datasets: The Role of Shape Bias in Deep Neural Network Generalization. In *NeurIPS Workshop on Synthetic Data Generation with Generative AI*, 2023. 131

Bengio, Y., Simard, P., and Frasconi, P. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 1994. 15

Bernhard, R., Moëllic, P., Mermillod, M., Bourrier, Y., Cohendet, R., Solinas, M., and Reyboz, M. Impact of Spatial Frequency Based Constraints on Adversarial Robustness. In *International Joint Conference on Neural Networks (IJCNN)*, 2021. 78, 86, 110

Beyer, L., Hénaff, O. J., Kolesnikov, A., Zhai, X., and van den Oord, A. Are we done with ImageNet? *arXiv preprint arXiv:2006.07159*, 2020. 28, 118, 128

Bhagoji, A. N., He, W., Li, B., and Song, D. Practical Black-Box Attacks on Deep Neural Networks Using Efficient Query Mechanisms. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 34

Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., Giacinto, G., and Roli, F. Evasion Attacks against Machine Learning at Test Time. In *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*, 2013. 2, 34, 65, 113, 115

Bishop, C. M. *Neural Networks for Pattern Recognition.* Clarendon Press, 1995. 29

Box, G. E. P. Science and Statistics. *Journal of the American Statistical Association*, 1976. 130

Boyd, R. Do People Only Use 10 Percent of Their Brains? *Scientific American*, 2008. 93

Branwen, G. The Scaling Hypothesis. `https://gwern.net/scaling-hypothesis`, 2020. [Online; accessed 30. Nov. 2024]. 2, 30

Brendel, W. and Bethge, M. Approximating CNNs with Bag-of-local-Features models works surprisingly well on ImageNet. In *International Conference on Learning Representations (ICLR)*, 2019. 104, 120

Brendel, W., Rauber, J., Kümmerer, M., Ustyuzhaninov, I., and Bethge, M. Accurate, reliable and fast robustness evaluation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 35

Brock, A., Lim, T., Ritchie, J. M., and Weston, N. Neural Photo Editing with Introspective Adversarial Networks. *arXiv preprint arXiv:1609.07093*, 2017. 66

Brock, A., Donahue, J., and Simonyan, K. Large Scale GAN Training for High Fidelity Natural Image Synthesis. In *International Conference on Learning Representations (ICLR)*, 2019. 66

Brown, T. B., Mané, D., Roy, A., Abadi, M., and Gilmer, J. Adversarial Patch. *arXiv preprint arXiv:1712.09665*, 2018. 34

Buda, M., Saha, A., and Mazurowski, M. A. Association of genomic subtypes of lower-grade gliomas with shape features automatically extracted by a deep learning algorithm. *Computers in Biology and Medicine*, 2019. 62

Buolamwini, J. and Gebru, T. Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification. In *Conference on Fairness, Accountability and Transparency (FAT)*, 2018. 131, 133, 157

Cammarata, N., Goh, G., Carter, S., Schubert, L., Petrov, M., and Olah, C. Curve Detectors. *Distill*, 2020. 22, 47

Cammarata, N., Goh, G., Carter, S., Voss, C., Schubert, L., and Olah, C. Curve Circuits. *Distill*, 2021. 22, 47

Cao, Y., Xiao, C., Cyr, B., Zhou, Y., Park, W., Rampazzi, S., Chen, Q. A., Fu, K., and Mao, Z. M. Adversarial Sensor Attack on LiDAR-Based Perception in Autonomous Driving. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2019. 113

Cao, Y., Jiang, D., and Yang, Q. Image classification based on enhanced learning of global and local features with structural priors. *Knowledge-Based Systems*, 2025. 154

Carlini, N. and Wagner, D. Towards Evaluating the Robustness of Neural Networks. *arXiv preprint arXiv:1608.04644*, 2016. 35, 36, 65

Carlini, N. and Wagner, D. Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods. In *Proceedings of the ACM Workshop on Artificial Intelligence and Security*, 2017. 36

Carlini, N., Athalye, A., Papernot, N., Brendel, W., Rauber, J., Tsipras, D., Goodfellow, I., Madry, A., and Kurakin, A. On Evaluating Adversarial Robustness. *arXiv preprint arXiv:1902.06705*, 2019. 35

Carlini, N., Tramer, F., Dvijotham, K. D., Rice, L., Sun, M., and Kolter, J. Z. (Certified!!) Adversarial Robustness for Free! In *International Conference on Learning Representations (ICLR)*, 2023. 36

Carmon, Y., Raghunathan, A., Schmidt, L., Liang, P., and Duchi, J. C. Unlabeled Data Improves Adversarial Robustness. *arXiv preprint arXiv:1905.13736*, 2022. 67, 161

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., and Joulin, A. Unsupervised Learning of Visual Features by Contrasting Cluster Assignments. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 95, 98, 120, 164, 165, 166

Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., and Joulin, A. Emerging Properties in Self-Supervised Vision Transformers. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021. 95, 98, 120, 164, 165, 166

Chameleon Team. Chameleon: Mixed-Modal Early-Fusion Foundation Models. *arXiv preprint arXiv:2405.09818*, 2024. 156

Chang, T., Wiens, J., Schnabel, T., and Swaminathan, A. Measuring Steerability in Large Language Models. In *NeurIPS Workshop on Safe Generative AI*, 2024. 154

Chao, P., Kao, C.-Y., Ruan, Y.-S., Huang, C.-H., and Lin, Y.-L. HarDNet: A Low Memory Traffic Network. *arXiv preprint arXiv:1909.00948*, 2019. 59

Chatterji, N., Neyshabur, B., and Sedghi, H. The intriguing role of module criticality in the generalization of deep networks. In *International Conference on Learning Representations (ICLR)*, 2020. 94, 96, 97, 99, 100

Cheinski, K. and Wawrzynski, P. DCT-Conv: Coding filters in convolutional networks with Discrete Cosine Transform. In *International Joint Conference on Neural Networks (IJCNN)*, 2020. 79

Chen, A., Lorenz, P., Yao, Y., Chen, P.-Y., and Liu, S. Visual Prompting for Adversarial Robustness. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023a. 156

Chen, E.-C. and Lee, C.-R. LTD: Low Temperature Distillation for Robust Adversarial Training. *arXiv preprint arXiv:2111.02331*, 2021. 67, 162, 163

Chen, F., Datta, G., Kundu, S., and Beerel, P. A. Self-Attentive Pooling for Efficient Deep Learning. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2023b. 19

Chen, J., Cheng, Y., Gan, Z., Gu, Q., and Liu, J. Efficient Robust Training via Backward Smoothing. *arXiv preprint arXiv:2010.01278*, 2021a. 67, 162, 163

Chen, P., Agarwal, C., and Nguyen, A. The shape and simplicity biases of adversarially robust ImageNet-trained CNNs. *arXiv preprint arXiv:2006.09373*, 2022a. 5, 103, 114

Chen, T., Kornblith, S., Swersky, K., Norouzi, M., and Hinton, G. E. Big Self-Supervised Models are Strong Semi-Supervised Learners. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020a. 95, 98, 120, 164, 165, 166

Chen, T., Liu, S., Chang, S., Cheng, Y., Amini, L., and Wang, Z. Adversarial Robustness: From Self-Supervised Pre-Training to Fine-Tuning. *arXiv preprint arXiv:2003.12862*, 2020b. 67, 161

Chen, W. *The Electrical Engineering Handbook*. Academic Press, 2004. 79

Chen, W., Wilson, J. T., Tyree, S., Weinberger, K. Q., and Chen, Y. Compressing Convolutional Neural Networks in the Frequency Domain. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2016. 79

Chen, X., Xie, S., and He, K. An Empirical Study of Training Self-Supervised Vision Transformers. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021b. 95, 98, 120, 164, 165, 166

Chen, Y., Zhou, R., Guo, B., Shen, Y., Wang, W., Wen, X., and Suo, X. Discrete cosine transform for filter pruning. *Applied Intelligence*, 2022b. 79

Chen, Y.-C., Li, L., Yu, L., El Kholy, A., Ahmed, F., Gan, Z., Cheng, Y., and Liu, J. UNITER: UNiversal Image-TExt Representation Learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020c. 25

Chen, Z., Wu, J., Wang, W., Su, W., Chen, G., Xing, S., Zhong, M., Zhang, Q., Zhu, X., Lu, L., Li, B., Luo, P., Lu, T., Qiao, Y., and Dai, J. InternVL: Scaling up Vision Foundation Models and Aligning for Generic Visual-Linguistic Tasks. *arXiv preprint arXiv:2312.14238*, 2024. 130, 132, 137, 138, 141, 142, 143, 148, 155, 167, 168

Chollet, F. Xception: Deep Learning With Depthwise Separable Convolutions. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 18, 21

Chrabaszcz, P., Loshchilov, I., and Hutter, F. A Downsampled Variant of ImageNet as an Alternative to the CIFAR datasets. *arXiv preprint arXiv:1707.08819*, 2017. 28

Cianfarani, C., Bhagoji, A. N., Sehwag, V., Zhao, B., Zheng, H., and Mittal, P. Understanding Robust Learning through the Lens of Representation Similarities. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 153

Clanuwat, T., Bober-Irizar, M., Kitamoto, A., Lamb, A., Yamamoto, K., and Ha, D. Deep Learning for Classical Japanese Literature. *arXiv preprint arXiv:1812.01718*, 2018. 48

Clark, K., Khandelwal, U., Levy, O., and Manning, C. D. What Does BERT Look at? An Analysis of BERT's Attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 2019. 156

Cohen, J. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 1960. 139

Cohen, J., Rosenfeld, E., and Kolter, Z. Certified Adversarial Robustness via Randomized Smoothing. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019. 36

Cohen, J. P., Viviano, J. D., Bertin, P., Morrison, P., Torabian, P., Guarrera, M., Lungren, M. P., Chaudhari, A., Brooks, R., Hashir, M., and Bertrand, H. TorchXRayVision: A library of chest X-ray datasets and models. In *Medical Imaging with Deep Learning (MIDL)*, 2022. 62

Cooley, J. W. and Tukey, J. W. An Algorithm for the Machine Calculation of Complex Fourier Series. *Mathematics of Computation*, 1965. 14

Croce, F. and Hein, M. Minimally distorted Adversarial Examples with a Fast Adaptive Boundary Attack. *arXiv preprint arXiv:1907.02044*, 2020a. 34, 35

Croce, F. and Hein, M. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2020b. 34, 35, 84

Croce, F., Andriushchenko, M., Sehwag, V., Debenedetti, E., Flammarion, N., Chiang, M., Mittal, P., and Hein, M. RobustBench: a standardized adversarial robustness benchmark. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 35, 48, 54, 67

Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., and Le, Q. V. AutoAugment: Learning Augmentation Strategies From Data. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 5, 30, 33, 95, 164

Cubuk, E. D., Zoph, B., Shlens, J., and Le, Q. RandAugment: Practical Automated Data Augmentation with a Reduced Search Space. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 33, 95, 164

Cui, J., Liu, S., Wang, L., and Jia, J. Learnable Boundary Guided Adversarial Training. *arXiv preprint arXiv:2011.11164*, 2021. 67, 162

Cui, J., Tian, Z., Zhong, Z., Qi, X., Yu, B., and Zhang, H. Decoupled Kullback-Leibler Divergence Loss. *arXiv preprint arXiv:2305.13948*, 2023. 36

Cybenko, G. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 1989. 15

Dai, S., Mahloujifar, S., and Mittal, P. Parameterizing Activation Functions for Adversarial Robustness. *arXiv preprint arXiv:2110.05626*, 2021. 67, 162

