A Scalable Walsh-Hadamard Regularizer to Overcome the Low-degree Spectral Bias of Neural Networks

Ali Gorji[*1] **Andisheh Amrollahi**[*1] **Andreas Krause**[1]

[1]Computer Science Department, ETH Zurich, Zurich, Switzerland

Abstract

Despite the capacity of neural nets to learn arbitrary functions, models trained through gradient descent often exhibit a bias towards "simpler" functions. Various notions of simplicity have been introduced to characterize this behavior. Here, we focus on the case of neural networks with discrete (zero-one), high-dimensional, inputs through the lens of their Fourier (Walsh-Hadamard) transforms, where the notion of simplicity can be captured through the *degree* of the Fourier coefficients. We empirically show that neural networks have a tendency to learn lower-degree frequencies. We show how this spectral bias towards low-degree frequencies can in fact *hurt* the neural network's generalization on real-world datasets. To remedy this we propose a new scalable functional regularization scheme that aids the neural network to learn higher degree frequencies. Our regularizer also helps avoid erroneous identification of low-degree frequencies, which further improves generalization. We extensively evaluate our regularizer on synthetic datasets to gain insights into its behavior. Finally, we show significantly improved generalization on four different datasets compared to standard neural networks and other relevant baselines.

1 INTRODUCTION

Classical work on neural networks shows that deep fully connected neural networks have the capacity to approximate arbitrary functions [Hornik et al., 1989, Cybenko, 1989]. However, in practice, neural networks trained through (stochastic) gradient descent have a "simplicity" bias. This notion of simplicity is not agreed upon and works such as [Arpit et al., 2017, Nakkiran et al., 2019, Valle-Perez et al., 2019, Kalimeris et al., 2019] each introduce a different notion of "simplicity". The simplicity bias can also be studied by considering the function the neural net represents (function space view) and modeling it as Gaussian processes (GP)[Rasmussen, 2004]. Daniely et al. [2016], Lee et al. [2018] show that a wide, randomly initialized, neural network in function space is a sample from a GP with a kernel called the "Conjugate Kernel" [Daniely, 2017]. Moreover, the evolution of gradient descent on a randomly initialized neural network can be described through the "Neural Tangent Kernel" Jacot et al. [2018], Lee et al. [2019]. These works open up the road for analyzing the simplicity bias of neural nets in terms of a *spectral* bias in Fourier space. Rahaman et al. [2019] show empirically that neural networks tend to learn sinusoids of lower frequencies earlier on in the training phase compared to those of higher frequencies. Through the GP perspective introduced by Jacot et al. [2018], Lee et al. [2019], among others, Ronen et al. [2019], Basri et al. [2020] were able to prove these empirical findings. These results focus on *continuous* domains and mainly emphasize the case where the input and output are both *one-dimensional*.

Here, we focus on *discrete* domains where the input is a *high-dimensional* zero-one vector and we analyze the function learned by the neural network in terms of the amount of interactions among its input features in a quantitative manner. Our work is complementary to the majority of the aforementioned work that has been done on the spectral bias of neural networks in the setting of *continuous*, *one-dimensional* inputs [Ronen et al., 2019, Basri et al., 2020, Rahaman et al., 2019]. Yang and Salman [2020], Valle-Perez et al. [2019] are the first to provide spectral bias results for the discrete, higher dimensional, setting (our setting). By viewing a fully connected neural network as a function that maps zero-one vectors to real values, one can expand this function in terms of the Fourier –a.k.a Walsh-Hadamard – basis functions. The Walsh-Hadamard basis functions have a natural ordering in terms of their complexity called their

*These authors contributed equally to this work

Accepted for the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023).

degree. The degree specifies how many features each basis function is dependent upon. For example, the zero-degree basis function is the constant function and the degree-one basis functions are functions that depend on exactly one feature. Through analysis of the NTK gram matrix on the Boolean cube, Yang and Salman [2020] theoretically show that, roughly speaking, neural networks learn the lower degree basis functions earlier in training.

This tendency to prioritize simpler functions in neural networks has been suggested as a cardinal reason for their remarkable generalization ability despite their over-parameterized nature [Neyshabur et al., 2017, Arpit et al., 2017, Kalimeris et al., 2019, Poggio et al., 2018]. However, much less attention has been given to the case where the simplicity bias can *hurt* generalization [Tancik et al., 2020, Shah et al., 2020]. Tancik et al. [2020] show how transforming the features with random Fourier features embedding helps the neural network overcome its spectral bias and achieve better performance in a variety of tasks. They were able to explain, in a unified way, many empirical findings in computer vision research such as sinusoidal positional embeddings through the lens of overcoming the spectral bias. In the same spirit as these works, we show that the spectral bias towards low-degree functions can hurt generalization and how to remedy this through our proposed regularizer.

In more recent lines of work, regularization schemes have been proposed to directly impose priors on the function the neural network represents [Benjamin et al., 2019, Sun et al., 2019, Wang et al., 2019]. This is in contrast to other methods such as dropout, batch normalization, or other methods that regularize the weight space. In this work, we also regularize neural networks in function space by imposing sparsity constraints on their Walsh-Hadamard transform. Closest to ours is the work of Aghazadeh et al. [2021]. Inspired by studies showing that biological landscapes are sparse and contain high-degree frequencies [Sailer and Harms, 2017, Yang et al., 2019, Brookes et al., 2022, Ballal et al., 2020, Poelwijk et al., 2019], they propose a functional regularizer to enforce sparsity in the Fourier domain and report improvements in generalization scores.

Our contributions:

- We analyze the spectral behavior of a simple MLP during training through extensive experiments. We show that the standard (unregularized) network not only is unable to learn (more complex) high-degree frequencies but it also starts learning erroneous low-degree frequencies and hence overfitting on this part of the spectrum.
- We propose a novel regularizer – HASHWH (Hashed Walsh Hadamard) – to remedy the aforementioned phenomenon. The regularizer acts as a "sparsifier" on the Fourier (Walsh-Hadamard) basis. In the most extreme cases, it reduces to simply imposing an L_1-norm on the Fourier transform of the neural network. Since computing the exact Fourier transform of the neural net is intractable, our regularizer hashes the Fourier coefficients to buckets and imposes an L1 norm on the buckets. By controlling the number of hash buckets, it offers a smooth trade-off between computational complexity and the quality of regularization.
- We empirically show that HASHWH aids the neural network in avoiding erroneous low-degree frequencies and also learning relevant high-degree frequencies. The regularizer guides the training procedure to allocate more energy to the high-frequency part of the spectrum when needed and allocate less energy to the lower frequencies when they are not present in the dataset.
- We show on real-world datasets that, contrary to popular belief of simplicity biases for neural networks, fitting a low degree function does not imply better generalization. Rather, what is more important, is keeping the *higher amplitude* coefficients regardless of their degree. We use our regularizer on four real-world datasets and provide state of the art results in terms of R^2 score compared to standard neural networks and other baseline ML models, especially for the low-data regime.

