On the non-universality of deep learning: quantifying the cost of symmetry

Anonymous Author(s)
Affiliation
Address
email

Abstract

Recent work [AS20] has shown that deep learning is a universal paradigm: it can emulate any efficient learning algorithm. However, this result relies on contrived neural network architectures that do not reflect architectures used in practice. In this paper, we prove that if one trains "regular" architectures and initializations, where the training algorithm satisfies a certain symmetry (a.k.a. equivariance), then deep learning is no longer universal. As our first main contribution, we prove a general computational lower bound for G-equivariant gradient descent in terms of the "G-orbit-alignment" of the data. We apply this bound in conjunction with representation theory tools to obtain two new results: (i) we characterize which functions fully-connected networks can weakly-learn on the binary hypercube and on the unit sphere, which demonstrates that as far as weak-learning and fully-connected networks are concerned, depth-2 is as powerful as any other depth; (ii) we take a further step towards characterizing which functions with latent low-dimensional structure neural networks can learn by strengthening the merged-staircase property necessity result of [ABM22] beyond the mean-field regime, to any architecture such that gradient descent is permutation-equivariant. As our second main contribution, we extend the ideas to stochastic gradient descent, proving hardness results for learning on fully-connected networks based on the Learning Parities with Noise cryptographic assumption.

1 Introduction

Over the last decade, deep learning has made impressive advances in a diverse range of areas, including image classification [KSH12], classical board games [SHS+18], language translation [BCB14], physics [RPK19], and, most recently, programming [LCC+22]. Deep learning trained with simple gradient-based optimizers has surpassed previous classical methods for these tasks, raising the question of how much further these advances can be pushed. Can we hope for deep learning methods to eventually replace all other learning algorithms? In other words, is deep learning a universal learning paradigm? Recently, [AS20] proved that in a certain sense the answer is yes: any PAC-learning algorithm [Val84] can be efficiently implemented by a neural network trained by stochastic gradient descent; analogously, any Statistical Query algorithm [Kea98] can be efficiently implemented by a neural network trained by noisy gradient descent. In short, for any learning algorithm one can construct a deep learning architecture capable of efficiently replacing it.

However, there is a catch: the result of [AS20] relies on a very carefully choice of neural network architecture, which is capable of emulating arbitrary learning algorithms using handcrafted memory and computation modules. This is far from the architectures which have been shown to be successful in practice. Indeed, although neural networks in practice do incorporate some domain knowledge (e.g., convolutional networks and transformers for image recognition and language processing

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

[LB+95, LKF10], graph neural networks for analyzing graph data [GMS05, BZSL13, VCC+17], and equivariant neural networks [CW16] such as Lorentz-equivariant networks for particle physics [BAO+20]), they have more "regularity" than the architectures of [AS20], in the sense that they do not rely on heterogeneous and carefully assigned initial weights. We therefore refine our question:

Is deep learning with "regular" architectures and initializations a universal learning paradigm? If not, can we quantify its limitations?

The above question is broad, and we would like an answer that is applicable to the incredible diversity of possible architectures. In order to formalize the problem and develop a general theory, we take an approach pioneered by [Ng04] and developed by [LZA21] of understanding deep learning on a certain architecture and initialization through the *symmetry group* G (otherwise known as equivariance group) of the learning algorithm. Namely, an algorithm $\mathcal{A}$, such as gradient descent training, that takes in a data distribution $\mathcal{D} \in \mathcal{P}(\mathcal{X} \times \mathbb{R})$[1] and outputs a function $\mathcal{A}(\mathcal{D}) : \mathcal{X} \to \mathbb{R}$ is said to be G-equivariant if for each $g \in G$

$$\mathcal{A}(g(\mathcal{D})) \stackrel{d}{=} \mathcal{A}(\mathcal{D}) \circ g. \tag{1}$$

Here g acts on the data space $\mathcal{X}$, and so is viewed as a function $g : \mathcal{X} \to \mathcal{X}$, and $g(\mathcal{D})$ is the distribution of $(g(\boldsymbol{x}), y)$, where $(\boldsymbol{x}, y) \sim \mathcal{D}$. The equivariance group depends on the particular optimizer used, the architecture, and the initialization [Ng04, LZA21]. We note that the equivariance group of a training algorithm is *distinct from, and should not be confused with,* the equivariance group of an architecture in the context of geometric deep learning [BBCV21].

We think of a deep learning algorithm that has a large equivariance group as being "regular", in opposition to the emulation architectures of [AS20], for which training has only the trivial equivariance. Studying the equivariance group of a training algorithm is a useful perspective since it is nontrivial in most practical cases: for example, we have orthogonal equivariance for gradient descent training of fully-connected networks with Gaussian initialization [Ng04], and permutation equivariance for gradient descent training of i.i.d. initialized ResNets and certain transformers. Furthermore, [LZA21, Theorem C.1] provides general conditions under which a deep learning training algorithm is G-equivariant.

In this work, we prove computational limitations on the performance of deep learning when the G-equivariance group of noisy gradient descent (GD) or stochastic gradient descent (SGD) training is not the trivial group. Our results show that deep learning with *regular* architectures and initializations is *not universal*, implying a separation with the carefully-crafted initializations and architectures considered in [AS20]. Specifically, the first contribution of this work is to develop a general theory for computational lower bounds for G-equivariant GD. This allows us (a) to fully characterize which functions can be weakly learned fully-connected networks on both the hypercube and the unit sphere, showing that, for fully-connected networks and weak learning, two-layer networks are as powerful as networks of arbitrary depth; and (b) to take a step towards characterizing which functions with latent low-dimensional structure can be learned, by extending the necessity result of [ABM22] for staircases beyond the two-layer mean-field regime. The second main contribution of our paper is to demonstrate nontrivial computational hardness results for fully-connected with Gaussian initialization trained by SGD, implying that these cannot learn universally, and providing a separation with [AS20]. An informal overview of these contributions is given in Sections 1.2 and 1.3, but first we cover prior work in more detail in Section 1.1.

1.1 Related work

The key challenge to proving limitations of deep learning is its nonlinear and nonconvex nature. Due to this, one direction of prior work has focused on proving limitations of linear methods such as kernel methods. Dimension lower bound arguments originating with [Bar93] have recently been refined and applied to show failure of general kernel methods on certain problems [WLLM19, AZL19, KMS20, AZL20, Hsu, HSSVG21, ABM22], which implies failure of neural networks in the Neural Tangent Kernel (NTK) regime [JGH18], but does not apply to general nonlinear training. Furthermore, for specific architectures such as fully-connected architectures [GMMM21, Mis22] and convolutional architectures [MM21], the kernels and random features models at random initialization are now well understood, yielding bounds for training in the NTK regime with those architectures.

[1] The set of probability distributions on Ω is denoted by $\mathcal{P}(\Omega)$.