Dai, W., Li, J., Li, D., Tiong, A., Zhao, J., Wang, W., Li, B., Fung, P., and Hoi, S. InstructBLIP: Towards General-purpose Vision-Language Models with Instruction Tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 132, 137, 138, 139, 145, 148, 167, 168

Darcet, T., Oquab, M., Mairal, J., and Bojanowski, P. Vision Transformers Need Registers. In *International Conference on Learning Representations (ICLR)*, 2024. 24

de Sa, V. Learning Classification with Unlabeled Data. In *Advances in Neural Information Processing Systems (NeurIPS)*, 1993. 1

Debenedetti, E., Sehwag, V., and Mittal, P. A Light Recipe to Train Robust Vision Transformers. In *Proceedings of the IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, 2023. 105

Dehghani, M., Djolonga, J., Mustafa, B., Padlewski, P., Heek, J., Gilmer, J., Steiner, A. P., Caron, M., Geirhos, R., Alabdulmohsin, I., Jenatton, R., Beyer, L., Tschannen, M., Arnab, A., Wang, X., Riquelme Ruiz, C., Minderer, M., Puigcerver, J., Evci, U., Kumar, M., Steenkiste, S. V., Elsayed, G. F., Mahendran, A., Yu, F., Oliver, A., Huot, F., Bastings, J., Collier, M., Gritsenko, A. A., Birodkar, V., Vasconcelos, C. N., Tay, Y., Mensink, T., Kolesnikov, A., Pavetic, F., Tran, D., Kipf, T., Lucic, M., Zhai, X., Keysers, D., Harmsen, J. J., and Houlsby, N. Scaling Vision Transformers to 22 Billion Parameters. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2023. 114, 127, 153, 155, 167

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 2, 3, 14, 20, 28, 32, 33, 37, 38, 40, 47, 48, 78, 83, 86, 93, 94, 96, 103, 105, 113, 118, 119, 132, 133

Deng, Y., Zheng, X., Zhang, T., Chen, C., Lou, G., and Kim, M. An Analysis of Adversarial Attacks and Defenses on Autonomous Driving Models. *arXiv preprint arXiv:2002.02175*, 2020. 65, 157

Dey, R. and Hong, Y. CompNet: Complementary Segmentation Network for Brain MRI Extraction. *arXiv preprint arXiv:1804.00521*, 2018. 61, 62, 63

Dictionary.com. shape. `https://www.dictionary.com/browse/shape`, 2024a. [Online; accessed 25. Feb. 2024]. 142

Dictionary.com. texture. `https://www.dictionary.com/browse/texture`, 2024b. [Online; accessed 25. Feb. 2024]. 142

Ding, G. W., Sharma, Y., Lui, K. Y. C., and Huang, R. MMA Training: Direct Input Space Margin Maximization through Adversarial Training. In *International Conference on Learning Representations (ICLR)*, 2020. 67, 161

Ding, X., Ding, G., Guo, Y., and Han, J. Centripetal SGD for Pruning Very Deep Convolutional Networks With Complicated Structure. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 47

Ding, X., Zhang, X., Han, J., and Ding, G. Scaling Up Your Kernels to 31x31: Revisiting Large Kernel Design in CNNs. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 22

Dodge, S. F. and Karam, L. J. A Study and Comparison of Human and Deep Learning Recognition Performance Under Visual Distortions. *arXiv preprint arXiv:1705.02498*, 2017. 78

Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., and Darrell, T. DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2014. 47

Dong, Y., Ruan, S., Su, H., Kang, C., Wei, X., and Zhu, J. ViewFool: Evaluating the Robustness of Visual Recognition to Adversarial Viewpoints. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 2, 5

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations (ICLR)*, 2021. 1, 2, 16, 23, 28, 105, 113, 132, 155

Dravid, A., Gandelsman, Y., Efros, A. A., and Shocher, A. Rosetta Neurons: Mining the Common Units in a Model Zoo. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2023. 4

Duan, R., Chen, Y., Niu, D., Yang, Y., Qin, A. K., and He, Y. AdvDrop: Adversarial Attack to DNNs by Dropping Information. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021. 78

Durall, R., Keuper, M., and Keuper, J. Watch Your Up-Convolution: CNN Based Generative Deep Neural Networks Are Failing to Reproduce Spectral Distributions. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 106

Elhage, N., Hume, T., Olsson, C., Schiefer, N., Henighan, T., Kravec, S., Hatfield-Dodds, Z., Lasenby, R., Drain, D., Chen, C., Grosse, R., McCandlish, S., Kaplan, J., Amodei, D., Wattenberg, M., and Olah, C. Toy Models of Superposition. *Transformer Circuits Thread*, 2022. 39

Engstrom, L., Ilyas, A., Salman, H., Santurkar, S., and Tsipras, D. Robustness (Python Library), 2019. 67, 161, 163

Eppel, S. Do large language vision models understand 3D shapes? *arXiv preprint arXiv:2412.10908*, 2025. 154

Erhan, D., Bengio, Y., Courville, A., and Vincent, P. Visualizing Higher-Layer Features of a Deep Network. Technical report, University of Montreal, 2009. 38

Erichson, N. B., Lim, S. H., Xu, W., Utrera, F., Cao, Z., and Mahoney, M. W. NoisyMix: Boosting Model Robustness to Common Corruptions. *arXiv preprint arXiv:2202.01263*, 2022. 33, 95, 119, 164, 165, 166

Fang, Y., Wang, W., Xie, B., Sun, Q., Wu, L., Wang, X., Huang, T., Wang, X., and Cao, Y. EVA: Exploring the Limits of Masked Visual Representation Learning at Scale. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 23

Feinman, R., Curtin, R. R., Shintre, S., and Gardner, A. B. Detecting Adversarial Samples from Artifacts. *arXiv preprint arXiv:1703.00410*, 2017. 36

Fellbaum, C. *WordNet: An Electronic Lexical Database*. Bradford Books, 1998. 28

Finder, S. E., Amoyal, R., Treister, E., and Freifeld, O. Wavelet Convolutions for Large Receptive Fields. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024. 154

Finlayson, S. G., Bowers, J. D., Ito, J., Zittrain, J. L., Beam, A. L., and Kohane, I. S. Adversarial attacks on medical machine learning. *Science*, 2019. 65, 113, 157

Fisher, R. B., Perkins, S., Walker, A., and Wolfart, E. *Hypermedia Image Processing Reference (HIPR)*. Wiley, 1997. 13, 14

Fourier, J. B. J. *Théorie Analytique de la Chaleur*. Firmin Didot, Père et Fils, 1822. 13

Frankle, J. and Carbin, M. The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks. In *International Conference on Learning Representations (ICLR)*, 2019. 50, 64, 75, 95

Fukushima, K. Neocognitron: A hierarchical neural network capable of visual pattern recognition. *Neural Networks*, 1988. 19

Gabor, D. Theory of communication. Part 1: The analysis of information. *Journal of the Institution of Electrical Engineers - Part III: Radio and Communication Engineering*, 1946. 22

Ganin, Y. and Lempitsky, V. Unsupervised Domain Adaptation by Backpropagation. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2015. 33

Gao, B. and Spratling, M. W. Filter competition results in more robust Convolutional Neural Networks. *Neurocomputing*, 2025. 153

Gatys, L. A., Ecker, A. S., and Bethge, M. Image Style Transfer Using Convolutional Neural Networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016. 61, 132

Gavrikov, P. and Keuper, J. An Empirical Investigation of Model-to-Model Distribution Shifts in Trained Convolutional Filters. In *NeurIPS Workshop on Distribution Shifts: Connecting Methods and Applications*, 2021. 8

Gavrikov, P. and Keuper, J. Adversarial Robustness Through the Lens of Convolutional Filters. In *Proceedings of the Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2022a. 9, 65

Gavrikov, P. and Keuper, J. CNN Filter DB: An Empirical Investigation of Trained Convolutional Filters. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022b. 8, 45, 48

Gavrikov, P. and Keuper, J. Does Medical Imaging learn different Convolution Filters? In *NeurIPS 2022 Workshop on Medical Imaging*, 2022c. 9, 45

Gavrikov, P. and Keuper, J. On the Interplay of Convolutional Padding and Adversarial Robustness. In *Proceedings of the International Conference on Computer Vision Workshops (ICCVW)*, 2023a. 10, 76

Gavrikov, P. and Keuper, J. The Power of Linear Combinations: Learning with Random Convolutions. *arXiv preprint arXiv:2301.11360*, 2023b. 97, 100, 120, 165, 166

Gavrikov, P. and Keuper, J. Can Biases in ImageNet Models Explain Generalization? In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 9, 98, 112

Gavrikov, P., Keuper, J., and Keuper, M. An Extended Study of Human-Like Behavior Under Adversarial Training. In *Proceedings of the Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2023. 9, 103, 114, 121, 132

Gavrikov, P., Agnihotri, S., Keuper, M., and Keuper, J. How Do Training Methods Influence the Utilization of Vision Models? In *NeurIPS 2024 Workshop on Interpretable AI: Past, Present and Future*, 2024a. 9, 93

Gavrikov, P., Lukasik, J., Jung, S., Geirhos, R., Lamm, B., Mirza, M. J., Keuper, M., and Keuper, J. Are Vision Language Models Texture or Shape Biased and Can We Steer Them? *arXiv preprint arXiv:2403.09193*, 2024b. 9, 114

Gavrikov, P., Lukasik, J., Jung, S., Geirhos, R., Mirza, M. J., Keuper, M., and Keuper, J. Can We Talk Models Into Seeing the World Differently? In *International Conference on Learning Representations (ICLR)*, 2025. 9, 130

Geirhos, R., Temme, C. R. M., Rauber, J., Schütt, H. H., Bethge, M., and Wichmann, F. A. Generalisation in humans and deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. 5, 6, 28, 32, 33, 103, 104, 105

Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F. A., and Brendel, W. ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations (ICLR)*, 2019. 5, 6, 28, 32, 33, 37, 38, 40, 41, 76, 77, 78, 80, 84, 88, 95, 102, 103, 104, 105, 110, 111, 112, 113, 114, 117, 118, 120, 121, 122, 127, 130, 131, 132, 133, 137, 144, 154, 156, 164, 165, 166

Geirhos, R., Jacobsen, J.-H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., and Wichmann, F. A. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2020a. 1, 5, 37, 131

Geirhos, R., Meding, K., and Wichmann, F. A. Beyond accuracy: quantifying trial-by-trial behaviour of CNNs and humans by measuring error consistency. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020b. 95, 103, 104, 111, 139, 140

Geirhos, R., Narayanappa, K., Mitzkus, B., Thieringer, T., Bethge, M., Wichmann, F. A., and Brendel, W. Partial success in closing the gap between human and machine vision. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 5, 78, 84, 103, 104, 105, 111, 114, 117, 118, 119, 121, 127, 132, 146

Geirhos, R., Zimmermann, R. S., Bilodeau, B., Brendel, W., and Kim, B. Don't trust your eyes: on the (un)reliability of feature visualizations. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2024. 4, 39

Gemini Team. Gemini: A Family of Highly Capable Multimodal Models. *arXiv preprint arXiv:2312.11805*, 2023. 1, 132, 137, 167, 168

Glorot, X. and Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010. 49, 95

Goh, G., Cammarata, N., Voss, C., Carter, S., Petrov, M., Schubert, L., Radford, A., and Olah, C. Multimodal Neurons in Artificial Neural Networks. *Distill*, 2021. 39, 41, 133

Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`. 5, 10, 16, 19, 25, 29, 30

