Why Do You Grok? A Theoretical Analysis of Grokking Modular Addition

Mohamad Amin Mohamadi^{†‡} Zhiyuan Li[†] Lei Wu[§] Danica J. Sutherland[‡]¶

Abstract

We present a theoretical explanation of the "grokking" phenomenon (Power et al., 2022), where a model generalizes long after overfitting, for the originally-studied problem of modular addition. First, we show that early in gradient descent, when the "kernel regime" approximately holds, no permutation-equivariant model can achieve small population error on modular addition unless it sees at least a constant fraction of all possible data points. Eventually, however, models escape the kernel regime. We show that two-layer quadratic networks that achieve zero training loss with bounded ℓ_{∞} norm generalize well with substantially fewer training points, and further show such networks exist and can be found by gradient descent with small ℓ_{∞} regularization. We further provide empirical evidence that these networks as well as simple Transformers, leave the kernel regime only after initially overfitting. Taken together, our results strongly support the case for grokking as a consequence of the transition from kernel-like behavior to limiting behavior of gradient descent on deep networks.

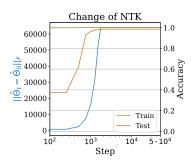
1 Introduction

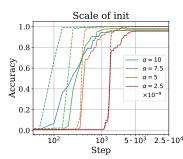
Understanding the generalization patterns of modern over-parameterized neural networks has been a long-standing goal of the deep learning community. Power et al. (2022) demonstrated an intriguing phenomenon they called "grokking" when learning transformers on small algorithmic tasks: neural networks can find a generalizing solution long after overfitting to the training dataset with poor generalization. This observation has lead to a stream of recent works aimed at uncovering the mechanisms that can lead a network to "grok," and properties of the final solutions, on various algorithmic tasks. Later, it was discovered that grokking can happen in tasks beyond modular arithmetic: in learning sparse parities (Barak et al., 2022; Bhattamishra et al., 2023), image classifiers (Liu et al., 2023), greatest common divisors (Charton, 2024), matrix completion (Lyu et al., 2024), and k-sparse linear predictors (Lyu et al., 2024).

Grokking has been variously attributed to difficulty of representation learning (Liu et al., 2022), the "slingshot" mechanism (Thilak et al., 2022), weight norm (Liu et al., 2023; Varma et al., 2023), properties of the loss landscape (Notsawo et al., 2023), simplicity of the learned solution (Nanda et al., 2023) and other feature learning mechanisms (Levi et al., 2024; Rubin et al., 2024). Theoretically, Gromov (2023) presented an analytical construction for a two-layer MLP that solves modular addition is compatible with a grokking pattern. Kumar et al. (2024) demonstrated grokking when training a two-layer MLP on a polynomial regression problem, as did Xu et al. (2024) for XOR data. The notion of delayed generalization was perhaps earlier observed by Li et al. (2022) when training diagonal

[†]Toyota Technological Institute at Chicago [‡]University of British Columbia [§]Peking University [¶]Amii Correspondence to {mohamadamin,zhiyuanli}@ttic.edu,dsuth@cs.ubc.ca.

¹Gromov (2023) claims this solution is the one found by gradient descent, but this did not seem to be the case in our experience.





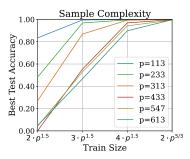


Figure 1: Empirical investigation into grokking modular addition on two-layer networks in the classification task with cross-entropy loss. Left: Change of empirical NTK¹ ($\|\hat{\Theta}_t - \hat{\Theta}_0\|_F$) is negligible before fitting the training data. NTK changes drastically after overfitting, implying that the delayed generalization might be caused by delayed a transitioning from kernel to rich regime. Middle: Reducing initialization scale can mitigate grokking, to the point of completely eliminating the gap between train and test curves. α denotes scale multiplied by θ_0 , the initial weights according to default PyTorch initialization (He et al., 2015). The dashed lines indicate train set statistics, and the solid lines correspond to the test set. Right: Empirical evaluations support a sample complexity of $\tilde{\mathcal{O}}(p^{5/3})$ on the classification task with cross-entropy loss. More details in Section 4.

linear networks with label noise SGD and through sharpness minimization, before it was known as grokking (Power et al., 2022).

