CONVOLUTIONAL NETWORKS INHERIT FREQUENCY SENSITIVITY FROM IMAGE STATISTICS

Anonymous authors
Paper under double-blind review

ABSTRACT

It is widely acknowledged that trained convolutional neural networks (CNNs) have different levels of sensitivity to signals of different frequency. In particular, a number of empirical studies have documented CNNs sensitivity to low-frequency signals. In this work we show with theory and experiments that this observed sensitivity is a consequence of the frequency distribution of natural images, which is known to have most of its power concentrated in low-to-mid frequencies. Our theoretical analysis relies on representations of the layers of a CNN in frequency space, an idea that has previously been used to accelerate computations and study implicit bias of network training algorithms, but to the best of our knowledge has not been applied in the domain of model robustness.

1 INTRODUCTION

Since their rise to prominence in the early 1990s, convolutional neural networks (CNNs) have formed the backbone of image and video recognition, object detection, and speech to text systems (Lecun et al., 1998). The success of CNNs has largely been attributed to their "hard priors" of spatial translation invariance and local receptive fields (Goodfellow et al., 2016, §9.3). On the other hand, more recent research has revealed a number of less desirable and potentially data-dependent biases of CNNs, such as a tendency to make predictions on the basis of texture features (Geirhos et al., 2019). Moreover, it has been repeatedly observed that CNNs are sensitive to perturbations in targeted ranges of the Fourier frequency spectrum (Guo et al., 2019; Sharma et al., 2019) and further investigation has shown that these frequency ranges are dependent on training data (Yin et al., 2019; Bernhard et al., 2021; Abello et al., 2021; Maiya et al., 2022). In this work, we provide a *mathematical explanation* for these frequency space phenomena, showing with theory and experiments that neural network training causes CNNs to be most sensitive to frequencies that are prevalent in the training data distribution.

Our theoretical results rely on representing an idealized CNN in frequency space, a strategy we borrow from (Gunasekar et al., 2018). This representation is built on the classical convolution theorem,

$$\widehat{w * x} = \hat{w} \cdot \hat{x} \tag{1.1}$$

where $\hat{x}$ and $\hat{w}$ denote the Fourier transform of x and w respectively, and $*$ denotes a convolution. Equation 1.1 demonstrates that a Fourier transform converts convolutions into products. As such, in a "cartoon" representation of a CNN in frequency space, the convolution layers become *coordinate-wise multiplications* (a more precise description is presented in section 3). This suggests that in the presence of some form of weight decay, the weights $\hat{w}$ for high-power frequencies in the training data distribution will grow during training, while weights corresponding to low-power frequencies in the training data will be suppressed. The resulting uneven magnitude of the weights $\hat{w}$ across frequencies can thus account for the observed uneven perturbation-sensitivity of CNNs in frequency space. We formalize this argument for linear CNNs (without biases) in sections 3 and 4.

One interesting feature of the framework set up in section 4 is that the discrete Fourier transform (DFT) representation of a linear CNN is *precisely* a feedforward network with block diagonal weight matrices, where each block corresponds to a spatial pixel index. For a linear CNN with one channel in each hidden layer and 1-dimensional output, training such a network of depth L with ℓ_2-weight decay is known to be equivalent to training a linear model with an ℓ_p weight penalty for $p = 2/L$ (Gunasekar et al., 2018; Tibshirani, 2021). In particular, the latter penalty is highly

sparsity-encouraging, suggesting as depth increases these linearly-activated CNNs have an even stronger incentive to prioritize frequencies present in the training data.

It has long been known that the frequency content of natural images is concentrated in low-to-mid frequencies, in the sense that the power in Fourier frequency f is well-described by $1/|f|^\alpha$ for a coefficient $\alpha \approx 1$ (Lee et al., 2001). Hence, when specialized to training data distributions of natural images, our results explain findings that CNNs are more susceptible to low frequency perturbations in practice (Guo et al., 2019; Sharma et al., 2019). We illustrate this by using our theoretical results to derive two specific predictions: deeper models, as well as models trained with substantial weight decay, exhibit frequency sensitivity more closely reflecting the statistics of the underlying images. We confirm these predictions for CNNs trained on the CIFAR10 dataset in section 5.

2 Related work

Perturbations in frequency components: (Guo et al., 2019) proposed algorithms for generating adversarial perturbations constrained to low frequency Fourier components, finding that they allowed for greater query efficiency and higher transferability between different neural networks. (Sharma et al., 2019) demonstrated empirically that constraining to high or or midrange frequencies did *not* produce similar effects, suggesting convolutional networks exhibit a particular sensitivity to low frequency perturbations.

(Yin et al., 2019) showed that different types of corruptions of natural images (e.g. blur, noise, fog) have quite different effects when viewed in frequency space, and that models trained with different augmentation strategies (e.g. adversarial training, gaussian noise augmentation) exhibit different sensitivities to perturbations along Fourier frequency components. (Bernhard et al., 2021) investigated the extent to which constraining models to use only the lowest (or highest) Fourier frequency components of input data provided perturbation robustness, also finding significant variability accross datasets. (Abello et al., 2021) tested the extent to which CNNs relied on various frequency bands by measuring model error on inputs where certain frequencies were removed, again finding a striking amount of variability accross datasets. (Maiya et al., 2022) analyzed the sensitivity of networks to perterbations in various frequncies, finding significant variation accross a variety of datasets and model architectures. All of these works suggest that model frequency sensitivity depends on the underlying training data. — our work began as an attempt to explain this phenomenon mathematically.

Implicit biases of CNNs: Our analysis of (linear) convolutional networks leans heavily on prior work on implicit bias of CNNs, especially (Gunasekar et al., 2018). Those authors found that for a linear convolutional network trained on a binary linear classification task with exponential loss, with linear effective predictor β, the Fourier transformed predictor $\hat{beta}$ converges in direction to a first-order stationary point of an optimization problem of the form

$$\min \frac{1}{2}|\hat{\beta}|_{2/L} \text{ such that } y^n \hat{\beta}^T x^n \geq 1 \text{ for all } n. \tag{2.1}$$

A generalization of this result to arbitrary group-equivariant CNNs (of which the usual CNNs are a special case for products of cyclic groups) appears in (Lawrence et al., 2022, Thm. 1) — while we suspect that some of our results generalize to more general equivariant networks we leave that to future work. For generalizations in different directions see also (Lyu & Li, 2020; Yun et al., 2021). Our general setup closely follows these authors', and our theorem 4.9 partially confirms a suspicion from their (Gunasekar et al., 2018, §6), that "with multiple outputs, as more layers are added, even fully connected networks exhibit a shrinking sparsity penalty on the singular values of the effective linear matrix predictor ..." in the case of two layers.

While (Gunasekar et al., 2018; Lawrence et al., 2022) study the *implicit* regularization imposed by gradient descent, we instead consider *explicit* regularization imposed by auxiliary ℓ_2 norm penalties in objective functions, and prove equivalences of minimization problems. While this approach lacks the intimate connection with the gradient descent algorithms used to train modern neural networks, it comes with some benefits: for example, our results are agnostic to the choice of per-sample loss function (in particular they apply to both squared error and cross entropy loss).

Other applications of Fourier transformed CNNs: (Zhu et al., 2021; Pratt et al., 2017; Mathieu et al., 2014; Vasilache et al., 2015) all, in one way or another, leverage frequency space representations

of convolutions to accellerate computations, e.g. neural network training. Since this is not our main focus, we omit a more detailed synopsis.

3 THE DISCRETE FOURIER TRANSFORM OF A CNN

In this section we fix the notation and structures we will be working with. We define a class of idealized, linear convolutional networks and derive a useful representation of these networks in the frequency space via the discrete Fourier transform.

Consider a linear, feedforward 2D-CNN $f(x)$ of the form

$$\mathbb{R}^{C\times H\times W} \xrightarrow{w^1 * -} \mathbb{R}^{C_1\times H\times W} \xrightarrow{w^2 * -} \mathbb{R}^{C_2\times H\times W} \xrightarrow{w^3 * -} \dots \xrightarrow{w^{L-1} * -} \mathbb{R}^{C_{L-1}\times H\times W} \xrightarrow{w^{L,T} -} \mathbb{R}^K \tag{3.1}$$

where $w^l * x$ denotes the convolution operation between tensors $w^l \in \mathbb{R}^{C_l\times H\times W\times C_{l-1}}$ and $x \in \mathbb{R}^{C_{l-1}\times H\times W}$, defined by

$$(w^l * x)_{cij} = \sum_{m+m'=i, n+n'=j} \Big(\sum_d w^l_{cmnd} x_{dm'n'} \Big) \tag{3.2}$$

and $w^{L,T}x$ denotes a contraction (a.k.a. Einstein summation) of the tensor $w^L \in \mathbb{R}^{K\times H\times W\times C_{L-1}}$ with the tensor $x \in \mathbb{R}^{C_{L-1}\times H\times W}$ over the last 3 indices (the $(-)^T$ denotes a transpose operation described momentarily). Explicitly,

$$(w^{L,T}x)_k = \sum_{l,m,n} w^L_{kmnl} x_{lmn}. \tag{3.3}$$

Thus, the model eq. (3.1) has weights $w_l \in \mathbb{R}^{C_l\times H\times W\times C_{l-1}}$ for $l = 1, \dots, L-1$ and $w_L \in \mathbb{R}^{K\times H\times W\times C_{L-1}}$.