2 BACKGROUND

In this section, we first review Walsh Hadamard transforms, and notions of degree and sparsity in the Fourier (Walsh-Hadamard) domain [O'Donnell, 2014]. Next, we review the notion of simplicity biases in neural networks and discuss why they are spectrally biased toward low-degree functions.

2.1 WALSH HADAMARD TRANSFORMS

Let $g : \{0,1\}^n \to \mathbb{R}$ be a function mapping Boolean zero-one vectors to the real numbers, also known as a "pseudo-boolean" function. The family of 2^n functions $\{\Psi_f : \{0,1\}^n \to \mathbb{R} \,|\, f \in \{0,1\}^n\}$ defined below consists of the Fourier basis functions. This family forms a basis over the vector space of all pseudo-boolean functions:

$$\Psi_f(x) = \frac{1}{\sqrt{2^n}}(-1)^{\langle f,x\rangle}, f, x \in \{0,1\}^n$$

where $\langle f, x\rangle = \sum_i f_i x_i$. Here, $f \in \{0,1\}^n$ is called the *frequency* of the basis function. For any frequency $f \in \{0,1\}^n$ we denote its *degree* by $\deg(f)$ which is defined as the number of non-zero elements. For example, $f_1 = [0,0,0,0,0]$ and $f_2 = [0,0,1,0,1]$ have degrees $\deg(f_1) = 0$ and $\deg(f_2) = 2$, respectively. One can think of the degree as a measure of the complexity of basis functions. For example, $\Psi_0(x)$ is constant, and $\Psi_{e_i}(x)$ where e_i is a standard basis vector ($\deg(e_i) = 1$) only depends on feature i of the input. It is equal to $+1$ when feature i is zero and equal to -1 when feature i is one. More generally, a degree d basis function depends on exactly d input features.

Since the Fourier basis functions form a basis for the vector space of all pseudo-boolean functions, any function $g : \{0,1\}^n \to \mathbb{R}$ can be written as a unique linear combination of these basis functions:

$$g(x) = \frac{1}{\sqrt{2^n}} \sum_{f \in \{0,1\}^n} \widehat{g}(f)(-1)^{\langle f,x \rangle}$$

The (unique) coefficients $\widehat{g}(f)$ are called the "Fourier coefficients" or "Fourier amplitudes" and are computed as $\widehat{g}(f) = \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} g(x)(-1)^{\langle f,x \rangle}$. The *Fourier spectrum* of g is the vector consisting of all of its 2^n Fourier coefficients, which we denote by the bold symbol $\widehat{\mathbf{g}} \in \mathbb{R}^{2^n}$. Assume $\mathbf{X} \in \{0,1\}^{2^n \times n}$ to be the matrix of an enumeration over all possible n-dimensional binary sequences ($\{0,1\}^n$), and $\mathbf{g}(\mathbf{X}) \in \mathbb{R}^{2^n}$ to be the vector of g evaluated on the rows of $\mathbf{X}$. We can compute the Fourier spectrum using Walsh-Hadamard transform as $\widehat{\mathbf{g}} = \frac{1}{\sqrt{2^n}} \mathbf{H}_n \mathbf{g}(\mathbf{X})$, where $\mathbf{H}_n \in \{\pm 1\}^{2^n \times 2^n}$ is the orthogonal Hadamard matrix (see Appendix A).

Lastly, we define the *support* of g as the set of frequencies with non-zero Fourier amplitudes $\text{supp}(g) := \{f \in \{0,1\}^n | \widehat{g}(f) \neq 0\}$. The function g is called *k-sparse* if $|\text{supp}(g)| \leq k$. The function g is called *of degree* d if all frequencies in its support have degree at most d.

2.2 SPECTRAL BIAS THEORY

The function that a ReLU neural network represents at initialization can be seen as a sample from a GP $N(0, K)$ in the infinite width limit [Daniely et al., 2016, Lee et al., 2018] (randomness is over the initialization of the weights and biases). The kernel K of the GP is called the "Conjugate Kernel" [Daniely et al., 2016] or the "nn-GP kernel" [Lee et al., 2018]. Let the kernel Gram matrix $\mathcal{K}$ be formed by evaluating the kernel on the Boolean cube i.e. $\{0,1\}^n$ and let $\mathcal{K}$ have the following spectral decomposition: $\mathcal{K} = \sum_{i=1}^{2^n} \lambda_i u_i u_i^\top$, where we assume that the eigenvalues $\lambda_1 \geq \cdots \geq \lambda_{2^n}$ are in decreasing order. Each sample of the GP can be obtained as $\sum_{i=1}^{2^n} \lambda_i \boldsymbol{w_i} u_i, \boldsymbol{w_i} \sim \mathcal{N}(0,1)$. Say that $\lambda_1 \gg \sum_{i \geq 2} \lambda_i$. Then a sample from the GP will, roughly speaking, look very much like u_1.

Let $u_f, f \in \{0,1\}^n$ be obtained by evaluating the Fourier basis function Ψ_f at the 2^n possible inputs on $\{0,1\}^n$. Yang and Salman [2020] show that u_f is a eigenvector for $\mathcal{K}$. Moreover, they show (weak) spectral bias results in terms of the degree of f. Namely, the eigenvalues corresponding to higher degrees have smaller values [1]. The result is *weak* as they do not provide a *rate* as to which the eigenvalues decrease with increasing degrees. Their results show that neural networks are similar to low-degree functions at initialization.

Other works show that in infinite-width neural networks weights after training via (stochastic) gradient descent do not end up too far from the initialization [Chizat et al., 2019, Jacot et al., 2018, Du et al., 2019, Allen-Zhu et al., 2019a,b], referred to as "lazy training" by Chizat et al. [2019]. Lee et al. [2018, 2019] show that training the last layer of a randomly initialized neural network via full batch gradient descent for an infinite amount of time corresponds to GP posterior inference with the kernel K. Jacot et al. [2018], Lee et al. [2019] proved that when training *all* the layers of a neural network (not just the final layer), the evolution can be described by a kernel called the "Neural Tangent Kernel" and the trained network yields the mean prediction of GP $N(0, K_{NTK})$ [Yang and Salman, 2020] after an infinite amount of time. Yang and Salman [2020] again show that u_f are eigenvectors and weak spectral bias holds. Furthermore, Yang and Salman [2020] provides empirical results for the generalization of neural nets of different depths on datasets arising from $k = 1$-sparse functions of varying degrees.