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and Harnessing Adversarial Examples. In *International Conference on Learning Representations (ICLR)*, 2015. 4, 30, 33, 34, 65

Gowal, S., Qin, C., Uesato, J., Mann, T., and Kohli, P. Uncovering the Limits of Adversarial Training against Norm-Bounded Adversarial Examples. *arXiv preprint arXiv:2010.03593*, 2020. 67, 161, 162

Gowal, S., Rebuffi, S.-A., Wiles, O., Stimberg, F., Calian, D. A., and Mann, T. Improving Robustness using Generated Data. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 36, 67, 162

Grabinski, J., Gavrikov, P., Keuper, J., and Keuper, M. Robust Models are less Over-Confident. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022a. 5, 10, 100, 153

Grabinski, J., Gavrikov, P., Keuper, J., and Keuper, M. Robust Models are less Over-Confident. In *ICML Workshop on New Frontiers in Adversarial Machine Learning*, 2022b. 10

Grabinski, J., Jung, S., Keuper, J., and Keuper, M. FrequencyLowCut Pooling - Plug and Play Against Catastrophic Overfitting. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022c. 76, 79, 85, 86, 90, 92, 110

Grabinski, J., Keuper, J., and Keuper, M. Aliasing and adversarial robust generalization of CNNs. *Machine Learning*, 2022d. 76, 110

Graves, A., Mohamed, A.-r., and Hinton, G. Speech recognition with deep recurrent neural networks. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2013. 1

Gretton, A., Smola, A., Huang, J., Schmittfull, M., Borgwardt, K., and Schölkopf, B. *Covariate shift and local learning by distribution matching*, pp. 131–160. MIT Press, 2009. 31

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On Calibration of Modern Neural Networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2017. 147

Guo, Y., Stutz, D., and Schiele, B. Improving Corruption and Adversarial Robustness by Enhancing Weak Subnets. *arXiv preprint arXiv:2201.12765*, 2022a. 75, 76

Guo, Y., Yang, Y., and Abbasi, A. Auto-Debias: Debiasing Masked Language Models with Automated Biased Prompts. In *Proceedings of the Annual Meeting of the Association for Computational Linguistic (ACL)*, 2022b. 133

Haller, P., Aynetdinov, A., and Akbik, A. OpinionGPT: Modelling Explicit Biases in Instruction-Tuned LLMs. *arXiv preprint arXiv:2309.03876*, 2023. 133

Han, S., Pool, J., Tran, J., and Dally, W. J. Learning both Weights and Connections for Efficient Neural Networks. *arXiv preprint arXiv:1506.02626*, 2015. 47

Harder, P., Pfreundt, F.-J., Keuper, M., and Keuper, J. SpectralDefense: Detecting Adversarial Attacks on CNNs in the Fourier Domain. In *International Joint Conference on Neural Networks (IJCNN)*, 2021. 36, 110

Hassibi, B., Stork, D., and Wolff, G. Optimal Brain Surgeon and general network pruning. In *IEEE International Conference on Neural Networks*, 1993. 93

He, K., Zhang, X., Ren, S., and Sun, J. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015. 2, 5, 16, 21, 49, 82, 95

He, K., Zhang, X., Ren, S., and Sun, J. Deep Residual Learning for Image Recognition. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1, 2, 19, 20, 21, 33, 48, 59, 67, 74, 76, 83, 93, 94, 95, 96, 98, 103, 113, 117, 119, 127, 137, 164, 165, 166

He, Y., Zhang, X., and Sun, J. Channel Pruning for Accelerating Very Deep Neural Networks. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017. 47

He, Y., Kang, G., Dong, X., Fu, Y., and Yang, Y. Soft Filter Pruning for Accelerating Deep Convolutional Neural Networks. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2018. 47

He, Y., Liu, P., Wang, Z., Hu, Z., and Yang, Y. Filter Pruning via Geometric Median for Deep Convolutional Neural Networks Acceleration. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 47

Hemmat, A., Davies, A., Lamb, T. A., Yuan, J., Torr, P., Khakzar, A., and Pinto, F. Hidden in Plain Sight: Evaluating Abstract Shape Recognition in Vision-Language Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 154, 156

Hendria, W. F. and Gavrikov, P. VisualTorch: Streamlining Visualization for PyTorch Neural Network Architectures. *Journal of Open Source Software*, 2024. 10

Hendrycks, D. and Dietterich, T. Benchmarking Neural Network Robustness to Common Corruptions and Perturbations. In *International Conference on Learning Representations (ICLR)*, 2019. 2, 5, 28, 32, 77, 78, 84, 100, 104, 113, 114, 115, 117, 118

Hendrycks, D. and Gimpel, K. Gaussian Error Linear Units (GELUs). *arXiv preprint arXiv:1606.08415*, 2016. 16

Hendrycks, D., Lee, K., and Mazeika, M. Using Pre-Training Can Improve Model Robustness and Uncertainty. *arXiv preprint arXiv:1901.09960*, 2019. 67, 161, 163

Hendrycks, D., Mu, N., Cubuk, E. D., Zoph, B., Gilmer, J., and Lakshminarayanan, B. AugMix: A Simple Data Processing Method to Improve Robustness and Uncertainty. In *International Conference on Learning Representations (ICLR)*, 2020. 5, 30, 33, 95, 117, 119, 164, 165, 166

Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., Song, D., Steinhardt, J., and Gilmer, J. The Many Faces of Robustness: A Critical Analysis of Out-of-Distribution Generalization. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021a. 2, 5, 28, 30, 33, 95, 98, 113, 118, 119, 126, 164, 165, 166

Hendrycks, D., Zhao, K., Basart, S., Steinhardt, J., and Song, D. Natural Adversarial Examples. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021b. 28, 118, 121

Hendrycks, D., Zou, A., Mazeika, M., Tang, L., Li, B., Song, D., and Steinhardt, J. PixMix: Dreamlike Pictures Comprehensively Improve Safety Measures. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 33, 95, 96, 97, 98, 117, 119, 164, 165, 166

Herculano-Houzel, S. The Human Brain in Numbers: A Linearly Scaled-up Primate Brain. *Frontiers in Human Neuroscience*, 2009. 93

Hermann, K., Chen, T., and Kornblith, S. The Origins and Prevalence of Texture Bias in Convolutional Neural Networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 114, 131, 132

Hinton, G., Vinyals, O., and Dean, J. Distilling the Knowledge in a Neural Network. *arXiv preprint arXiv:1503.02531*, 2015. 1, 93

Hinton, G. E., Osindero, S., and Teh, Y.-W. A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*, 2006. 1

Ho, S.-T., Vo, T. V., Ebrahimkhani, S., and man Cheung, N. Vision Transformer Neural Architecture Search for Out-of-Distribution Generalization: Benchmark and Insights. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 154

Hochreiter, S. and Schmidhuber, J. Long Short-Term Memory. *Neural Computation*, 1997. 1, 15

Hoffman, J., Tzeng, E., Park, T., Zhu, J.-Y., Isola, P., Saenko, K., Efros, A., and Darrell, T. CyCADA: Cycle-Consistent Adversarial Domain Adaptation. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018. 33

Hofmanninger, J., Prayer, F., Pan, J., Röhrich, S., Prosch, H., and Langs, G. Automatic lung segmentation in routine imaging is primarily a data diversity problem, not a methodology problem. *European Radiology Experimental*, 2020. 61

Honegger, D., Schürholt, K., and Borth, D. Sparsified Model Zoo Twins: Investigating Populations of Sparsified Neural Network Models. In *ICLR Workshop on Sparsity in Neural Networks*, 2023. 152

Hong, W., Wang, W., Lv, Q., Xu, J., Yu, W., Ji, J., Wang, Y., Wang, Z., Zhang, Y., Li, J., Xu, B., Dong, Y., Ding, M., and Tang, J. CogAgent: A Visual Language Model for GUI Agents. *arXiv preprint arXiv:2312.08914*, 2023. 137, 167, 168

Hopfield, J. J. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 1982. 15

Hornik, K., Stinchcombe, M., and White, H. Multilayer feedforward networks are universal approximators. *Neural Networks*, 1989. 15

Hossain, M. T., Teng, S. W., Zhang, D., Lim, S., and Lu, G. Distortion Robust Image Classification Using Deep Convolutional Neural Network with Discrete Cosine Transform. In *IEEE International Conference on Image Processing (ICIP)*, 2019. 79

Howard, A., Sandler, M., Chu, G., Chen, L.-C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., Le, Q. V., and Adam, H. Searching for MobileNetV3. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2019. 59

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv preprint arXiv:1704.04861*, 2017. 18, 21

Howard, J. Imagenette. `https://github.com/fastai/imagenette`, 2019. [Online; accessed 10. Dec. 2024]. 28

Huang, G., Liu, Z., van der Maaten, L., and Weinberger, K. Q. Densely Connected Convolutional Networks. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2, 19, 21, 48, 59

Huang, H., Wang, Y., Erfani, S. M., Gu, Q., Bailey, J., and Ma, X. Exploring Architectural Ingredients of Adversarially Robust Deep Neural Networks. *arXiv preprint arXiv:2110.03825*, 2022. 67, 162

Huang, L., Zhang, C., and Zhang, H. Self-Adaptive Training: beyond Empirical Risk Minimization. *arXiv preprint arXiv:2002.10319*, 2020. 67, 161

Huang, S., Dong, L., Wang, W., Hao, Y., Singhal, S., Ma, S., Lv, T., Cui, L., Mohammed, O. K., Patra, B., Liu, Q., Aggarwal, K., Chi, Z., Bjorck, J., Chaudhary, V., Som, S., Song, X., and Wei, F. Language Is Not All You Need: Aligning Perception with Language Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 131, 132

Hubel, D. H. and Wiesel, T. N. Receptive fields of single neurones in the cat's striate cortex. *The Journal of Physiology*, 1959. 16, 97

Hubel, D. H. and Wiesel, T. N. Brain Mechanisms of Vision. *Scientific American*, 1979. 97

Hussain, M., Bird, J. J., and Faria, D. R. A Study on CNN Transfer Learning for Image Classification. In *Advances in Computational Intelligence Systems*, 2019. 47

Ilyas, A., Engstrom, L., Athalye, A., and Lin, J. Black-box Adversarial Attacks with Limited Queries and Information. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018. 34

Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., and Madry, A. Adversarial Examples Are Not Bugs, They Are Features. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 37, 121

Ioffe, S. and Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2015. 19, 20, 94, 95, 96

Irvin, J., Rajpurkar, P., Ko, M., Yu, Y., Ciurea-Ilcus, S., Chute, C., Marklund, H., Haghgoo, B., Ball, R., Shpanskaya, K., Seekins, J., Mong, D. A., Halabi, S. S., Sandberg, J. K., Jones, R., Larson, D. B., Langlotz, C. P., Patel, B. N., Lungren, M. P., and Ng, A. Y. CheXpert: A Large Chest Radiograph Dataset with Uncertainty Labels and Expert Comparison. *arXiv preprint arXiv:1901.07031*, 2019. 62

Islam, M. A., Kowal, M., Esser, P., Jia, S., Ommer, B., Derpanis, K. G., and Bruce, N. Shape or Texture: Understanding Discriminative Features in CNNs. In *International Conference on Learning Representations (ICLR)*, 2021. 114, 131, 132, 141, 154, 156

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., and Hinton, G. E. Adaptive Mixtures of Local Experts. *Neural Computation*, 1991. 99

Jain, S. and Wallace, B. C. Attention is not Explanation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2019. 38