Remarks 3.4. For tensors with at least 3 indices (such as x and the weights w_l above) we will always use the transpose notation $-^T$ to denote reversing the second and third tensor indices, which will always be the 2D spatial indices. For matrices and vectors it will be used according to standard practice. In eq. (3.3) the transpose ensures that the indices in Einstein sums move from "inside to out."

Equivalently, $w^{L,T}x$ can be described as a usual matrix product $\tilde{w}_L \mathrm{vec}(x)$ where $\mathrm{vec}(x)$ is the vectorization (flattening) of x and $\tilde{w}_L$ is obtained by flattening the last 3 tensor indices of w_L (compatibly with those of x as dictated by eq. (3.3)). Hence it represents a typical "flatten and then apply a linear layer" architecture component. Our reason for adopting the tensor contraction perspective is that it is more amenable to the Fourier analysis described below.

Note that in this network the number of channels is allowed to vary but the heights and widths remain fixed, and that we use full $H \times W$ convolutions throughout as opposed to the local (e.g. 3×3) convolutions often occurring in practice.

Given an array $x \in \mathbb{R}^{C_l\times H\times W}$, we may consider its discrete Fourier transform (DFT) $\hat{x}$, whose entries are computed as

$$\hat{x}_{cij} = \frac{1}{\sqrt{HW}} \sum_{m,n} x_{cmn} \exp(-\frac{2\pi\imath}{H} mi - \frac{2\pi\imath}{W} nj). \tag{3.5}$$

Similarly, for an array $w \in \mathbb{R}^{C_l\times H\times W\times C_{l-1}}$ the DFT $\hat{w}$ is defined to be

$$\hat{w}_{cijd} = \frac{1}{\sqrt{HW}} \sum_{m,n} w_{cmnd} \exp(-\frac{2\pi\imath}{H} mi - \frac{2\pi\imath}{W} nj). \tag{3.6}$$

In what follows the DFT will always be taken with respect to the two spatial dimensions and no others. The mapping $x \mapsto \hat{x}$ defines an orthogonal linear transformation of $\mathbb{R}^{C_l\times H\times W}$. In addition, it satisfies the following two properties:

$$\widehat{w * x} = \hat{w} \cdot \hat{x} \text{ for } w \in \mathbb{R}^{C_l\times H\times W\times C_{l-1}}, x \in \mathbb{R}^{C_l\times H\times W} \text{ (convolution theorem)} \tag{3.7}$$

$$w^T x = \hat{w}^T \hat{x} \text{ for } w \in \mathbb{R}^{C_l\times H\times W\times C_{l-1}}, x \in \mathbb{R}^{C_{l-1}\times H\times W} \text{ (Parseval's theorem).} \tag{3.8}$$

Explicitly, the products on the right hand sides of eqs. (3.7) and (3.8) are defined as

$$(\hat{w} \cdot \hat{x})_{cij} = \sum_d \hat{w}_{cijd}\hat{x}_{dij} \text{ and } (\hat{w}^T\hat{x})_c = \sum_{d,i,j} \hat{w}_{cijd}\hat{x}_{dij} \text{ respectively.} \tag{3.9}$$

Remark 3.10. In the terminology of CNNs, this says that the DFT converts full $H \times W$ convolutions to spatially pointwise matrix multiplications, and preserves dot products.

Our first lemma is a mild generalization of (Gunasekar et al., 2018, Lem. 3); we defer all proofs to appendix B.

Lemma 3.11. *The CNN $f(x)$ is functionally equivalent to the network $\hat{f}(\hat{x})$ defined as*

$$\mathbb{C}^{C\times H\times W} \xrightarrow{\hat{-}} \mathbb{C}^{C\times H\times W} \xrightarrow{\hat{w}^1\cdot -} \mathbb{C}^{C_1\times H\times W} \cdots \xrightarrow{\hat{w}^{L-1}\cdot -} \mathbb{C}^{C_{L-1}\times H\times W} \xrightarrow{\hat{w}^{L,T}-} \mathbb{C}^K, \tag{3.12}$$

where the first map $\hat{-}$ denotes the DFT $x \mapsto \hat{x}$.

4 Regularized CNN optimization problems in frequency space

Consider a supervised-learning style optimization problem described as follows: given a dataset $\mathcal{D} = \{(x^n, y^n) \in \mathbb{R}^{C\times H\times W} \times \mathbb{R}^K \mid i = 1, \dots, N\}$, and a loss function $\ell : \mathbb{R}^K \times \mathbb{R}^K \to \mathbb{R}$ (for example, cross-entropy or squared error), we seek weights w solving the ℓ_2-regularized empirical risk minimization problem

$$\begin{aligned} &\min_w \frac{1}{N}\sum_n \ell(f(x^n), y^n) + \lambda \sum_l |w^l|_2^2 \text{ or equivalently given lemma 3.11,} \\ &\min_{\hat{w}} \frac{1}{N}\sum_n \ell(\hat{f}(\hat{x}^n), y^n) + \lambda \sum_l |\hat{w}^l|_2^2 \end{aligned} \tag{4.1}$$

According to lemma 3.11,

$$\hat{f}(\hat{x}) = \hat{w}^{L,T}\big(\hat{w}^{L-1}\cdots\hat{w}^1\cdot\hat{x}\big). \tag{4.2}$$

In the case where the numbers of channels $C, C_1, \dots, C_{L-1}$ are all 1 and the number of classes $K = 1$, networks of this form were studied in (Tibshirani, 2021) where they were termed "simply connected." In the case where the the number of classes $K = 1$ but the numbers of channels C_l may be larger, such networks were studied in (Gunasekar et al., 2018). Of course, eq. (4.2) is just an over-parametrized linear function. We can describe it more succinctly by introducing a new tensor $\hat{v} \in \mathbb{R}^{K\times H\times W\times C}$ such that $\hat{f}(\hat{x}) = \hat{v}^T\hat{x}$. With a little manipulation of eq. (4.2), we can obtain a formula for $\hat{v}$ in terms of the $\hat{w}_l$ for $l = 1, \dots, L$.

Lemma 4.3.

$$\hat{v} = \hat{w}^{L,T}\cdot\hat{w}^{L-1}\cdots\hat{w}^1 \tag{4.4}$$

Remark 4.5. One can view lemma 4.3 an analogue of a much simpler identity for functions $f^1, \dots, f^L, g$ on the real line:

$$\int f^L(t)(f^{L-1}\cdots f^1\cdot g)(t)\,dt = \int f^L(t)\cdot f^{L-1}(t)\cdots f^1(t)\cdot g(t)\,dt = \int (f^L\cdot f^{L-1}\cdots f^1)(t)g(t)\,dt$$

The following theorem states that the regularization term of eq. (4.1), which penalizes the ℓ_2-norms of the factors $\hat{w}^l$, is equivalent to a penalty using more sparsity-encouraging norms of $\hat{v}$. To state it we need a definition.

Definition 4.6 (C_p-norms)**.** Let $A = (a_{ij}) \in M(n \times n, \mathbb{C})$ be a square matrix and let $A = UDV^T$ be a singular value decomposition of A, where $U, V \in U(n)$ and $D = \mathrm{diag}(\lambda_i)$ is a non-negative diagonal matrix with diagonal entries $\lambda_1, \dots, \lambda_n \geq 0$. For any $p > 0$ the C_p**-norm of** A is

$$|A|_p = \Big(\sum_n |\lambda_i|^p\Big)^{\frac{1}{p}}. \tag{4.7}$$

Remarks 4.8. The norms in definition 4.6 are also known as **Schatten** p-norms.

In the case $p = 2$, the 2-norm of definition 4.6 agrees with the usual Euclidean 2-norm $(\sum_{i,j}|a_{ij}|^2)^{\frac{1}{2}}$, since left (resp. right) multiplication by a unitary matrix U^T (resp. V) preserves the Euclidean 2-norms of the columns (resp. rows) of A. Hence there is no notational clash between the 2-norms of definition 4.6 and those occuring previously.

Theorem 4.9. *The optimization problem eq. (4.1) is equivalent to an optimization problem for $\hat{v}$ of the form*

$$\min_{\hat{v}} \frac{1}{N} \sum_n \ell(\hat{v}^T \hat{x}^n, y^n) + \lambda L \sum_{i,j} |\hat{v}_{ij}|_{\frac{2}{L}}^{\frac{2}{L}}, \tag{4.10}$$

where[1] *$\hat{v}_{ij}$ denotes the $K \times C$ matrix obtained by fixing the spatial indices of $\hat{v} = (\hat{v}_{cijd})$,* in the following 2 cases*:*

(i) $C = K = 1$ and $C_l = 1$ for all l.
(ii) $L = 2$.