3 LOW-DEGREE SPECTRAL BIAS

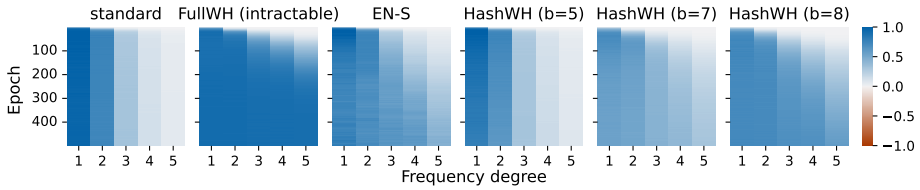
In this section, we conduct experiments on synthetically generated datasets to show neural networks' spectral bias and their preference toward learning lower-degree functions over higher-degree ones. Firstly, we show that the neural network is not able to pick up the high-degree frequency components. Secondly, it can learn erroneous lower-degree frequency components. To address these issues, in Section 4, we introduce our regularization scheme called HASHWH (Hashed Walsh Hadamard) and demonstrate how it can remedy both problems.

3.1 FOURIER SPECTRUM EVOLUTION

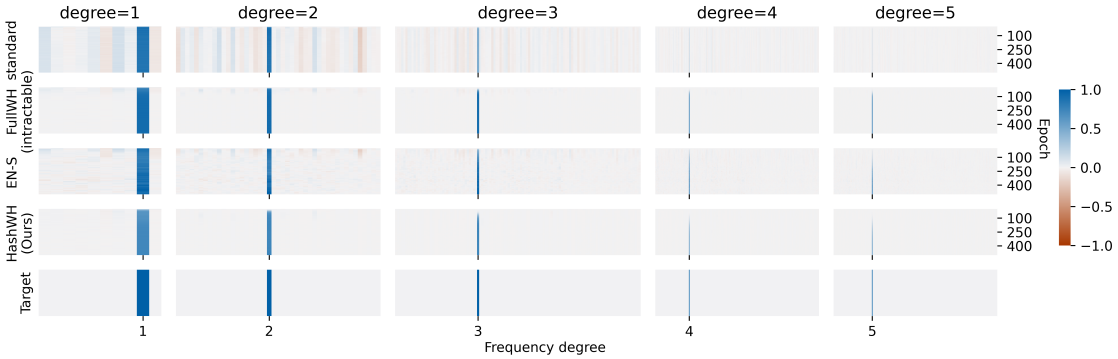
We analyze the evolution of the function learned by neural networks during training. We train a neural network on a dataset arising from a synthetically generated sparse function with a low-dimensional input domain. Since the input is low-dimensional it allows us to calculate the Fourier spectrum of the network (exactly) at the end of each epoch.

Setup. Let $g^* : \{0,1\}^{10} \to \mathbb{R}$ be a synthetic function with five frequencies in its support with degrees 1 to 5 ($\text{supp}(g^*) = \{f_1, f_2, f_3, f_4, f_5\}, \deg(f_i) = i$), all having equal Fourier amplitudes of $\widehat{g}^*(f_i) = 1$. Each f_i is sampled uniformly at random from all possible frequencies of degree i. The training set is formed by drawing uniform samples from the Boolean cube $x \sim \mathcal{U}_{\{0,1\}^{10}}$ and evaluating $g^*(x)$.

[1] To be more precise, they show that the eigenvalues corresponding to even and odd degree frequencies form decreasing sequences. That is, even and odd degrees are considered separately.



(a) Target support



(b) Whole Fourier spectrum

Figure 1: Evolution of the Fourier spectrum during training. STANDARD is the unregularized neural network. FULLWH imposes L_1-norm regularization on the exact Fourier spectrum and is intractable. EN-S alternates between computing a sparse Fourier approximation (computationally very expensive) and regularization. HASHWH (ours) imposes L_1 regularization on the hashed spectrum. Figure (a) is limited to the target support. The standard neural network is unable to learn higher degree frequencies. Our regularizer fixes this. Figure (b) is on the whole spectrum. The standard neural network picks up erroneous low-degree frequencies while not being able to learn the higher-degree frequencies. Our regularizer fixes both problems.

We draw five such target functions g^* (with random support frequencies). For each draw of the target function, we create five different datasets all with 200 training points and sampled uniformly from the input domain but with different random seeds. We then train a standard five-layer fully connected neural network using five different random seeds for the randomness in the training procedure (such as initialization weights and SGD). We aggregate the results over the 125 experiments by averaging. We experiment the same setting with three other training set sizes. Results with training set size other than 200 and further setup details are reported in Appendices F.1 and D, respectively.

Results. We first inspect the evolution of the learned Fourier spectrum over different epochs and limited to the target support ($\text{supp}(g^*)$). Figure 1a shows the learned amplitudes for frequencies in the target support at each training epoch. Aligned with the literature on simplicity bias [Valle-Perez et al., 2019, Yang and Salman, 2020], we observe that neural networks learn the low-degree frequencies earlier in the epochs. Moreover, we can see in the left-most figure in Figure 1a that despite eventually learning low-degree frequencies, the standard network is unable to learn high-degree frequencies.

Next, we expand the investigation to the whole Fourier spectrum instead of just focusing on the support frequencies. The first row of Figure 1b shows the evolution of the Fourier spectrum during training and compares it to the spectrum of the target function on the bottom row. We average the spectrum linked to one of the five target synthetic functions (over the randomness of the dataset sampling and training procedure) and report the other four in Appendix F.1. We observe that in addition to the network not being able to learn the high-degree frequencies, the standard network is prone to learning incorrect low-degree frequencies as well.

4 OVERCOMING THE SPECTRAL BIAS VIA REGULARIZATION

Now, we introduce our regularization scheme HASHWH (Hashed Walsh-Hadamard). Our regularizer is essentially a "sparsifier" in the Fourier domain. That is, it guides the neural network to have a sparse Fourier spectrum. We empirically show later how sparsifying the Fourier spectrum can both stop the network from learning erroneous low-degree frequencies and aid it in learning the higher-degree ones, hence remedying the two aforementioned problems.

Assume $\mathcal{L}_{net}$ is the loss function that a standard neural network minimizes, e.g., the MSE loss in the above case. We modify it by adding a regularization term $\lambda \mathcal{L}_{sparsity}$.

Hence the total loss is given by: $\mathcal{L} = \mathcal{L}_{net} + \lambda \mathcal{L}_{sparsity}$.