For nonlinear training, considerably less is known. The idea of studying equivariance groups of training was introduced in [Ng04], which used it to show a task with a $\Omega(d)$ versus $O(1)$ sample complexity separation for learning a certain distribution on d-dimensional space with an SGD-trained, Gaussian-initialized fully-connected architecture versus an arbitrary algorithm. More recently, [LZA21] built on [Ng04] to show a $O(1)$ versus $\Omega(d^2)$ sample-complexity separation between convolutional and fully-connected architectures that are initialized i.i.d. Gaussian and trained by SGD. In this paper, we also analyze the equivariance group of training, but with the different focus of proving computational lower bounds. For nonlinear training, there are known lower bounds for nonlinear stochastic gradient descent (SGD) training of two-layer mean-field networks [ABM22], but these only apply when SGD converges to the mean-field limit, and so do not apply to general architectures. It is also known that networks trained with noisy[2] gradient descent (GD) fall under the Statistical Query (SQ) framework [Kea98], which allows to show lower bounds for GD training based on SQ lower bounds. This has been combined in [SSSS17, MSS20, ACHM22] with the permutation symmetry of GD-training of i.i.d. fully-connected networks to prove lower bounds for learning high-degree parities and polynomials. Our work reveals that their arguments can be fruitfully viewed in the broader context of more general group symmetries.

We also note the sample-complexity benefits of invariant and equivariant *architectures* have been studied for linear methods in [MMM21, EZ21, Ele21, BVB21], and recently for non-linear PAC learning by [Ele22]. However, we remind the reader that equivariant *training algorithms* and equivariant *architectures* are distinct, and the former is the focus of this work.

1.2 Contribution 1: Lower bounds for noisy gradient descent (GD)

We consider the supervised learning setup where we train a neural network $f_{\mathsf{NN}}(\cdot;\boldsymbol{\theta}) : \mathcal{X} \to \mathbb{R}$ parametrized by $\boldsymbol{\theta} \in \mathbb{R}^p$ to minimize the mean-squared error on a data distribution $\mathcal{D} \in \mathcal{P}(\mathcal{X} \times \mathbb{R})$,

$$\ell_{\mathcal{D}}(\boldsymbol{\theta}) = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}[\ell(\boldsymbol{\theta};\boldsymbol{x},y)], \text{ where } \ell(\boldsymbol{\theta};\boldsymbol{x},y) = (y - f_{\mathsf{NN}}(\boldsymbol{x};\boldsymbol{\theta}))^2. \tag{2}$$

The noisy Gradient Descent (GD) training algorithm randomly initializes $\boldsymbol{\theta}^0 \sim \mu_{\boldsymbol{\theta}}$ for some initialization distribution $\mu_{\boldsymbol{\theta}} \in \mathcal{P}(\mathbb{R}^p)$, and then iteratively updates the parameters with step size $\eta > 0$ in the direction of the population loss gradient, plus Gaussian noise $\boldsymbol{\xi}^k \sim \mathcal{N}(0, \tau^2 \boldsymbol{I})$:

$$\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^k - \eta \nabla_{\boldsymbol{\theta}} \ell_{\mathcal{D}}(\boldsymbol{\theta}^k) + \boldsymbol{\xi}_k. \tag{GD}$$

Roughly speaking, GD training is considered efficient if the network is Lipschitz, the step size is bounded, and the noise magnitude τ is lower-bounded, since then it can be efficiently implemented with noisy minibatch SGD. Assuming that the GD training algorithm is G-equivariant as in (1) for some nontrivial group G, then we show a general result that certain data distributions cannot be efficiently learned. For our result, we introduce a quantity called the G-orbit-alignment:

Definition 1.1 (G-orbit-alignment)**.** *Let G be a compact group, let $\mu_{\mathcal{X}} \in \mathcal{P}(\mathcal{X})$ be a distribution over data points, and let $f : \mathcal{X} \to \mathbb{R}$ be a labeling function. The G-orbit-alignment of $(\mu_{\mathcal{X}}, f)$ is:*

$$\mathcal{C}((\mu_{\mathcal{X}}, f); G) = \sup_h \mathbb{E}_{g\sim\mu_G}[\mathbb{E}_{\boldsymbol{x}\sim\mu_{\mathcal{X}}}[f(g(\boldsymbol{x}))h(\boldsymbol{x})]^2],$$

where μ_G is the Haar measure of G and the supremum is over $h \in L^2(\mu_{\mathcal{X}})$ such that $\|h\|^2 = 1$.

In our applications, we use tools from representation theory (see e.g., [Kna96]) to evaluate the G-orbit-alignment. Using the G-orbit-alignment, we can prove a master theorem for lower bounds, stated here informally:

Theorem 1.2 (GD lower bound, informal statement of Theorem 3.1)**.** *Let $\mathcal{D}_f \in \mathcal{P}(\mathcal{X} \times \mathbb{R})$ be the distribution of $(\boldsymbol{x}, f(\boldsymbol{x}))$ for $\boldsymbol{x} \sim \mu_{\mathcal{X}}$. If $\mu_{\mathcal{X}}$ is G-invariant, and the G-orbit-alignment of $(\mu_{\mathcal{X}}, f)$ is small, then networks cannot efficiently learn f when trained by (GD) on data distribution $\mathcal{D} = \mathcal{D}_f$ if the GD algorithm is G-equivariant.*

The first insight behind this theorem can be traced back at least to [Ng04]: if a G-equivariant algorithm can learn the function f by training on the distribution $\mathcal{D}_f$, then for any group element $g \in G$, it can learn $f \circ g$ by training on the distribution $\mathcal{D}_{f\circ g}$. In other words, the algorithm can learn

[2] Here the noise is used to control the gradients' precision as in [AS20, AKM+21].

the class of functions $\mathcal{F} = \{f \circ g : g \in g\}$, which can potentially be much larger than just the single function $\{f\}$. Therefore, if we could prove that the class of functions $\mathcal{F}$ cannot be efficiently learned by *any* network with GD, then this would show that f cannot be efficiently learned by any network where GD is G-equivariant.

The second insight of the theorem is to show that if the G-orbit-alignment of $(\mu_{\mathcal{X}}, f)$ is small, then $\mathcal{F}$ cannot be efficiently learned by GD on *any* network. Why does the G-orbit-alignment arise? The intuition is that the G-orbit-alignment measures the diversity of the functions in $\mathcal{F}$: if the G-orbit-alignment of f is large, then there must be some function h that has high correlation with most of the functions in $\mathcal{F}$, and therefore the functions in $\mathcal{F}$ are not very diverse. On the other hand, if the G-orbit-alignment is small, then this must mean that there is no h that correlates with most of them, and so $\mathcal{F}$ is very diverse, which can be used to show that it is hard to learn by gradient descent. This type of argument appears in [AS20, ACHM22] in the specific case of Boolean functions and for permutation equivariance; our proof both applies to a more general setting (beyond Boolean functions and permutations) and sharpens the bounds in these papers.[3]

While the idea behind Theorem 1.2 is intuitively simple, the theorem is very broadly applicable, to arbitrary groups and beyond functions defined on the Boolean hypercube. We demonstrate its power and ease-of-use by deriving two novel applications that are interesting in their own right, and that showcase the limitations of fully-connected networks with standard initializations. We expect that Theorem 1.2 will find further applications in future work.