Jaini, P., Clark, K., and Geirhos, R. Intriguing Properties of Generative Classifiers. In *International Conference on Learning Representations (ICLR)*, 2024. 95, 114, 119, 132, 142, 164, 165, 166

Jia, C., Yang, Y., Xia, Y., Chen, Y.-T., Parekh, Z., Pham, H., Le, Q., Sung, Y.-H., Li, Z., and Duerig, T. Scaling up visual and vision-language representation learning with noisy text supervision. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021. 25, 132

Jiang, A. Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., Chaplot, D. S., de las Casas, D., Hanna, E. B., Bressand, F., Lengyel, G., Bour, G., Lample, G., Lavaud, L. R., Saulnier, L., Lachaux, M.-A., Stock, P., Subramanian, S., Yang, S., Antoniak, S., Scao, T. L., Gervet, T., Lavril, T., Wang, T., Lacroix, T., and Sayed, W. E. Mixtral of Experts. *arXiv preprint arXiv:2401.04088*, 2024. 134

Johnson, A. E. W., Pollard, T. J., Greenbaum, N. R., Lungren, M. P., Deng, C.-y., Peng, Y., Lu, Z., Mark, R. G., Berkowitz, S. J., and Horng, S. MIMIC-CXR-JPG, a large publicly available database of labeled chest radiographs. *arXiv preprint arXiv:1901.07042*, 2019. 62

Jolliffe, I. *Principal Component Analysis.* Springer New York, 1986. 48

Juodelyte, D., Jiménez-Sánchez, A., and Cheplygina, V. Revisiting Hidden Representations in Transfer Learning for Medical Imaging. *Transactions on Machine Learning Research (TMLR)*, 2023. 152

Kanbak, C., Moosavi-Dezfooli, S.-M., and Frossard, P. Geometric Robustness of Deep Networks: Analysis and Improvement. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 5

Kar, O. F., Yeo, T., Atanov, A., and Zamir, A. 3D Common Corruptions and Data Augmentation. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2, 32

Karpathy, A. What I learned from competing against a ConvNet on ImageNet. `https://karpathy.github.io/2014/09/02/what-i-learned-from-competing-against-a-convnet-on-imagenet`, 2014. [Online; accessed 29. Feb. 2024]. 2, 21

Kataoka, H., Okayasu, K., Matsumoto, A., Yamagata, E., Yamada, R., Inoue, N., Nakamura, A., and Satoh, Y. Pre-training without Natural Images. In *Asian Conference on Computer Vision (ACCV)*, 2020. 46, 58

Khaligh-Razavi, S.-M. and Kriegeskorte, N. Deep Supervised, but Not Unsupervised, Models May Explain IT Cortical Representation. *PLOS Computational Biology*, 2014. 47

Kim, H., Lee, W., and Lee, J. Understanding Catastrophic Overfitting in Single-step Adversarial Training. *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2021. 36

Kim, M., Orshulevich, V., and Vardanian, A. UForm by Unum Cloud, 2023. 137, 167, 168

Kireev, K., Andriushchenko, M., and Flammarion, N. On the effectiveness of adversarial training against common corruptions. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2022. 36, 76

Kirichenko, P., Izmailov, P., and Wilson, A. G. Last Layer Re-Training is Sufficient for Robustness to Spurious Correlations. In *International Conference on Learning Representations (ICLR)*, 2023. 154

Kobayashi, T. and Ye, J. Spatio-Temporal Filter Analysis Improves 3D-CNN for Action Classification. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2024. 152

Koenderink, J., Valsecchi, M., van Doorn, A., Wagemans, J., and Gegenfurtner, K. Eidolons: Novel stimuli for vision research. *Journal of Vision*, 2017. 104

Kolesnikov, A., Beyer, L., Zhai, X., Puigcerver, J., Yung, J., Gelly, S., and Houlsby, N. Big Transfer (BiT): General Visual Representation Learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 105

Kornblith, S., Norouzi, M., Lee, H., and Hinton, G. Similarity of Neural Network Representations Revisited. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019a. 4

Kornblith, S., Shlens, J., and Le, Q. V. Do Better ImageNet Models Transfer Better? In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019b. 28

Kriegeskorte, N., Mur, M., and Bandettini, P. A. Representational similarity analysis - connecting the branches of systems neuroscience. *Frontiers in Systems Neuroscience*, 2008. 4

Krizhevsky, A. Learning Multiple Layers of Features from Tiny Images. Technical report, University of Toronto, 2009. 15

Krizhevsky, A., Nair, V., and Hinton, G. CIFAR-10. Technical report, Canadian Institute for Advanced Research, 2009. 28, 48, 66, 67, 78, 83

Krizhevsky, A., Sutskever, I., and Hinton, G. E. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2012. 1, 2, 5, 18, 19, 20, 22, 25, 30, 48, 98, 113, 118, 133

Krogh, A. and Hertz, J. A. A Simple Weight Decay Can Improve Generalization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 1991. 94

Kullback, S. and Leibler, R. A. On Information and Sufficiency. *The Annals of Mathematical Statistics*, 1951. 51

Kurakin, A., Goodfellow, I. J., and Bengio, S. Adversarial Machine Learning at Scale. In *International Conference on Learning Representations (ICLR)*, 2017. 35, 117

Laidlaw, C. and Feizi, S. Functional Adversarial Attacks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 34

Laidlaw, C., Singla, S., and Feizi, S. Perceptual Adversarial Robustness: Defense Against Unseen Threat Models. In *International Conference on Learning Representations (ICLR)*, 2021. 34

Landau, B., Smith, L. B., and Jones, S. S. The importance of shape in early lexical learning. *Cognitive Development*, 1988. 114

Lauscher, A., Lüken, T., and Glavas, G. Sustainable Modular Debiasing of Language Models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021. 133

Le, Y. and Yang, X. S. Tiny ImageNet Visual Recognition Challenge, 2015. 28, 78, 83

LeCun, Y., Boser, B., Denker, J., Henderson, D., Howard, R., Hubbard, W., and Jackel, L. Handwritten Digit Recognition with a Back-Propagation Network. In *Advances in Neural Information Processing Systems (NeurIPS)*, 1989a. 1, 19, 97

LeCun, Y., Denker, J., and Solla, S. Optimal Brain Damage. In *Advances in Neural Information Processing Systems (NeurIPS)*, 1989b. 93

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998a. 1, 20

LeCun, Y., Cortes, C., and Burges, C. J. The MNIST database of handwritten digits. Technical report, Courant Institute, Google Labs, Microsoft Research, 1998b. 27, 48, 78, 83

Lee, K., Lee, K., Lee, H., and Shin, J. A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. 36

Li, C., Gan, Z., Yang, Z., Yang, J., Li, L., Wang, L., and Gao, J. Multimodal Foundation Models: From Specialists to General-Purpose Assistants. *arXiv preprint arXiv:2309.10020*, 2023a. 133

Li, H., Kadav, A., Durdanovic, I., Samet, H., and Graf, H. P. Pruning Filters for Efficient ConvNets. *arXiv preprint arXiv:1608.08710*, 2017. 47

Li, J., Li, D., Xiong, C., and Hoi, S. BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2022. 132

Li, J., Li, D., Savarese, S., and Hoi, S. BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models. *arXiv preprint arXiv:2301.12597*, 2023b. 132

Li, Q., Shen, L., Guo, S., and Lai, Z. Wavelet Integrated CNNs for Noise-Robust Image Classification. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 76, 85, 86

Li, X., Chen, Y., Zhu, Y., Wang, S., Zhang, R., and Xue, H. ImageNet-E: Benchmarking Neural Network Robustness via Attribute Editing. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023c. 33

Li, Y., Lin, S., Zhang, B., Liu, J., Doermann, D., Wu, Y., Huang, F., and Ji, R. Exploiting Kernel Sparsity and Entropy for Interpretable CNN Compression. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 23, 47, 50

Li, Y., Yu, Q., Tan, M., Mei, J., Tang, P., Shen, W., Yuille, A., and Xie, C. Shape-Texture Debiased Neural Network Training. In *International Conference on Learning Representations (ICLR)*, 2021. 95, 113, 114, 119, 132, 149, 164, 165, 166

Li, Z., Evtimov, I., Gordo, A., Hazirbas, C., Hassner, T., Ferrer, C. C., Xu, C., and Ibrahim, M. A Whac-a-Mole Dilemma: Shortcuts Come in Multiples Where Mitigating One Amplifies Others. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023d. 6, 28, 37

Liang, S., Liang, J., Pang, T., Du, C., Liu, A., Zhu, M., Cao, X., and Tao, D. Revisiting Backdoor Attacks against Large Vision-Language Models from Domain Shift. *arXiv preprint arXiv:2406.18844*, 2024. 155

Lim, S., Kim, I., Kim, T., Kim, C., and Kim, S. Fast AutoAugment. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 95, 164

Lin, B., Tang, Z., Ye, Y., Cui, J., Zhu, B., Jin, P., Huang, J., Zhang, J., Ning, M., and Yuan, L. MoE-LLaVA: Mixture of Experts for Large Vision-Language Models. *arXiv preprint arXiv:2401.15947*, 2024. 137, 139, 141, 167, 168

Lin, M., Chen, Q., and Yan, S. Network In Network. In *International Conference on Learning Representations (ICLR)*, 2014. 20

Lin, S., Ji, R., Li, Y., Wu, Y., Huang, F., and Zhang, B. Accelerating Convolutional Networks via Global & Dynamic Filter Pruning. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2018. 47

Lin, S., Ji, R., Yan, C., Zhang, B., Cao, L., Ye, Q., Huang, F., and Doermann, D. Towards Optimal Structured CNN Pruning via Generative Adversarial Learning. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 47

Linse, C., Barth, E., and Martinetz, T. Convolutional Neural Networks Do Work with Pre-Defined Filters. In *International Joint Conference on Neural Networks (IJCNN)*, 2023. 152

Linse, C., Brückner, B., and Martinetz, T. Enhancing Generalization in Convolutional Neural Networks Through Regularization with Edge and Line Features. In *Artificial Neural Networks and Machine Learning (ICANN)*, 2024. 152

Lipton, Z., Wang, Y.-X., and Smola, A. Detecting and Correcting for Label Shift with Black Box Predictors. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018. 31

Liu, C., Dong, Y., Xiang, W., Yang, X., Su, H., Zhu, J., Chen, Y., He, Y., Xue, H., and Zheng, S. A Comprehensive Study on Robustness of Image Classification Models: Benchmarking and Rethinking. *arXiv preprint arXiv:2302.14301*, 2023a. 117

Liu, H., Li, C., Li, Y., and Lee, Y. J. Improved Baselines with Visual Instruction Tuning. *arXiv preprint arXiv:2310.03744*, 2023b. 137, 139, 145, 167, 168

Liu, H., Li, C., Wu, Q., and Lee, Y. J. Visual Instruction Tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023c. 132, 137, 138

Liu, H., Li, C., Li, Y., Li, B., Zhang, Y., Shen, S., and Lee, Y. J. LLaVA-NeXT: Improved reasoning, OCR, and world knowledge. `https://llava-vl.github.io/blog/2024-01-30-llava-next/`, 2024. 6, 137, 139, 141, 143, 145, 167, 168

Liu, J. and Yang, X. WTT: combining wavelet transform with transformer for remote sensing image super-resolution. *Machine Vision and Applications*, 2025. 154

Liu, N., Li, S., Du, Y., Tenenbaum, J., and Torralba, A. Learning to Compose Visual Relations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021a. 133

