Spectral Bias in Practice: the Role of Function Frequency in Generalization

Anonymous Author(s)
Affiliation
Address
email

Abstract

Despite their ability to represent highly expressive functions, deep learning models seem to find simple solutions that generalize surprisingly well. Spectral bias – the tendency of neural networks to prioritize learning low frequency functions – is one possible explanation for this phenomenon, but so far spectral bias has primarily been observed in theoretical models and simplified experiments. In this work, we propose methodologies for measuring spectral bias in modern image classification networks on CIFAR-10 and ImageNet. We find that these networks indeed exhibit spectral bias, and that interventions that improve test accuracy on CIFAR-10 tend to produce learned functions that have higher frequencies overall but lower frequencies in the vicinity of examples from each class. This trend holds across variation in training time, model architecture, number of training examples, data augmentation, and self-distillation. We also explore the connections between function frequency and image frequency and find that spectral bias is sensitive to the low frequencies prevalent in natural images. On ImageNet, we find that learned function frequency also varies with internal class diversity, with higher frequencies on more diverse classes. Our work enables measuring and ultimately influencing the spectral behavior of neural networks used for image classification, and is a step towards understanding why deep models generalize well.

1 Introduction

Two fundamental questions in machine learning are why overparameterized models generalize and how to make them more robust to distribution shift and adversarial examples. Resolving both of these questions requires understanding how complex our models should be.

For instance, it is thought that overparameterized models generalize well because there are implicit regularizers that constrain the complexity of the learned functions. However, the precise nature of these implicit regularizers, and their importance in practice, remains unclear. As for how to achieve robustness, no consensus has emerged regarding function complexity. On the one hand, Cranko et al. [4] and Leino et al. [22] argue that Lipschitz smoothness of the learned function offers a guarantee of robustness; on the other, Shah et al. [35] and Madry et al. [23] argue that simplicity can be harmful and a robust model must actually be more complex than its non-robust counterpart.

One window into function complexity is spectral bias – the tendency of neural networks to learn low frequency (simple and smooth) functions first, then gradually increase the frequency (complexity) of the learned function as training proceeds. Foundational work in this area has shown theoretical evidence of spectral bias by analyzing convergence rates of neural networks towards functions of different frequencies [1, 34].

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

In practice, spectral bias is difficult to measure: the most direct method involves taking a Fourier transform with respect to the input, which is impractical to compute due to the high dimensionality of images. Early experimental work focused on low-dimensional synthetic data [1, 2] or used proxy measurements of spectral bias [41], for instance by inserting label noise of various frequencies during training [34]. However, the label noise method is limited to binary classification and involves modifying the training data, making it difficult to disentangle how other changes to the training data (such as augmentation) affect the spectral content of the learned function. It remains an open question whether modern neural networks exhibit spectral bias and what role it plays in generalization.

In this work, we investigate model complexity through the lens of spectral bias, by introducing experimental methods to study the frequency decomposition of the functions learned by modern image classification networks. We find that high-accuracy models learn a function that is high-frequency in regions between different image classes but low-frequency within each class.

Contributions. We extend the label noise procedure of Rahaman et al. [34] to enable measuring spectral bias in multi-class classification via label smoothing, and apply this technique to understand the function frequencies present in high-accuracy models on CIFAR-10 [21]. We also introduce a second method for measuring the smoothness of a learned function via linear interpolation between test examples, and apply this method to both CIFAR-10 and ImageNet [8] models. We use this method to probe the effects of the training data on the learned function frequencies, and to distinguish the spectral content in paths between images from the same class ("within-class paths") and paths between images from different classes ("between-class paths").

Using our label smoothing methodology, we find that, as expected, models are capable of learning higher frequency target functions the longer they train. However, using our linear interpolation methodology we find that throughout training the learned function becomes lower-frequency on within-class paths even as it becomes higher-frequency on between-class paths. Indeed, we find that higher-accuracy models consistently demonstrate greater separation in frequency content between within-class and between-class paths, with lower frequencies within-class and higher-frequencies between-class. On CIFAR-10, this trend holds as we improve the model architecture, increase the size of the training dataset, improve the data augmentation strategy, and apply self-distillation. On ImageNet, this trend appears again as we consider models with more diverse architectures.

We also explore the relationship between image frequency and function frequency. By further extending the label noise methodology of Rahaman et al. [34] to study spectral bias in directions of interest through the input space, we find that CIFAR-10 models most readily learn functions of the low image frequencies common in natural images.

Via linear interpolation within ImageNet classes, we find that many models are higher-frequency in more internally diverse classes and lower-frequency in more internally consistent classes.

2 Related Work

Implicit bias. A common belief is that some form of implicit bias imposed by the training procedure may account for the generalization ability of overparameterized neural networks, and accordingly, much research has been directed towards understanding these implicit biases [36, 45, 20, 13, 14, 15, 17, 3, 30, 28, 29, 24, 13, 27]. One of the most well-known forms of implicit bias comes from the optimization algorithm itself, which can guide networks to solutions that generalize well [36, 20, 40].

Spectral bias. Spectral bias, also called the frequency principle, is a form of implicit bias with much recent attention including theoretical and experimental approaches [33, 41, 34, 1, 47]. Basri et al. [1] studied spectral bias using a linear model of SGD training dynamics to show that models learn low frequency (simple) functions faster, assuming training data that is distributed uniformly on the hypersphere. Basri et al. [2] extended the analysis to consider nonuniformly spaced training data, and found that learning is faster where samples are denser; if sampling is nonuniform, then during training, the learned function will be higher frequency in regions with denser samples.

Rahaman et al. [34] used a more direct analysis to show the same spectral bias toward low frequency functions and also suggested that when training data lie on a manifold (as natural images are believed to), increasing the complexity of the manifold makes it easier for models to fit high frequency functions. Additionally, Rahaman et al. [34] posited that high frequency components of the learned

function are most sensitive to perturbations in the model parameters, connecting back to the idea of flat optimization minima [29] or smooth loss landscape [25]. They proposed experimental methods to study spectral bias in image classification, but focused on a binary subset of the relatively simple MNIST dataset [9], with mean square error loss.

Our label smoothing experiments are a direct extension of Rahaman et al. [34] to multiclass classification with the more common cross-entropy loss. Our linear interpolation experiments are somewhat similar to the recently-proposed experimental methods in Zhang et al. [47], except that we sample along paths between images rather than in regions surrounding each image; this allows for a more global measurement of spectral bias that we apply to a broad and distinct set of training procedures beyond the double descent regime studient in Zhang et al. [47].

Model sensitivity to image frequency. While we focus on *function frequency* in this work, prior research aimed to understand model sensitivity to image frequencies. Jo and Bengio [19] found that CNNs are sensitive to Fourier statistics of the training data, even those irrelevant to human viewers. Ortiz-Jimenez et al. [32] studied the image frequency bias induced by using a convolutional architecture, and Ortiz-Jimenez et al. [31] argued that models are sensitive primarily to discriminative Fourier directions in the training data. They posit that adversarial training induces large margins in orthogonal directions, thereby improving robustness. Yin et al. [43] introduced a procedure to measure the sensitivity of trained models to Fourier image perturbations, and showed that adversarial training and Gaussian data augmentation increase robustness to high frequency perturbations but increase sensitivity to low frequency perturbations. Our work adds to this line of research a study of the interplay between two notions of frequency: the function frequency involved in spectral bias, and the image frequencies present in the data.

3 Methodology

The goal of our work is to measure the complexity of neural network functions through the lens of frequency. In low dimensions, measuring the frequency decomposition of a function is straightforward and tractable: evaluate the function at dense, uniform sampling positions and compute its discrete Fourier transform. However, the function of interest in image classification is unavoidably high-dimensional, mapping images (with thousands of pixel values) to object classes. It would be intractable even to collect sufficient samples of this function to compute a discrete Fourier transform, let alone compute the transform itself. Instead, we employ two complementary approaches to measure informative proxies of this frequency decomposition.

3.1 Label Smoothing

The core idea for measuring function frequency introduced by Rahaman et al. [34] is to construct a sinusoid over the space of images, and to use that sinusoid as a form of label noise during training. Let $\mathcal{D} = \{(\mathbf{X}_i, \boldsymbol{y}_i)\}_{i=1}^{n_{train}}$ be the training examples and $\mathcal{D}_{\text{valid}} = \{(\mathbf{X}_j, \boldsymbol{y}_j)\}_{j=1}^{n_{val}}$ be the validation examples, with $\mathbf{X}_i$ an image and $\boldsymbol{y}_i$ a one-hot class encoding, where n_{train} is the number of training examples, n_{val} is the number of validation examples, d is the side length of an image (assumed to be square for simplicity), c is the number of color channels, and M is the number of classes, so $\mathbf{X}_i \in \mathbb{R}^{d \times d \times c}$ and $\boldsymbol{y}_i \in \mathbb{R}^M$.

To extend this procedure to the multi-class setting, we add noise of various frequencies to an M-dimensional label vector via label smoothing [37]. Let $S : \mathbb{R}^{d \times d \times c} \to [0, 1]$ be a target function that maps an input image $\mathbf{X}_i$ to a scalar value between 0 and 1. We apply label smoothing to each label $\boldsymbol{y}_i$, mapping it to $\bar{\boldsymbol{y}}_i = \boldsymbol{y}_i(1 - S(\mathbf{X}_i)) + \frac{1}{M} S(\mathbf{X}_i)$ to retain a valid probability distribution. We then train from scratch using the original examples $\mathbf{X}_i$ and their smoothed labels $\bar{\boldsymbol{y}}_i$. Finally, we evaluate on the validation images $\mathbf{X}_j$, comparing to both their original one-hot labels $\boldsymbol{y}_j$ and smoothed labels $\bar{\boldsymbol{y}}_j = \boldsymbol{y}_j(1 - S(\mathbf{X}_j)) + \frac{1}{M} S(\mathbf{X}_j)$. Our label smoothing experiments use the CIFAR-10 dataset [21], where $n_{train} = 50000$, $n_{val} = 10000$, $d = 32$, $c = 3$, and $M = 10$; however, our experimental setup is sufficiently general as to transfer easily to any image classification task.