Application: Characterization of weak-learnability by fully-connected networks In our first application, we consider weak-learnability: i.e., when can a function be learned non-negligibly better than just outputting the estimate $f_{\mathsf{NN}} \equiv 0$? Using Theorem 1.2, we characterize which functions over the binary hypercube $f : \{+1, -1\}^d \to \mathbb{R}$ and over the sphere $f : \mathbb{S}^{d-1} \to \mathbb{R}$ are efficiently weakly-learnable by GD-trained fully-connected networks with i.i.d. symmetric and i.i.d. Gaussian initialization, respectively. The takeaway is that a function $f : \{+1, -1\}^d \to \mathbb{R}$ is weakly-learnable if and only if it has a nonnegligible Fourier coefficient of order $O(1)$ or $d - O(1)$. Similarly, a function $f : \mathbb{S}^{d-1} \to \mathbb{R}$ is weakly-learnable if and only if it has nonnegligible projection onto the degree-$O(1)$ spherical harmonics. Such functions can be weakly-learned by 2-layer fully-connected networks, which shows that adding more depth does not help for weak learning. This application is presented in Section 3.1.

Application: Evidence for the staircase property In our second application, we consider learning a target function $f : \{+1, -1\}^d \to \mathbb{R}$ that only depends on the first P coordinates, $f(\boldsymbol{x}) = h(x_1, \ldots, x_P)$. Our regime of interest here is when the function $h : \{+1, -1\}^P \to \mathbb{R}$ remains fixed and the dimension d grows, since this models the situation where a latent low-dimensional space determines the labels in a high-dimensional dataset. Recently, [ABM22] studied SGD-training of mean-field two-layer networks, and gave a near-characterization of which functions can be learned to arbitrary accuracy ϵ in $O_{h,\epsilon}(d)$ samples, in terms of the *merged-staircase property* (MSP). Using Theorem 1.2, we prove that the MSP is necessary for GD-learnability by any permutation-equivariant architecture, well beyond the 2-layer mean-field regime, and also generalize it beyond leaps of size 1. This application is presented in Section 3.2.

1.3 Contribution 2: Hardness for stochastic gradient descent (SGD)

Our second main contribution concerns Stochastic Gradient Descent (SGD) training, which randomly initializes the weights $\boldsymbol{\theta}^0 \sim \mu_{\boldsymbol{\theta}}$, and then iteratively trains the parameters with the following update rule to try to minimize the loss (2):

$$\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^k - \eta \nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}^k; y_{k+1}, \boldsymbol{x}_{k+1}), \tag{SGD}$$

where $(y_{k+1}, \boldsymbol{x}_{k+1}) \sim \mathcal{D}$ is a fresh sample on each iteration, and $\eta > 0$ is the learning rate.[4]

Proving lower bounds for SGD is a notoriously difficult problem [AKM+21], exacerbated by the fact that for general architectures SGD can be used to simulate any polynomial-time learning algorithm

[3] Our bound can also be interpreted in terms of the Statistical Query framework, as we discuss in Appendix A.4.

[4] For brevity, we focus on one-pass SGD with a single fresh sample per iteration. Our results extend to empirical risk minimization (ERM) setting and to mini-batch SGD, see Remark E.1.

[AS20]. However, we demonstrate that one can prove hardness results for SGD training based off of cryptographic assumptions when the training has large equivariance groups. This demonstrates the *non-universality* of SGD in certain practical architectures.

Theorem 1.3 (Hardness for SGD, informal statement of Theorem 4.4)**.** *Under the assumption that the Learning Parities with Noise (LPN) problem*[5] *is hard, fully-connected neural networks with Gaussian initialization trained by SGD cannot learn* $f_{\mathrm{mod8}} : \{+1, -1\}^d \to \{0, \ldots, 7\}$*,*

$$f_{\mathrm{mod8}}(\boldsymbol{x}) \equiv \sum_{i=1}^{d} x_i \pmod 8,$$

in polynomial time from noisy samples $(\boldsymbol{x}, f_{\mathrm{mod8}}(\boldsymbol{x}) + \xi)$ *where* $\boldsymbol{x} \sim \{+1, -1\}^d$ *and* $\xi \sim \mathcal{N}(0, 1)$*.*

This result is the first of its kind, in the sense that it proves nontrivial limitations of SGD training based on an average-case reduction from a cryptographic problem. The architecture and initialization play a crucial role: a network whose first layer is initialized and fixed to all 1's, and whose second layer is trained, would be able to easily learn f_{mod8}, or indeed any other bounded function depending only on $\sum_i x_i$. On the other hand, SGD on fully-connected, Gaussian-initialized architectures is *sign-flip* equivariant, and we can use this to show that f_{mod8} is hard to learn.

2 Preliminaries

Notation The law of a random variable X is denoted by $\mathcal{L}(X)$. If S is a finite set, then $X \sim S$ is shorthand for $X \sim \mathrm{Unif}[S]$. For any set Ω, we let $\mathcal{P}(\Omega)$ denote the set of distributions on Ω. Let $\mathcal{H}_d = \{+1, -1\}^d$ denote the binary hypercube, and let $\mathbb{S}^{d-1} = \{\boldsymbol{x} \in \mathbb{R}^d : \|\boldsymbol{x}\|_2 = 1\}$ denote the unit sphere. We use $\odot$ to denote the elementwise product of vectors. For any $\mu_{\mathcal{X}} \in \mathcal{P}(\mathcal{X})$, and any group G acting on $\mathcal{X}$, we say $\mu_{\mathcal{X}}$ is G-invariant if $g(\boldsymbol{x}) \overset{d}{=} \boldsymbol{x}$ for $\boldsymbol{x} \sim \mu_{\mathcal{X}}$.

2.1 Equivariance of GD and SGD

We define GD and SGD equivariance separately.