Liu, S., Chen, T., Chen, X., Chen, X., Xiao, Q., Wu, B., Kärkkäinen, T., Pechenizkiy, M., Mocanu, D. C., and Wang, Z. More ConvNets in the 2020s: Scaling up Kernels Beyond 51x51 using Sparsity. In *International Conference on Learning Representations (ICLR)*, 2023d. 22

Liu, Y., Chen, X., Liu, C., and Song, D. Delving into Transferable Adversarial Examples and Black-box Attacks. In *International Conference on Learning Representations (ICLR)*, 2017. 34

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021b. 23, 28

Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., and Xie, S. A ConvNet for the 2020s. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 16, 18, 21, 22, 28, 76, 80, 83, 84

LLaVA Team. LLaVA/docs/Evaluation.md at main · haotian-liu/LLaVA. `https://github.com/haotian-liu/LLaVA/blob/main/docs/Evaluation.md`, 2024. [Online; accessed 6. Mar. 2024. 145

LLMRails. llmrails/ember-v1 · Hugging Face. `https://huggingface.co/llmrails/ember-v1`, 2024. [Online; accessed 27. Feb. 2024]. 135

Lo, S. and Hang, H. Exploring Semantic Segmentation on the DCT Representation. In *ACM Multimedia Asia (MMAsia)*, 2019. 79

Lonnqvist, B., Machiraju, H., and Herzog, M. H. A comment on Guo et al. [arXiv:2206.11228]. *arXiv preprint arXiv:2208.01456*, 2022. 153

Lopes, R. G., Yin, D., Poole, B., Gilmer, J., and Cubuk, E. D. Improving Robustness Without Sacrificing Accuracy with Patch Gaussian Augmentation. *arXiv preprint arXiv:1906.02611*, 2020. 33, 76, 79, 85, 86, 92, 110, 113, 115

Loshchilov, I. and Hutter, F. SGDR: Stochastic Gradient Descent with Warm Restarts. In *International Conference on Learning Representations (ICLR)*, 2017. 74

Loshchilov, I. and Hutter, F. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations (ICLR)*, 2019. 29, 84

Lu, L., Shin, Y., Su, Y., and Karniadakis, G. E. Dying ReLU and Initialization: Theory and Numerical Examples. *Communications in Computational Physics*, 2020. 15

Lukasik, J. *Topology Learning for Prediction, Generation, and Robustness in Neural Architecture Search*. PhD thesis, University of Mannheim, 2023. 9

Lukasik, J., Gavrikov, P., Keuper, J., and Keuper, M. Improving Native CNN Robustness with Filter Frequency Regularization. *Transactions on Machine Learning Research (TMLR)*, 2023. 8, 36, 76, 114, 132

Luo, G., Zhou, Y., Ren, T., Chen, S., Sun, X., and Ji, R. Cheap and Quick: Efficient Vision-Language Instruction Tuning for Large Language Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 132

Luo, J.-H., Wu, J., and Lin, W. ThiNet: A Filter Level Pruning Method for Deep Neural Network Compression. *arXiv preprint arXiv:1707.06342*, 2017. 47

Ma, X., Li, B., Wang, Y., Erfani, S. M., Wijewickrema, S., Schoenebeck, G., Houle, M. E., Song, D., and Bailey, J. Characterizing Adversarial Subspaces Using Local Intrinsic Dimensionality. In *International Conference on Learning Representations (ICLR)*, 2018. 36

Ma, X., Niu, Y., Gu, L., Wang, Y., Zhao, Y., Bailey, J., and Lu, F. Understanding adversarial attacks on deep learning based medical image analysis systems. *Pattern Recognition*, 2021. 65, 157

Ma, X., Ni, Z., and Chen, X. TinyViM: Frequency Decoupling for Tiny Hybrid Vision Mamba. *arXiv preprint arXiv:2411.17473*, 2024. 154

Maas, A. L., Hannun, A. Y., and Ng, A. Y. Rectifier nonlinearities improve neural network acoustic models. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2013. 15

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards Deep Learning Models Resistant to Adversarial Attacks. In *International Conference on Learning Representations (ICLR)*, 2018. 3, 4, 5, 30, 34, 35, 36, 44, 54, 65, 76, 81, 92, 94, 95, 97, 98, 114, 115, 117, 119, 151, 164, 165, 166

Maiya, S. R., Ehrlich, M., Agarwal, V., Lim, S., Goldstein, T., and Shrivastava, A. A Frequency Perspective of Adversarial Robustness. *arXiv preprint arXiv:2111.00861*, 2021. 78, 86, 110

Majkowska, A., Mittal, S., Steiner, D. F., Reicher, J. J., McKinney, S. M., Duggan, G. E., Eswaran, K., Cameron Chen, P.-H., Liu, Y., Kalidindi, S. R., Ding, A., Corrado, G. S., Tse, D., and Shetty, S. Chest Radiograph Interpretation with Deep Learning Models: Assessment with Radiologist-adjudicated Reference Standards and Population-adjusted Evaluation. *Radiology*, 2020. 62

Mangal, R., Leino, K., Wang, Z., Hu, K., Yu, W., Pasareanu, C., Datta, A., and Fredrikson, M. Is Certifying ℓ_p Robustness Still Worthwhile? *arXiv preprint arXiv:2310.09361*, 2023. 153

Mania, H. and Sra, S. Why do classifier accuracies show linear trends under distribution shift? *arXiv preprint arXiv:2012.15483*, 2021. 117

Marcus, D. S., Wang, T. H., Parker, J., Csernansky, J. G., Morris, J. C., and Buckner, R. L. Open Access Series of Imaging Studies (OASIS): cross-sectional MRI data in young, middle aged, nondemented, and demented older adults. *Journal of Cognitive Neuroscience*, 2007. 61

Meade, N., Poole-Dayan, E., and Reddy, S. An Empirical Survey of the Effectiveness of Debiasing Techniques for Pre-trained Language Models. In *Proceedings of the Annual Meeting of the Association for Computational Linguistic (ACL)*, 2022. 133

Mehrer, J., Spoerer, C. J., Kriegeskorte, N., and Kietzmann, T. C. Individual differences among deep neural network models. *Nature Communications*, 2020. 4

Meng, R., Liu, Y., Joty, S. R., Xiong, C., Zhou, Y., and Yavuz, S. SFR-Embedded-Mistral. Salesforce AI Research Blog, 2024. 147, 148

Michel, P., Levy, O., and Neubig, G. Are Sixteen Heads Really Better than One? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 156

Micikevicius, P., Narang, S., Alben, J., Diamos, G., Elsen, E., Garcia, D., Ginsburg, B., Houston, M., Kuchaiev, O., Venkatesh, G., and Wu, H. Mixed Precision Training. In *International Conference on Learning Representations (ICLR)*, 2018. 48

Miller, J. P., Taori, R., Raghunathan, A., Sagawa, S., Koh, P. W., Shankar, V., Liang, P., Carmon, Y., and Schmidt, L. Accuracy on the Line: on the Strong Correlation Between Out-of-Distribution and In-Distribution Generalization. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021. 28, 117, 128

Mintun, E., Kirillov, A., and Xie, S. On Interaction Between Augmentations and Corruptions in Natural Corruption Robustness. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 28, 32, 33, 118

Mirza, M. J., Zhao, M., Mao, Z., Doveh, S., Lin, W., Gavrikov, P., Dorkenwald, M., Yang, S., Jha, S., Wakaki, H., Mitsufuji, Y., Possegger, H., Feris, R., Karlinsky, L., and Glass, J. GLOV: Guided Large Language Models as Implicit Optimizers for Vision Language Models. *arXiv preprint arXiv:2410.06154*, 2024. 10, 26, 156

Moayeri, M., Banihashem, K., and Feizi, S. Explicit Tradeoffs between Adversarial and Natural Distributional Robustness. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 36

Modas, A., Rade, R., Ortiz-Jiménez, G., Moosavi-Dezfooli, S.-M., and Frossard, P. PRIME: A Few Primitives Can Boost Robustness to Common Corruptions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 33, 95, 117, 119, 164, 165, 166

Mooney, C. M. Age in the development of closure ability in children. *Canadian Journal of Psychology / Revue canadienne de psychologie*, 1957. 154

Mordvintsev, A., Olah, C., and Tyka, M. Inceptionism: Going Deeper into Neural Networks. `https://research.google/blog/inceptionism-going-deeper-into-neural-networks`, 2015. [Online; accessed 15. Oct. 2024]. 39

Moreno-Torres, J. G., Raeder, T., Alaiz-Rodríguez, R., Chawla, N. V., and Herrera, F. A unifying view on dataset shift in classification. *Pattern Recognition*, 2012. 30

Muennighoff, N., Tazi, N., Magne, L., and Reimers, N. MTEB: Massive Text Embedding Benchmark. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2023. 147

Müller, P., Braun, A., and Keuper, M. Classification Robustness to Common Optical Aberrations. In *Proceedings of the International Conference on Computer Vision Workshops (ICCVW)*, 2023. 95, 119, 164, 165, 166

Mummadi, C. K., Subramaniam, R., Hutmacher, R., Vitay, J., Fischer, V., and Metzen, J. H. Does enhanced shape bias improve neural network robustness to common corruptions? In *International Conference on Learning Representations (ICLR)*, 2021. 114, 121

Myrtle AI Team. How to train your ResNet 4: Architecture. `https://myrtle.ai/learn/how-to-train-your-resnet-4-architecture/`, 2019. [Online; accessed 22. June 2019]. 56, 83

Nair, V. and Hinton, G. E. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2010. 15

Naseer, M. M., Ranasinghe, K., Khan, S. H., Hayat, M., Shahbaz Khan, F., and Yang, M.-H. Intriguing Properties of Vision Transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 24, 114, 131, 132, 141

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading Digits in Natural Images with Unsupervised Feature Learning. *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011. 27, 78, 83

Neuhaus, Y., Augustin, M., Boreiko, V., and Hein, M. Spurious Features Everywhere - Large-Scale Detection of Harmful Spurious Features in ImageNet. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2023. 38

Nguyen, A., Yosinski, J., and Clune, J. Multifaceted Feature Visualization: Uncovering the Different Types of Features Learned By Each Neuron in Deep Neural Networks. *arXiv preprint arXiv:1602.03616*, 2016. 39

Nie, W., Guo, B., Huang, Y., Xiao, C., Vahdat, A., and Anandkumar, A. Diffusion Models for Adversarial Purification. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2022. 36

Olah, C., Mordvintsev, A., and Schubert, L. Feature Visualization. *Distill*, 2017. https://distill.pub/2017/feature-visualization. 22, 39

Olah, C., Cammarata, N., Schubert, L., Goh, G., Petrov, M., and Carter, S. An Overview of Early Vision in InceptionV1. *Distill*, 2020a. 22, 47

Olah, C., Cammarata, N., Schubert, L., Goh, G., Petrov, M., and Carter, S. Zoom In: An Introduction to Circuits. *Distill*, 2020b. 4, 22, 39, 47

Olah, C., Cammarata, N., Voss, C., Schubert, L., and Goh, G. Naturally Occurring Equivariance in Neural Networks. *Distill*, 2020c. 22, 47

Oliva, A., Torralba, A., and Schyns, P. G. Hybrid images. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference (SIGGRAPH)*, 2006. 40, 144

Olshausen, B. A. and Field, D. J. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Research*, 1997. 22, 97

OpenAI. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774*, 2023. 131, 132, 137, 138, 148, 167, 168

OpenAI. GPT-4. `https://openai.com/research/gpt-4`, 2024. [Online; accessed 6. Mar. 2024]. 1

Oquab, M., Bottou, L., Laptev, I., and Sivic, J. Learning and Transferring Mid-level Image Representations Using Convolutional Neural Networks. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 45