Given section 3, both (i) and (ii) are straightforward applications of (Tibshirani, 2021). Moreover, case (ii) hinges on a result of (Srebro et al., 2004): for a matrix factorization $V = W_1 W_2$, the inequality $|V|_1 \leq |W_1|_2 |W_2|_2$ holds, and moreover

$$|V|_1 = \min_{W_1 W_2 = V} |W_1|_2 |W_2|_2 = \min_{W_1 W_2 = V} \frac{1}{2}(|W_1|_2^2 + |W_2|_2^2) \tag{4.11}$$

In the case where the numbers of channels $C, C_1, \ldots, C_{L-1}$ are all 1 and and the number of outputs $K = 1$, *and* where the loss ℓ is squared error, the problem eq. (4.10) reduces to

$$\min_{\hat{v}} \frac{1}{N} \sum_n |y^n - \sum_{i,j} \hat{v}_{ij} \hat{x}_{ij}|_2^2 + \lambda L \sum_{i,j} |\hat{v}_{ij}|^{\frac{2}{L}}. \tag{4.12}$$

The sum in the regularization term of eq. (4.12) is $|\hat{v}|_p^p$ where $p = \frac{2}{L}$ — in particular when $L = 1$, eq. (4.12) is a ridge regression problem (Hastie et al., 2001; Tibshirani & Wasserman) and when $L = 2$ (the one hidden layer case) eq. (4.12) is a LASSO problem. We can analyze these two tractable cases to obtain qualitative predictions which will be tested empirically in section 5. Since the qualitative predictions from both cases are similar, we devote the following section to ridge, and defer LASSO to appendix A.

We conjecture that theorem 4.9 is true without restrictions on C, K and C_l for $l = 1, \ldots, L-1$. We are only able to prove this *conditional on* a separate conjecture, which asserts a generalized non-commutative Hölder inequality:

Conjecture 4.13. *If $B \in M(m \times n, \mathbb{C})$ is a matrix with complex entries, $A_1, \ldots, A_k$ is a composable sequence of complex matrices such that $A_1 \cdots A_L = B$ and $\sum_i \frac{1}{p_1} = \frac{1}{r}$ where $p_1, \ldots, p_L, r > 0$ are positive real numbers,*

$$|B|_r \leq \prod_i |A_i|_{p_i} \tag{4.14}$$

In the case $r = 1$, conjecture 4.13 follows from (Dixmier, 1953, Thm. 6). To the best of our knowledge for $r \neq 1$ it is open.

4.1 $L = 1$: RIDGE REGRESSION

In this case, eq. (4.12) is the usual ridge regression objective; the closed-form solution is

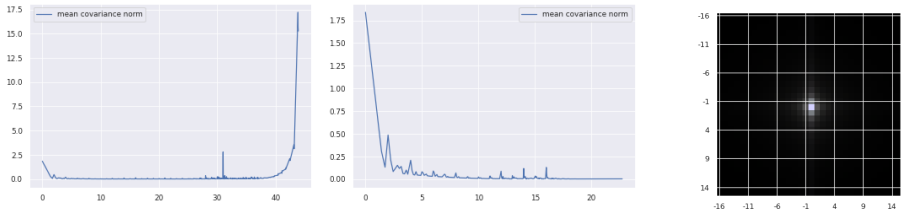
$$(\lambda + \frac{1}{N} \hat{X}^T \hat{X}) \hat{v} = \frac{1}{N} \hat{X}^T Y \tag{4.15}$$

where $\hat{X}$ is a $N \times C \times H \times W$ batch tensor with "rows" the $\hat{x}^n$ and the entries of Y are the y^n (see e.g. (Hastie et al., 2001)). When $\lambda = 0$ this reduces to the usual (unpenalized) least squares solution $\hat{v}_{\text{LS}} := (\hat{X}^T \hat{X})^{-1} \hat{X}^T Y$, and substituting $\hat{X}^T Y = \hat{X}^T \hat{X} \hat{v}_{\text{LS}}$ in eq. (4.15) we obtain

$$(\lambda + \frac{1}{N} \hat{X}^T \hat{X}) \hat{v} = \frac{1}{N} \hat{X}^T \hat{X} \hat{v}_{\text{LS}} \tag{4.16}$$

where strictly speaking $\hat{X}^T \hat{X}$ is a tensor product of $\hat{X}$ with itself in which we contract over the batch index of length N, hence it is of shape $W \times H \times C \times C \times H \times W$.

[1] by an abuse of notation for which we beg your forgiveness.



(a) (b) (c)

Figure 1: Testing the assumption of eq. (4.17) on the CIFAR10 dataset. **(a)** Average covariance entry norm $E[|\Sigma_{jicc'i'j'}| \mid |(c,i,j)-(c,i',j')| = r]$ plotted with respect to $r = |(i,j)-(i',j')|$. **(b)** Same as (a), except the i,j part of the distance r is computed *modularly*, i.e. on the discrete torus $\mathbb{Z}/H \times \mathbb{Z}/W$ instead of the discrete square $\{0,\dots,H-1\} \times \{0,\dots,W-1\}$. For more details see appendix C. **(c)** Variance of the CIFAR10 dataset along Fourier basis vectors, i.e. the "diagonal" entries Σ_{jiccij} (viewed as an RGB image). The origin corresponds to the lowest frequency basis vectors (i.e. constant images).

The frequency properties of images enter into the structure of the symmetric tensor $\frac{1}{N}\hat{X}^T\hat{X}$, which (if the dataset $\mathcal{D}$ is centered, i.e. preprocessed by subtracting the mean $\frac{1}{N}X^T\mathbf{1}_N$) serves as a generalized *covariance matrix* for the frequency space representation of $\mathcal{D}$. To ease notation, let $\Sigma = \frac{1}{N}\hat{X}^T\hat{X}$, and *suppose* that

$$\Sigma_{whcc'h'w'} \approx \begin{cases} \tau_{cij} & \text{if } (c,i,j) = (c',i',j') \\ 0 \text{ otherwise.} \end{cases} \tag{4.17}$$

In other words, proper covariances between distinct frequency components are negligible and we retain only the *variances*, i.e. the diagonal entries of the covariance matrix. In fig. 1 we demonstrate that this assumption is not unrealistic in the case where X is a dataset of natural images.

With the assumption of eq. (4.17), eq. (4.16) reduces to

$$\hat{v}_{cij} = \frac{\tau_{cij}}{\lambda + \tau_{cij}}\hat{v}_{\mathrm{LS},cij} = \frac{1}{1+\frac{\lambda}{\tau_{cij}}}\hat{v}_{\mathrm{LS},cij} \text{ for all } cij. \tag{4.18}$$

Equation (4.18) is an instance of the classic fact that ridge regression shrinks coefficients more in directions of low input variance. In words, τ_{cij} is the variance of training images in Fourier component cij, and eq. (4.18) says $|\hat{v}_{cij}|$ shrinks more when the variance τ_{cij} is low; in the limiting case $\tau_{cij} \to 0$ the coefficient $\hat{v}_{cij} \to 0$ as well.

Returning to the subject of frequency sensitivity, observe that $\hat{v}_{cij}$ is the directional derivative of f with respect to the cij-th Fourier component.

Proposition 4.19 (Data-dependent frequency sensitivity, $L = 1$)**.** *With the notations and assumptions introduced above, the magnitude of the directional derivative of f with respect to the cij-th Fourier component scales with λ and τ according to $\frac{1}{1+\frac{\lambda}{\tau_{cij}}}$.*

Empirically it has been found that for natural distributions of images the variances τ_{cij} follow a power law of the form $\tau_{cij} \approx \frac{\gamma}{|i|^\alpha + |j|^\beta}$ (Lee et al., 2001; Baradad et al., 2021).[2] Under this model, eq. (4.18) becomes

$$\hat{v}_{cij} = \frac{1}{1+\frac{\lambda}{\gamma}(|i|^\alpha + |j|^\beta)}\hat{v}_{\mathrm{LS},cij} \text{ for all } cij, \tag{4.20}$$

that is, sensitivity is monotonically decreasing with respect to both frequency magnitude and the regularization coefficient λ. This is consistent with findings that CNNs trained on natural images are vulnerable to low frequency perturbations (Guo et al., 2019; Sharma et al., 2019).

[2] In particular, under this approximation τ_{cij} is independent of c.