Definition 2.1. *Let* $\mathcal{A}^{\mathrm{GD}}$ *be the algorithm that takes in data distribution* $\mathcal{D} \in \mathcal{P}(\mathcal{X} \times \mathbb{R})$*, runs* (GD) *on initialization* $\boldsymbol{\theta}^0 \sim \mu_{\boldsymbol{\theta}}$ *for* k *steps, and outputs the function* $\mathcal{A}^{\mathrm{GD}}(\mathcal{D}) = f_{\mathsf{NN}}(\cdot; \boldsymbol{\theta}^k)$

We say "$(f_{\mathsf{NN}}, \mu_{\boldsymbol{\theta}})$*-GD is* G*-equivariant" if* $\mathcal{A}^{\mathrm{GD}}$ *is* G*-equivariant in the sense of* (1)*.*

Definition 2.2. *Let* $\mathcal{A}^{\mathrm{SGD}}$ *be the algorithm that takes in samples* $\{(\boldsymbol{x}_i, y_i)\}_{i \in [n]}$*, runs* (SGD) *on initialization* $\boldsymbol{\theta}^0 \sim \mu_{\boldsymbol{\theta}}$ *for* n *steps, and outputs* $\mathcal{A}^{\mathrm{SGD}}(\{(\boldsymbol{x}_i, y_i)\}_{i \in [n]}) = f_{\mathsf{NN}}(\cdot; \boldsymbol{\theta}^k)$*.*

We say "$(f_{\mathsf{NN}}, \mu_{\boldsymbol{\theta}})$*-SGD is* G*-equivariant" if* $\mathcal{A}^{\mathrm{SGD}}(\{(g(\boldsymbol{x}_i), y_i)\}_{i \in [n]}) \overset{d}{=} \mathcal{A}^{\mathrm{SGD}}(\{(\boldsymbol{x}_i, y_i)\}_{i \in [n]}) \circ g$ *for any* $g \in G$*, and any samples* $\{(\boldsymbol{x}_i, y_i)\}_{i \in [n]}$*.*

2.2 Regularity conditions on networks imply equivariances of GD and SGD

We take the data space $\mathcal{X} \subseteq \mathbb{R}^d$, and consider the following groups that act on $\mathbb{R}^d$.

Definition 2.3. *Define the following groups and actions:*

- *Let* $G_{perm} = S_d$ *denote the group of permutations on* $[d]$*. An element* $\sigma \in G_{perm}$ *acts on* $\boldsymbol{x} \in \mathbb{R}^d$ *in the standard way:* $\sigma(\boldsymbol{x}) = (x_{\sigma(1)}, \ldots, x_{\sigma(d)})$*.*
- *Let* $G_{sign,perm}$ *denote the group of signed permutations, an element* $g = (\boldsymbol{s}, \sigma) \in G_{sign,perm}$ *is given by a sign-flip vector* $\boldsymbol{s} \in \{+1, -1\}^d$ *and a permutation* $\sigma \in G_{perm}$*. It acts on* $\boldsymbol{x} \in \mathbb{R}^d$ *by* $g(\boldsymbol{x}) = \boldsymbol{s} \odot \sigma(\boldsymbol{x}) = (s_1 x_{\sigma(1)}, \ldots, s_d x_{\sigma(d)})$*.*[6]
- *Let* $G_{rot} = G_{rot} \subseteq GL(d, \mathbb{R})$ *denote the rotation group. An element* $g \in G_{rot}$ *is a rotation matrix that acts on* $\boldsymbol{x}$ *by matrix multiplication.*

[5] See Section 4 and Appendix D.3 for definitions and discussion on LPN.

[6] The group product is $g_1 g_2 = (\boldsymbol{s}_1, \sigma_1)(\boldsymbol{s}_2, \sigma_2) = (\boldsymbol{s}_1 \odot \sigma_1(\boldsymbol{s}_2), \sigma_1 \circ \sigma_2)$.

Under mild conditions on the neural network architecture and initialization, GD and SGD training are known to be G_{perm}-, $G_{sign,perm}$-, or G_{rot}-equivariant [Ng04, LZA21].

Assumption 2.4 (Fully-connected i.i.d. first layer and no skip connections from the input)**.** *We can decompose the parameters as $\boldsymbol{\theta} = (\boldsymbol{W}, \boldsymbol{\psi})$, where $\boldsymbol{W} \in \mathbb{R}^{m \times d}$ is the matrix of the first-layer weights, and there is a function $g_{\mathsf{NN}}(\cdot; \tilde{\boldsymbol{\theta}}) : \mathbb{R}^m \to \mathbb{R}$ such that $f_{\mathsf{NN}}(\boldsymbol{x}; \boldsymbol{\theta}) = g_{\mathsf{NN}}(\boldsymbol{W}\boldsymbol{x}; \boldsymbol{\psi})$. Furthermore, the initialization distribution is $\mu_{\boldsymbol{\theta}} = \mu_{\boldsymbol{W}} \times \mu_{\boldsymbol{\psi}}$, where $\mu_{\boldsymbol{W}} = \mu_w^{m \otimes d}$ for $\mu_w \in \mathcal{P}(\mathbb{R})$.*

Notice that Assumption 2.4 is satisfied by fully-connected networks with i.i.d. initialization. Under assumptions on μ_w, we obtain equivariances of GD and SGD (see Appendix E for proofs.)

Proposition 2.5 ([Ng04, LZA21])**.** *Under Assumption 2.4, GD and SGD are G_{perm}-equivariant. If μ_w is sign-flip symmetric, then GD and SGD are $G_{sign,perm}$-equivariant. If $\mu_w = \mathcal{N}(0, \sigma^2)$ for some σ, then GD and SGD are G_{rot}-equivariant.*

3 Lower bounds for learning with GD

In this section, let $\mathcal{D}(f, \mu_{\mathcal{X}}) \in \mathcal{P}(\mathcal{X} \times \mathbb{R})$ denote the distribution of $(\boldsymbol{x}, f(\boldsymbol{x}))$ where $\boldsymbol{x} \sim \mu_{\mathcal{X}}$.

We give a master theorem for lower bounds for learning with G-equivariant GD, depending on the G-orbit-alignment from Definition 1.1.

Theorem 3.1 (Lower bound for GD based on G-orbit-alignment)**.** *Let G be a compact group, and let $f_{\mathsf{NN}}(\cdot; \boldsymbol{\theta}) : \mathcal{X} \to \mathbb{R}$ be an architecture and $\mu_{\boldsymbol{\theta}} \in \mathcal{P}(\mathbb{R}^p)$ be an initialization such that GD is G-equivariant and the architecture has bounded Lipschitz constant $L = \sup_{\boldsymbol{\theta}, \boldsymbol{x}} \|\nabla_{\boldsymbol{\theta}} f_{\mathsf{NN}}(\boldsymbol{x}; \boldsymbol{\theta})\|$.*

Fix any G-invariant distribution $\mu_{\mathcal{X}} \in \mathcal{P}(\mathcal{X})$, any label function $f \in L^2(\mu_{\mathcal{X}})$, and any baseline function $\alpha \in L^2(\mu_{\mathcal{X}})$ satisfying $\alpha \circ g = \alpha$ for all $g \in G$. Let $\boldsymbol{\theta}^k$ be the random weights after k time-steps of GD training with noise parameter $\tau > 0$ and step size $\eta > 0$ on the distribution $\mathcal{D} = \mathcal{D}(f_*, \mu_{\mathcal{X}})$. Then, for any $\epsilon > 0$,

$$\mathbb{P}_{\boldsymbol{\theta}^k}[\ell_{\mathcal{D}}(\boldsymbol{\theta}^k) \leq \|f_* - \alpha\|^2_{L^2(\mu_{\mathcal{X}})} - \epsilon] \leq \frac{\eta L \sqrt{k\mathcal{C}}}{2\tau} + \frac{\mathcal{C}}{\epsilon},$$

where $\mathcal{C} = \mathcal{C}((f_* - \alpha, \mu_{\mathcal{X}}); G)$ is the G-orbit-alignment of Definition 1.1.