Oquab, M., Darcet, T., Moutakanni, T., Vo, H. V., Szafraniec, M., Khalidov, V., Fernandez, P., HAZIZA, D., Massa, F., El-Nouby, A., Assran, M., Ballas, N., Galuba, W., Howes, R., Huang, P.-Y., Li, S.-W., Misra, I., Rabbat, M., Sharma, V., Synnaeve, G., Xu, H., Jegou, H., Mairal, J., Labatut, P., Joulin, A., and Bojanowski, P. DINOv2: Learning Robust Visual Features without Supervision. *Transactions on Machine Learning Research (TMLR)*, 2024. 23, 97, 155

Ortiz-Jiménez, G., Modas, A., Moosavi-Dezfooli, S., and Frossard, P. Hold me tight! Influence of discriminative features on deep network boundaries. In *Chinese Control Conference (CCC)*, 2020. 78, 86

Ourghi, F., Amamra, A., and Azri, Y. Layer-Wise Compression Analysis of CNN Architectures: Insights from VGG16. In *Advances in Computing Systems and Applications*, 2024. 152

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P. F., Leike, J., and Lowe, R. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 132

Pan, S. J. and Yang, Q. A Survey on Transfer Learning. In *IEEE Transactions on Knowledge and Data Engineering*, 2010. 45, 47

Pang, T., Xu, K., Dong, Y., Du, C., Chen, N., and Zhu, J. Rethinking Softmax Cross-Entropy Loss for Adversarial Robustness. In *International Conference on Learning Representations (ICLR)*, 2020a. 36

Pang, T., Yang, X., Dong, Y., Xu, K., Zhu, J., and Su, H. Boosting Adversarial Training with Hypersphere Embedding. *arXiv preprint arXiv:2002.08619*, 2020b. 67, 161

Papernot, N., McDaniel, P., Wu, X., Jha, S., and Swami, A. Distillation as a Defense to Adversarial Perturbations against Deep Neural Networks. *arXiv preprint arXiv:1511.04508*, 2015. 65

Papernot, N., McDaniel, P., and Goodfellow, I. Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples. *arXiv preprint arXiv:1605.07277*, 2016. 35

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 19, 48, 95, 120, 164, 165, 166

Peng, S., Xu, W., Cornelius, C., Hull, M., Li, K., Duggal, R., Phute, M., Martin, J., and Chau, D. H. Robust Principles: Architectural Design Principles for Adversarially Robust CNNs. *arXiv preprint arXiv:2308.16258*, 2023. 4

Petrov, M., Voss, C., Schubert, L., Cammarata, N., Goh, G., and Olah, C. Weight Banding. *Distill*, 2021. 22, 47

Picard, D. Torch.manual_seed(3407) is all you need: On the influence of random seeds in deep learning architectures for computer vision. *arXiv preprint arXiv:2109.08203*, 2023. 75

Pinsky, M. A. *Introduction to Fourier Analysis and Wavelets.* American Mathematical Society, 2008. 13

Quiñonero-Candela, J., Sugiyama, M., Schwaighofer, A., and Lawrence, N. D. *Dataset Shift in Machine Learning.* The MIT Press, 2008. 30

Qwen Team. Introducing Qwen-VL. `https://qwenlm.github.io/blog/qwen-vl/`, 2024. [Accessed: 12 Feb. 2024]. 132, 137, 167, 168

Rade, R. and Moosavi-Dezfooli, S.-M. Helper-based Adversarial Training: Reducing Excessive Margin to Achieve a Better Accuracy vs. Robustness Trade-off. In *ICML Workshop on Adversarial Machine Learning*, 2021. 67, 162, 163

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. Learning Transferable Visual Models From Natural Language Supervision. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021. 1, 25, 26, 105, 132, 133, 135, 137, 139, 146

Radford, B. The Ten-Percent Myth. *Skeptical Inquirer*, 1999. 93

Raghunathan, A., Xie, S. M., Yang, F., Duchi, J., and Liang, P. Adversarial Training Can Hurt Generalization. In *ICML 2019 Workshop on Identifying and Understanding Deep Learning Phenomena*, 2019. 117

Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F., Bengio, Y., and Courville, A. On the Spectral Bias of Neural Networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019. 81

Raji, I. D. and Buolamwini, J. Actionable Auditing: Investigating the Impact of Publicly Naming Biased Performance Results of Commercial AI Products. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society (AIES)*, 2019. 131, 133

Rauber, J., Brendel, W., and Bethge, M. Foolbox: A Python toolbox to benchmark the robustness of machine learning models. In *ICML Workshop on Reliable Machine Learning in the Wild*, 2017. 84

Razavian, A. S., Azizpour, H., Sullivan, J., and Carlsson, S. CNN features off-the-shelf: An astounding baseline for recognition. In *Proceedings of the Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2014. 45

Rebuffi, S.-A., Gowal, S., Calian, D. A., Stimberg, F., Wiles, O., and Mann, T. Data Augmentation Can Improve Robustness. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021a. 36

Rebuffi, S.-A., Gowal, S., Calian, D. A., Stimberg, F., Wiles, O., and Mann, T. Fixing Data Augmentation to Improve Adversarial Robustness. *arXiv preprint arXiv:2103.01946*, 2021b. 67, 162, 163

Recht, B., Roelofs, R., Schmidt, L., and Shankar, V. Do ImageNet Classifiers Generalize to ImageNet? In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019. 28, 32, 98, 118, 128

Reed, S., Zolna, K., Parisotto, E., Colmenarejo, S. G., Novikov, A., Barth-maron, G., Giménez, M., Sulsky, Y., Kay, J., Springenberg, J. T., Eccles, T., Bruce, J., Razavi, A., Edwards, A., Heess, N., Chen, Y., Hadsell, R., Vinyals, O., Bordbar, M., and de Freitas, N. A Generalist Agent. *Transactions on Machine Learning Research (TMLR)*, 2022. 1

Ribeiro, M. T., Singh, S., and Guestrin, C. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016. 155

Rice, L., Wong, E., and Kolter, Z. Overfitting in adversarially robust deep learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2020. 4, 36, 67, 76, 92, 161, 163

Ridnik, T., Ben-Baruch, E., Noy, A., and Zelnik, L. ImageNet-21K Pretraining for the Masses. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 28

Rosenblatt, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 1958. 1, 14

Rosenfeld, A., Zemel, R., and Tsotsos, J. K. The Elephant in the Room. *arXiv preprint arXiv:1808.03305*, 2018. 2, 5, 6, 41

Roth, K., Kim, J. M., Koepke, A. S., Vinyals, O., Schmid, C., and Akata, Z. Waffling Around for Performance: Visual Classification with Random Words and Broad Concepts. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2023. 26

Ruderman, D. L. The statistics of natural images. *Network: Computation In Neural Systems*, 1994. 14, 79, 115

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. Learning representations by back-propagating errors. *Nature*, 1986. 1, 16

Rusak, E., Schott, L., Zimmermann, R. S., Bitterwolf, J., Bringmann, O., Bethge, M., and Brendel, W. A Simple Way to Make Neural Networks Robust Against Diverse Image Corruptions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 33

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 2015. 2, 21, 28, 32, 63, 66, 67, 78, 94, 95, 96, 103, 119

Sagawa, S., Koh, P. W., Hashimoto, T. B., and Liang, P. Distributionally Robust Neural Networks. In *International Conference on Learning Representations (ICLR)*, 2020. 41

Saikia, T., Schmid, C., and Brox, T. Improving robustness against common corruptions with frequency biased models. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021. 76, 79, 80, 90, 109, 110, 113, 115

Salman, H., Ilyas, A., Engstrom, L., Kapoor, A., and Madry, A. Do Adversarially Robust ImageNet Models Transfer Better? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 5, 67, 95, 100, 105, 119, 129, 163, 164, 165, 166

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 48

Scherrer, N., Shi, C., Feder, A., and Blei, D. Evaluating the Moral Beliefs Encoded in LLMs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 133

Schubert, L., Voss, C., Cammarata, N., Goh, G., and Olah, C. High-Low Frequency Detectors. *Distill*, 2021. 22, 47

Schürholt, K., Knyazev, B., Giró-i Nieto, X., and Borth, D. Hyper-Representations as Generative Models: Sampling Unseen Neural Network Weights. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022a. 152

Schürholt, K., Taskiran, D., Knyazev, B., Giró-i Nieto, X., and Borth, D. Model Zoos: A Dataset of Diverse Populations of Neural Network Models. In *Advances in Neural Information Processing Systems*, 2022b. 152

Schürholt, K. *Hyper-Representations: Learning from Populations of Neural Networks.* PhD thesis, University of St. Gallen, 2024. 152

Sehwag, V., Wang, S., Mittal, P., and Jana, S. HYDRA: Pruning Adversarially Robust Neural Networks. *arXiv preprint arXiv:2002.10509*, 2020. 67, 161

Sehwag, V., Mahloujifar, S., Handina, T., Dai, S., Xiang, C., Chiang, M., and Mittal, P. Robust Learning Meets Generative Models: Can Proxy Distributions Improve Adversarial Robustness? *arXiv preprint arXiv:2104.09425*, 2021. 67, 162

Serrano, S. and Smith, N. A. Is Attention Interpretable? In *Proceedings of the Annual Meeting of the Association for Computational Linguistic (ACL)*, 2019. 38

Shafahi, A., Najibi, M., Ghiasi, A., Xu, Z., Dickerson, J., Studer, C., Davis, L. S., Taylor, G., and Goldstein, T. Adversarial Training for Free! *arXiv preprint arXiv:1904.12843*, 2019. 65

Shankar, V., Roelofs, R., Mania, H., Fang, A., Recht, B., and Schmidt, L. Evaluating Machine Accuracy on ImageNet. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2020. 28

Shankar, V., Dave, A., Roelofs, R., Ramanan, D., Recht, B., and Schmidt, L. Do Image Classifiers Generalize Across Time? In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021. 28

Shannon, C. E. A mathematical theory of communication. *The Bell System Technical Journal*, 1948. 25, 50

Shi, B., Zhang, D., Dai, Q., Zhu, Z., Mu, Y., and Wang, J. Informative Dropout for Robust Representation Learning: A Shape-bias Perspective. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2020. 114, 131, 141

Shi, C., Holtz, C., and Mishne, G. Online Adversarial Purification based on Self-supervised Learning. In *International Conference on Learning Representations (ICLR)*, 2021. 36

Shih, G., Wu, C. C., Halabi, S. S., Kohli, M. D., Prevedello, L. M., Cook, T. S., Sharma, A., Amorosa, J. K., Arteaga, V., Galperin-Aizenberg, M., Gill, R. R., Godoy, M. C. B., Hobbs, S., Jeudy, J., Laroia, A., Shah, P. N., Vummidi, D., Yaddanapudi, K., and Stein, A. Augmenting the National Institutes of Health Chest Radiograph Dataset with Expert Annotations of Possible Pneumonia. *Radiology: Artificial Intelligence*, 2019. 62

Shimodaira, H. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 2000. 31

Shrikumar, A., Greenside, P., and Kundaje, A. Learning important features through propagating activation differences. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2017. 38

Sietsma, J. and Dow, R. J. Creating artificial neural networks that generalize. *Neural Networks*, 1991. 30

Simonyan, K. and Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *International Conference on Learning Representations (ICLR)*, 2015. 2, 20, 25, 48, 59

Simonyan, K., Vedaldi, A., and Zisserman, A. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. In *International Conference on Learning Representations (ICLR)*, 2014. 19, 38