Lyu et al. (2024) present a rigorous theoretical framework in which grokking can be provably demonstrated through a dichotomy of early and late implicit biases of the training algorithm. More specifically, they attribute the overfitting stage of grokking to the initial behavior of gradient descent being similar to a kernel predictor (Jacot et al., 2018; Arora et al., 2019b; Lee et al., 2019), and the generalization stage is to different late-phase implicit biases such as sharpness minimization (Blanc et al., 2020; Li et al., 2022; Damian et al., 2021; HaoChen et al., 2020), margin maximization (Soudry et al., 2018; Nacson et al., 2019; Wei et al., 2019; Lyu and Li, 2020), or parameter norm minimization (Gunasekar et al., 2017; Gunasekar et al., 2018; Arora et al., 2019a). This transition from kernel to rich regime has been widely observed (Chizat et al., 2019; Moroshko et al., 2020; Geiger et al., 2020; Telgarsky, 2022; Lyu et al., 2024). Consistent with this framework, Kumar et al. (2024) hypothesized that grokking can be explained through the transition from the kernel regime to the "rich" regime, as long as the size of the training dataset is neither too small (where generalization would be impossible) nor too large (where generalization would be easy). They provided empirical support by considering scaling the model output, which is a rough proxy for the rate of feature learning in modular addition on two-layer MLPs.

This dichotomy between early kernel regime and late feature learning triggered by weak implicit or explicit regularization (i.e. the transition from lazy to rich regime) seems to be the most promising theory to explain grokking. Even so, two fundamental questions as to *why* grokking occurs on the original problem of **modular addition** have remained unanswered:

- 1. Why do a wide variety of architectures all fail to generalize in the initial phase of training, i.e. in the kernel regime? Is it because their kernels accidentally share some common property, or it is indeed a property of the modular addition task itself?
- 2. How does weak regularization encourage the network to learn generalizable features later in

 $^{{}^1\}hat{\Theta}_t$ is the NTK of the model output on the training data using the parameter at step t. When the model output f is a vector, we use the NTK of its first coordinate as an approximation, following (Mohamadi et al., 2023), where $\hat{\Theta}_t \triangleq [\nabla_{\theta} f_1(\theta_t; \mathcal{X}_{\text{train}})][\nabla_{\theta} f_1(\theta_t; \mathcal{X}_{\text{train}})]^{\top}$.

training?

Our Contributions. In this work, we address these questions with rigorous theoretical analyses of learning modular addition with gradient descent on two-layer MLPs.

We specifically focus on the problem

$$a + b \equiv c \pmod{p} \tag{1.1}$$

where $a,b,c\in\mathbb{Z}_p$ for a fixed $p\in\mathbb{N}$. We use (regularized) gradient descent on a two-layer MLP with quadratic activations, the same architecture as Gromov (2023). We consider two different tasks: it is somewhat easier to analyze a "regression" task where we use square loss to learn a function of (a,b,c) that indicates whether (1.1) holds, but we also study the "classification" task in which we use cross-entropy loss to learn a p-way classifier to predict which c value satisfies (1.1) for a given (a,b). This discrete, noiseless setting has only a finite number of possible distinct data points: p^3 for regression, p^2 for classification.

To address the first question, we prove in Sections 3.1 and 4.1 that this task is *fundamentally hard* for permutation-equivariant kernel methods, due to inherent symmetries. Thus, permutation-equivariant networks which are well-approximated by their neural tangent kernel approximation cannot generalize well. We also prove that, although no such method can generalize, neural tangent kernel approaches based on our architecture with realistic widths *can* achieve zero training error. This result is highly suggestive of why drastic overfitting with poor generalization has been empirically observed on this problem across a wide variety of architectures, losses, and learning algorithms (e.g. Power et al., 2022; Liu et al., 2022; Gromov, 2023).

To prove the generalization lower bounds, we develop a novel general technique to analyze the population ℓ_2 loss of learning general function classes with predictors of intrinsic dimension n (for instance kernel predictors with n training points), presented in Appendix D.4. This framework allows us to prove lower bounds on population ℓ_2 loss for the general case of learning modular addition on m summands (rather than 2 or 3) with kernels, and might be of independent interest.

To address the second question – why this occurs – we identify ℓ_{∞} norm as an effective and practically-relevant complexity measure for generalization, which is closely related to the implicit bias of Adam (Xie and Li, 2024; Zhang et al., 2024), and show that networks with small ℓ_{∞} norm in the regression setting (Section 3.2) and large ℓ_{∞} -normalized margin in the classification setting (Section 4.2) can both provably generalize with far fewer samples than required in the kernel regime. Thus, models with corresponding implicit biases can generalize well, answering the second question. In regression, our proofs are based on "optimistic rates" (Srebro et al., 2010) for the smooth ℓ_2 loss in terms of the Rademacher complexity of networks with bounded ℓ_{∞} parameter norm. In classification, our proof applies the PAC-Bayesian framework of McAllester (2003) to networks with bounded ℓ_{∞} parameter norm.