As discussed in Section 1.2, the theorem states that if the G-orbit-alignment is very small, then GD training cannot efficiently improve on the trivial loss. Appendix A shows a generalization of the theorem for learning a class of functions $\mathcal{F} = \{f_1, \ldots, f_m\}$ instead of just a single function f_*. This generalization provides strictly tighter bounds than [AS20], even in the case where G is the trivial group with one element. We demonstrate the usefulness of the theorem through two novel applications in Sections 3.1 and 3.2.

3.1 Application: Characterizing weak-learnability by fully-connected networks

In our first application of Theorem 3.1, we consider fully-connected architectures with i.i.d. initialization, and show how to use their training equivariances to characterize what functions they can weakly-learn: i.e., for what target functions f_* they can efficiently achieve a non-negligible correlation after training.

Definition 3.2 (Weak learnability)**.** *Let $\{\mu_d\}_{d \in \mathbb{N}}$ be a family of distributions $\mu_d \in \mathcal{P}(\mathcal{X}_d)$, and let $\{f_d\}_{d \in \mathbb{N}}$ be a family of functions $f_d \in L^2(\mu_d)$. Finally, let $\{\tilde{f}_d\}_{d \in \mathbb{N}}$ be a family of estimators, where $\tilde{f}_d$ is a random function in $L^2(\mu_d)$. We say that $\{f_d, \mu_d\}_{d \in \mathbb{N}}$ is "weakly-learned" by the family of estimators $\{\tilde{f}_d\}_{d \in \mathbb{N}}$ if there are constants $d_0, C > 0$ such that for all $d > d_0$,*

$$\mathbb{P}_{\tilde{f}_d}[\|f_d - \tilde{f}_d\|^2_{L^2(\mu_d)} \leq \|f_d\|^2_{L^2(\mu_d)} - d^{-C}] \geq 9/10. \tag{3}$$

In words, weak-learning measures whether the family of estimators $\{\tilde{f}_d\}$ has a non-negligible edge over simply estimating with the identically zero functions $\tilde{f}_d \equiv 0$. We study weak-learnability by GD-trained fully-connected networks.

Definition 3.3. *We say that $\{f_d, \mu_d\}_{d \in \mathbb{N}}$ is efficiently weakly-learnable by GD-trained fully-connected networks if there are fully-connected networks and initializations $\{f_{\mathsf{NN},d}, \mu_{\boldsymbol{\theta},d}\}$, and hyperparameters $\{\eta_d, k_d, \tau_d\}$ such that for some constant $c > 0$,*

- *Hyperparameters are polynomial size:* $0 \leq \eta_d, k_d, 1/\tau_d, L_d \leq O(d^c)$;
- *Networks are polynomially Lipschitz:* $L_d = \sup_{\boldsymbol{x}} \|\nabla_{\boldsymbol{\theta}} f_{\mathsf{NN},d}(\boldsymbol{x};\boldsymbol{\theta})\| \leq O(d^c)$;
- $\{\tilde{f}_d\}$ *weakly-learns* $\{f_d, \mu_d\}$ *in the sense of Definition 3.2, where* $\tilde{f}_d = f_{\mathsf{NN}}(\cdot;\boldsymbol{\theta}_d)$ *for weights* $\boldsymbol{\theta}_d$ *that are GD-trained on* $\mathcal{D}(f_d, \mu_d)$ *for* k_d *steps with step size* η_d *and noise* τ_d*, starting from initialization* $\mu_{\boldsymbol{\theta},d}$.

If $\mu_{\boldsymbol{\theta},d}$ *is i.i.d copies of a symmetric distribution, we say that the fully-connected networks are symmetrically-initialized, and Gaussian-initialized if* $\mu_{\boldsymbol{\theta},d}$ *is i.i.d. copies of a Gaussian distribution.*

3.1.1 Functions on hypercube, fully-connected networks with i.i.d. symmetric initialization

Let us first consider functions on the Boolean hypercube $f : \mathcal{H}_d \to \mathbb{R}$. An important fact is that these can be uniquely written as a multilinear polynomial $f(\boldsymbol{x}) = \sum_{S \subseteq [d]} \hat{f}(S) \prod_{i \in S} x_i$, where $\hat{f}(S)$ are the Fourier coefficients of f. We characterize weak learnability of functions on the hypercube in terms of their Fourier coefficients. The proof is deferred to Appendix B.1.

Theorem 3.4. *Let* $\{f_d\}_{d \in \mathbb{N}}$ *be a family of functions* $f_d : \mathcal{H}_d \to \mathbb{R}$ *with* $\|f_d\|_{L^2(\mathcal{H}_d)} \leq 1$*. Then* $\{f_d, \mathcal{H}_d\}$ *is efficiently weakly-learnable by GD-trained symmetrically-initialized fully-connected networks if and only if there is a constant* $C > 0$ *such that for each* $d \in \mathbb{N}$ *there is* $S_d \subseteq [d]$ *with* $|S_d| \leq C$ *or* $|S_d| \geq d - C$*, and* $|\hat{f}_d(S_d)| \geq \Omega(d^{-C})$.

The algorithmic result can be achieved by two-layer fully-connected networks, and relies on random features analysis where each network weight is initialized to 0 with probability $1 - p$, and $+1$ or -1 with equal probability $p/2$.[7] Therefore, for weak learning, two-layer networks are as good as networks of any depth. For the converse impossibility result, we apply Theorem 3.1, recalling that GD is $G_{sign,perm}$-equivariant by Proposition 2.5, and noting that $G_{sign,perm}$-orbit-alignment is:

Lemma 3.5. *Let* $f : \mathcal{H}_d \to \mathbb{R}$*. Then* $\mathcal{C}((f, \mathcal{H}_d); G_{sign,perm}) = \max_{k \in [d]} \binom{d}{k}^{-1} \sum_{\substack{S \subseteq [d] \\ |S| = k}} \hat{f}(S)^2$.

This lemma can be proved via direct manipulation or using the representation theory of $G_{sign,perm}$. It implies that if the Fourier coefficients of f are negligible for all S s.t. $\min(|S|, d - |S|) \leq O(1)$, then the $G_{sign,perm}$-orbit-alignment of f is negligible and f cannot be learned efficiently.