Singh, N. C. and Theunissen, F. Modulation spectra of natural sounds and ethological theories of auditory processing. *The Journal of the Acoustical Society of America*, 2004. 79, 115

Singh, N. D., Croce, F., and Hein, M. Revisiting Adversarial Training for ImageNet: Architectures, Training and Generalization across Threat Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 24

Singla, S. and Feizi, S. Salient ImageNet: How to discover spurious features in Deep Learning? In *International Conference on Learning Representations (ICLR)*, 2022. 38

Sinha, S., Garg, A., and Larochelle, H. Curriculum By Smoothing. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 110

Sitawarin, C., Chakraborty, S., and Wagner, D. SAT: Improving Adversarial Training via Curriculum-Based Loss Smoothing. *arXiv preprint arXiv:2003.09347*, 2021. 67, 162, 163

Sivaprasad, S., Kaushik, P., Abdelnabi, S., and Fritz, M. Exploring Value Biases: How LLMs Deviate Towards the Ideal. *arXiv preprint arXiv:2402.11005*, 2024. 6, 133

Slack, D., Hilgard, S., Jia, E., Singh, S., and Lakkaraju, H. Fooling LIME and SHAP: Adversarial Attacks on Post hoc Explanation Methods. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, 2020. 155

Smilkov, D., Thorat, N., Kim, B., Viégas, F., and Wattenberg, M. SmoothGrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017. 88, 89

Sridhar, K., Sokolsky, O., Lee, I., and Weimer, J. Improving Neural Network Robustness via Persistency of Excitation. *arXiv preprint arXiv:2106.02078*, 2021. 67, 162

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research (JMLR)*, 2014. 20, 25, 30

Srivastava, S. and Sharma, G. OmniVec: Learning robust representations with cross modal sharing. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2024. 2

Storkey, A. *When Training and Test Sets Are Different: Characterizing Learning Transfer*, pp. 3–28. The MIT Press, 2008. 30

Stutz, D., Hein, M., and Schiele, B. Disentangling Adversarial Robustness and Generalization. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 117

Su, J., Vargas, D. V., and Sakurai, K. One Pixel Attack for Fooling Deep Neural Networks. *IEEE Transactions on Evolutionary Computation*, 2019. 34

Subramanian, A., Sizikova, E., Majaj, N., and Pelli, D. Spatial-frequency channels, shape bias, and adversarial robustness. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 6, 38, 40, 41, 102, 112, 113, 116, 117, 123, 124, 125, 131, 144, 154, 156

Sucholutsky, I., Muttenthaler, L., Weller, A., Peng, A., Bobu, A., Kim, B., Love, B. C., Cueva, C. J., Grant, E., Groen, I., Achterberg, J., Tenenbaum, J. B., Collins, K. M., Hermann, K. L., Oktar, K., Greff, K., Hebart, M. N., Cloos, N., Kriegeskorte, N., Jacoby, N., Zhang, Q., Marjieh, R., Geirhos, R., Chen, S., Kornblith, S., Rane, S., Konkle, T., O'Connell, T. P., Unterthiner, T., Lampinen, A. K., Müller, K.-R., Toneva, M., and Griffiths, T. L. Getting aligned on representational alignment. *arXiv preprint arXiv:2310.13018*, 2024. 4

Sun, B. and Saenko, K. Deep CORAL: Correlation Alignment for Deep Domain Adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016. 33

Sun, C., Shrivastava, A., Singh, S., and Gupta, A. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017. 45

Sun, Q., Cui, Y., Zhang, X., Zhang, F., Yu, Q., Luo, Z., Wang, Y., Rao, Y., Liu, J., Huang, T., and Wang, X. Generative Multimodal Models are In-Context Learners. *arXiv preprint arXiv:2312.13286*, 2023a. 137, 139, 167, 168

Sun, Q., Fang, Y., Wu, L., Wang, X., and Cao, Y. EVA-CLIP: Improved Training Techniques for CLIP at Scale. *arXiv preprint arXiv:2303.15389*, 2023b. 132, 139, 146

Sun, Q., Wang, J., Yu, Q., Cui, Y., Zhang, F., Zhang, X., and Wang, X. EVA-CLIP-18B: Scaling CLIP to 18 Billion Parameters. *arXiv preprint arXiv:2402.04252*, 2024. 146, 155

Sun, Y., Wang, X., Liu, Z., Miller, J., Efros, A., and Hardt, M. Test-Time Training with Self-Supervision for Generalization under Distribution Shifts. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2020. 33

Sun, Z., Shen, S., Cao, S., Liu, H., Li, C., Shen, Y., Gan, C., Gui, L.-Y., Wang, Y.-X., Yang, Y., Keutzer, K., and Darrell, T. Aligning Large Multimodal Models with Factually Augmented RLHF. *arXiv preprint arXiv:2309.14525*, 2023c. 137, 138, 167, 168

Suresh, J., Nayak, N., and Kalyani, S. First line of defense: A robust first layer mitigates adversarial attacks. *arXiv preprint arXiv:2408.11680*, 2024. 153

Sutskever, I., Vinyals, O., and Le, Q. V. Sequence to Sequence Learning with Neural Networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014. 1

Sutton, R. The Bitter Lesson. `http://www.incompleteideas.net/IncIdeas/BitterLesson.html`, 2019. [Online; accessed 30. Nov. 2024]. 2, 30

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. Going Deeper with Convolutions. *arXiv preprint arXiv:1409.4842*, 2014a. 20

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. In *International Conference on Learning Representations (ICLR)*, 2014b. 2, 30, 34, 65, 113, 115

Szegedy, C., Ioffe, S., Vanhoucke, V., and Alemi, A. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. *arXiv preprint arXiv:1602.07261*, 2016a. 22, 47, 48, 117, 119

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the Inception Architecture for Computer Vision. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016b. 20, 48

Szeliski, R. *Computer Vision: Algorithms and Applications.* Springer Cham, 2022. 13, 14

Tan, M. and Le, Q. EfficientNetV2: Smaller Models and Faster Training. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021. 21, 59

Tan, M. and Le, Q. V. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *arXiv preprint arXiv:1905.11946*, 2020. 18, 21, 76, 80, 83

Tang, S., Gong, R., Wang, Y., Liu, A., Wang, J., Chen, X., Yu, F., Liu, X., Song, D., Yuille, A., Torr, P. H. S., and Tao, D. RobustART: Benchmarking Robustness on Architecture Design and Training Techniques. *arXiv preprint arXiv:2109.05211*, 2021. 117

Taori, R., Dave, A., Shankar, V., Carlini, N., Recht, B., and Schmidt, L. Measuring Robustness to Natural Distribution Shifts in Image Classification. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 113, 117

Tayyab, M. and Mahalanobis, A. BasisConv: A method for compressed representation and learning in CNNs. *arXiv preprint arXiv:1906.04509*, 2019. 47

Teknium, theemozilla, karan4d, and huemin_art. Nous Hermes 2 Mistral 7B DPO. `https://huggingface.co/NousResearch/Nous-Hermes-2-Mistral-7B-DPO`, 2024. [Online; accessed 27. Feb. 2024]. 135

Teney, D., Lin, Y., Oh, S. J., and Abbasnejad, E. ID and OOD Performance Are Sometimes Inversely Correlated on Real-world Datasets. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 128

Thomee, B., Shamma, D. A., Friedland, G., Elizalde, B., Ni, K., Poland, D., Borth, D., and Li, L.-J. YFCC100M: The New Data in Multimedia Research. *Communications of the ACM*, 2016. 127

Thrush, T., Jiang, R., Bartolo, M., Singh, A., Williams, A., Kiela, D., and Ross, C. Winoground: Probing Vision and Language Models for Visio-Linguistic Compositionality. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 133

Tokozume, Y., Ushiku, Y., and Harada, T. Between-Class Learning for Image Classification. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 5, 30, 33

Tomsett, R., Harborne, D., Chakraborty, S., Gurram, P., and Preece, A. Sanity Checks for Saliency Metrics. *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2020. 38, 40

Torralba, A., Fergus, R., and Freeman, W. T. 80 Million Tiny Images: A Large Data Set for Nonparametric Object and Scene Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008. 28

Tramèr, F. and Boneh, D. Adversarial Training and Robustness for Multiple Perturbations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 76

Trockman, A. and Kolter, J. Z. Patches Are All You Need? *Transactions on Machine Learning Research (TMLR)*, 2023. 21, 22, 23, 103

Trockman, A., Willmott, D., and Kolter, J. Z. Understanding the Covariance Structure of Convolutional Filters. In *International Conference on Learning Representations (ICLR)*, 2023. 23

Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., and Madry, A. Robustness May Be at Odds with Accuracy. In *International Conference on Learning Representations (ICLR)*, 2019. 5, 36, 117

Tsymbal, A. The Problem of Concept Drift: Definitions and Related Work. Technical report, Trinity College Dublin, 2004. 31

Tuli, S., Dasgupta, I., Grant, E., and Griffiths, T. L. Are Convolutional Neural Networks or Transformers more like human vision? *arXiv preprint arXiv:2105.07197*, 2021. 132

Tzeng, E., Hoffman, J., Saenko, K., and Darrell, T. Adversarial Discriminative Domain Adaptation. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 33

Uesato, J., O'Donoghue, B., Kohli, P., and van den Oord, A. Adversarial Risk and the Dangers of Evaluating Against Weak Attacks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018. 36

Ulicny, M., Krylov, V. A., and Dahyot, R. Harmonic convolutional networks based on discrete cosine transform. *Pattern Recognition*, 2022. 79, 81

Unterthiner, T., Keysers, D., Gelly, S., Bousquet, O., and Tolstikhin, I. Predicting Neural Network Accuracy from Weights. *arXiv preprint arXiv:2002.11448*, 2021. 46

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is All you Need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 1, 23

Veerabadran, V., Goldman, J., Shankar, S., Cheung, B., Papernot, N., Kurakin, A., Goodfellow, I., Shlens, J., Sohl-Dickstein, J., Mozer, M. C., and Elsayed, G. F. Subtle adversarial image manipulations influence both human and machine perception. *Nature Communications*, 2023. 2

Veit, A., Wilber, M. J., and Belongie, S. Residual Networks Behave Like Ensembles of Relatively Shallow Networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016. 93

Villalobos, P., Ho, A., Sevilla, J., Besiroglu, T., Heim, L., and Hobbhahn, M. Will we run out of data? Limits of LLM scaling based on human-generated data. *arXiv preprint arXiv:2211.04325*, 2024. 2

Vinyals, O., Toshev, A., Bengio, S., and Erhan, D. Show and Tell: A Neural Image Caption Generator. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 133

Voita, E., Talbot, D., Moiseev, F., Sennrich, R., and Titov, I. Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned. In *Proceedings of the Annual Meeting of the Association for Computational Linguistic (ACL)*, 2019. 156

Voss, C., Cammarata, N., Goh, G., Petrov, M., Schubert, L., Egan, B., Lim, S. K., and Olah, C. Visualizing Weights. *Distill*, 2021a. 22, 47

Voss, C., Goh, G., Cammarata, N., Petrov, M., Schubert, L., and Olah, C. Branch Specialization. *Distill*, 2021b. 22, 47

Vries, S. d. The Effect of Shape Selective Kernels in Convolutional Neural Networks . Master's thesis, University of Groningen, 2024. 153

Vryniotis, V. How to Train State-Of-The-Art Models Using TorchVision's Latest Primitives. `https://pytorch.org/blog/how-to-train-state-of-the-art-models-using-torchvision-latest-primitives`, 2023. [Online; accessed 15. Nov. 2023]. 95, 98, 120, 164, 165, 166