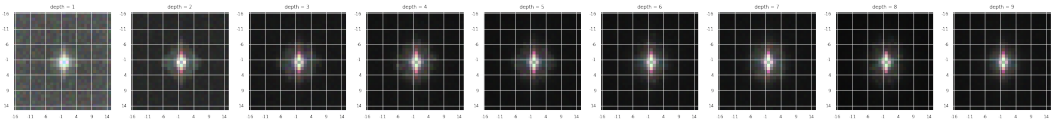


Figure 2: Frequency sensitivity $E[|\nabla_x f(x)^T \hat{e}_{cij}|]$ of ConvActually models of varying depth. We see that as depth increases, the frequency sensitivity images more and more closely resemble the statistics of the training data images shown in fig. 1c. Accuracy of the models ranged from 57% (depth = 1) to 66% (depth=9), roughly monotonically increasing with depth.

5 Experiments

When $L > 2$ (that is, when there are more than 1 convolutional layers), the C_p "norm" of eq. (4.10) is non-convex (hence the quotes), and in the limit as $L \to \infty$ we obtain the C_0 "norm," which is simply the number of non-0 singular values of $\hat{v}$.[3] Moreover, we see that the regularization coefficient of eq. (4.10) is effectively multiplied by L.

Even in the case where $K = 1$ so that $\hat{v}$ is a vector, it is known that solving eq. (4.10) for $L > 1$ is NP-hard (Chen et al., 2017), so we have no hope of finding closed form solutions as in section 4.1. However, we can use the analysis in section 4 to derive two testable hypotheses:

I. The fact that the regularization term of eq. (4.10) becomes more sparsity-encouraging as L increases suggests that the data-dependent frequency sensitivity observed in section 4.1 and appendix A becomes *even more pronounced* as the number of convolutional layers increases.
II. Moreover, the functional forms of eqs. (4.18) and (A.7) suggest that the data-dependent frequency sensitivity will increase monotonically with the weight decay parameter λ.

We empirically validate these hypotheses with experiments using CNNs trained on the CIFAR10 dataset. We measure frequency sensitivity of a CNN f in terms of the magnitudes of the directional derivatives $\nabla_x f(x)^T \hat{e}_{cij}$, where $\hat{e}_{cij}$ is the cij-th Fourier basis vector. These magnitudes are averaged over all the input images x in the CIFAR10 validation set. Thus, by "data-dependent frequency sensitivity" we mean the extent to which the frequency sensitivity of a CNN f reflects the statistics of the DFTed images $\hat{x}$. Figure 1c shows that the variance of DFTed CIFAR10 images is heavily concentrated at low frequencies, in agreement with the power law form $\tau_{cij} \approx \frac{\gamma}{|i|^\alpha + |j|^\beta}$ described in section 4. Hence in the absence of any modifications to the underlying images, we expect that increasing model depth and the weight decay parameter λ will emphasize sensitivity of f to perturbations along the lowest frequency Fourier basis vectors.

The CNNs used in these experiments are:

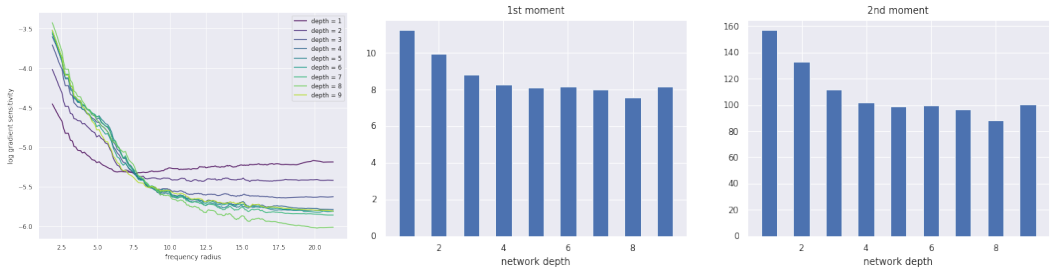
- A CNN, which we refer to as ConvActually, that closely approximates eq. (3.1), the only differences being the addition of biases and ReLU non-linearities. This is accomplished by applying convolutions with 32×32 kernels (i.e., kernels of the same size as the input images) with circular padding. This network has constant channel dimension 64.
- A small feed-forward CNN known to achieve high performance on CIFAR10 obtained from (Page), which we refer to as MyrtleCNN. This CNN has small kernels, ReLU non-linearities and $\max$-pooling, as well as exponentially varying channel dimension.

For more detailed architecture summaries we refer to appendix C.5.

5.1 Frequency sensitivity and depth

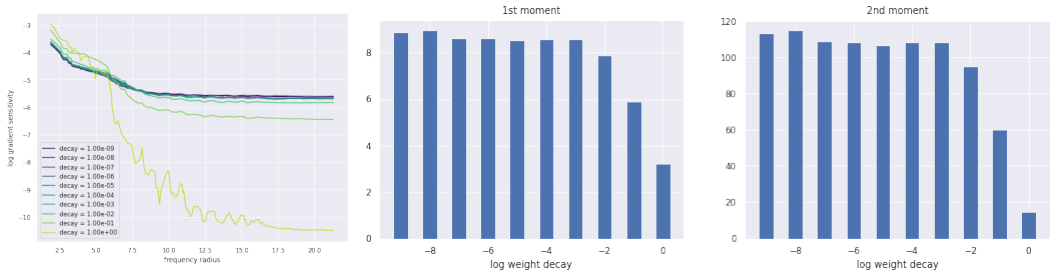
Figure 2 shows the raw frequency sensitivity images of ConvActually models of varying depth, and visually corroborates hypotheses I. We can further summarize the data displayed in fig. 2 by averaging over circles with varying radii r, to obtain the frequency sensitivity curves shown in fig. 3a. These curves illustrate that as depth increases, frequency sensitivity $E[|\nabla_x f(x)^T \hat{e}_{cij}|]$ decreases more and more rapidly with respect to $r = |(i, j)|$. We can capture this phenomena concisely with a single

[3] The special case where $K = 1$ is the penalty of the *subset selection* problem in the field of sparse linear models.



(a) (b)

Figure 3: **(a)** Radial averages $E[|\nabla_x f(x)^T \hat{e}_{cij}| \,|\, |(i,j)| = r]$ of the images in fig. 2, further post-processed by dividing each curve by its integral (to obtain probability distributions), taking logarithms (to obtain log-ed probability distributions) and finally smoothing by averaging with 3 neighbors on either side. **(b)** First and second moments of the radially averaged frequency sensitivities $E[|\nabla_x f(x)^T \hat{e}_{cij}| \,|\, |(i,j)| = r]$ (*without* the post-processing used in (a)).



(a) (b)

Figure 4: **(a)** Radial averages $E[|\nabla_x f(x)^T \hat{e}_{cij}| \,|\, |(i,j)| = r]$ of the ConvActually frequency sensitivities in fig. 7a, further post-processed as described in fig. 3a. **(b)** First and second moments of the radially averaged frequency sensitivities $E[|\nabla_x f(x)^T \hat{e}_{cij}| \,|\, |(i,j)| = r]$ (*without* the post-processing used in (a)).

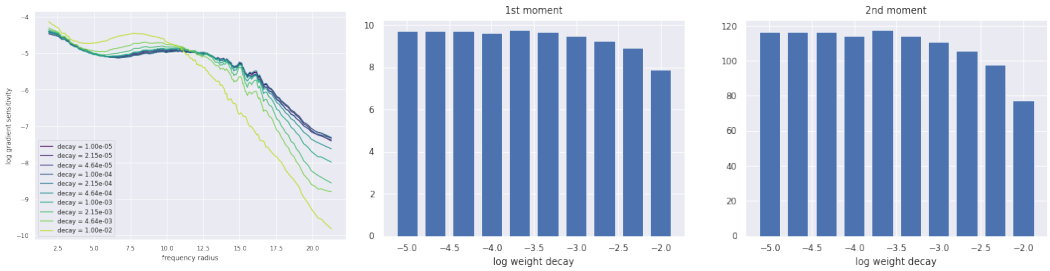
summary statistic: by definition for a (well-behaved) non-negative function $g : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$, the **$k$-th moment of** g is

$$\frac{\int_0^\infty r^k g(r)\, dr}{\int_0^\infty g(r)\, dr} \tag{5.1}$$

(equivalently the expectation of r^k with respect to the distribution obtained by dividing $g(r)$ by its integral). The first moment can be viewed as a center of mass, the second as a moment of inertia about $r = 0$. In fig. 3b we see that the first and second moments of the radial frequency sensitivity curves in fig. 3a are roughly monotonically decreasing with respect to depth.

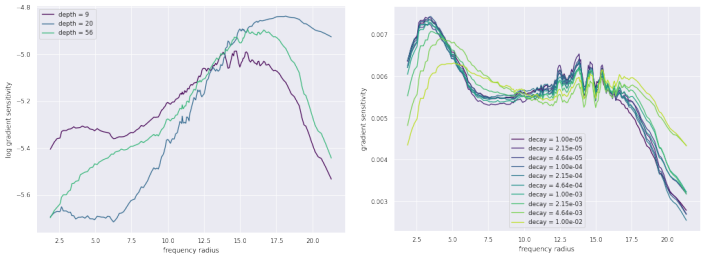
5.2 Frequency sensitivity and weight decay

Figures 4 and 5 show radial frequency sensitivity curves, as well as their first and second moments, for ConvActually and MyrtleCNN models trained with varying weight decay coefficient λ. For both models, we see that the first and second moments of the radial frequency sensitivity curves are roughly monotonically decreasing with respect to λ. This suggests that as λ increases, model frequency sensitivity more and more closely reflects the statistics of the training data images shown in fig. 1c, which would corroborate hypothesis II.. The frequency sensitivity images for these models can be found in Figure 7 in the appendix.