The most intuitive choice is $\mathcal{L}_{sparsity} = \|\widehat{\mathbf{g_N}}\|_0$, where $\widehat{\mathbf{g_N}}$ is the Fourier spectrum of the neural network function $g_N : \{0,1\}^n \to \mathbb{R}$. Since the L_0-penalty's derivative is zero almost everywhere, one can use its tightest convex relaxation, the L_1-norm, which is also sparsity-inducing, as a surrogate loss. Aghazadeh et al. [2021] use this idea and name it as Epistatic-Net or "EN" regularization: $\mathcal{L}_{EN} := \mathcal{L}_{net} + \lambda \|\widehat{\mathbf{g_N}}\|_1$. In this work, we call this regularization FULLWH (Full Walsh Hadamard transform).

FULLWH requires the evaluation of the network output on all 2^n possible inputs at each iteration of back-prop. Therefore, the computational complexity grows *exponentially* with the number of dimensions n, making it computationally intractable for $n > 20$ in all settings of practical importance.

Aghazadeh et al. [2021] also suggest a more scalable version of FULLWH, called "EN-S", which roughly speaking, alternates between computing the sparse *approximate* Fourier transform of the network at the end of each epoch and doing normal back-prop, as opposed to the exact computation of the exact Fourier spectrum when back-propagating the gradients. In our experiments, we show EN-S can be computationally expensive because the sparse Fourier approximation primitive can be time-consuming. For a comprehensive comparison see Appendix B.3. Later, we show that empirically, it is also less effective in overcoming the spectral bias as measured by achievable final generalization error.

4.1 HASHWH

We avoid the exponentially complex burden of computing the exact Fourier spectrum of the network by employing a hashing technique to approximate the regularization term $\lambda \|\widehat{\mathbf{g_N}}\|_1$. Let $g : \{0,1\}^n \to \mathbb{R}$ be a pseudo-boolean function. We define the lower dimensional function $u_\sigma : \{0,1\}^b \to \mathbb{R}$, where $b \ll n$, by sub-sampling g on its domain: $u_\sigma(\tilde{x}) \triangleq \sqrt{\frac{2^n}{2^b}}\, g(\sigma \tilde{x}),\ \tilde{x} \in \{0,1\}^b$ where $\sigma \in \{0,1\}^{n \times b}$ is some matrix which we call the *hashing matrix*. The matrix-vector multiplication $\sigma\tilde{x}$is taken modulo 2. u_σ is defined by sub-sampling g on all the points lying on the (at most) b-dimensional subspace spanned by the columns of the hashing matrix σ. The special property of sub-sampling the input space from this subspace is in the arising Fourier transform of u_σ which we will explain next.

The Fourier transform of u_σ can be derived as (see Appendix B.1):

$$\widehat{u}_\sigma(\tilde{f}) = \sum_{f \in \{0,1\}^n :\, \sigma^\top f = \tilde{f}} \widehat{g}(f),\ \tilde{f} \in \{0,1\}^b \qquad (1)$$

One can view $\widehat{u}_\sigma(\tilde{f})$ as a "bucket" containing the sum of all Fourier coefficients $\widehat{g}(\tilde{f})$ that are "hashed" (mapped) into it by the linear hashing function $h(f) = \sigma^\top f$. There are 2^b such buckets and each bucket contains frequencies lying in the kernel (null space) of the hashing map plus some shift.

In practice, we let $\sigma \sim \mathcal{U}_{\{0,1\}^{n \times b}}$ be a uniformly sampled hash matrix that is re-sampled after each iteration of back-prop. Let $\mathbf{X}_b \in \{0,1\}^{2^b \times b}$ be a matrix containing as rows the enumeration over all points on the Boolean cube $\{0,1\}^b$. Our regularization term approximates (4) and is given by:

$$\mathcal{L}_{\text{HASHWH}} \triangleq \mathcal{L}_{net} + \lambda \|\mathbf{H}_b \mathbf{g_N}(\mathbf{X}_b \sigma^T)\|_1 = \mathcal{L}_{net} + \lambda \|\widehat{\mathbf{u}_\sigma}\|_1$$

That is, instead of imposing the L_1-norm directly on the whole spectrum, this procedure imposes the norm on the "bucketed" (or partitioned) spectrum where each bucket (partition) contains sums of coefficients mapped to it. The larger b is the more partitions we have and the finer-grained the sparsity-inducing procedure is. Therefore, the quality of the approximation can be controlled by the choice of b. Larger b allows for a finer-grained regularization but, of course, comes at a higher computational cost because a Walsh-Hadamard transform is computed for a higher dimensional sub-sampled function u. Note that $b = n$ corresponds to hashing to 2^n buckets. As long as the hashing matrix is invertible, this precisely is the case of FULLWH regularization.

The problem with the above procedure arises when, for example, two "important" frequencies f_1 and f_2 are hashed into the same bucket, i.e., $\sigma^\top f_1 = \sigma^\top f_2$, an event which we call a "collision". This can be problematic when the absolute values $|\widehat{g}(f_1)|$ and $|\widehat{g}(f_2)|$ are large (hence they are important frequencies) but their sum can cancel out due to differing signs. In this case, the hashing procedure can zero out the sum of these coefficients. We can reduce the probability of a collision by increasing the number of buckets, i.e., increasing b [Alon et al., 1999].

In Appendix B.2 we show that the expected number of collisions C is given by: $\mathbb{E}[C] = \frac{(k-1)^2}{2^b}$ which decreases linearly with the number of buckets 2^b. Furthermore, we can upper bound the probability p that a given important frequency f_i collides with any other of the $k-1$ important frequencies in one round of hashing. Since we are independently sampling a new hashing matrix σ at each round of back-prop, the number of collisions of a given frequency over the different rounds has a binomial distribution. In Appendix B.2 we show that picking $b \geq \log_2(\frac{k-1}{\epsilon}), \epsilon > 0$ guarantees that collision of a given frequency happens approx. an ϵ-fraction of the T rounds, and not much more.

Fourier spectrum evolution of different regularization methods. We analyze the effect of regularizing the network with various Fourier sparsity regularizers in the setting of the previous section. Our regularizers of interest are FULLWH, EN-S with $m = 5$ (2^m is the number of buckets their sparse Fourier approximation algorithm hashes into), and HASHWH with $b \in \{5, 7, 8\}$.