In summary, our **main contributions** are as follows:

- 1. We prove that networks in the kernel regime can only generalize if trained on at least a constant fraction of all possible data points, i.e. an $\Omega(1)$ portion, for regression (Section 3.1) and classification (Section 4.1).
- 2. We prove that networks with appropriate regularization can generalize with many fewer samples: $\mathcal{O}(1/p)$ portion of all possible data points for square loss generalization on the regression task (Section 3.2), and $\tilde{\mathcal{O}}(1/p^{1/3})$ for zero-one loss generalization on classification (Section 4.2).

2 Preliminaries and Setup

Notations. We use [p] to denote the set $\{1,\ldots,p\}$. We use e_i to denote the i standard basis vector, i.e., the vector with 1 in its ith component and 0 elsewhere. For vector a, we use $a^{\odot 2}$ denotes the element-wise square of the vector a For any nonempty set \mathcal{X} , a symmetric function $K: \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is called a positive semi-definite kernel (p.s.d.) kernel on \mathcal{X} if for all $n \in \mathbb{N}$, all $x_1, \ldots, x_n \in \mathcal{X}$, and all $\lambda_1, \ldots, \lambda_n \in \mathbb{R}$, it holds that $\sum_{i=1}^n \sum_{j=1}^n \lambda_i \lambda_j K(x_i, x_j) \geq 0$. Given two non-negative functions f, g, we say that f(n) = O(g(n)) (resp. $f(n) = \Omega(g(n))$) iff. there exists absolute constant C > 0, such that for all $n \geq 0$, $f(n) \leq Cg(n)$ (resp. $f(n) \geq Cg(n)$). We use $f(n) = \omega(g(n))$ to denote that for all C > 0, there exists $n_0 > 0$ such that for all $n \geq n_0$, f(n) > Cg(n).

Setup. We focus on learning modular addition, (1.1), with a two-layer network with no biases and quadratic activation, following Gromov (2023). More specifically, given parameters $\theta = (W, V)$, the model f maps the pair of integers (a, b) – represented as the vector $(e_a, e_b) \in \mathbb{R}^{2p}$ – to a vector in \mathbb{R}^p . We use the form

$$f(\theta; (e_a, e_b)) = V(W(e_a, e_b))^{\odot 2},$$
 (2.1)

where $W \in \mathbb{R}^{h \times 2p}$, $V \in \mathbb{R}^{p \times h}$ for some integer h. We call h the width of the hidden layer and it will be set later.

In the classification setting (Section 4), we train f with cross-entropy loss to identify the c such that $a+b\equiv c\pmod p$ in a multi-classification setting. Letting $\mathcal{Z}=\{e_i:i\in[p]\}$, the set of all possible inputs is $\mathcal{X}=\mathcal{Z}\times\mathcal{Z}$ and outputs is $\mathcal{Y}=\mathcal{Z}$; there are $N=p^2$ distinct data points.

In the regression setting (Section 3), we instead aim to learn for each tuple (a,b,c), if $a+b\equiv c\pmod p$, that is, map $\boldsymbol x=(e_a,e_b,e_c)$ to the scalar $y=p\,\mathbf 1(a+b\equiv c\pmod p)$ using $g(\theta;(e_a,e_b,e_c)))\triangleq e_c^{\top}f(\theta;(e_a,e_b))$. Here $\mathcal X=\mathcal Z^3,\,\mathcal Y=\{0,p\}$, and $N=p^3$; we train the model $g(\theta;\boldsymbol x)=\langle e_c,f(\theta;(e_a,e_b))\rangle$ to minimize the square loss.

In either setting, we use \mathcal{D} to denote the distribution over $\mathcal{X} \times \mathcal{Y}$ which is uniform over all N possible input-output pairs, while $\mathcal{D}_{\text{train}} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ is the training dataset of size n. Given parameters θ , we use $\Psi(\theta;\cdot): \mathcal{X} \to \hat{\mathcal{Y}}$ to denote the predictor $(g(\theta;\cdot) \text{ for regression}, f(\theta;\cdot) \text{ for classification})$, we train the model with gradient descent with tiny ℓ_{∞} regularization,³ which is intended to emulate the implicit bias of Adam (AdamW) which might lead to grokking. In general, we define loss as

$$\mathcal{L}_{\ell}^{\lambda}(\Psi, \theta, \mathcal{D}_{\text{train}}) \triangleq \frac{1}{n} \sum_{(\boldsymbol{x}, y) \in \mathcal{D}_{\text{train}}} \ell(\Psi(\theta; \boldsymbol{x}), y) + \lambda \|\theta\|_{\infty},$$
 (2.2)

where $\ell: \hat{\mathcal{Y}} \times \mathcal{Y} \to \mathbb{R}_{\geq 0}$ denotes either square (ℓ_2) or cross-entropy loss and $\lambda \geq 0$ controls the strength of regularization. Omitting λ refers to the case where no regularization is used and omitting \mathcal{D}_{train} means the whole population is used in evaluating the loss. For non-parametric functions (like Ψ_0 denoted as the predictor which returns zero on any input) we drop θ as well.