(a) (b)

Figure 5: **(a)** Radial averages $E[|\nabla_x f(x)^T \hat{e}_{cij}| \,|\, |(i,j)| = r]$ of the of the MyrtleCNN frequency sensitivities in fig. 7b, further postprocessed as described in fig. 3a. **(b)** First and second moments of the radially averaged frequency sensitivities $E[|\nabla_x f(x)^T \hat{e}_{cij}| \,|\, |(i,j)| = r]$ (*without* the postprocessing used in (a)).



(a) (b)

Figure 6

6 Limitations and open questions

In order to obtain an optimization problem with some level of analytical tractibility, we made many simplifying assumptions in sections 3 and 4, most notably omitting nonlinearities from our idealized CNNs. While the experimental results of section 5 illustrate that multiple predictions derived from sections 3 and 4 hold true for CNNs more closely resembling those used in practice, fig. 6 shows that hypotheses I-II can fail in the presence of residual connections. There is a simple heuristic explanation for this fact: if we add residual connections to eq. (3.1), then as the weights $w^l \to 0$, $f \to \mathrm{id}$ (the identity) as a function — in contrast, without residual connections $f \to 0$. Since the identity function has equal sensitivity to all frequencies, we would not expect model sensitivity concentrated in low frequencies.

While our theoretical results suggest that CNN frequency sensitivity reflects the variance of DFTed image datasets in general, that is, even when they are not concentrated in low-to-mid frequencies like natural image datasets such as CIFAR10, our experiments are regretably limited to one dataset of natural images. Worthwhile experiments for future work include training on high-pass-filtered images as in (Yin et al., 2019), incorporating higher resolution image datasets, and training on computer-generated images with prescribed frequency statistics as in (Baradad et al., 2021).

Perhaps more significantly, it must be emphasized that model sensitivity as measured by gradients represents a very small corner of a broader picture of model robustness (or lack therof). For example, it does not encompass model behaviour on corruptions (see e.g. (Hendrycks & Dietterich, 2019)) or shifted distributions (see e.g. (Recht et al., 2019)).

References

Antonio A. Abello, Roberto Hirata, and Zhangyang Wang. Dissecting the high-frequency bias in convolutional neural networks. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 863–871, 2021. doi: 10.1109/CVPRW53098.2021.00096.

Manel Baradad, Jonas Wulff, Tongzhou Wang, Phillip Isola, and Antonio Torralba. Learning to see by looking at noise. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=RQUl8gZnN7O.

Rémi Bernhard, Pierre-Alain Moëllic, Martial Mermillod, Yannick Bourrier, Romain Cohendet, Miguel Solinas, and Marina Reyboz. Impact of spatial frequency based constraints on adversarial robustness. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2021. doi: 10.1109/IJCNN52387.2021.9534307.

Yichen Chen, Dongdong Ge, Mengdi Wang, Zizhuo Wang, Yinyu Ye, and Hao Yin. Strong NP-hardness for sparse optimization with concave penalty functions. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 740–747. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/chen17d.html.

Jacques Dixmier. Formes linéaires sur un anneau d'opérateurs. *Bulletin de la Société Mathématique de France*, 81:9–39, 1953. doi: 10.24033/bsmf.1436.

Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bygh9j09KX.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

Suriya Gunasekar, Jason D Lee, Daniel Soudry, and Nati Srebro. Implicit bias of gradient descent on linear convolutional networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Chuan Guo, Jared S. Frank, and Kilian Q. Weinberger. Low Frequency Adversarial Perturbation. In *UAI*, 2019.

Trevor J. Hastie, Robert Tibshirani, and Jerome H. Friedman. The elements of statistical learning: Data mining, inference, and prediction, 2nd edition. In *Springer Series in Statistics*, 2001.

Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HJz6tiCqYm.

Hannah Lawrence, Bobak Kiani, Kristian G. Georgiev, and Andrew K. Dienes. Implicit Bias of Linear Equivariant Networks. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 12096–12125. PMLR, June 2022.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

Ann B. Lee, David Mumford, and Jinggang Huang. Occlusion Models for Natural Images: A Statistical Study of a Scale-Invariant Dead Leaves Model. *International Journal of Computer Vision*, 41(1):35–59, January 2001. ISSN 1573-1405. doi: 10.1023/A:1011109015675.

Kaifeng Lyu and Jian Li. Gradient Descent Maximizes the Margin of Homogeneous Neural Networks, December 2020.

Shishira Maiya, Max Ehrlich, Vatsal Agarwal, Ser-Nam Lim, Tom Goldstein, and Abhinav Shrivastava. A frequency perspective of adversarial robustness. 2022. URL https://openreview.net/forum?id=7gRvcAulxa.

Michael Mathieu, Mikael Henaff, and Yann LeCun. Fast training of convolutional networks through ffts: International conference on learning representations (iclr2014), cbls, april 2014. January 2014. 2nd International Conference on Learning Representations, ICLR 2014 ; Conference date: 14-04-2014 Through 16-04-2014.

David Page. How to Train Your ResNet. URL https://myrtle.ai/learn/how-to-train-your-resnet/.

Harry Pratt, Bryan M. Williams, Frans Coenen, and Yalin Zheng. Fcnn: Fourier convolutional neural networks. In *ECML/PKDD*, 2017.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet classifiers generalize to ImageNet? In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5389–5400. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/recht19a.html.

Yash Sharma, G. Ding, and Marcus A. Brubaker. On the Effectiveness of Low Frequency Perturbations. 2019. doi: 10.24963/ijcai.2019/470.

Nathan Srebro, Jason Rennie, and Tommi Jaakkola. Maximum-Margin Matrix Factorization. In *Advances in Neural Information Processing Systems*, volume 17. MIT Press, 2004.

The Mosaic ML Team. composer. https://github.com/mosaicml/composer/, 2021.

Ryan Tibshirani and Larry Wasserman. Sparsity, the Lasso, and Friends. pp. 34.

Ryan J Tibshirani. Equivalences Between Sparse Models and Neural Networks. pp. 8, 2021. URL https://www.stat.cmu.edu/~ryantibs/papers/sparsitynn.pdf.

Nicolas Vasilache, Jeff Johnson, Michael Mathieu, Soumith Chintala, Serkan Piantino, and Yann LeCun. Fast convolutional nets with fbfft: A gpu performance evaluation. January 2015. 3rd International Conference on Learning Representations, ICLR 2015 ; Conference date: 07-05-2015 Through 09-05-2015.

Dong Yin, Raphael Gontijo Lopes, Jonathon Shlens, Ekin Dogus Cubuk, and Justin Gilmer. A fourier perspective on model robustness in computer vision. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 13255–13265, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/b05b57f6add810d3b7490866d74c0053-Abstract.html.

Chulhee Yun, Shankar Krishnan, and Hossein Mobahi. A unifying view on implicit bias in training linear neural networks. In *International Conference on Learning Representations*, March 2021.

Xiaohan Zhu, Zhen Cui, Tong Zhang, Yong Li, and Jian Yang. Going deeper in frequency convolutional neural network: A theoretical perspective. *ArXiv*, abs/2108.05690, 2021.