This method is introduced in Figure 1. As training proceeds, we see both the training loss (between predictions and smoothed labels $\bar{\boldsymbol{y}}_i$) and noisy validation loss (between predictions and smoothed labels $\bar{\boldsymbol{y}}_j$) decrease. The clean validation loss (between predictions and one-hot labels $\boldsymbol{y}_j$), however, initially decreases but at some point in training plateaus or begins to increase. At

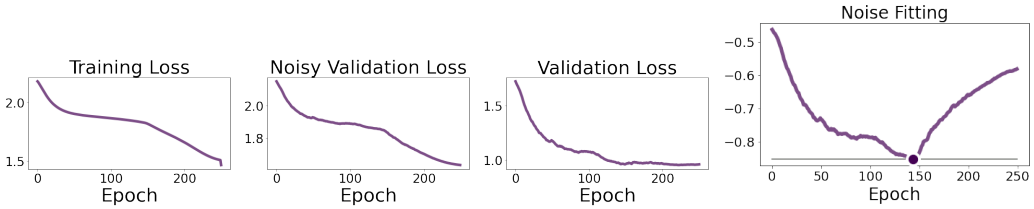


Figure 1: **Noise fitting shows when and how much a network fits a target frequency function.** At roughly epoch 150, noise fitting *(Right)*, the difference between clean and noisy validation loss, exhibits a clear "dip" when the model begins to fit the target function: training *(Left)* and validation *(Center Left)* loss on the perturbed function drop, while improvement stalls on clean validation data *(Center Right)*. By comparing the minimum value of noise fitting achieved by different models throughout training (the gray line in the right plot), we can compare the relative degree to which different models fit noise functions of varying frequency. A model with higher **min noise fitting** more readily fits the target noise function. Here, we train a `WideResNet32` model (wide-resnet with width 32) with radial wave label smoothing at frequency 0.04. For visual clarity, we apply exponential averaging to all curves.

this point, the model has begun to learn the target function S. Accordingly, we introduce **noise fitting**, defined as the difference between the validation loss on one-hot labels and the validation loss on labels smoothed with the same function S that was applied to the training labels: **noise fitting** $=$ **clean validation loss** $-$ **noisy validation loss**. We compare different models by computing the minimum noise fitting achieved throughout training. A model with higher **min noise fitting** more readily fits the target function S than a model with lower min noise fitting.

By choosing different functions S, we can probe nuances of spectral bias. Typically (inspired by Rahaman et al. [34]) we choose a radial wave: $S(\mathbf{X}) = \sin(2\pi f(\|\mathbf{X}\| - \mathbb{E}_{\mathcal{D}}\|\mathbf{X}\|))$, where $\|\mathbf{X}\|$ denotes the Euclidean norm of the vectorized image. We can vary the frequency f to understand spectral bias at this global scale. We can also choose more targeted functions S; for instance, if $\mathbf{V}$ is a direction of interest through the space of images (*i.e.* $\mathbf{V}$ is an image-shaped vector of unit norm), we can construct $S(\mathbf{X}) = \sin(2\pi f \langle \mathbf{X}, \mathbf{V} \rangle)$. This allows us to vary both the frequency and direction of the target sinusoid, to understand models' learnability along different directions through image space.

However, label smoothing has a few limitations. Because it involves perturbing the training labels, it cannot directly study the interaction between the training dataset and spectral bias, such as the effects of data augmentation. It also requires retraining each model from scratch, with substantial investment of time and computational resources. We address these limitations with our linear interpolation technique in the next section.

3.2 Linear Interpolation

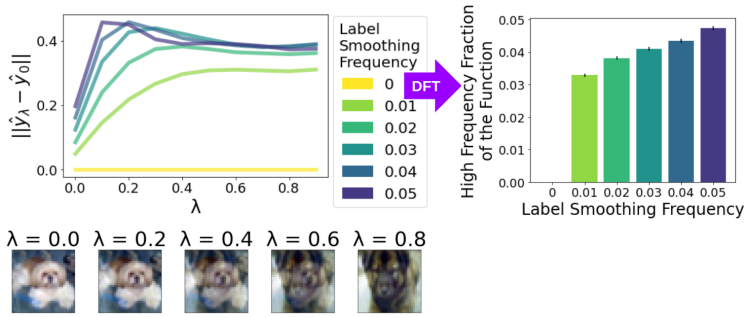


Figure 2: **Linear interpolation measurements on functions of varying frequency.** *Left*: Interpolation between images of the same class on an oracle function: the true one-hot label perturbed by radial wave label smoothing of variable frequency. As the oracle function increases in frequency, the interpolating paths become less smooth. *Right*: Summary of this interpolation experiment via a discrete Fourier transform; as the oracle frequency increases, so does the proportion of the DFT magnitude allocated to the high frequency components. In all interpolation figures, error bars report the standard error of the mean high frequency fraction over all the interpolating paths measured.

To complement the label smoothing approach, we also propose a methodology based on linear interpolation between validation images. Although we cannot take dense, regularly-spaced samples that cover the entire input (image) space, we can sample along specific paths and thereby glean glimpses into the spectral content of the learned function.

We consider two types of paths: those between images from the same class, and those between images from different classes. When considering paths between CIFAR-10 images in the same class, we choose 200 random, distinct pairs of validation images from each of the 10 CIFAR-10 classes and average over the paths defined by these pairs. We do the same for ImageNet, choosing 50 random, distinct paths for each of the 1000 ImageNet classes. When interpolating between CIFAR-10 classes, we choose 200 random, distinct validation image pairs from each of the 45 (10 choose 2) pairs of distinct classes, and average over the resulting paths; for ImageNet we choose 1000 random, distinct between-class paths, ensuring that each class is represented in exactly two paths.

For each pair of images $(\mathbf{X}_0, \mathbf{X}_1)$, we vary $\lambda \in [0, 1]$ to trace out a path, where each image in the path is given by $\mathbf{X}_\lambda = \lambda \mathbf{X}_1 + (1 - \lambda)\mathbf{X}_0$. We choose λs so that the distance between adjacent images on the path is constant; each path may produce a different number of samples depending on the total distance between $\mathbf{X}_0$ and $\mathbf{X}_1$. Figure 2 *(Bottom Left)* shows an illustrative example of two CIFAR-10 images and a corresponding interpolation path between them. For our experiments, we typically have 50 to 100 images along the interpolation path.

Relationship between interpolation and label smoothing experiments. In order to validate our interpolation methodology, we consider the oracle function that maps directly from the interpolated image $\mathbf{X}_\lambda$ to the target label used in our label smoothing experiments. By increasing the frequency of the label smoothing noise, we create oracle functions with higher frequencies. We then verify that our interpolation measurement indeed measures higher frequencies on the higher frequency oracle functions.

In Figure 2 *(Left)*, we show an example where we vary the label smoothing frequency from $f = 0$ to $f = 0.05$. We plot the value of λ along the interpolating path on the x axis, and the norm of the difference between the model output on the interpolated image $\hat{\boldsymbol{y}}_\lambda$ and the model output on the original image $\hat{\boldsymbol{y}}_0$ on the y axis. The norm of the prediction difference allows us to visualize how smooth the function is along the interpolating path; smoother prediction difference norm indicates a lower frequency function.

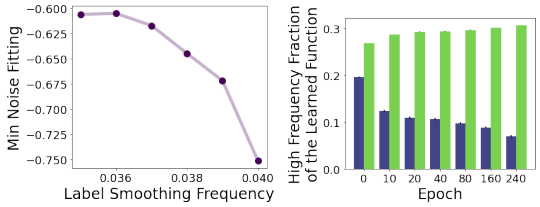
To quantify the frequency of the function along the interpolating path, we compute the per-class discrete Fourier transform (DFT). Letting $\{\mathbf{X}_{\lambda_t}\}_{t=1}^T$ denote the set of interpolated images, the DFT coefficient for frequency f is given by $\hat{\mathbf{Y}}_f[m] = \sum_{t=1}^T \hat{\boldsymbol{y}}_{\lambda_t}[m] \exp(-i2\pi f t)$, where i is the imaginary number and $m \in \{1, \ldots, M\}$ denotes the class. To summarize how the magnitudes of the DFT coefficients $\hat{\mathbf{Y}}_f$ are distributed between low and high frequencies, we average the coefficient magnitudes for each of the classes and compute the fraction of the total average coefficient magnitudes above a threshold. In Figure 2 *(Right)* we show this summary metric averaged over all within-class interpolating paths for the oracle functions, along with its standard error. In Figure 2 *(Left)*, the higher frequency oracle functions have less smooth prediction norm differences, and in Figure 2 *(Right)*, they have a higher fraction of DFT coefficient magnitudes in the high frequencies.

3.3 Data and Models

Our experiments use the CIFAR-10 [21] dataset of low-resolution (32×32) natural images from ten animal and object classes, as well as the ImageNet [8] dataset of higher-resolution (224×224) natural images from 1000 animal and object classes. We consider a diversity of model architectures on CIFAR-10 (6 total) and ImageNet (10 total). See Section A.1 for more details on the data processing, models, and their accuracies.