3.1.2 Functions on sphere, fully-connected networks with i.i.d. Gaussian initialization

We now study learning a target function on the unit sphere, $f \in L^2(\mathbb{S}^{d-1})$, where we take the standard Lebesgue measure on $\mathbb{S}^{d-1}$. A key fact in harmonic analysis is that $L^2(\mathbb{S}^{d-1})$ can be written as the direct sum of subspaces spanned by spherical harmonics of each degree: $L^2(\mathbb{S}^{d-1}) = \oplus_{l=0}^{\infty} \mathcal{V}_{d,l}$, where $\mathcal{V}_{d,l} \subseteq L^2(\mathbb{S}^{d-1})$ is the space of degree-ℓ spherical harmonics (see, e.g., [Hoc12]). Let $P_{\mathcal{V}_{d,l}} : L^2(\mathbb{S}^{d-1}) \to \mathcal{V}_{d,l}$ be the projection operator to the space of degree-l spherical harmonics, which has dimension $\dim(\mathcal{V}_{d,l}) = \frac{2l+d-2}{l}\binom{l+d-3}{l-1}$. In Appendix B.2, we prove the following characterization of when functions on the sphere are weakly-learnable.

Theorem 3.6. *Let* $\{f_d\}_{d \in \mathbb{N}}$ *be a family of functions* $f_d : \mathbb{S}^{d-1} \to \mathbb{R}$ *with* $\|f_d\|_{L^2(\mathbb{S}^{d-1})} \leq 1$*. Then* $\{f_d, \mathbb{S}^{d-1}\}$ *is efficiently weakly-learnable by GD-trained Gaussian-initialized fully-connected networks if and only if there is a constant* $C > 0$ *such that* $\sum_{l=0}^{C} \|P_{\mathcal{V}_{d,l}} f_d\|^2 \geq d^{-C}$.

The algorithmic result can be achieved by two-layer fully-connected networks, and is a consequence of the analysis of the random feature kernel in [GMMM21], which shows that the projection of f_d onto the low-degree spherical harmonics can be efficiently learned. For the impossibility result, we apply Theorem 3.1, noting that GD is G_{rot}-equivariant by Proposition 2.5, and the G_{rot}-orbit-alignment is:

Lemma 3.7. *Let* $f \in L^2(\mathbb{S}^{d-1})$*. Then* $\mathcal{C}((\mathbb{S}^{d-1}, f); G_{rot}) = \max_{l \in \mathbb{Z}_{\geq 0}} \|P_{\mathcal{V}_{d,l}} f\|^2 / \dim(\mathcal{V}_{d,l})$.

The G_{rot}-orbit-alignment is computed using the representation theory of G_{rot}. In particular, we use the Schur orthogonality theorem, and the fact that for any degree $l \in \mathbb{Z}_{\geq 0}$ the representation $\Phi_l : \mathcal{V}_{d,l} \to \mathcal{V}_{d,l}$ of G_{rot} given by $\Phi_l(g) f = f \circ g$ is irreducible (see, e.g., [Kna96]).

[7] Surprisingly, this means that the full parity function $f_*(\boldsymbol{x}) = \prod_{i=1}^{d} x_i$ can be efficiently learned with such initializations. See Appendix B.

3.2 Application: Extending the merged-staircase property necessity result

In our second application, we study the setting of learning a sparse function on the binary hypercube (a.k.a. a junta) that depends on only $P \leq d$ coordinates of the input $\boldsymbol{x}$, i.e.,

$$f_*(\boldsymbol{x}) = h_*(x_1, \ldots, x_P),$$

where $h_* : \mathcal{H}_P \to \mathbb{R}$. The regime of interest to us is when h_* is fixed and $d \to \infty$, representing a hidden signal in a high-dimensional dataset. This setting was studied by [ABM22], who identified the "merged-staircase property", which we generalize below.

Definition 3.8 (l-MSP)**.** *For $l \in \mathbb{Z}_+$ and $h_* : \mathcal{H}_P \to \mathbb{R}$, we say that h_* satisfies the merged staircase property with leap l (i.e., l-MSP) if its set of nonzero Fourier coefficients $\mathcal{S} = \{S : \hat{h}_*(S) \neq \emptyset\}$ can be ordered as $\mathcal{S} = \{S_1, \ldots, S_m\}$ such that for all $i \in [m]$, $|S_i \setminus \cup_{j<i} S_j| \leq l$.*

For example, $h_*(\boldsymbol{x}) = x_1 + x_1x_2 + x_1x_2x_3$ satisfies 1-MSP; $h_*(\boldsymbol{x}) = x_1x_2 + x_1x_2x_3$ satisfies 2-MSP, but not 1-MSP because of the leap required to learn x_1x_2; similarly $h_*(\boldsymbol{x}) = x_1x_2x_3 + x_4$ satisfies 3-MSP but not 2-MSP. If h_* satisfies l-MSP for some small l, then the function f_* can be learned greedily in an efficient manner, by iteratively discovering the coordinates on which it depends. In [ABM22] it was proved that the 1-MSP property nearly characterized which sparse functions could be ϵ-learned in $O_{\epsilon,h_*}(d)$ samples by one-pass SGD training in the mean-field regime.

In this work, we prove the MSP necessity result for GD training. On the one hand, our necessity result is for a different training algorithm, GD, which injects noise during training. On the other hand, our result is much more general in the sense that it applies to any Lipschitz, permutation-equivariant architecture (whereas the necessity result of [ABM22] applies only to two-layer architectures in the mean-field regime). We also generalize the result to any leap l.

Theorem 3.9 (l-MSP necessity)**.** *Let $f_{\mathsf{NN}}(\cdot; \boldsymbol{\theta}) : \mathcal{H}_d \to \mathbb{R}$ be an architecture and $\mu_{\boldsymbol{\theta}} \in \mathcal{P}(\mathbb{R}^p)$ be an initialization such that GD is G_{perm}-equivariant. Let $L = \sup_{\boldsymbol{\theta},\boldsymbol{x}} \|\nabla_{\boldsymbol{\theta}} f_{\mathsf{NN}}(\boldsymbol{x}; \boldsymbol{\theta})\|$ be the Lipschitz constant, and let $\boldsymbol{\theta}^k$ be the random weights after k steps of GD training with noise parameter $\tau > 0$ and step size η on the distribution $\mathcal{D} = \mathcal{D}(f_*, \mathcal{H}_d)$. Suppose that $f_*(\boldsymbol{x}) = h_*(\boldsymbol{z})$ where $h_* : \mathcal{H}_P \to \mathbb{R}$ does not satisfy l-MSP for some $l \in \mathbb{Z}_+$. Then there are constants $C, \epsilon_0 > 0$ depending on h_* such that $\mathbb{P}_{\boldsymbol{\theta}^k}[\ell_{\mathcal{D}}(\boldsymbol{\theta}^k) \leq \epsilon_0] \leq \frac{C\eta L}{2\tau}\sqrt{\frac{k}{d^{l+1}}} + \frac{C}{d^{l+1}}$.*

The interpretation is that if h_* does not satisfy l-MSP, then to learn f_* to better than ϵ_0 error with constant probability, we need at least $\Omega_{h_*,\epsilon}(d^{l+1})$ steps of GD on a $O_{h_*,\epsilon}(1)$-Lipschitz network with step size $\eta = O_{h_*,\epsilon}(1)$ and noise level $\tau = \Omega_{h_*,\epsilon}(1)$. The proof is deferred to Appendix C. It proceeds by first isolating the "easily-reachable" coordinates $T \subseteq [P]$, and subtracting their contribution from f_*. We then bound G-orbit-alignment of the resulting function, where G is the permutation group on $[d] \setminus T$.