A $L = 2$: LASSO PARAMETER SHRINKING AND SELECTION

When $C_l = 1$ for all l and $L = 1$, we may simplify eqs. (4.10) and (4.12) to

$$\min_{\hat{v}} \frac{1}{N}|Y - \hat{X}\hat{v}|_2^2 + 2\lambda|\hat{v}|_1 \tag{A.1}$$

where $\hat{X}$ and Y are as in section 4.1. The optimality criterion for eq. (4.10) becomes (see e.g. (Tibshirani & Wasserman))

$$\frac{1}{N}(\hat{X}^T\hat{X}\hat{v} - \hat{X}^T Y) + \lambda\nabla|\hat{v}|_1 = 0, \tag{A.2}$$

where $\nabla|\hat{v}|_1$ is the *sub-gradient* of the ℓ_1-norm:

$$(\nabla|\hat{v}|_1)_i \begin{cases} = \operatorname{sign}(\hat{v}_i) & \text{if } \hat{v}_i \neq 0 \\ \in [-1, 1] & \text{if } \hat{v}_i = 0. \end{cases} \tag{A.3}$$

When $\lambda = 0$ this again reduces to the unpenalized least squares solution $\hat{v}_{\text{LS}} := (\hat{X}^T\hat{X})^{-1}\hat{X}^T Y$, and substituting this in eq. (A.2) we obtain

$$\frac{1}{N}(\hat{X}^T\hat{X}\hat{v} - \hat{X}^T\hat{X}\hat{v}_{\text{LS}}) + \lambda\nabla|\hat{v}|_1 = 0. \tag{A.4}$$

If we again make the assumption that $\Sigma = \frac{1}{N}X^T X$ is "diagonal" as in eq. (4.17), eq. (A.2) simplifies to

$$\tau_{cij}(\hat{v}_{cij} - \hat{v}_{\text{LS},cij}) + \lambda\frac{d|\hat{v}_{cij}|}{d\hat{v}_{cij}} = 0 \text{ for all } cij, \tag{A.5}$$

where strictly speaking $\frac{d|\hat{v}_{cij}|}{d\hat{v}_{cij}}$ is a subgradient as in eq. (A.3). From this we conclude

$$\hat{v}_{cij} \begin{cases} = \hat{v}_{\text{LS},cij} - \frac{\lambda}{\tau_{cij}}\operatorname{sign}(\hat{v}_{cij}), & \text{if } \hat{v}_{cij} \neq 0 \\ \in [\hat{v}_{\text{LS},cij} - \frac{\lambda}{\tau_{cij}}, \hat{v}_{\text{LS},cij} + \frac{\lambda}{\tau_{cij}}] & \text{if } \hat{v}_{cij} = 0 \end{cases} \text{ for all } cij \tag{A.6}$$

(the second case is equivalent to: if the LASSO solution $\hat{v}_{cij} = 0$, it must be that the least squares solution satisfies $|\hat{v}_{\text{LS},cij}| \leq \frac{\lambda}{\tau_{cij}}$). In the case where $\operatorname{sign}(\hat{v}_{cij}) = \operatorname{sign}(\hat{v}_{\text{LS},cij})$, we obtain a particularly nice conclusion:

$$|\hat{v}_{cij}| = |\hat{v}_{\text{LS},cij}| - \frac{\lambda}{\tau_{cij}} \text{ for all } cij \tag{A.7}$$

Proposition A.8 (Data-dependent frequency sensitivity, $L = 2$)**.** *With the notations and assumptions introduced above, the magnitude of the directional derivative of f with respect to the cij-th Fourier component is linear in $\frac{\lambda}{\tau_{cij}}$ with slope -1.*

If we again plug in the empirically determined power law observed in natural imagery, $\tau_{cij} \approx \frac{\gamma}{|i|^\alpha + |j|^\beta}$, eq. (A.7) becomes

$$|\hat{v}_{cij}| = |\hat{v}_{\text{LS},cij}| - \frac{\lambda}{\gamma}(|i|^\alpha + |j|^\beta) \text{ for all } cij. \tag{A.9}$$

Here, the sensitivity decreases monotonically with respect to both frequency magnitude and the regularization coefficient λ. Note that compared to eq. (4.20) from section 4, eq. (A.9) implies a greater shrinking effect when $|\hat{v}_{cij}| < 1$, and less shrinking when $|\hat{v}_{cij}| > 1$.[4] We conjecture that shrinkage due to $|\hat{v}_{cij}| < 1$ is the dominant effect, due to the initial distribution of weights w in modern neural networks, which are often sampled from uniform or normal distributions with variance $\ll 1$ (e.g., according to He initialization; He et al., 2016).

B Proofs

B.1 Proof of lemmas 3.11 and 4.3

Proof of lemmas 3.11 and 4.3. We proceed by induction on L. In the base case where $L = 0$, lemma 3.11 is equivalent to Parseval's theorem eq. (3.8). So, suppose $L > 0$. We may decompose f as a convolution followed by a linear CNN of the form eq. (3.1), say g, with one fewer layers. Explicitly, g has weights $w_2, \ldots, w_L$ and

$$f(x) = g(w_1 * x). \tag{B.1}$$

By inductive hypothesis, we may assume that

$$g(w_1 * x) = \hat{g}(\widehat{w_1 * x}) = \hat{w}^{L,T}\big((\prod_{l=1}^{L-1}\hat{w}_{L-l,\ldots ij})\widehat{w_1 * x}_{:,ij}\big) \tag{B.2}$$

[4]This remark applies generally to LASSO vs. ridge regression, and is perhaps most easily explained by comparing the derivatives of $|x|$ and x^2.

Applying the convolution theorem eq. (3.7) to obtain

$$\widehat{w_1 * x}_{:ij} = \hat{w}_{1,\dots,ij}\hat{x}_{:,ij} \tag{B.3}$$

completes the proof. □

Proof of lemma 4.3. For each weight $\hat{w}^l$ let $\hat{w}^l_{ij}$ denote the $C_l \times C_{l-1}$ matrix obtained by fixing the spatial indices of $\hat{w}^l$ to ij. Unpacking definitions,

$$\begin{aligned}\left(\hat{w}^{L-1}\cdot\hat{w}^{L-1}\cdots\hat{w}^1\cdot\hat{x}\right)_{ij} &= \hat{w}^{L-1}_{ij}\cdot\hat{w}^{L-1}_{ij}\cdots\hat{w}^1_{ij}\cdot\hat{x}_{ij} \text{ and so}\\ \hat{w}^{L,T}\left(\hat{w}^{L-1}\cdot\hat{w}^{L-1}\cdots\hat{w}^1\cdot\hat{x}\right)_c &= \sum_{i,j}\sum_d (\hat{w}^L_{ij})_{cd}\left(\hat{w}^{L-1}_{ij}\cdot\hat{w}^{L-1}_{ij}\cdots\hat{w}^1_{ij}\cdot\hat{x}_{ij}\right)_d\end{aligned} \tag{B.4}$$

We can recognize the inside sum as performing a matrix product of $\hat{w}^L_{ij}$ with $\hat{w}^{L-1}_{ij}\cdot\hat{w}^{L-1}_{ij}\cdots\hat{w}^1_{ij}\cdot\hat{x}_{ij}$. Since matrix multiplication is associative, we can just as well multiply the $\hat{w}^l_{ij}$ first and *then* act on the vector $\hat{x}_{ij}$. Thus

$$\sum_{i,j}\sum_d (\hat{w}^L_{ij})_{cd}\left(\hat{w}^{L-1}_{ij}\cdot\hat{w}^{L-1}_{ij}\cdots\hat{w}^1_{ij}\cdot\hat{x}_{ij}\right)_d = \sum_{i,j}\sum_d \left(\hat{w}^L_{ij}\cdot\hat{w}^{L-1}_{ij}\cdot\hat{w}^{L-1}_{ij}\cdots\hat{w}^1_{ij}\right)_{cd}\cdot(\hat{x}_{ij})_d \tag{B.5}$$

Now we can recognize the right hand side as $(\hat{w}^L\cdot\hat{w}^{L-1}\cdots\hat{w}^1)^T\hat{x}$, as claimed. □

B.2 Proof of theorem 4.9 (assuming conjecture 4.13 when necessary)

Recall we aim to prove: the optimization problem

$$\min_{\hat{w}} \frac{1}{N}\sum_n \ell\left((\hat{w}^L\cdot\hat{w}^{L-1}\cdot\hat{w}^{L-1}\cdots\hat{w}^1)^T\hat{x}^n, y^n\right) + \lambda\sum_{l,i,j}|\hat{w}^l_{ij}|_2^2 \tag{B.6}$$

is equivalent to the following optimization problem for the product $\hat{v}_{\dots ij} = \prod_{l=0}^L \hat{w}_{L-l,\dots ij}$:

$$\min_{\hat{v}} \frac{1}{N}\sum_n \ell(\hat{v}^T\hat{x}^n, y^n) + \lambda L\sum_{i,j}|\hat{v}_{ij}|^{\frac{2}{L}}_{\frac{2}{L}} \tag{B.7}$$

in the following 2 cases:

(i) $C = K = 1$ and $C_l = 1$ for all l and
(ii) $L = 2$.

Moreover, we will prove this equivalence in general assuming conjecture 4.13.

We proceed by a series of reductions; as a first step we observe

$$\begin{aligned}&\min_{\hat{w}} \frac{1}{N}\sum_n \ell\left((\hat{w}^L\cdot\hat{w}^{L-1}\cdot\hat{w}^{L-1}\cdots\hat{w}^1)^T\hat{x}^n, y^n\right) + \lambda\sum_{l,i,j}|\hat{w}^l_{ij}|_2^2\\ =&\min_{\hat{v}} \min_{\hat{w}^L\cdot\hat{w}^{L-1}\cdot\hat{w}^{L-1}\cdots\hat{w}^1=\hat{v}} \frac{1}{N}\sum_n \ell(\hat{v}^T\hat{x}^n, y^n) + \lambda\sum_{l,i,j}|\hat{w}^l_{ij}|_2^2,\end{aligned} \tag{B.8}$$

and hence theorem 4.9 will follow if we can prove

$$\sum_{i,j}|\hat{v}_{ij}|^{\frac{2}{L}}_{\frac{2}{L}} = \min \frac{1}{L}\sum_{l,i,j}|\hat{w}^l_{ij}|_2^2, \tag{B.9}$$

where the min on the right hand side runs over all $\hat{w}$ such that $\hat{w}^L\cdot\hat{w}^{L-1}\cdot\hat{w}^{L-1}\cdots\hat{w}^1 = \hat{v}$. We can make life slightly simpler by noticing eq. (B.9) decomposes over the i, j index, and will follow from

$$|\hat{v}_{ij}|^{\frac{2}{L}}_{\frac{2}{L}} = \min \frac{1}{L}\sum_l |\hat{w}^l_{ij}|_2^2 \text{ for all } i, j, \tag{B.10}$$

which we will show in lemma B.13 — note that at this point we are arriving at a statement about norms of matrix products and can dispense with the baggage of i, j indices and $\hat{\text{s}}$. To formally state that lemma, we introduce a convenient definition.