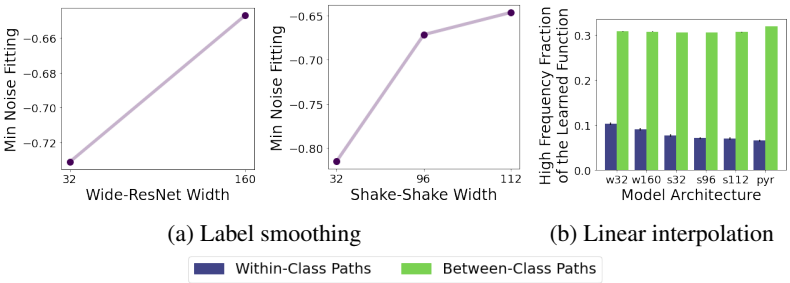
4 Results on CIFAR-10

Throughout this section we show representative examples from our CIFAR-10 experiments; full results (on all six models we tested) are included in Section A.3.



(a) Label smoothing (b) Linear interpolation

Figure 3: **Modern CNNs have spatially-dependent spectral bias.** Higher min noise fitting denotes that the label smoothing noise function is learned more readily. *Left*: Min noise fitting for a `ShakeShake96` model with variable-frequency radial wave label smoothing; lower frequencies are easier to learn. *Right*: Linear interpolation for a `ShakeShake96` model throughout training; as training proceeds, the learned function becomes lower-frequency within-class and higher-frequency between-class.



(a) Label smoothing (b) Linear interpolation

Figure 4: **Larger models learn high frequencies more readily; higher-accuracy models are higher-frequency between classes and lower-frequency within each class.** *Left*: Min noise fitting for Wide-ResNets and Shake-Shake models of variable width at frequency 0.039; the larger models learn this frequency more readily. *Right*: Linear interpolation frequencies for all six models we tested on CIFAR-10, ordered by test accuracy; higher-accuracy models are lower-frequency within-class and higher-frequency between-class.

4.1 Spectral Bias

We begin by applying our label smoothing methodology to find that modern image classification CNNs exhibit spectral bias, learning low frequency target functions early in training and learning higher frequency functions as training proceeds. Note that Figure 3 *(Left)* shows spectral bias over a small but illustrative range of frequencies (the full range of frequencies we tested was 0 to 0.1); target functions of sufficiently low frequency are fit almost immediately and target functions of sufficiently high frequency are never learned during the 250 epochs of training we tested.

Using our linear interpolation methodology and distinguishing between within-class paths (in which both endpoint images are of the same class) and between-class paths (where the path must traverse a class boundary) in Figure 3 *(Right)*, we find that this spectral bias is highly variable over the domain of images. In particular, while the learned function does increase in frequency for between-class paths, it decreases in frequency for within-class paths, indicating that the model is learning to cluster and separate the different classes. Throughout our CIFAR-10 experiments, we find that higher-accuracy models have greater separation between the frequencies of their within-class and between-class paths, with lower-frequency behavior inside each class and higher-frequency behavior in between.

4.2 Spectral Bias and Model Architecture

Although all models we tested exhibit spectral bias, we found that the precise nature of the bias depends on the choice of model. For example, with all else fixed, increasing the width of a model decreases its spectral bias, enabling it to more readily fit target frequency functions. This trend is evident in Figure 4 for the Wide-ResNet family *(Left)* and the Shake-Shake family *(Center)*.

Although larger models learn high frequencies faster, we found that higher-accuracy models are not uniformly higher-frequency. Instead, using linear interpolation in Figure 4 *(Right)* we find that higher-

accuracy models (ordered left to right by increasing test accuracy) are higher-frequency along between-class paths but lower-frequency along within-class paths, perhaps evident of a better class clustering.

4.3 Sensitivity to Natural Image Directions

Label smoothing with a radial wave offers a convenient global picture of spectral bias, by replacing the radial wave with other target functions we can use the same methodology to test more localized aspects of spectral bias. We take a step in this direction by considering the family of target label smoothing functions $S(\mathbf{X}; k) = \sin(2\pi f \langle \mathbf{X}, \mathbf{F}_k \rangle)$, where $\mathbf{F}_k$ is a diagonal Fourier basis image with frequency k and the same dimensions as $\mathbf{X}$, visualized in Figure 5 *(Bottom)*.

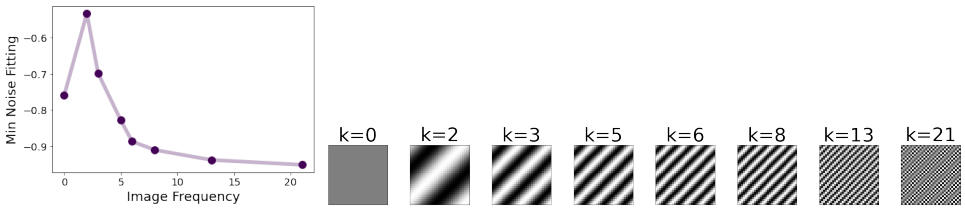


Figure 5: **Models are most sensitive to variations of low (but nonzero) image frequency.** *Left*: Effective noise fitting of a `WideResNet32` model with label smoothing of frequency 0.038 in various unit norm directions corresponding to Fourier basis images *(Right)* indexed by image frequency k (scaled to [0, 1] for visualization).

We consider (a subset of) Fourier basis images because the Fourier spectra statistics of natural images are well studied (see, *e.g.* Tolhurst et al. [38]): natural images tend to be composed of Fourier basis images with amplitude proportional to their inverse spatial frequency. Indeed, in Figure 5 we find that `ShakeShake96` is more sensitive to label smoothing in low image frequency directions.

This finding is consistent with theoretical predictions of Basri et al. [2] that models learn faster in regions of higher density of training examples: since low image frequencies are more common in natural images, the effective sampling density enjoyed by a label smoothing target is higher in these directions. It is also possible that this bias is inherent to the convolutional model architecture regardless of the data distribution, as discussed in Ortiz-Jimenez et al. [32], or that a combination of both effects is at play. Determining the precise cause of this image frequency finding is an interesting direction for further study.

4.4 Spectral Bias and Training Data

In our CIFAR-10 interpolation experiments, we consistently observed that variations in training that improve accuracy create greater disparity between the frequencies of within-class and between-class paths, with lower frequencies within-class and higher frequencies between-class. This trend is evident in Figure 3 *(Right)* as a model trains for more epochs and in Figure 4 *(Right)* as the model architecture improves. In Figure 6 we find that the same trend also holds as we train on more data *(Left)*, apply more effective data augmentation *(Center)*, or apply self-distillation *(Right)*.

Training Set Size. As shown in Figure 6 *(Left)* on `WideResNet160`, we find that increasing the number of training examples increases the difference in frequency content between within-class interpolating paths and between-class interpolating paths. Training with more data makes our CIFAR-10 models behave lower-frequency within each class but higher-frequency between classes, perhaps indicating that the models are learning a better clustering of the classes as they are provided with additional data.

Data Augmentation. Data augmentation is a common strategy to increase the effective training set size without the expense of actually collecting additional examples. In Figure 6 *(Center)* we consider the effects of Mixup (with strength 0.1) [46], AutoAugment [5], and RandAugment [6], each of which improves the final accuracy of the trained model (with RandAugment producing the most substantial benefits in our experiments, followed by AutoAugment and then Mixup). RandAugment and AutoAugment generate new images by applying geometric and lighting transformations, whereas Mixup linearly interpolates between pairs of existing images. Much like our experiments with training set size, we find that augmentations that improve accuracy more also produce greater frequency separation between within-class paths and between-class paths.

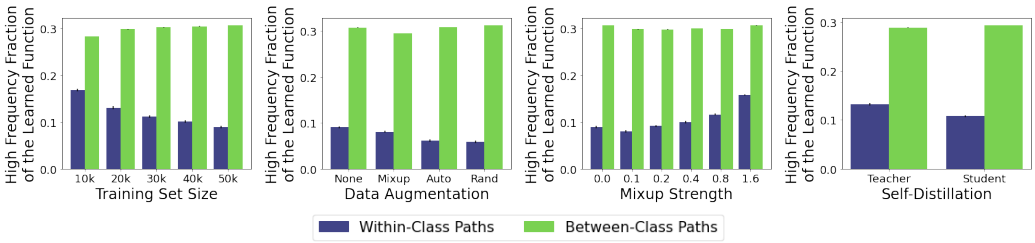


Figure 6: **Higher-accuracy models are lower-frequency within-class and higher-frequency between-class.** Test accuracy increases from left to right in each figure as we train `WideResNet160` with more data *(Left)*, better data augmentation *(Center Left)*, and self-distillation *(Right)*; in each case the frequency separation between within-class and between-class paths increases as accuracy improves. We also find that varying the Mixup strength *(Center Right)* within roughly the range 0.1 to 0.4 recommended by the original authors [46] tends to increase this frequency separation, but using stronger Mixup has the opposite effect.

Mixup Strength. Mixup augmentation [46] perturbs each batch of training data as follows: each example is matched with a partner based on a random permutation of the batch, and each example is then perturbed towards its partner by an interpolation amount λ drawn from a symmetric beta distribution (with the same λ used for all examples in the batch). We refer to the parameter of this beta distribution as the Mixup strength, as it controls the degree to which the augmented images tend to lie close to an original training image or close to the average of two training images. A parameter of 0 corresponds to no augmentation ($\lambda = 0$ or $\lambda = 1$, always using the original images), a parameter of 1 corresponds to the uniform distribution over $\lambda \in [0, 1]$, and a parameter of ∞ corresponds to $\lambda = 0.5$, the exact midpoint between a pair of training images. Figure 6 *(Center)* compares a range of Mixup strengths and finds that relatively low values that only slightly perturb the original images tend to produce greater frequency separation (and higher accuracy), whereas perturbing the images too much with stronger Mixup decreases the frequency separation between within-class and between-class paths (and hurts final accuracy). Of note, the range of Mixup strengths that increase or do not affect frequency separation in our experiments are well aligned with those recommended by the original authors [46]: 0.1 to 0.4.