Wallace, G. The JPEG still picture compression standard. *IEEE Transactions on Consumer Electronics*, 1992. 79

Wang, H., Ge, S., Lipton, Z., and Xing, E. P. Learning Robust Global Representations by Penalizing Local Predictive Power. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 2, 28, 78, 98, 104, 105, 118

Wang, H., Wu, X., Huang, Z., and Xing, E. P. High-Frequency Component Helps Explain the Generalization of Convolutional Neural Networks. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020a. 6, 36, 41, 76, 78, 80, 102, 112, 113, 115, 117, 121, 131, 144, 154

Wang, M. and Deng, W. Deep visual domain adaptation: A survey. *Neurocomputing*, 2018. 33

Wang, P., Wang, S., Lin, J., Bai, S., Zhou, X., Zhou, J., Wang, X., and Zhou, C. ONE-PEACE: Exploring One General Representation Model Toward Unlimited Modalities. *arXiv preprint arXiv:2305.11172*, 2023a. 23

Wang, S., Veldhuis, R., Brune, C., and Strisciuglio, N. What do neural networks learn in image classification? A frequency shortcut perspective. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2023b. 126, 129

Wang, W., Lv, Q., Yu, W., Hong, W., Qi, J., Wang, Y., Ji, J., Yang, Z., Zhao, L., Song, X., Xu, J., Xu, B., Li, J., Dong, Y., Ding, M., and Tang, J. CogVLM: Visual Expert for Pretrained Language Models. *arXiv preprint arXiv:2311.03079*, 2023c. 137, 167, 168

Wang, X., Peng, Y., Lu, L., Lu, Z., Bagheri, M., and Summers, R. M. ChestX-Ray8: Hospital-Scale Chest X-Ray Database and Benchmarks on Weakly-Supervised Classification and Localization of Common Thorax Diseases. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 62

Wang, Y., Zou, D., Yi, J., Bailey, J., Ma, X., and Gu, Q. Improving Adversarial Robustness Requires Revisiting Misclassified Examples. In *International Conference on Learning Representations (ICLR)*, 2020b. 67, 161

Wang, Y., Li, Y., Gong, R., Xiao, T., and Yu, F. Real World Robustness from Systematic Noise. In *Proceedings of the International Workshop on Adversarial Learning for Multimedia*, 2021. 117, 118

Wang, Z., Pang, T., Du, C., Lin, M., Liu, W., and Yan, S. Better Diffusion Models Further Improve Adversarial Training. *arXiv preprint arXiv:2302.04638*, 2023d. 36

Wenzel, F., Dittadi, A., Gehler, P., Simon-Gabriel, C.-J., Horn, M., Zietlow, D., Kernert, D., Russell, C., Brox, T., Schiele, B., Schölkopf, B., and Locatello, F. Assaying Out-Of-Distribution Generalization in Transfer Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 128

Widmer, G. and Kubat, M. Learning in the Presence of Concept Drift and Hidden Contexts. *Machine Learning*, 1996. 31

Wiegreffe, S. and Pinter, Y. Attention is not not Explanation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019. 38

Wightman, R. PyTorch Image Models. `https://github.com/rwightman/pytorch-image-models`, 2019. 48, 67, 95, 105, 120, 164, 165, 166

Wightman, R., Touvron, H., and Jegou, H. ResNet strikes back: An improved training procedure in timm. In *NeurIPS 2021 Workshop on ImageNet: Past, Present, and Future*, 2021. 3, 95, 98, 120, 121, 164, 165, 166

Wong, E., Schmidt, F., and Kolter, Z. Wasserstein Adversarial Examples via Projected Sinkhorn Iterations. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019. 34

Wong, E., Rice, L., and Kolter, J. Z. Fast is better than free: Revisiting adversarial training. In *International Conference on Learning Representations (ICLR)*, 2020. 36, 67, 161, 163

Woo, S., Debnath, S., Hu, R., Chen, X., Liu, Z., Kweon, I. S., and Xie, S. ConvNeXt V2: Co-Designing and Scaling ConvNets With Masked Autoencoders. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 22

Wood, C. Evaluating Neural Networks as Models of Human Language-Vision Feedback. Technical report, Brown University, 2024. 154

Wu, D., tao Xia, S., and Wang, Y. Adversarial Weight Perturbation Helps Robust Generalization. *arXiv preprint arXiv:2004.05884*, 2020. 67, 161, 163

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv preprint arXiv:1708.07747*, 2017. 48

Xiao, K. Y., Engstrom, L., Ilyas, A., and Madry, A. Noise or Signal: The Role of Image Backgrounds in Object Recognition. In *International Conference on Learning Representations (ICLR)*, 2021a. 2, 5, 41

Xiao, T., Singh, M., Mintun, E., Darrell, T., Dollar, P., and Girshick, R. Early Convolutions Help Transformers See Better. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021b. 24

Xie, C., Tan, M., Gong, B., Yuille, A., and Le, Q. V. Smooth Adversarial Training. *arXiv preprint arXiv:2006.14536*, 2021. 119

Xie, S., Girshick, R., Dollar, P., Tu, Z., and He, K. Aggregated Residual Transformations for Deep Neural Networks. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 18

Xu, K., Qin, M., Sun, F., Wang, Y., Chen, Y., and Ren, F. Learning in the Frequency Domain. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 79

Yalniz, I. Z., Jégou, H., Chen, K., Paluri, M., and Mahajan, D. Billion-scale semi-supervised learning for image classification. *arXiv preprint arXiv:1905.00546*, 2019. 105, 127

Yan, W., Zhang, Y., Abbeel, P., and Srinivas, A. VideoGPT: Video Generation using VQ-VAE and Transformers. *arXiv preprint arXiv:2104.10157*, 2021. 135

Yang, C., Wang, X., Lu, Y., Liu, H., Le, Q. V., Zhou, D., and Chen, X. Large Language Models as Optimizers. In *International Conference on Learning Representations (ICLR)*, 2024. 133, 134

Yaroslavsky, L. Fast Transforms in Image Processing: Compression, Restoration, and Resampling. *Advances in Electrical Engineering*, 2014. 79

Ye, W., Zheng, G., Ma, Y., Cao, X., Lai, B., Rehg, J. M., and Zhang, A. MM-SpuBench: Towards Better Understanding of Spurious Biases in Multimodal LLMs. *arXiv preprint arXiv:2406.17126*, 2024. 154

Yin, D., Gontijo Lopes, R., Shlens, J., Cubuk, E. D., and Gilmer, J. A Fourier Perspective on Model Robustness in Computer Vision. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 33, 86, 115, 121, 122

Yoon, J., Hwang, S. J., and Lee, J. Adversarial Purification with Score-based Generative Models. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021. 36

Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014. 5, 16, 22, 45, 47, 61, 70, 81, 97

Yu, C., Han, B., Shen, L., Yu, J., Gong, C., Gong, M., and Liu, T. Understanding Robust Overfitting of Adversarial Training and Beyond. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2022a. 76

Yu, J., Wang, Z., Vasudevan, V., Yeung, L., Seyedhosseini, M., and Wu, Y. CoCa: Contrastive Captioners are Image-Text Foundation Models. In *Transactions on Machine Learning Research (TMLR)*, 2022b. 132

Yu, R., Li, A., Chen, C.-F., Lai, J.-H., Morariu, V. I., Han, X., Gao, M., Lin, C.-Y., and Davis, L. S. NISP: Pruning Networks Using Neuron Importance Score Propagation. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 47

Yun, S., Han, D., Chun, S., Oh, S. J., Yoo, Y., and Choe, J. CutMix: Regularization Strategy to Train Strong Classifiers With Localizable Features. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2019. 5, 30, 33

Yun, S., Oh, S. J., Heo, B., Han, D., Choe, J., and Chun, S. Re-Labeling ImageNet: From Single to Multi-Labels, From Global to Localized Labels. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 28

Zagoruyko, S. and Komodakis, N. Wide Residual Networks. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2016. 67, 103

Zech, J. R., Badgeley, M. A., Liu, M., Costa, A. B., Titano, J. J., and Oermann, E. K. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: A cross-sectional study. *PLOS Medicine*, 2018. 38

Zeiler, M. D. and Fergus, R. Visualizing and Understanding Convolutional Networks. *arXiv preprint arXiv:1311.2901*, 2013. 16, 22, 47, 61, 70, 97, 98

Zeiler, M. D., Krishnan, D., Taylor, G. W., and Fergus, R. Deconvolutional networks. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. 22

Zhai, X., Kolesnikov, A., Houlsby, N., and Beyer, L. Scaling Vision Transformers. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022a. 2, 105, 132

Zhai, X., Wang, X., Mustafa, B., Steiner, A., Keysers, D., Kolesnikov, A., and Beyer, L. LiT: Zero-Shot Transfer with Locked-image Text Tuning. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022b. 132, 133

Zhang, A., Lipton, Z. C., Li, M., and Smola, A. J. Dive into deep learning. *arXiv preprint arXiv:2106.11342*, 2021a. 31

Zhang, C., Bengio, S., and Singer, Y. Are All Layers Created Equal? *Journal of Machine Learning Research*, 2022. 7, 93, 94, 95, 96, 152

Zhang, C., Pan, F., Kim, J., Kweon, I. S., and Mao, C. ImageNet-D: Benchmarking Neural Network Robustness on Diffusion Synthetic Object. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 33

Zhang, D., Zhang, T., Lu, Y., Zhu, Z., and Dong, B. You Only Propagate Once: Accelerating Adversarial Training via Maximal Principle. *arXiv preprint arXiv:1905.00877*, 2019a. 67, 161

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond Empirical Risk Minimization. In *International Conference on Learning Representations (ICLR)*, 2018. 5, 30, 33

Zhang, H., Yu, Y., Jiao, J., Xing, E. P., Ghaoui, L. E., and Jordan, M. I. Theoretically Principled Trade-off between Robustness and Accuracy. *arXiv preprint arXiv:1901.08573*, 2019b. 67, 161

Zhang, J., Xu, X., Han, B., Niu, G., zhen Cui, L., Sugiyama, M., and Kankanhalli, M. S. Attacks Which Do Not Kill Training Make Adversarial Learning Stronger. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2020. 36, 67, 161

Zhang, J., Zhu, J., Niu, G., Han, B., Sugiyama, M., and Kankanhalli, M. Geometry-aware Instance-reweighted Adversarial Training. *arXiv preprint arXiv:2010.01736*, 2021b. 67, 162

Zhang, K., Schölkopf, B., Muandet, K., and Wang, Z. Domain Adaptation under Target and Conditional Shift. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2013. 31

Zhang, R. Making Convolutional Networks Shift-Invariant Again. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019. 76, 85, 86

Zhang, T. and Zhu, Z. Interpreting Adversarially Trained Convolutional Neural Networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019. 5, 103, 114, 141

Zhou, K., Yang, J., Loy, C. C., and Liu, Z. Conditional Prompt Learning for Vision-Language Models. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 26

Zhou, K., Liu, Z., Qiao, Y., Xiang, T., and Loy, C. C. Domain Generalization: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2023. 33

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., and He, Q. A Comprehensive Survey on Transfer Learning. *arXiv preprint arXiv:1911.02685*, 2020. 47

Zimmermann, R. S., Borowski, J., Geirhos, R., Bethge, M., Wallis, T., and Brendel, W. How Well do Feature Visualizations Support Causal Understanding of CNN Activations? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 39

Zou, X., Xiao, F., Yu, Z., and Lee, Y. J. Delving Deeper into Anti-aliasing in ConvNets. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2020. 76, 85, 86