Definition B.11. A sequence of matrices $A_1 \in M(m_1 \times n_1, \mathbb{C}), \ldots, A_L \in M(m_L \times n_L, \mathbb{C})$ is **composable** if and only if

$$m_l = n_{l-1} \text{ for } l = 2, \ldots, L \tag{B.12}$$

In other words, $A_1, \ldots, A_L$ is composable if and only if the product $A_L \cdots A_1$ makes sense.

Lemma B.13. *If $B \in M(m \times n, \mathbb{C})$ is a matrix with complex entries and $L \in \mathbb{N}$ is a non-negative integer, then*

$$|B|_{\frac{2}{L}}^{\frac{2}{L}} = \min(\prod_l |A_l|_2^2)^{\frac{1}{L}} = \min \frac{1}{L} \sum_l |A_l|_2^2, \tag{B.14}$$

where both minima are taken over all composable sequences of complex matrices $A_1, \ldots, A_L$ such that $A_L \cdots A_1 = B$, provided that *either*

(i) all of the matrices are 1-by-1 or
(ii) $L = 2$.

Moreover eq. (B.14) holds in general assuming conjecture 4.13.

We first deal with the elementary aspects of lemma B.13: it will suffice to show that whenever $A_L \cdots A_1 = B$,

$$|B|_{\frac{2}{L}}^{\frac{2}{L}} \leq (\prod_l |A_l|_2^2)^{\frac{1}{L}} \leq \frac{1}{L} \sum_l |A_l|_2^2 \textit{ and that} \tag{B.15}$$

$$|B|_{\frac{2}{L}}^{\frac{2}{L}} = \frac{1}{L} \sum_l |A_l|_2^2 \tag{B.16}$$

for *some* composable sequence $A_1, \ldots, A_L$ such that $A_L \cdots A_1 = B$. As noted in (Gunasekar et al., 2018) the second inequality of eq. (B.15) is simply the arithmetic-geometric mean inequality applied to $|A_1|_2^2, \ldots |A_L|_2^2 \in \mathbb{R}_{\geq 0}$. Furthermore, we can obtain eq. (B.16) using the singular value decomposition of B: let $B = USV^*$ where, letting $r := \min\{m, n\}$, $U \in U(r, m)$ and $V \in U(n, r)$ are unitary and $S = \operatorname{diag}(\lambda_1, \ldots, \lambda_r)$, where $\lambda_i \geq 0$ for all i. We may decompose B into L factors like

$$B = (US^{\frac{1}{L}}) \cdot (\prod_{i=1}^{L-1} S^{\frac{1}{L}}) \cdot (S^{\frac{1}{L}} V^*), \tag{B.17}$$

and by unitary invariance of the C_2 (a.k.a. Frobenius) norm,

$$\begin{aligned} &|US^{\frac{1}{L}}|_2^2 + \sum_{i=1}^{L-1} |S^{\frac{1}{L}}|_2^2 + |S^{\frac{1}{L}} V^*|_2^2 = |S^{\frac{1}{L}}|_2^2 + \sum_{i=1}^{L-1} |S^{\frac{1}{L}}|_2^2 + |S^{\frac{1}{L}}|_2^2 \\ &= L|S^{\frac{1}{L}}|_2^2. \end{aligned} \tag{B.18}$$

We now note that *by definition 4.6*,

$$\begin{aligned} |B|_{\frac{2}{L}}^{\frac{2}{L}} &= \sum_i |\lambda_i|^{\frac{2}{L}} \text{ and on the other hand} \\ |S^{\frac{1}{L}}|_2^2 &= \sum_i |\lambda_i|^{\frac{1}{L} \cdot 2}, \end{aligned} \tag{B.19}$$

so that combining eqs. (B.18) and (B.19) gives eq. (B.16) for the composable sequence of eq. (B.17).

It remains to prove the first inequality of eq. (B.15), in the cases needed for theorem 4.9. In the case where $C = C_1 = \cdots = C_{L-1} = K = 1$, all of the matrices $\hat{w}_{ij}^1, \ldots, \hat{w}_{ij}^L, \hat{v}_{ij}$ are 1-by-1, and in this case our desired inequality follows from (Tibshirani, 2021, §2.2). In the case where $L = 2$, it follows from (Dixmier, 1953, Thm. 6).[5]

In general, the first inequality of eq. (B.15) would follow from conjecture 4.13, which we recall for convenience. We note that this conjectured non-commutative generalized Hölder inequality is "non-commutative" since we work with products of matrices as opposed to inner products of vectors, and "generalized" since we consider ℓ_p exponents p_i where $\sum \frac{1}{p_i} > 1$; in the application $p_i = \frac{1}{2}$ for all i.

[5] strictly speaking Dixmier works with square matrices, but since our matrix norms are insensitive to adding rows or columns of zeros, we may zero-pad our matrices to shape $M \times M$ where $M = \max\{C, C_1, K\}$ and apply Dixmier's theorem

Conjecture B.20. *If $B \in M(m \times n, \mathbb{C})$ is a matrix with complex entries, $A_1, \ldots, A_L$ is a composable sequence of complex matrices such that $A_1 \cdots A_L = B$ and $\sum_i \frac{1}{p_1} = \frac{1}{r}$ where $p_1, \ldots, p_L > 0$ are positive real numbers,*

$$|B|_r \leq \prod_i |A_i|_{p_i} \tag{B.21}$$

As mentioned in section 4, when $r = 1$ this is a theorem of Dixmier and to the best of the authors knowledge it is open for any $r \neq 1$.

We briefly note that while for complex vectors there is a standard trick that extends the $r = 1$ Hölder inequality to all values of $r > 0$, this trick is not straightforward to apply for complex matrices. The trick in question hinges on the following identity: if $x^1, x^2 \in \mathbb{C}^n$ are complex vectors, functions, then

$$|x^1 \cdot x^2|_r^r = \sum_i |x_i^1 x_i^2|^r = \sum_i |x_i^1|^r |x_i^2|^r = |{x^1}^r {x^2}^r|_1 \tag{B.22}$$

where denotes the coordinate-wise absolute value function. Observe that this manipulation involves *commuting* x^1, x^2 — for example, if x^1, x^2 are real and $r = 2$, we have effectively written $x^1 x^1 x^2 x^2 = x^1 x^2 x^1 x^2$. A possible path forward would be to try to prove for complex matrices A_1, A_2 with polar decompositions $A_1 = U_1 H_1$ and $A_2 = U_2 H_2$ that $|A_1 A_2|_r^r \leq |H_1^r H_2^r|_1$, where H_1^r denotes the operation of raising singular values (equivalently eigenvalues in the case of the hermitian H_1) to the r-th power and similarly for H_2. We were unable to prove such an inequality.

C Experimental Details

C.1 Covariance Matrices of DFTed Image Datasets

In this section we provide further details on our computations of (co)variances and standard deviations of DFTed image datasets.

To compute the covariances $\Sigma_{jicc'i'j'}$ for fig. 1, we begin with a dataset of natural images, say X. We subtract its mean RGB pixel value (a vector in $\mathbb{R}^3$) and divide by the standard deviation of RGB pixel values (also a vector in $\mathbb{R}^3$) as is standard. We next apply the DFT to every image in X, to obtain a DFTed dataset $\hat{X}$. We then compute the mean $\hat{\mu}$ of the *images* in $\hat{X}$, not the RGB pixel values — this is a $3 \times H \times W$ tensor, where H, W are the heights/widths of the images in X (e.g., 32 for CIFAR10). This mean is then subtracted from $\hat{X}$ to obtain a centered dataset. Next, we sample batches of size B, say $\hat{x}_1, \ldots, \hat{x}_B$, from $\hat{X}$ (these are tensors of shape $B \times C \times H \times W$). For each batch, we subtract the mean, contract over the batch index and divide by B to obtain an estimate for Σ, say $\tilde{\Sigma}$ like so:

$$\tilde{\Sigma} = \frac{1}{B} \sum_{b=1}^{B} (\hat{x}_b - \hat{\mu})^T \otimes (\hat{x}_b - \hat{\mu}); \text{ explicitly } \tilde{\Sigma}_{jicc'i'j'} = \frac{1}{B} \sum_{b=1}^{B} (\hat{x}_{b,jic} - \hat{\mu}_{jic})(\hat{x}_{b,c'i'j'} - \hat{\mu}_{c'i'j'}) \tag{C.1}$$

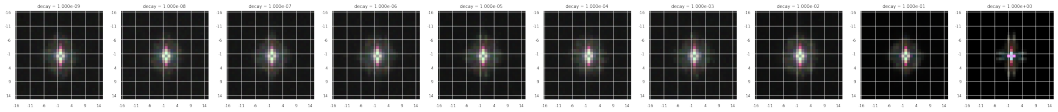
Finally, we average over an entire dataset's worth of batches to get our final estimate of $\Sigma_{jicc'i'j'}$.