Self-Distillation. Finally, we consider the effect of self-distillation on the frequency content of the learned function. In self-distillation, a "teacher" model is trained with some form of strong regularization, in our case a combination of weight decay and early stopping based on training loss. A "student" model with the same architecture is then trained from scratch to fit the pseudolabels produced by the teacher, instead of the original one-hot training labels (alternatively, an interpolation between the pseudolabels and the original labels may be used). Prior research [11] found that this procedure can train student models that outperform both their teachers and a baseline (the same architecture trained to completion on the one-hot labels).

Prior research has also sought to understand the mechanism behind knowledge distillation and self-distillation in particular. Mobahi et al. [26] finds that self-distillation acts as a regularizer by limiting the basis functions available to the student to learn. We complement their theoretical work with our interpolation experimental methodology in Figure 6 *(Right)*, where indeed we find that the student model learns a greater frequency separation between within-class and between-class paths than its teacher, while also achieving higher test accuracy. We conjecture that this effect is analogous to low-pass prefiltering common in digital signal processing: a high-frequency target function that we cannot adequately sample is first smoothed (in this case by being approximated by a regularized teacher) and then it can be modeled via samples without further loss in fidelity. Without this prefiltering, our samples are inadequate to capture the complexity of the target function, so we reconstruct an imperfect version corrupted by aliasing.

5 Results on ImageNet

Finally, we apply our linear interpolation methodology to study the frequency content of a range of pretrained models on ImageNet [8]. Here we show results on `ResNet50`; results on all ten models we tested on ImageNet are included in Section A.4.

5.1 Training Time

As we observed for CIFAR-10, our ImageNet models tend to increase the frequency gap between the two types of interpolating paths (within-class and between-class) as training proceeds, with lower-frequencies within-class and higher-frequencies between-class. We visualize this trend for ResNet50 in Figure 7 *(Left)*. For our CIFAR-10 models, this increase in frequency difference between path types was achieved by a combination of increasing the average high frequency content on between-class paths and decreasing the average high frequency content on within-class paths. On ImageNet, we again find that average frequency content on between-class paths increases during training, but that within-class frequencies sometimes decrease during training (*e.g.* for ResNets) but sometimes increase (slower than between-class paths) during training (*e.g.* for CoATNets). We speculate that the trend of decreasing within-class frequency as training time progresses is less consistent on ImageNet than CIFAR-10 because, as we explore in the next section, ImageNet classes are more internally diverse than CIFAR-10 classes.

5.2 Class Coherence

In addition to containing 100× as many classes as CIFAR-10, ImageNet classes can also be more internally diverse. This notion of within-class "consistency" is captured by C-Scores [18], which assign to each training example a value between 0 and 1 denoting how typical that example is. One way of capturing the internal consistency or diversity of a class is by studying the distribution of its C-Scores. In Figure 7 *(Right)* we plot the mean and standard deviation of these scores for each of the 1000 ImageNet classes in gray and highlight the 10 classes whose ResNet50 interpolating paths between validation images are most high-frequency (red stars) and least high-frequency (blue squares). We find that ResNet50 is lower-frequency on more internally consistent classes and higher-frequency on more diverse classes, supporting the intuition that more diverse classes require more complex functions.

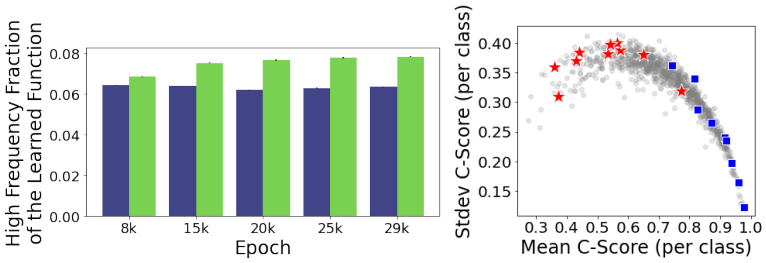


Figure 7: **Within vs. between-class frequency gap widens during training on ImageNet, and internally diverse classes have higher within-class function frequencies.** *Left*: The frequency separation between within-class and between-class paths for ResNet50 grows during ImageNet training. *Right*: Highlighted points show the 10 classes for which ResNet50 is most (red stars) and least (blue squares) high-frequency. Gray points show the distribution of C-Scores among all 1000 ImageNet classes.

6 Conclusions

In this paper, we introduced two methods to measure spectral bias in modern image classification neural networks, and applied these methods towards the central question:

What kinds of function frequencies are needed for modern neural networks to generalize?

Specifically, we applied these methods to examine the impact of a variety of training choices on the learned frequencies. On CIFAR-10, we found that higher-accuracy models typically have greater frequency separation between within-class and between-class paths, with lower frequencies within-class and higher frequencies between-class. This trend holds regardless of whether high accuracy is achieved through longer training, choice of model architecture, increase in dataset size, data augmentation, or self-distillation. On ImageNet, we again found that this frequency separation increases during training, and noted that models are higher-frequency within more internally diverse classes. Our experimental methods offer a window onto the frequency structure of neural networks for image classification, and its relationship to model performance.

References

[1] Ronen Basri, David Jacobs, Yoni Kasten, and Shira Kritchman. The convergence rate of neural networks for learned functions of different frequencies, 2019.

[2] Ronen Basri, Meirav Galun, Amnon Geifman, David Jacobs, Yoni Kasten, and Shira Kritchman. Frequency bias in neural networks for input of non-uniform density, 2020.

[3] Mikhail Belkin, Siyuan Ma, and Soumik Mandal. To understand deep learning we need to understand kernel learning. In *International Conference on Machine Learning*, pages 541–549. PMLR, 2018.

[4] Zac Cranko, Simon Kornblith, Zhan Shi, and Richard Nock. Lipschitz networks and distributional robustness, 2018.

[5] Ekin D. Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V. Le. Autoaugment: Learning augmentation policies from data, 2019.

[6] Ekin D. Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V. Le. Randaugment: Practical automated data augmentation with a reduced search space, 2019.

[7] Zihang Dai, Hanxiao Liu, Quoc V. Le, and Mingxing Tan. Coatnet: Marrying convolution and attention for all data sizes, 2021.

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

[9] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

[11] Tommaso Furlanello, Zachary C. Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. Born again neural networks, 2018.

[12] Xavier Gastaldi. Shake-shake regularization, 2017.

[13] Suriya Gunasekar, Jason Lee, Daniel Soudry, and Nathan Srebro. Implicit bias of gradient descent on linear convolutional networks, 2019.

[14] Suriya Gunasekar, Jason Lee, Daniel Soudry, and Nathan Srebro. Characterizing implicit bias in terms of optimization geometry, 2020.

[15] Moritz Hardt, Benjamin Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent, 2016.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[17] Elad Hoffer, Itay Hubara, and Daniel Soudry. Train longer, generalize better: closing the generalization gap in large batch training of neural networks, 2018.

[18] Ziheng Jiang, Chiyuan Zhang, Kunal Talwar, and Michael C. Mozer. Characterizing structural regularities of labeled data in overparameterized models, 2021.

[19] Jason Jo and Yoshua Bengio. Measuring the tendency of cnns to learn surface statistical regularities, 2017.

[20] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima, 2017.

[21] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). URL `http://www.cs.toronto.edu/~kriz/cifar.html`.

[22] Klas Leino, Zifan Wang, and Matt Fredrikson. Globally-robust neural networks, 2021.

[23] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks, 2019.

[24] Horia Mania, John Miller, Ludwig Schmidt, Moritz Hardt, and Benjamin Recht. Model similarity mitigates test set overuse, 2019.

[25] Christian H.X. Ali Mehmeti-Göpel, David Hartmann, and Michael Wand. Ringing re{lu}s: Harmonic distortion analysis of nonlinear feedforward networks. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=TaYhv-q1Xit`.

[26] Hossein Mobahi, Mehrdad Farajtabar, and Peter L. Bartlett. Self-distillation amplifies regularization in hilbert space, 2020.

[27] Preetum Nakkiran, Gal Kaplun, Dimitris Kalimeris, Tristan Yang, Benjamin L. Edelman, Fred Zhang, and Boaz Barak. Sgd on neural networks learns functions of increasing complexity, 2019.

[28] Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. In search of the real inductive bias: On the role of implicit regularization in deep learning, 2015.

[29] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nathan Srebro. Exploring generalization in deep learning, 2017.

[30] Behnam Neyshabur, Zhiyuan Li, Srinadh Bhojanapalli, Yann LeCun, and Nathan Srebro. Towards understanding the role of over-parametrization in generalization of neural networks, 2018.

[31] Guillermo Ortiz-Jimenez, Apostolos Modas, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Hold me tight! influence of discriminative features on deep network boundaries, 2020.

[32] Guillermo Ortiz-Jimenez, Apostolos Modas, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Neural anisotropy directions, 2020.

[33] Samet Oymak, Zalan Fabian, Mingchen Li, and Mahdi Soltanolkotabi. Generalization guarantees for neural networks via harnessing the low-rank structure of the jacobian, 2019.

[34] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred A. Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks, 2019.

[35] Harshay Shah, Kaustav Tamuly, Aditi Raghunathan, Prateek Jain, and Praneeth Netrapalli. The pitfalls of simplicity bias in neural networks, 2020.

[36] Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data, 2018.

[37] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision, 2015.

[38] DJ Tolhurst, Y_ Tadmor, and Tang Chao. Amplitude spectra of natural images. *Ophthalmic and Physiological Optics*, 12(2):229–232, 1992.

[39] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers and distillation through attention, 2021.

[40] Ashia C Wilson, Rebecca Roelofs, Mitchell Stern, Nathan Srebro, and Benjamin Recht. The marginal value of adaptive gradient methods in machine learning. *arXiv preprint arXiv:1705.08292*, 2017.