Returning to Figure 1a, we see that despite the inability of

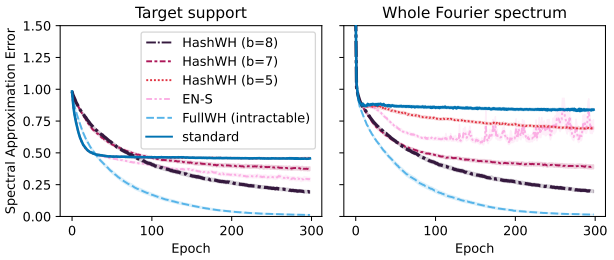


Figure 2: Evolution of the spectral approximation error during training. The left plot limits the error to the target support, while the right one considers the whole Fourier spectrum. For the standard neural network, the SAE is considerably worse on the full spectrum which shows the importance of eliminating the erroneous frequencies that are not in the support of the target function. We also see the graceful scaling of SAE of HASHWH (ours) with the hashing matrix size.

the standard neural network in picking up the *high-degree* frequencies, all sparsity-inducing regularization methods display the capacity for learning them. FULLWH is capable of perfectly learning the entire target support. It can also be seen that increasing the size of the hashing matrix in HASHWH (ours) boosts the learning of high-degree frequencies. Furthermore, Figure 1b shows that in addition to the better performance of the sparsity-inducing methods in learning the target support, they are also better at filtering out non-relevant *low-degree* frequencies.

We define a notion of approximation error which is basically the normalized energy of the error in the learned Fourier spectrum on an arbitrary subset of frequencies.

Metric 4.1 (Spectral Approximation Error (SAE))**.** *Let $g_N : \{0,1\}^n \to \mathbb{R}$ be an approximation of the target function $g^* : \{0,1\}^n \to \mathbb{R}$. Consider a subset of frequencies $S \subseteq \{0,1\}^n$, and assume $\widehat{\mathbf{g_N}}_S$ and $\widehat{\mathbf{g}^*}_S$ to be the vector of Fourier coefficients of frequencies in S, for g_N and g^* respectively. As a measure of the distance between g_N and g on the subset of frequencies S, we define Spectral Approximation Error as: SAE $= \frac{\|\widehat{\mathbf{g_N}}_S - \widehat{\mathbf{g}^*}_S\|_2^2}{\|\widehat{\mathbf{g}^*}_S\|_2^2}$*

Figure 2 shows the SAE of the trained network using different regularization methods over epochs, for both when S is target support as well as when $S = \{0,1\}^n$ (whole Fourier spectrum). The standard network displays a significantly higher (worse) SAE on the whole Fourier spectrum compared to the target support, while Walsh-Hadamard regularizers exhibit consistent performance across both. This shows the importance of enforcing the neural network to have zero Fourier coefficients on the non-target frequencies. Moreover, we can see HASHWH (ours) leads to a reduction in SAE that can be smoothly controlled by the size of its hashing matrix.

To gain more insight, we split the frequencies into subsets S consisting of frequencies with the same degree. We visualize the evolution of SAE and also the Fourier energy of the network defined as $\|\widehat{\mathbf{g_N}}_S\|_2^2$ in Figure 3. Firstly, the energy of high-degree frequencies is essentially zero for the standard neural network when compared to the low-degree frequencies, which further substantiates the claim that standard neural network training does not learn any high-degree frequencies. We can see that our HASHWH regularization scheme helps the neural network learn higher degree frequencies as there is more energy in the high degree components. Secondly, looking at the lower degrees 2 and 3 we can see that the standard neural network reduces the SAE up to some point but then starts overfitting. Looking at the energy plot one can attribute the overfitting to picking up irrelevant degree 2 and 3 frequencies. We see that the regularization scheme helps prevent the neural net from overfitting on the low-degree frequencies and their SAE reduces roughly monotonously. We observe that HASHWH (ours) with a big enough hashing matrix size exhibits the best performance among tractable methods in terms of SAE on all degrees. Finally, we can see HASHWH is distributing the energy to where it should be for this dataset: less in the low-degree and more in the high-degree frequencies.

Finally, it is worth noting that our regularizer makes the neural network behave more like a *decision tree*. It is well known that ensembles of decision tree models have a sparse and low-degree Fourier transform [Kushilevitz and Mansour, 1991]. Namely, let $g : \{0,1\}^n \to \mathbb{R}$ be a function that can be represented as an ensemble of T trees each of depth at most d. Then g is $k = O(T \cdot 4^d)$-sparse and of degree at most d (Appendix E.1). Importantly, their spectrum is *exactly sparse* and unlike standard neural networks, which seem to "fill up" the spectrum on the low-degree end, i.e., learn irrelevant low-degree coefficients, decision trees avoid this. Decision trees are well-known to be effective on discrete/tabular data [Arik and Pfister, 2021], and our regularizer prunes the spectrum of the neural network so it behaves similarly.

5 EXPERIMENTS

In this section, we first evaluate our regularization method on higher dimensional input spaces (higher n) on synthetically generated datasets. In this setting, FULLWH is not applicable due to its exponential runtime in n. In addition, we allow varying training set sizes to showcase the efficacy of the regularizer in improving generalization at varying levels in terms of the number of training points in the dataset and especially in the low-data sample regime. Next, we move on to four real-world datasets. We first show the efficacy of our proposed regularizer HASHWH on real-world datasets in terms of achieving better generalization errors, especially in the low-data sample regimes. Finally, using an ablation

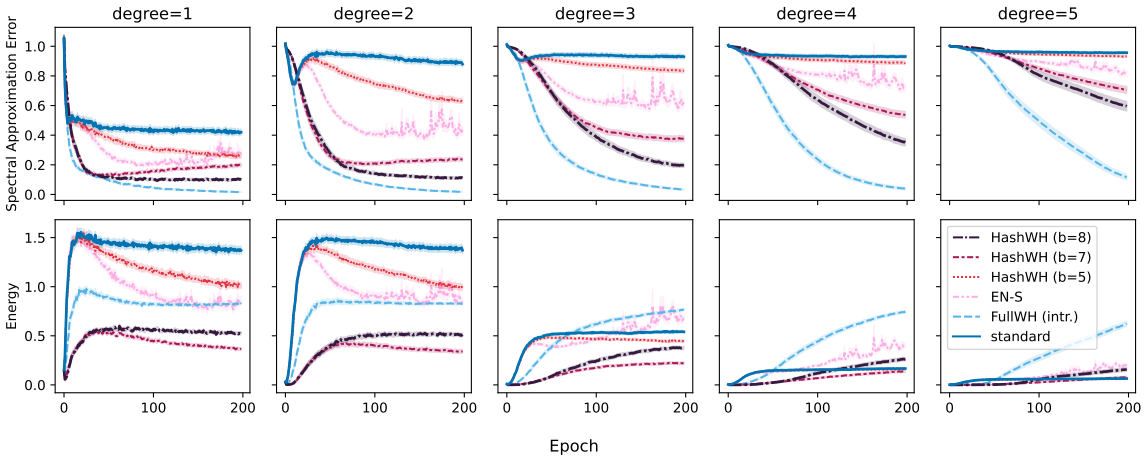


Figure 3: Evolution of the Spectral Approximation Error (SAE) and energy of the network during training, split by frequency degree. Firstly, in a standard neural network, the energy of high-degree frequencies is essentially zero compared to low-degree frequencies. Secondly, for low degrees (2 and 3) the energy continues to increase while the SAE exhibits overfitting behavior. This implies the neural network starts learning erroneous low-degree frequencies after some epochs. Our regularizer prevents overfitting in lower degrees and enforces higher energy on higher-degree frequencies. Regularized networks show lower energies for lower degrees and higher energy for higher degrees when compared to the standard neural network.

study, we experimentally convey that the low-degree bias does not result in lower generalization error.