To estimate the expectations $E[|\Sigma_{jicc'i'j'}| \,|\, |(c,i,j) - (c,i',j')| = r]$, we average absolute values $|\Sigma_{jicc'i'j'}|$ over all indices $jicc'i'j'$ satisfying $|(c,i,j) - (c',i',j')| = r$. The number of such entries varies significantly with r, which is why we have not explicitly written down the average. We compute "modular" distances, i.e. distances on the discrete torus, using the formulae

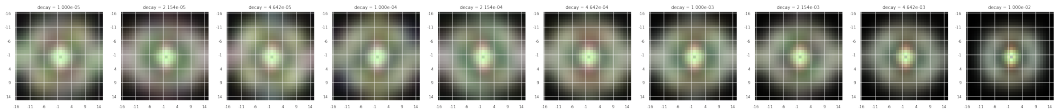
$$\begin{aligned} d(i,j') &= \min\{|(h-h')\text{mod}H|, |H - ((h-h')\text{mod}H)|\} \\ d(w,w') &= \min\{|(w-w')\text{mod}W|, |W - ((w-w')\text{mod}W)|\} \end{aligned} \tag{C.2}$$

and finally $d((c,h,w),(c',h',w')) = \sqrt{(c-c')^2 + d(h,h')^2 + d(w,w')^2}$.

In fig. 1c, we begin with a dataset of natural images, say X. We subtract its mean RGB pixel value (a vector in $\mathbb{R}^3$) and divide by the standard deviation of RGB pixel values (also a vector in $\mathbb{R}^3$) as is standard. We next apply the DFT to every image in X, to obtain a DFTed dataset $\hat{X}$. We then compute the standard deviation of the *images* in $\hat{X}$, not the RGB pixel values — this is a $3 \times H \times W$ tensor, where H, W are the heights/widths of the images in X (e.g., 32 for CIFAR10).



(a) Frequency sensitivity $E[|\nabla_x f(x)^T \hat{e}_{cij}|]$ of ConvActually models of depth 4 trained with varying weight decay. Accuracy of the models ranged from 52% ($\lambda = 1$) to 67% ($\lambda = 0.01$).



(b) Frequency sensitivity of MyrtleCNN models trained with varying weight decay. Accuracy of the models ranged from 83% ($\lambda = 0.01$) to 87% ($\lambda = 0.002$).

Figure 7: For both ConvActually (a) and MyrtleCNN models (b), as the weight decay coefficient λ increases the frequency sensitivity images more and more closely resemble the statistics of the training data images shown in fig. 1c.

C.2 Gradient sensitivity images

To compute the sensitvity maps in figs. 2 and 7, we subsample 5,000 images from the underlying dataset. For each such image x, we compute the DFT $\hat{x}$, and then backpropagate gradients through the composition $\hat{x} \to \hat{\hat{x}} = x \to f(x)$. The result is a $C \times H \times W \times K$ Jacobian matrix expressing the derivative of f with respect to Fourier basis vectors. We take ℓ_2 norms over the class index (corresponding to K) to obtain a $C \times H \times W$ gradient norm image, with c, h, w component $|\nabla_x f(x)^T \hat{e}_{cij}|$. Finally, we average these gradient norms over the subsampled dataset.

For the radial frequency sensitivity curves in figs. 3a, 4a and 5a, we average radially much as we did in fig. 1. Given an expected gradient norm image $E[|\nabla_x f(x)^T \hat{e}_{cij}|]$ obtained as above, we further average over all indices cij such that $\sqrt{i^2 + j^2} = r$. Again, the number of such indices is highly variable.

C.3 More frequency sensitivity images

C.4 Datasets

CIFAR-10: We use canonical train/test splits (imported using torchvision).

C.5 Model architectures and training parameters

For the experiments with ConvActuallys of varying depth, we train using SGD with momentum 0.9, batch size of 1028, learning rate starting at 10^{-4} and following a "reduce on plateau" schedule with drops by a factor of 0.1 after 50 epochs with no improvement, for a maximum of 500 epochs or until the learning rate hits 10^{-6}. These parameters were chosen to avoid early stopping and allow SGD to run to convergence. We use weight decay of 10^{-9} (i.e., basically no decay). For the experiments training ResNets of varying depth, we train using SGD with momentum 0.9, batch size 32, initieal learning rate $32/512$ with drops by a factor of 0.1 after 10 epochs with no improvement, for a maximum of 500 epochs or until the learning rate hits 10^{-6}

For the experiments with ConvActuallys trained with varying decay, we use the same optimizer, batch size and learning rate schedule, however we fix the depth to 4 and vary decay as indicated. For the experiments with MyrtleCNNs trained with varying decay, we use SGD with momentum 0.9, batch size of 512, initial learning rate 10^{-4} and a reduce on plateau schedule with drops by a factor of 0.1 after 20 epochs with no improvement, for a maximum of 500 epochs or until the learning rate hits 10^{-6}. Again, our motivation was to allow SGD to run to convergence. For the experiments with ResNets trained with varying decay, we use SGD with momentum 0.9, batch size of 512, initial learning rate 10^{-1} and a reduce on plateau schedule with drops by a factor of 0.1 after 20 epochs with no improvement, for a maximum of 500 epochs or until the learning rate hits 10^{-6}.

We use the implementation of CIFAR10 resnets from (Team, 2021).

```
==========================================================================================
Layer (type:depth-idx)                   Output Shape              Param #
==========================================================================================
ConvActually                             [1, 64, 32, 32]           --
├─Conv2d: 1-1                            [1, 64, 32, 32]           196,672
├─Identity: 1-2                          [1, 64, 32, 32]           --
├─ReLU: 1-3                              [1, 64, 32, 32]           --
==========================================================================================
Total params: 196,672
Trainable params: 196,672
Non-trainable params: 0
Total mult-adds (M): 201.39
==========================================================================================
Input size (MB): 0.01
Forward/backward pass size (MB): 0.52
Params size (MB): 0.79
Estimated Total Size (MB): 1.32
==========================================================================================
```

(a) A single block of ConvActually.

```
==========================================================================================
Layer (type:depth-idx)                   Output Shape              Param #
==========================================================================================
mCNN_k                                   [1, 10]                   --
├─Sequential: 1-1                        [1, 64, 32, 32]           --
│    └─Conv2d: 2-1                       [1, 64, 32, 32]           1,792
│    └─ReLU: 2-2                         [1, 64, 32, 32]           --
├─ModuleList: 1-2                        --                        --
│    └─Sequential: 2-3                   [1, 128, 16, 16]          --
│    │    └─Conv2d: 3-1                  [1, 128, 32, 32]          73,856
│    │    └─ReLU: 3-2                    [1, 128, 32, 32]          --
│    │    └─MaxPool2d: 3-3               [1, 128, 16, 16]          --
│    └─Sequential: 2-4                   [1, 256, 8, 8]            --
│    │    └─Conv2d: 3-4                  [1, 256, 16, 16]          295,168
│    │    └─ReLU: 3-5                    [1, 256, 16, 16]          --
│    │    └─MaxPool2d: 3-6               [1, 256, 8, 8]            --
│    └─Sequential: 2-5                   [1, 512, 4, 4]            --
│    │    └─Conv2d: 3-7                  [1, 512, 8, 8]            1,180,160
│    │    └─ReLU: 3-8                    [1, 512, 8, 8]            --
│    │    └─MaxPool2d: 3-9               [1, 512, 4, 4]            --
├─Sequential: 1-3                        [1, 10]                   --
│    └─MaxPool2d: 2-6                    [1, 512, 1, 1]            --
│    └─Flatten: 2-7                      [1, 512]                  --
│    └─Linear: 2-8                       [1, 10]                   5,130
==========================================================================================
Total params: 1,556,106
Trainable params: 1,556,106
Non-trainable params: 0
Total mult-adds (M): 228.56
==========================================================================================
Input size (MB): 0.01
Forward/backward pass size (MB): 2.36
Params size (MB): 6.22
Estimated Total Size (MB): 8.60
==========================================================================================
```

(b) Architecture of the MyrtleCNN.