[41] Zhi-Qin John Xu, Yaoyu Zhang, and Yanyang Xiao. Training behavior of deep neural network in frequency domain, 2019.

[42] Yoshihiro Yamada, Masakazu Iwamura, Takuya Akiba, and Koichi Kise. Shakedrop regularization for deep residual learning. *IEEE Access*, 7:186126–186136, 2019. ISSN 2169-3536. doi: 10.1109/access.2019.2960566. URL `http://dx.doi.org/10.1109/ACCESS.2019.2960566`.

[43] Dong Yin, Raphael Gontijo Lopes, Jonathon Shlens, Ekin D. Cubuk, and Justin Gilmer. A fourier perspective on model robustness in computer vision, 2020.

[44] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks, 2017.

[45] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization, 2017.

[46] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization, 2018.

[47] Xiao Zhang, Haoyi Xiong, and Dongrui Wu. Rethink the connections among generalization, memorization and the spectral bias of dnns, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] Limitations of the label smoothing methodology are discussed at the end of Section 3.1. Limitations of the interpolation methodology are discussed in Section A.2.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Code is provided in the supplement.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Some of these details are provided in the appendix, and the rest are evident from the code, provided in the supplement.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] Standard errors are reported for linear interpolation experiments.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No] These results did require substantial GPU compute, but we do not quantify the exact amount. Each point in a label smoothing figure required training a model from scratch, and each value in an interpolation figure required evaluating a pretrained model on hundreds of thousands of images.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We cite the creators of CIFAR-10 [21] and ImageNet [8], the image datasets we use, as well as the creators of the models we use [44, 12, 42, 5, 16, 10, 39, 7].
   (b) Did you mention the license of the assets? [No] The licenses of each dataset and model we use are available on their websites.
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Appendix

A.1 Data, Models, and Model Accuracies

Images are pixel-wise normalized by the mean and standard deviation of the training images for each dataset, and for ImageNet all images are center cropped and resized to 224×224; this preprocessing is done before any interpolating paths are constructed. For CIFAR-10, we consider six different convolutional neural networks: two Wide-ResNets [44] of different sizes (WideResNet32 and WideResNet160), three Shake-Shake regularized networks [12] of different sizes (ShakeShake32, ShakeShake96, and ShakeShake112), and PyramidNet, which uses the even stronger Shake-Drop regularization [42]. Our implementations are based on Cubuk et al. [5]; we use the same optimizer (stochastic gradient descent with momentum) and cosine learning rate schedule. We train without data augmentation (to ensure all models are trained on exactly the same examples), except for experiments that explicitly vary data augmentation. Without data augmentation, the test accuracies of our models are shown in Table 1.

Model	CIFAR-10 Test Accuracy (%)
WideResNet32	89.4
WideResNet160	90.2
ShakeShake32	92.2
ShakeShake96	93.5
ShakeShake112	93.6
PyramidNet	95.8

Table 1: Test accuracies (in increasing order) of our six CIFAR-10 models, trained without data augmentation.

For ImageNet, we consider ten different pretrained models: three ResNets [16] of different sizes (ResNet50, ResNet101, and ResNet152), two Vision Transformers [10] of different sizes (ViT and ViTSmall), three distilled transformers [39] (DeiT, DeiTSmall, and DeiTClsToken), and two CoAtNets [7] (CoAtNet0 and CoAtNet0BF16). The top-1 classification accuracies of these models are presented in Table 2.

Model	ImageNet Test Accuracy (%)
ViTSmall	64.3
ViT	75.1
ResNet50	76.1
CoAtNet0BF16	78.4
ResNet101	78.5
ResNet152	79.0
DeiTSmall	80.1
DeiTClsToken	81.5
DeiT	81.7
CoAtNet0	81.7

Table 2: Test accuracies of our ten pretrained ImageNet models, in increasing order.

A.2 Linear Interpolation: Methodological Details

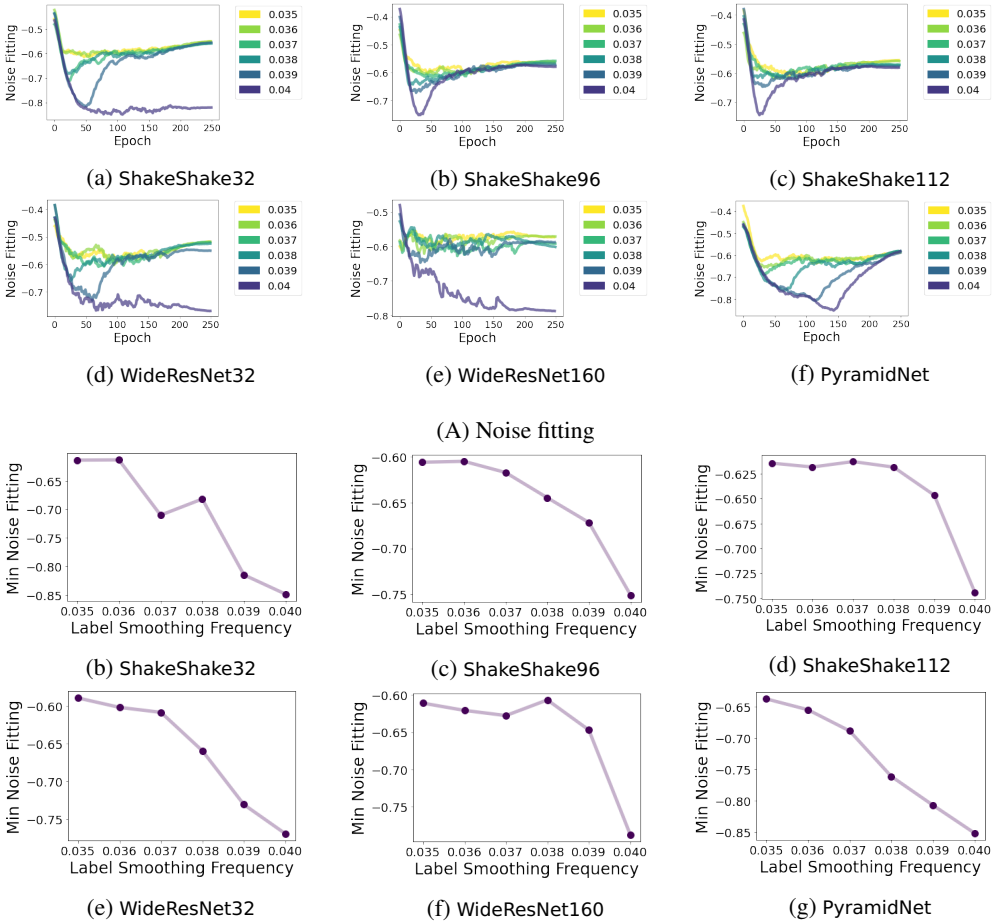
For each sampled path, we compute the discrete Fourier transform (DFT) of the prediction function along the path separately for each of the M class predictions, take the (real) magnitude of the resulting (complex) DFT coefficients, and average them among the M classes. We then compute the fraction of this averaged DFT magnitude that is allocated to the high frequency components, and aggregate over the many paths by calculating the mean high frequency fraction and its standard error. We use a simple threshold frequency of 0.05 for both CIFAR10 and ImageNet, although our paths are sampled with a distance of 1 for CIFAR10 and 7 for ImageNet (since ImageNet images are larger and thus farther apart). This precise threshold is not critical to our qualitative results, and different tradeoffs of frequency resolution and computational expense are possible.

Limitations of linear interpolation. Like all proxy measurements of spectral bias, linear interpolation is a coarse metric. We only measure the Fourier content of the learned function along specific paths, and aggregate results across many paths, across all classes, and only summarize frequencies into coarse "low" and "high" bins. An additional limitation is that the paths we define interpolate in pixel space, which means that there are images along our interpolating paths that are off the hypothesized manifold of natural images.

A.3 CIFAR-10 full results

A.3.1 Label Smoothing

Figure 8 shows the same results as Figure 3 *(Left)* on all six CIFAR-10 models we tested.



(a) ShakeShake32 (b) ShakeShake96 (c) ShakeShake112

(d) WideResNet32 (e) WideResNet160 (f) PyramidNet

(A) Noise fitting

(b) ShakeShake32 (c) ShakeShake96 (d) ShakeShake112

(e) WideResNet32 (f) WideResNet160 (g) PyramidNet

(B) Min noise fitting summaries

Figure 8: **All six CIFAR-10 models we tested exhibit spectral bias.** Here we show noise fitting when training each model with different frequencies of radial wave label smoothing.

A.3.2 Sensitivity to Natural Image Directions

Figure 9 shows the same results as Figure 5 on all six CIFAR-10 models we tested.

A.3.3 Agreement Between Label Smoothing and Linear Interpolation

Varying the frequency of the radial wave label smoothing When the frequency of the radial wave used for label smoothing increases, models take more time to fit the smoothing noise. Figure 3 *(Left)* and Figure 8 show this using label smoothing; the corresponding interpolation experiment is

shown in Figure 10. The target frequencies between 0.035 and 0.04 are close enough that, were the model to fit each perfectly, the interpolation curves would be nearly visually indistinguishable, as we can tell from Figure 2 *(Left)*. However, the model is actually smoother when trying to fit higher frequency label smoothing noise, because it fits this noise less well (in addition to fitting it later in training). We can see this effect, for instance, by noting that `WideResNet32`, `WideResNet160`, and `ShakeShake32` fail to fit frequency 0.04, both in Figure 8 and Figure 10.

It is also worth noting that, in Figure 10 and repeatedly across our interpolation experiments, the learned function is smoother (lower-frequency) within-class and less smooth (higher-frequency) between-class. This is to be expected since the model must change predictions somewhere along the path between examples from different classes.