5.1 SYNTHETIC DATA

Setup. Again, we consider a synthetic pseudo-boolean target function $g^* : \{0,1\}^n \to \mathbb{R}$, which has 25 frequencies in its support $|\text{supp}(g^*)| = 25$, with the degree of maximum five, i.e., $\forall f \in \text{supp}(g^*) : \deg(f) \leq 5$. To draw a g^*, we sample each of its support frequencies f_i by first uniformly sampling its degree $d \sim \mathcal{U}_{\{1,2,3,4,5\}}$, based on which we then sample $f_i \sim \{f \in \{0,1\}^n | \deg(f) = d\}$ and its corresponding amplitude uniformly $\widehat{g^*}(f_i) \sim \mathcal{U}_{[-1,1]}$.

We draw g^* as above for different input dimensions $n \in \{25, 50, 100\}$. We pick points uniformly at random from the input domain $\{0,1\}^n$ and evaluate g^* to generate datasets of various sizes: we generate five independently sampled datasets of size $c \cdot 25n$, for different multipliers $c \in \{1, .., 8\}$ (40 datasets for each g^*). We train a 5-layer fully-connected neural network on each dataset using five different random seeds to account for the randomness in the training procedure. Therefore, for each g^* and dataset size, we train and average over 25 models to capture variance arising from the dataset generation, and also the training procedure.

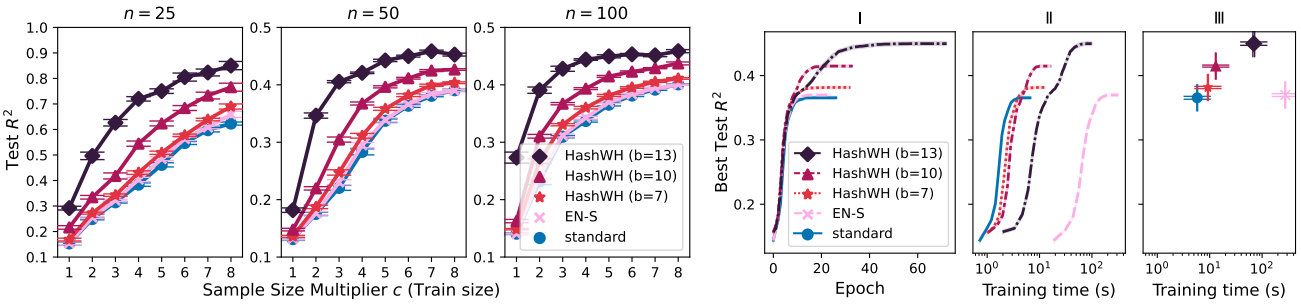
Results. Figure 4a shows the generalization performance of different methods in terms of their R^2 score on a hold-out dataset (details of dataset splits in Appendix D) for different dataset sizes. Our regularization method, HashWH, outperforms the standard network and EN-S in all possible combinations of input dimension, and dataset size. Here, EN-S does not show any significant improvements over the standard neural network, while HASHWH (ours) improves generalization by a large margin. Moreover, its performance is tunable via the hashing matrix size b.

To stress the computational scalability of HASHWH (ours), Figure 4b shows the achievable R^2-score by the number of training epochs and training time for different methods, when $n = 50$ and $c = 5$ (see Appendix F.2 for other settings). The trade-off between the training time and generalization can be directly controlled with the choice of the hashing size b. More importantly, comparing HASHWH with EN-S, we see that for any given R^2 we have runtimes that are orders of magnitude smaller. This is primarily due to the very time-consuming approximation of the Fourier transform of the network at each epoch in EN-S.

5.2 REAL DATA

Next, we assess the performance of our regularization method on four different real-world datasets of varying nature and dimensionality. For baselines, we include not only standard neural networks and EN-S regularization, but also other popular machine learning methods that work well on discrete data, such as ensembles of trees. Three of our datasets are related to protein landscapes [Poelwijk et al., 2019, Sarkisyan et al., 2016, Wu et al., 2016] which are identical to the ones used by the proposers of EN-S [Aghazadeh et al., 2021], and one is a GPU-tuning [Nugteren and Codreanu, 2015] dataset. See Appendix C for dataset details.

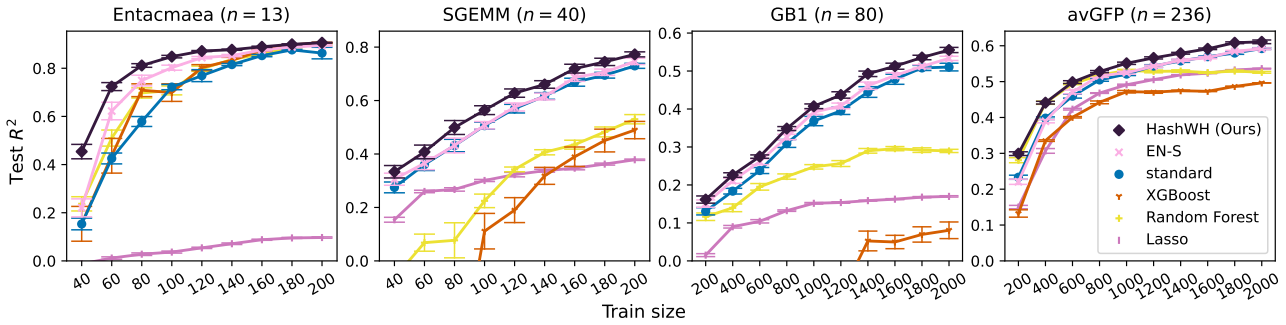
Results. Figure 5a displays the generalization performance of different models in learning the four datasets mentioned,