4 Hardness for learning with SGD

In this section, for $\gamma > 0$, we let $\mathcal{D}(f, \mu_{\mathcal{X}}, \gamma) \in \mathcal{P}(\mathcal{X} \times \mathbb{R})$ denote the distribution of $(\boldsymbol{x}, f(\boldsymbol{x}) + \xi)$ where $\boldsymbol{x} \sim \mu_{\mathcal{X}}$ and $\xi \sim \mathcal{N}(0, \gamma^2)$ is independent noise.

We show that the equivariance of SGD implies that the function $f_{\mathrm{mod8}} : \mathcal{H}_d \to \{0, \ldots, 7\}$ given by

$$f_{\mathrm{mod8}}(\boldsymbol{x}) \equiv \sum_i x_i \pmod 8 \tag{4}$$

is hard for SGD-trained, i.i.d. symmetrically-initialized fully-connected networks. Our hardness result is the first of its kind for neural network training, as it relies on a cryptographic assumption to prove superpolynomial lower bounds for SGD learning. We introduce the assumption.

For any $S \subseteq [d]$, let $\chi_S : \mathcal{H}_d \to \{+1, -1\}^d$ be the parity function $\chi_S(\boldsymbol{x}) = \prod_{i \in S} x_i$.

Definition 4.1. *The learning parities with Gaussian noise, (d, n, γ)-LPGN, problem is parametrized by $d, n \in \mathbb{Z}_{>0}$ and $\gamma \in \mathbb{R}_{>0}$. An instance $(S, \boldsymbol{q}, (\boldsymbol{x}_i, y_i)_{i \in [n]})$ consists of (i) an unknown subset $S \subseteq [d]$ of size $|S| = \lfloor d/2 \rfloor$, and (ii) a known query vector $\boldsymbol{q} \sim \{+1, -1\}^d$, and i.i.d. samples $(\boldsymbol{x}_i, y_i)_{i \in [n]} \sim \mathcal{D}(\chi_S, \mathcal{H}_d, \gamma)$. The task is to return $\chi_S(\boldsymbol{q}) \in \{+1, -1\}$.*[8]

[8] More formally, one would express this as a probabilistic promise problem [Ale03].

Our cryptographic assumption is that $\mathrm{poly}(d)$-size circuits cannot succeed on LPGN.

Definition 4.2. *Let $\gamma > 0$. We say γ-LPGN is $\mathrm{poly}(d)$-time solvable if there is a sequence of sample sizes $\{n_d\}_{d\in\mathbb{N}}$ and circuits $\{\mathcal{A}_d\}_{d\in\mathbb{N}}$ such that $n_d, \mathrm{size}(\mathcal{A}_d) \leq \mathrm{poly}(d)$, and $\mathcal{A}_d$ solves (d, n_d, γ)-LPGN with success probability at least $9/10$, when inputs are rounded to $\mathrm{poly}(d)$ bits.*

Assumption 4.3. *Fix γ. The γ-LPGN-hardness assumption is: γ-LPGN is not $\mathrm{poly}(d)$-time solvable.*

The LPGN problem is the simply standard Learning Parities with Noise problem (LPN) [BKW03], except with Gaussian noise instead of binary classification noise, and we are also promised that $|S| = \lfloor d/2 \rfloor$. In Appendix D.3, we derive Assumption 4.3 from the standard hardness of LPN. We now state our SGD hardness result.

Theorem 4.4. *Let $\{f_{\mathsf{NN},d}, \mu_{\boldsymbol{\theta},d}\}_{d\in\mathbb{N}}$ be a family of networks and initializations satisfying Assumption 2.4 (fully-connected) with i.i.d. symmetric initialization. Let $\gamma > 0$, and let $\{n_d\}$ be sample sizes such that $(f_{\mathsf{NN},d}, \mu_{\boldsymbol{\theta},d})$-SGD training on n_d samples from $\mathcal{D}(f_{\mathrm{mod8}}, \mathcal{H}_d, \gamma)$ rounded to $\mathrm{poly}(d)$ bits yields parameters $\boldsymbol{\theta}_d$ with*

$$\mathbb{E}_{\boldsymbol{\theta}_d}[\|f_{\mathrm{mod8}} - f_{\mathsf{NN}}(\cdot; \boldsymbol{\theta}_d)\|^2] \leq 0.0001.$$

Then, under $(\gamma/2)$-LPGN hardness, $(f_{\mathsf{NN},d}, \mu_{\boldsymbol{\theta},d})$-SGD on n_d samples cannot run in $\mathrm{poly}(d)$ time.

In order to prove Theorem 4.4, we use the sign-flip equivariance of gradient descent guaranteed by the symmetry in the initialization. Indeed, a sign-flip equivariant network that learns $f_{\mathrm{mod8}}(\boldsymbol{x})$ from γ-noisy samples, is capable of solving the harder problem of learning $f_{\mathrm{mod8}}(\boldsymbol{x} \odot \boldsymbol{s})$ from γ-noisy samples, where $\boldsymbol{s} \in \mathcal{H}_d$ is an unknown sign-flip vector. However, through an average-case reduction we show that this problem is $(\gamma/2)$-LPGN-hard.

5 Discussion

Our general GD lower bound in Theorem 3.1, and our idea of basing hardness for SGD training on cryptographic assumptions could be further developed to other settings, such as architectures and initializations beyond those considered in this paper.

An outstanding open question is whether computational lower bounds for SGD/GD training can be shown beyond those implied by equivariance. For example, consider the function $f : \mathcal{H}_d \to \{+1, -1\}$ that computes the "full parity", i.e., the parity of all of the inputs $f(\boldsymbol{x}) = \prod_{i=1}^{d} x_i$. Past work has empirically shown that SGD on fully-connected networks with Gaussian initialization [SSSS17, AS20, NY21] fails to learn this function. Proving this would represent a significant advance in our theory for lower bounds, since there is no obvious equivariance that implies that the full parity is hard to learn. Indeed, we have shown here that the full parity can be learned by fully-connected networks with symmetric $\mathrm{Rad}(1/2)$ initialization in which case training is $G_{sign,perm}$-equivariant.

References

[ABM22] E. Abbe, E. Boix-Adsera, and T. Misiakiewicz. The merged-staircase property: a necessary and nearly sufficient condition for sgd learning of sparse functions on two-layer neural networks. *arXiv preprint arXiv:2202.08658*, 2022.