Learning low frequencies first The label smoothing experiment presented in Figure 3 *(Left)* and Figure 8 shows that low frequency target functions are learned earlier than higher frequency targets; we can also confirm this finding using interpolation with model checkpoints saved at different epochs throughout training (without any label smoothing) in Figure 11. We see that spectral bias in image classification is more nuanced than simply increasing in frequency over time: as training proceeds, models become higher-frequency along between-class paths but lower-frequency along within-class

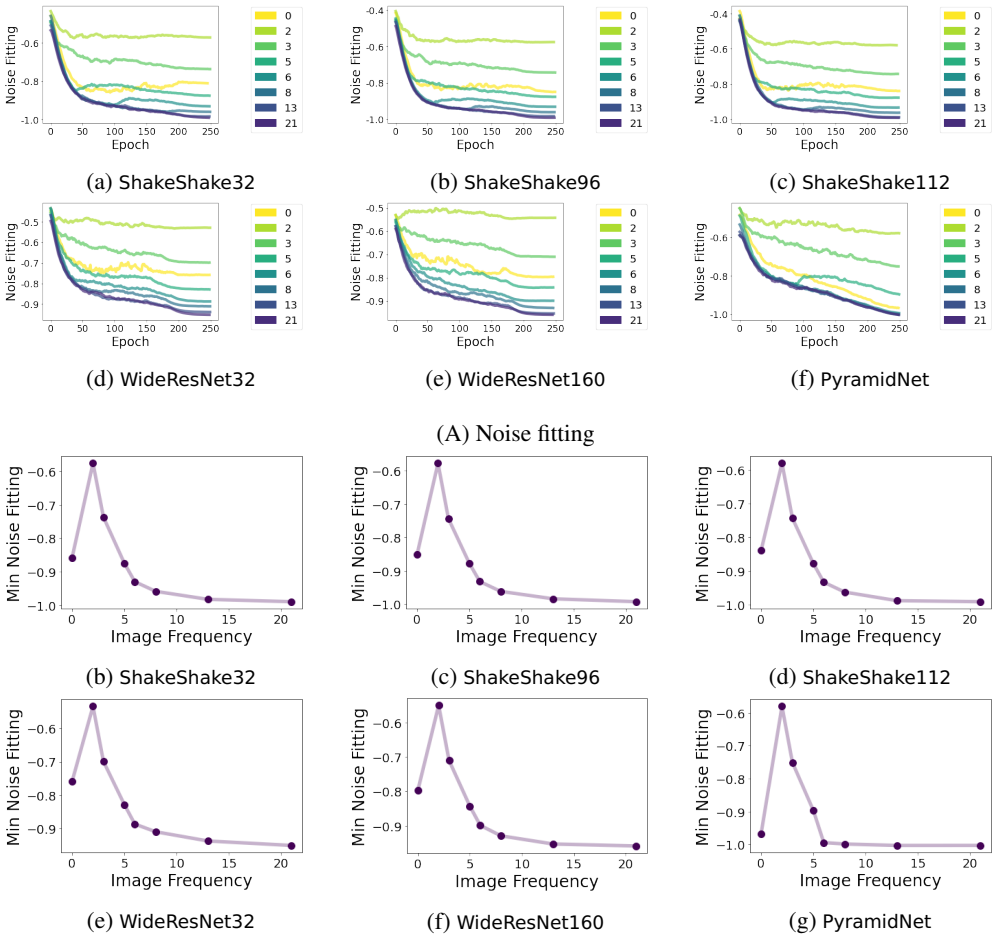


Figure 9: **All six CIFAR-10 models we tested exhibit sensitivity to variations of low (but nonzero) image frequency, which are dominant in natural images.** Here we show noise fitting when training each model with label smoothing of frequency 0.038 in various unit norm direction corresponding to Fourier basis images indexed by frequency k.

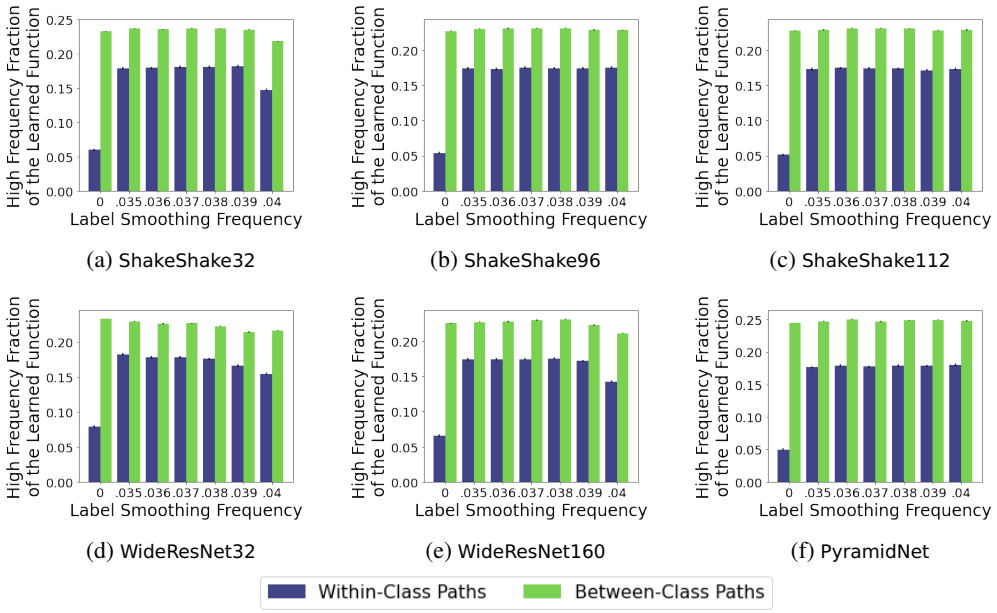


Figure 10: **Spectral bias is evident via interpolation when training with radial wave label smoothing at various frequencies.** Results here parallel those in Figure 8 but using the linear interpolation methodology.

paths. This may indicate that the model is learning an appropriate clustering of the data into its classes, and reducing the functional variation within each cluster.

A.3.4 Weight Decay

Figure 12 uses the label smoothing methodology on our six CIFAR-10 models to show that weight decay delays the learning of high frequencies.

A.3.5 Training Set Size

Figure 6 *(Left)* uses linear interpolation to show the within-class regularization (frequency reduction) effect of increasing dataset size for `WideResNet160`. Figure 13 shows the same effect on all six CIFAR-10 models we tested.

A.3.6 Data Augmentation

Figure 6 *(Center)* uses linear interpolation to study the effect of common data augmentation procedures on the learned frequencies for `WideResNet160`. Figure 14 shows the same experiment on all six models we tested.

Mixup Strength Figure 6 *(Center)* uses linear interpolation to show that training with Mixup data augmentation [46] causes `WideResNet160` to learn a within-class lower-frequency function, but too much Mixup can produce a higher-frequency function within-class. Figure 15 repeats the experiment on all six CIFAR-10 models we tested.

A.3.7 Self-Distillation

Figure 6 *(Right)* uses linear interpolation to show that self-distillation with `ShakeShake96` produces a student model that is lower-frequency within-class (and slightly higher-frequency between-class) than its teacher, and has higher validation accuracy than the teacher. Figure 16 shows the same result across all six CIFAR-10 models we tested.

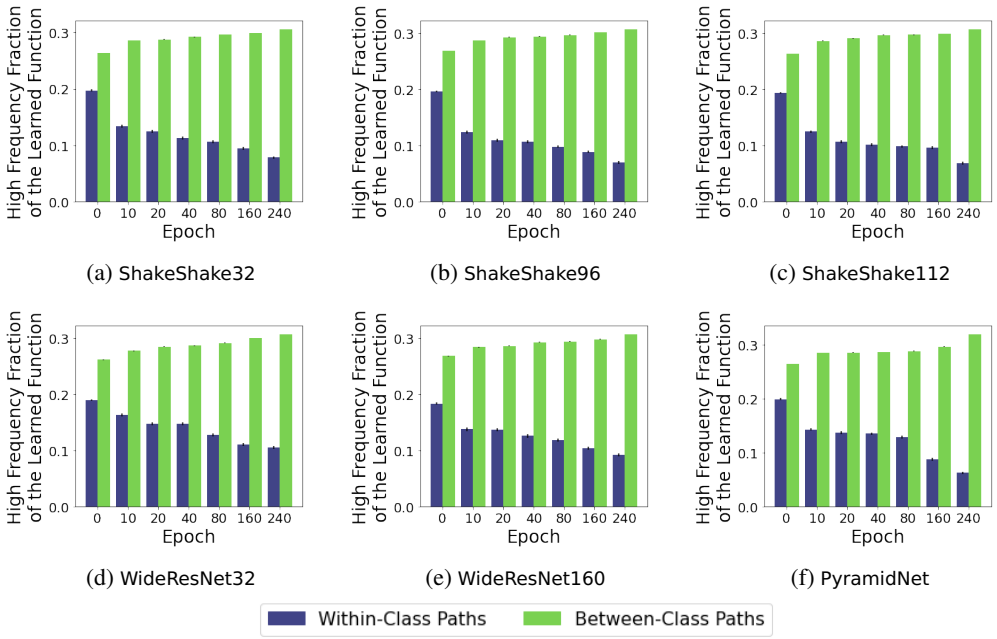


Figure 11: **All six CIFAR-10 models become higher-frequency between-class and lower-frequency within-class throughout training.** As accuracy improves, so does the frequency separation between within-class and between-class paths.