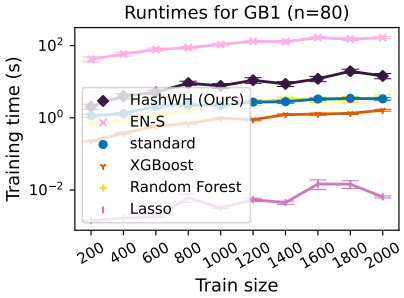
(a) Generalization comparison

(b) Runtime comparison

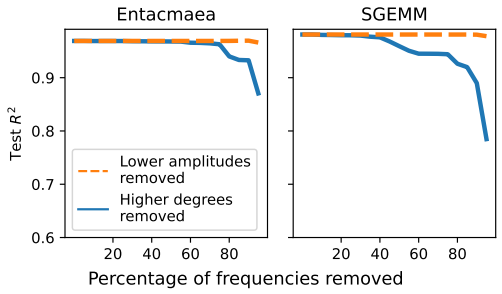
Figure 4: (a) Generalization performance on learning a synthetic function $g^* : \{0,1\}^n \to \mathbb{R}$ with train set size: $c \cdot 25n$ (b) Best achievable test R^2 (I) at end of each epoch (II) up to a certain time (seconds). (III) Shows the early stopped R^2 score vs. time (seconds). We provide significant improvements across all training sizes over EN-S and standard neural networks, while also showing an order of magnitude speed-up compared to EN-S.



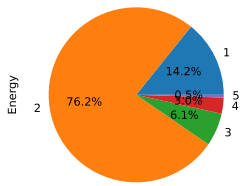
(a) Performance on learning real datasets



(b) Runtimes for GB1



(c) Ablation studies



(d) Energy distribution over degrees in Entacmaea

Figure 5: (a) Generalization performance of standard and regularized neural networks and benchmark ML models on four real datasets. (b) Training times of different models on the GB1 dataset (c) Results of an ablation study on the potential effect of simplicity bias in the generalization error. This figure shows picking higher amplitude coefficients results in better generalization compared to picking the lower degree terms (d) Distribution of the energy over degree-based sets of frequencies in Entacmaea's top 100 Fourier coefficients. This shows high-degree components constitute a non-negligible portion of the energy of the function.

using training sets of small sizes. For each given dataset size we randomly sample the original dataset with five different random seeds to account for the randomness of the dataset sub-sampling. Next, we fit five models with different random seeds to account for the randomness of the training procedure. One standard deviation error bars and averages are plotted accordingly over the 25 runs. It can be seen that our regularization method significantly outperforms the standard neural network as well as popular baseline methods on nearly all datasets and dataset sizes. The margin, however, is somewhat smaller than on the synthetic experiments in some cases. This may be partially explained by the distribu-

tion of energy in a real dataset (Figure 5d), compared to the uniform distribution of energy over different degrees in our synthetic setting.

To highlight the importance of higher degree frequencies, we compute the exact Fourier spectrum of the Entacmaea dataset (which is possible, since all possible input combinations are evaluated in the dataset). Figure 5d shows the energy of 100 frequencies with the highest amplitude (out of 8192 total frequencies) categorized into varying degrees. This shows that the energy of the higher degree frequencies 3 and 4 is comparable to frequencies of degree 1. However, as we showed in the previous section, the standard neural network may not be able to pick up the higher degree frequencies due to its simplicity bias (while also learning erroneous low-degree frequencies).

We also study the relationship between the low-degree spectral bias and generalization in Figure 5c. The study is conducted on the two datasets "Entacmaea" and "SGEMM". We first fit a sparse Fourier function to our training data (see Appendix E). We then start deleting coefficients once according to their degree (highest to lowest and ties are broken randomly) and in another setting according to their amplitude (lowest to highest). To assess generalization, we evaluate the R^2 of the resulting function on a hold-out (test) dataset. This study shows that among functions of equal complexity (in terms of size of support), functions that keep the higher amplitude frequencies as opposed to ones that keep the low-degree ones exhibit better generalization. This might seem evident according to Parseval's identity, which states that time energy and Fourier energy of a function are equal. However, considering the fact that the dataset distribution is not necessarily uniform, there is no reason for this to hold in practice. Furthermore, it shows the importance of our regularization scheme: deviating from low-degree functions and instead aiding the neural network to learn higher amplitude coefficients *regardless* of the degree.

Conclusion We showed through extensive experiments how neural networks have a tendency to not learn high-degree frequencies and overfit in the low-degree part of the spectrum. We proposed a computationally efficient regularizer that aids the network in not overfitting in the low-degree frequencies and also picking up the high-degree frequencies. Finally, we exhibited significant improvements in terms of R^2 score on four real-world datasets compared to various popular models in the low-data regime.

Acknowledgements

This research was supported in part by the NCCR Catalysis (grant number 180544), a National Centre of Competence in Research funded by the Swiss National Science Foundation. We would also like to thank Lars Lorch and Viacheslav Borovitskiy for their detailed and valuable feedback in writing the paper.

References

A. Aghazadeh, H. Nisonoff, O. Ocal, D. H. Brookes, Y. Huang, O. O. Koyluoglu, J. Listgarten, and K. Ramchandran. Epistatic Net allows the sparse spectral regularization of deep neural networks for inferring fitness functions. *Nature Communications*, 12(1):5225, Sept. 2021. ISSN 2041-1723. doi: 10.1038/s41467-021-25371-3. Number: 1 Publisher: Nature Publishing Group.

Z. Allen-Zhu, Y. Li, and Z. Song. A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning*, pages 242–252. PMLR, 2019a.

Z. Allen-Zhu, Y. Li, and Z. Song. On the convergence rate of training recurrent neural networks. *Advances in neural information processing systems*, 32, 2019b.

N. Alon, M. Dietzfelbinger, P. B. Miltersen, E. Petrank, and G. Tardos. Linear hash functions. *Journal of the ACM (JACM)*, 46(5):667–683, 1999.

S. Ö. Arik and T. Pfister. Tabnet: Attentive interpretable tabular learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 6679–6687, 2021.

D. Arpit, S. Jastrzębski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio, and S. Lacoste-Julien. A Closer Look at Memorization in Deep Networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 233–242. PMLR, July 2017. ISSN: 2640-3498.

A. Ballal, C. Laurendon, M. Salmon, M. Vardakou, J. Cheema, M. Defernez, P. E. O'Maille, and A. V. Morozov. Sparse Epistatic Patterns in the Evolution of Terpene Synthases. *Molecular Biology and Evolution*, 37(7):1907–1924, July 2020. ISSN 0737-4038. doi: 10.1093/molbev/msaa052.

R. Basri, M. Galun, A. Geifman, D. Jacobs, Y. Kasten, and S. Kritchman. Frequency bias in neural networks for input of non-uniform density. In *International Conference on Machine Learning*, pages 685–694. PMLR, 2020.

A. Benjamin, D. Rolnick, and K. Kording. Measuring and regularizing networks in function space. In *International Conference on Learning Representations*, 2019.