[ACHM22] E. Abbe, E. Cornacchia, J. Hązła, and C. Marquis. An initial alignment between neural network and target is needed for gradient descent to learn. *arXiv preprint arXiv:2202.12846*, 2022.

[AKM+21] E. Abbe, P. Kamath, E. Malach, C. Sandon, and N. Srebro. On the power of differentiable learning versus PAC and SQ learning. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

[Ale03] M. Alekhnovich. More on average case vs approximation complexity. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 298–307. IEEE, 2003.

[AS20] E. Abbe and C. Sandon. Poly-time universality and limitations of deep learning. *arXiv preprint arXiv:2001.02992*, 2020.

[AZL19] Z. Allen-Zhu and Y. Li. What can resnet learn efficiently, going beyond kernels? *Advances in Neural Information Processing Systems*, 32, 2019.

[AZL20] Z. Allen-Zhu and Y. Li. Backward feature correction: How deep learning performs deep learning. *arXiv preprint arXiv:2001.04413*, 2020.

[BAO+20] A. Bogatskiy, B. Anderson, J. Offermann, M. Roussi, D. Miller, and R. Kondor. Lorentz group equivariant neural network for particle physics. In *International Conference on Machine Learning*, pages 992–1002. PMLR, 2020.

[Bar93] A. R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information theory*, 39(3):930–945, 1993.

[BBCV21] M. M. Bronstein, J. Bruna, T. Cohen, and P. Velicković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021.

[BCB14] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[BKW03] A. Blum, A. Kalai, and H. Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *Journal of the ACM (JACM)*, 50(4):506–519, 2003.

[Boi20] E. Boix-Adsera. Average-case statistical query algorithms, 2020. Report for MDS Lab Internship.

[BVB21] A. Bietti, L. Venturi, and J. Bruna. On the sample complexity of learning under geometric stability. *Advances in Neural Information Processing Systems*, 34, 2021.

[BZSL13] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

[CW16] T. Cohen and M. Welling. Group equivariant convolutional networks. In *International conference on machine learning*, pages 2990–2999. PMLR, 2016.

[Ele21] B. Elesedy. Provably strict generalisation benefit for invariance in kernel methods. *Advances in Neural Information Processing Systems*, 34, 2021.

[Ele22] B. Elesedy. Group symmetry in pac learning. In *ICLR 2022 Workshop on Geometrical and Topological Representation Learning*, 2022.

[EZ21] B. Elesedy and S. Zaidi. Provably strict generalisation benefit for equivariant models. In *International Conference on Machine Learning*, pages 2959–2969. PMLR, 2021.

[GMMM21] B. Ghorbani, S. Mei, T. Misiakiewicz, and A. Montanari. Linearized two-layers neural networks in high dimension. *The Annals of Statistics*, 49(2):1029–1054, 2021.

[GMS05] M. Gori, G. Monfardini, and F. Scarselli. A new model for learning in graph domains. In *Proceedings. 2005 IEEE international joint conference on neural networks*, volume 2, pages 729–734, 2005.

[Hoc12] H. Hochstadt. *The functions of mathematical physics*. Courier Corporation, 2012.

[HSSVG21] D. Hsu, C. Sanford, R. A. Servedio, and E.-V. Vlatakis-Gkaragkounis. On the approximation power of two-layer networks of random relus. *arXiv preprint arXiv:2102.02336*, 2021.

[Hsu] D. Hsu. Dimension lower bounds for linear approaches to function approximation.

[HZRS16] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[JGH18] A. Jacot, F. Gabriel, and C. Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

[Kea98] M. Kearns. Efficient noise-tolerant learning from statistical queries. *Journal of the ACM (JACM)*, 45(6):983–1006, 1998.

[KMS20] P. Kamath, O. Montasser, and N. Srebro. Approximate is good enough: Probabilistic variants of dimensional and margin complexity. In *Conference on Learning Theory*, pages 2236–2262. PMLR, 2020.

[Kna96] A. W. Knapp. *Lie groups beyond an introduction*, volume 140. Springer, 1996.

[KSH12] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[LB+95] Y. LeCun, Y. Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[LCC+22] Y. Li, D. Choi, J. Chung, N. Kushman, J. Schrittwieser, R. Leblond, T. Eccles, J. Keeling, F. Gimeno, A. D. Lago, et al. Competition-level code generation with alphacode. *arXiv preprint arXiv:2203.07814*, 2022.

[LKF10] Y. LeCun, K. Kavukcuoglu, and C. Farabet. Convolutional networks and applications in vision. In *Proceedings of 2010 IEEE international symposium on circuits and systems*, pages 253–256. IEEE, 2010.

[LZA21] Z. Li, Y. Zhang, and S. Arora. Why are convolutional nets more sample-efficient than fully-connected nets? In *International Conference on Learning Representations*, 2021.

[Mis22] T. Misiakiewicz. Spectrum of inner-product kernel matrices in the polynomial regime and multiple descent phenomenon in kernel ridge regression. *arXiv preprint arXiv:2204.10425*, 2022.

[MM21] T. Misiakiewicz and S. Mei. Learning with convolution and pooling operations in kernel methods. *arXiv preprint arXiv:2111.08308*, 2021.

[MMM21] S. Mei, T. Misiakiewicz, and A. Montanari. Learning with invariances in random features and kernel models. In *Conference on Learning Theory*, pages 3351–3418. PMLR, 2021.

[MSS20] E. Malach and S. Shalev-Shwartz. Computational separation between convolutional and fully-connected networks. *arXiv preprint arXiv:2010.01369*, 2020.

[Ng04] A. Y. Ng. Feature selection, l 1 vs. l 2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning*, page 78, 2004.

[NY21] I. Nachum and A. Yehudayoff. On symmetry and initialization for neural networks. In *Latin American Symposium on Theoretical Informatics*, pages 401–412. Springer, 2021.

[Rey20] L. Reyzin. Statistical queries and statistical algorithms: Foundations and applications. *arXiv preprint arXiv:2004.00557*, 2020.

[RPK19] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

[SHS+18] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

[SSSS17] S. Shalev-Shwartz, O. Shamir, and S. Shammah. Failures of gradient-based deep learning. In *International Conference on Machine Learning*, pages 3067–3075. PMLR, 2017.

[Val84] L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[VCC+17] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. *stat*, 1050:20, 2017.

[WLLM19] C. Wei, J. D. Lee, Q. Liu, and T. Ma. Regularization matters: Generalization and optimization of neural nets vs their induced kernel. *Advances in Neural Information Processing Systems*, 32, 2019.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]

(c) Did you discuss any potential negative societal impacts of your work? [N/A]

(d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...

   (a) Did you state the full set of assumptions of all theoretical results? [Yes]

   (b) Did you include complete proofs of all theoretical results? [Yes]

3. If you ran experiments...

   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [N/A]

   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [N/A]

   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]

   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [N/A]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

   (a) If your work uses existing assets, did you cite the creators? [N/A]

   (b) Did you mention the license of the assets? [N/A]

   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]

   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]