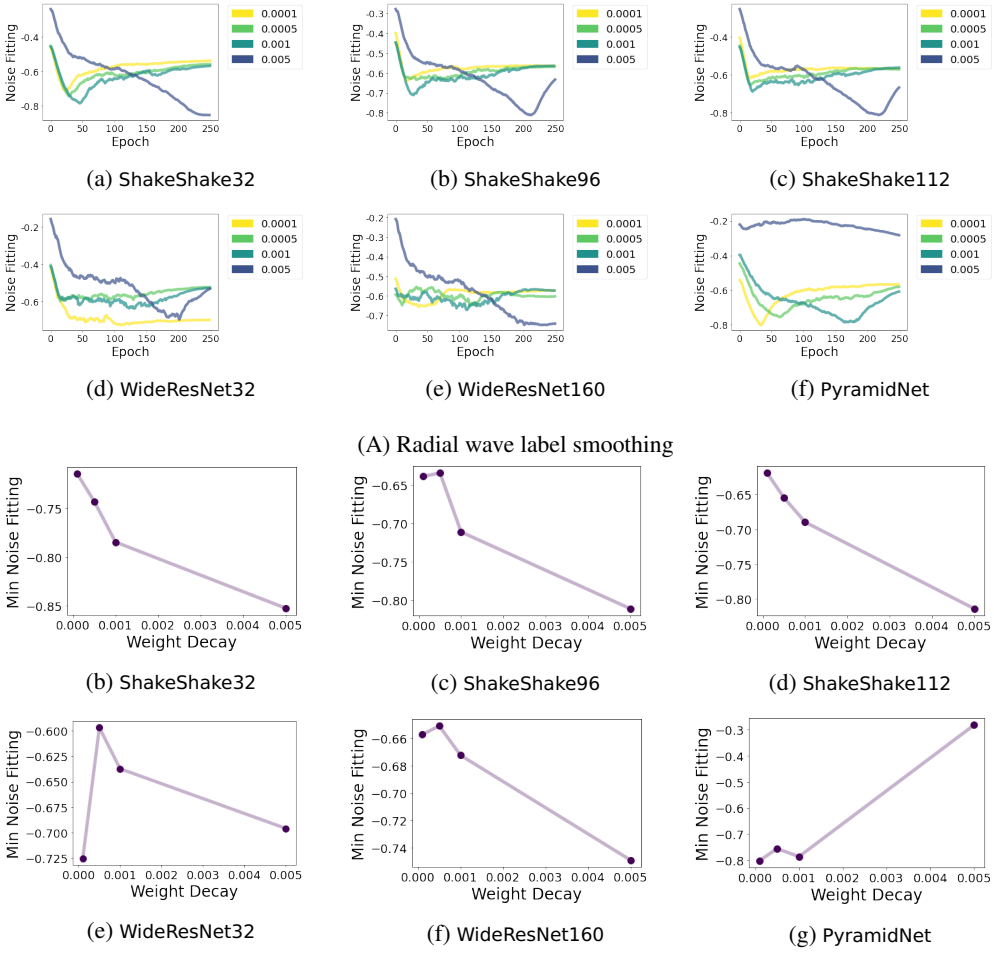


Figure 12: **All six CIFAR-10 models we tested are slower to learn high frequency target functions when trained with stronger weight decay.** Note that for PyramidNet, min noise fitting does not fully capture the results of noise fitting; the "dips" are clearly separated across epochs, with higher weight decay inducing delayed dips, even if they achieve similar minimum noise fitting values.

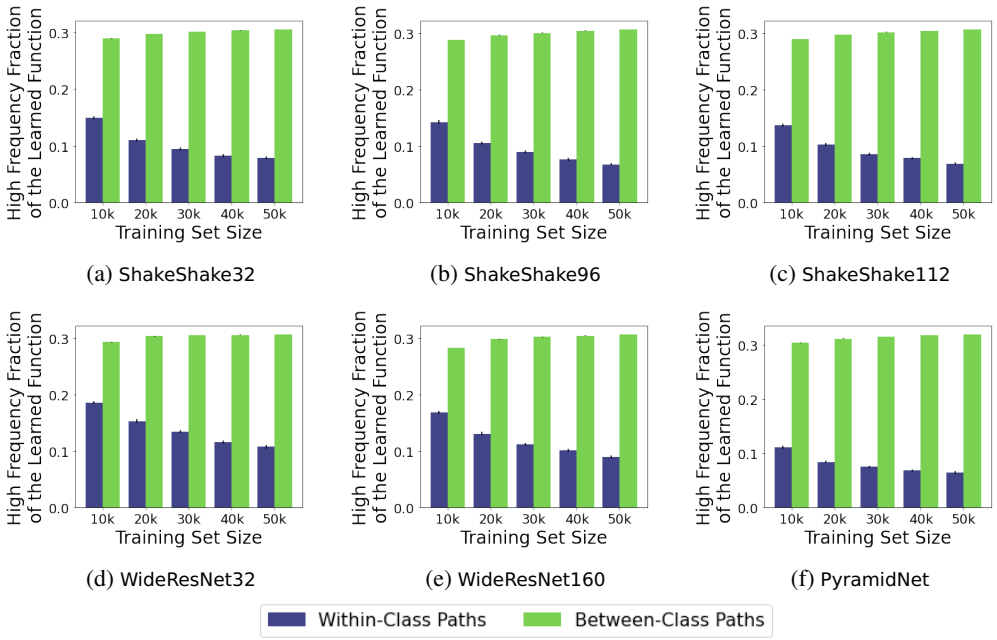


Figure 13: **For all six CIFAR-10 models we tested, increasing the number of training examples decreases the function frequency along within-class paths while slightly increasing the function frequency along between-class paths.**

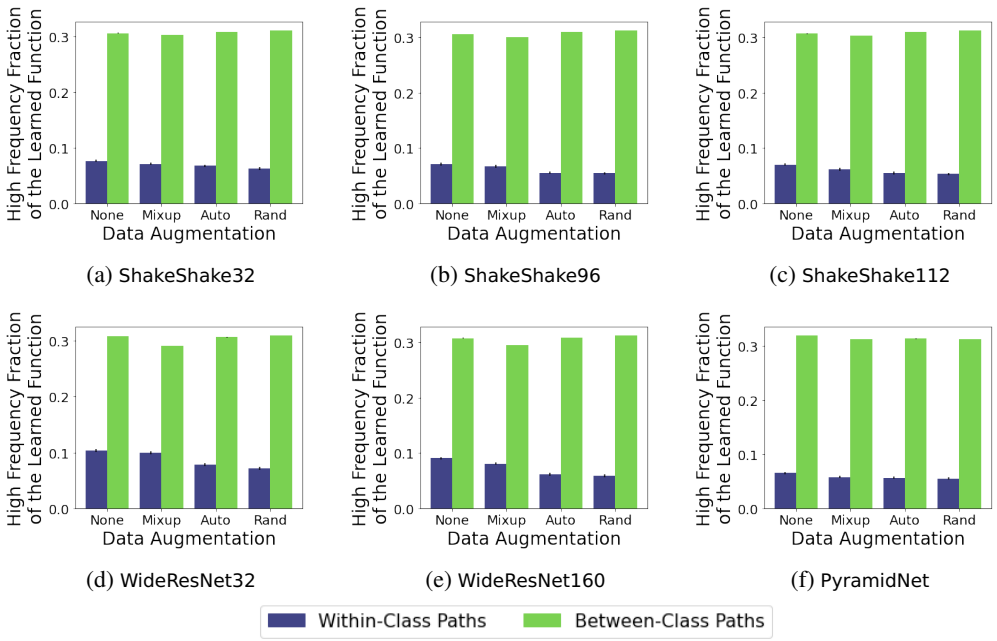


Figure 14: **On all six CIFAR-10 models, more effective data augmentation produces a model that is lower-frequency within-class.** In each figure, model test accuracy increases from left to right, as does the frequency separation between within-class and between-class paths. This trend parallels what we observe when increasing the number of training examples.

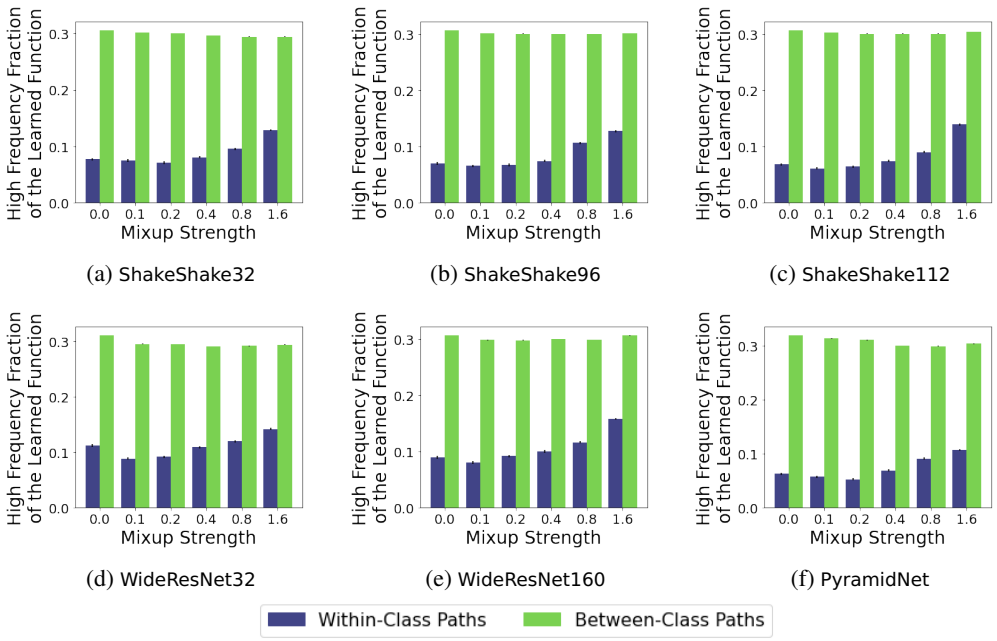


Figure 15: **On all six CIFAR10 models, modest Mixup augmentation produces a within-class lower-frequency learned function, but Mixup that is too strong can induce higher frequencies within-class.**