D. H. Brookes, A. Aghazadeh, and J. Listgarten. On the sparsity of fitness functions and implications for learning. *Proceedings of the National Academy of Sciences of the United States of America*, 119(1):e2109649118, Jan. 2022. ISSN 1091-6490. doi: 10.1073/pnas.2109649118.

L. Chizat, E. Oyallon, and F. Bach. On lazy training in differentiable programming. *Advances in Neural Information Processing Systems*, 32, 2019.

G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

A. Daniely. Sgd learns the conjugate kernel class of the network. *Advances in Neural Information Processing Systems*, 30, 2017.

A. Daniely, R. Frostig, and Y. Singer. Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. *Advances in neural information processing systems*, 29, 2016.

S. S. Du, X. Zhai, B. Poczos, and A. Singh. Gradient descent provably optimizes over-parameterized neural networks. In *International Conference on Learning Representations*, 2019.

K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

A. Jacot, F. Gabriel, and C. Hongler. Neural Tangent Kernel: Convergence and Generalization in Neural Networks. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

D. Kalimeris, G. Kaplun, P. Nakkiran, B. Edelman, T. Yang, B. Barak, and H. Zhang. SGD on Neural Networks Learns Functions of Increasing Complexity. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

E. Kushilevitz and Y. Mansour. Learning decision trees using the fourier spectrum. In *Proceedings of the twenty-third annual ACM symposium on Theory of computing*, pages 455–464, 1991.

J. Lee, Y. Bahri, R. Novak, S. S. Schoenholz, J. Pennington, and J. Sohl-Dickstein. Deep neural networks as gaussian processes. In *International Conference on Learning Representations*, 2018.

J. Lee, L. Xiao, S. Schoenholz, Y. Bahri, R. Novak, J. Sohl-Dickstein, and J. Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. *Advances in neural information processing systems*, 32, 2019.

P. Nakkiran, G. Kaplun, D. Kalimeris, T. Yang, B. L. Edelman, F. Zhang, and B. Barak. Sgd on neural networks learns functions of increasing complexity. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 3496–3506, 2019.

B. Neyshabur, S. Bhojanapalli, D. McAllester, and N. Srebro. Exploring Generalization in Deep Learning, July 2017. arXiv:1706.08947 [cs].

C. Nugteren and V. Codreanu. CLTune: A Generic Auto-Tuner for OpenCL Kernels. In *2015 IEEE 9th International Symposium on Embedded Multicore/Many-core Systems-on-Chip*, pages 195–202, Sept. 2015. doi: 10.1109/MCSoC.2015.10.

R. O'Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.

F. J. Poelwijk, M. Socolich, and R. Ranganathan. Learning the pattern of epistasis linking genotype and phenotype in a protein. *Nature Communications*, 10(1): 4213, Sept. 2019. ISSN 2041-1723. doi: 10.1038/s41467-019-12130-8. Number: 1 Publisher: Nature Publishing Group.

T. Poggio, K. Kawaguchi, Q. Liao, B. Miranda, L. Rosasco, X. Boix, J. Hidary, and H. Mhaskar. Theory of Deep Learning III: explaining the non-overfitting puzzle, Jan. 2018. arXiv:1801.00173 [cs].

N. Rahaman, A. Baratin, D. Arpit, F. Draxler, M. Lin, F. Hamprecht, Y. Bengio, and A. Courville. On the Spectral Bias of Neural Networks. In *Proceedings of the 36th International Conference on Machine Learning*, pages 5301–5310. PMLR, May 2019. ISSN: 2640-3498.

C. E. Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*, pages 63–71. Springer, 2004.

B. Ronen, D. Jacobs, Y. Kasten, and S. Kritchman. The Convergence Rate of Neural Networks for Learned Functions of Different Frequencies. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Z. R. Sailer and M. J. Harms. Detecting High-Order Epistasis in Nonlinear Genotype-Phenotype Maps. *Genetics*, 205(3):1079–1088, Mar. 2017. ISSN 1943-2631. doi: 10.1534/genetics.116.195214.

K. S. Sarkisyan, D. A. Bolotin, M. V. Meer, D. R. Usmanova, A. S. Mishin, G. V. Sharonov, D. N. Ivankov, N. G. Bozhanova, M. S. Baranov, O. Soylemez, N. S. Bogatyreva, P. K. Vlasov, E. S. Egorov, M. D. Logacheva, A. S. Kondrashov, D. M. Chudakov, E. V. Putintseva, I. Z. Mamedov, D. S. Tawfik, K. A. Lukyanov, and F. A. Kondrashov. Local fitness landscape of the green fluorescent protein. *Nature*, 533(7603):397–401, May 2016. ISSN 1476-4687. doi: 10.1038/nature17995. Number: 7603 Publisher: Nature Publishing Group.

H. Shah, K. Tamuly, A. Raghunathan, P. Jain, and P. Netrapalli. The Pitfalls of Simplicity Bias in Neural Networks.

In *Advances in Neural Information Processing Systems*, volume 33, pages 9573–9585. Curran Associates, Inc., 2020.

S. Sun, G. Zhang, J. Shi, and R. Grosse. Functional Variational Bayesian Neural Networks. In *International Conference on Learning Representations*, 2019.

M. Tancik, P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. Barron, and R. Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems*, 33: 7537–7547, 2020.

G. Valle-Perez, C. Q. Camargo, and A. A. Louis. Deep learning generalizes because the parameter-function map is biased towards simple functions. In *International Conference on Learning Representations*, 2019.

Z. Wang, T. Ren, J. Zhu, and B. Zhang. Function space particle optimization for bayesian neural networks. In *International Conference on Learning Representations*, 2019.

N. C. Wu, L. Dai, C. A. Olson, J. O. Lloyd-Smith, and R. Sun. Adaptation in protein fitness landscapes is facilitated by indirect paths. *eLife*, 5:e16965, July 2016. ISSN 2050-084X. doi: 10.7554/eLife.16965. Publisher: eLife Sciences Publications, Ltd.

G. Yang and H. Salman. A Fine-Grained Spectral Perspective on Neural Networks, Apr. 2020. arXiv:1907.10599 [cs, stat].

G. Yang, D. W. Anderson, F. Baier, E. Dohmen, N. Hong, P. D. Carr, S. C. L. Kamerlin, C. J. Jackson, E. Bornberg-Bauer, and N. Tokuriki. Higher-order epistasis shapes the fitness landscape of a xenobiotic-degrading enzyme. *Nature Chemical Biology*, 15(11):1120–1128, Nov. 2019. ISSN 1552-4469. doi: 10.1038/s41589-019-0386-3. Number: 11 Publisher: Nature Publishing Group.