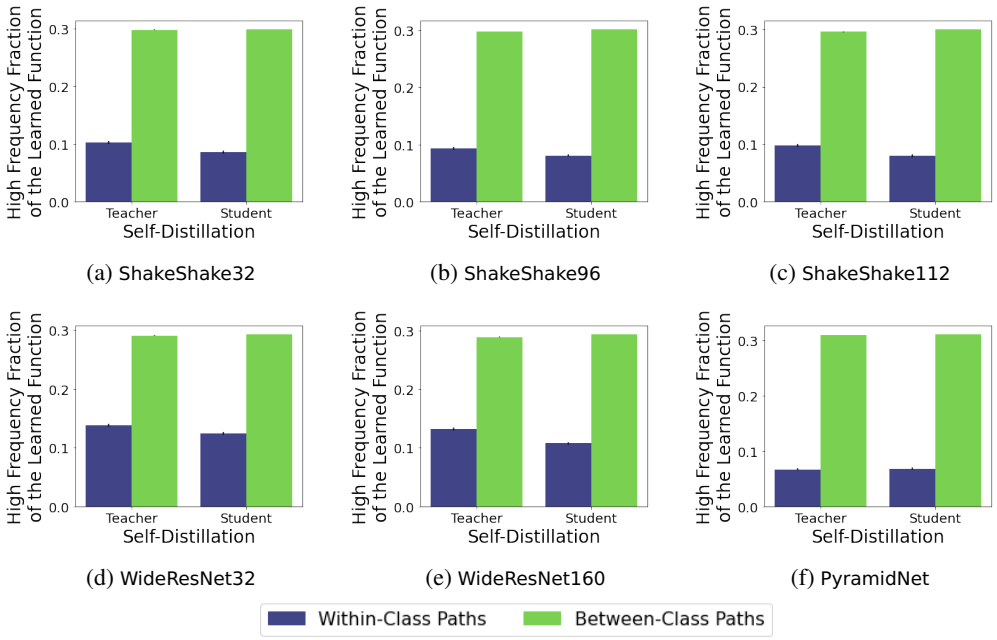


Figure 16: **In our CIFAR-10 interpolation experiments, self-distillation produces a student that is lower-frequency within-class, and has higher test accuracy, compared to its teacher.**

A.4 ImageNet Full Results

A.4.1 Training Time

Figure 7 *(Left)* shows that `ResNet50` increases in between-class frequency as training proceeds, while within-class frequencies remain constant or slightly decrease, causing increasing frequency separation between these two types of paths. In Figure 17 we show the same experiment for all ten ImageNet models we tested. We find that the overall frequency trends vary between models, but all models increase in frequency separation between path types during training.

A.4.2 Class Coherence

Figure 7 *(Right)* shows that `ResNet50` has higher frequencies along paths within more internally diverse classes. In Figure 18 we show the same experiment for all ten ImageNet models we tested. Interestingly, the two CoATNets exhibit no clear correlation between class diversity and function frequency, and also dramatically increase in function frequency during training (Figure 17), unlike other models.

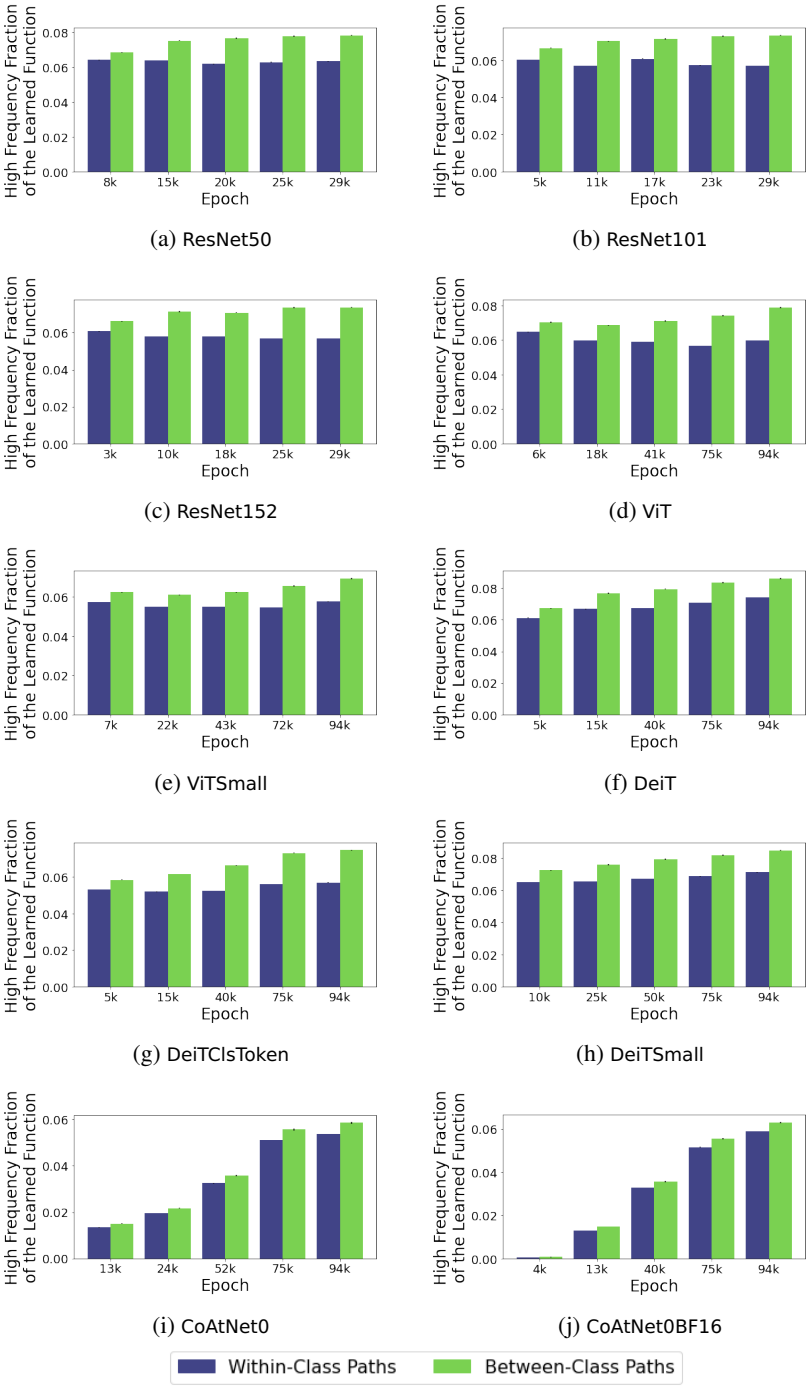
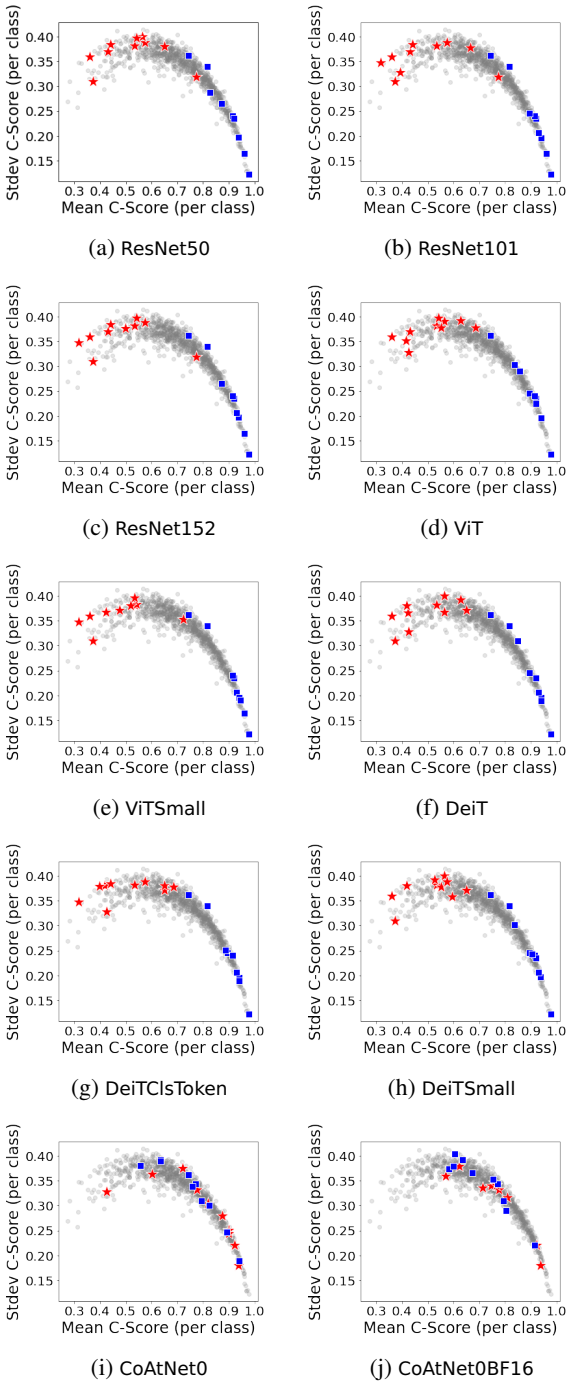


Figure 17: **On all ten ImageNet models, training longer increases the frequency separation between within-class and between-class paths.**



(a) ResNet50 (b) ResNet101

(c) ResNet152 (d) ViT

(e) ViTSmall (f) DeiT

(g) DeiTClsToken (h) DeiTSmall

(i) CoAtNet0 (j) CoAtNet0BF16

Figure 18: **On most of our ImageNet models, paths within more diverse classes have higher function frequencies.** Red stars show the 10 classes with the largest fraction of high-frequency content along within-class paths; blue squares show the 10 classes with the lowest fraction of high-frequency content along within-class paths.