Connection between ℓ_∞ -norm Regularization and Grokking. Power et al. (2022) shows that grokking phenomenon happens for Adam and gets more prominent with decoupled weight decay (AdamW). Recent works by Xie and Li (2024) and Zhang et al. (2024) show that Adam implicitly regularizes the ℓ_∞ norm of the weights. In detail, Xie and Li (2024) shows that AdamW can only converge to KKT points of ℓ_∞ -norm constrained optimization. Zhang et al. (2024) shows that Adam converges to max-margin solutions w.r.t. ℓ_∞ norm for linear models on separable datasets and

This scaling implies that the predictor $\Psi_0(\cdot)=0$ has population square loss p; this scaling allows bounded $\|\theta\|_{\infty}$.

³More details about the regularization strength can be found in Appendix A.

conjecture this result could be generalized to general homogeneous models (including our model f, which is 3-homogeneous). This connection motivates us to use the explicit ℓ_{∞} -norm regularization on top of gradient descent to understand grokking in the modular addition setting.

Definition 2.1. A deterministic supervised *learning algorithm* \mathcal{A} is a mapping from a sequence of training data, $\mathcal{D}_{\text{train}} \in (\mathcal{X} \times \mathcal{Y})^n$, to a hypothesis $\mathcal{A}(\mathcal{D}_{\text{train}}) : \mathcal{X} \to \hat{\mathcal{Y}}$. The algorithm \mathcal{A} could also be randomized, in which case the output $\mathcal{A}(\mathcal{D}_{\text{train}})$ is a distribution over hypotheses. Two randomized algorithms \mathcal{A} and \mathcal{A}' are the same if for any input, their outputs have the same distribution in function space, written as $\mathcal{A}(\mathcal{D}_{\text{train}}) \stackrel{d}{=} \mathcal{A}'(\mathcal{D}_{\text{train}})$.

We further define equivariance of learning algorithms which is the main component of our analysis in deriving lower bounds. This definition is used in Theorem 3.3 and Proposition 4.2 to prove equivariance of GD in learning modular addition.

Definition 2.2 (Equivariant Algorithms). A learning algorithm is *equivariant* under group $\mathcal{G}_{\mathcal{X}}$ (or $\mathcal{G}_{\mathcal{X}}$ -equivariant) if for any dataset $\mathcal{D}_{\text{train}} \in (\mathcal{X} \times \mathcal{Y})^n$ and for all $T \in \mathcal{G}_{\mathcal{X}}$, it holds that $\mathcal{A}(\{(T(\boldsymbol{x}_i),y_i)\}_{i=1}^n) \circ T \stackrel{d}{=} \mathcal{A}(\{(\boldsymbol{x}_i,y_i)\}_{i=1}^n)$. For deterministic learning algorithms, this is equivalent to saying that for all $\boldsymbol{x} \in \mathcal{X}$, $\mathcal{A}(\{(T(\boldsymbol{x}_i),y_i)\}_{i=1}^n)(T(\boldsymbol{x})) = [\mathcal{A}(\{(\boldsymbol{x}_i,y_i)\}_{i=1}^n)](\boldsymbol{x})$.

Definition 2.3 (Kernel Methods). For $\hat{\mathcal{Y}} \subseteq \mathbb{R}$, we say a learning algorithm \mathcal{A} is a *kernel method* if it first picks a (potentially random) positive semi-definite kernel K on \mathcal{X} before seeing the data,⁴ and then outputs some hypothesis Ψ such that there exist $\{\lambda_i\}_{i=1}^n \in \mathbb{R}$ for which $\Psi(\cdot) = \sum_{i=1}^n K(\cdot, \boldsymbol{x}_i)\lambda_i$, where λ_i can depend on the training dataset $\mathcal{D}_{\text{train}}$.

In particular, when $\lambda(\mathcal{D}_{\text{train}}) = \mathbf{K}^{\dagger}\mathbf{y}$, where $\mathbf{K} = (K(\mathbf{x}_i, \mathbf{x}_j))_{i,j=1}^n$, $\mathbf{y} = (y_i)_{i=1}^n$, the kernel method is called a *(ridgeless) kernel regression* method.

Theorem 2.4. For any p.s.d. kernel $K: \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ and transformation group \mathcal{G}_gX , kernel regression (Definition 2.3) with respect to kernel K is $\mathcal{G}_{\mathcal{X}}$ -equivariant if and only if kernel K is equivariant to \mathcal{G}_gX , i.e., K(T(x), T(x')) = K(x, x') for any $T \in \mathcal{G}_gX$ and $x, x' \in \mathcal{X}$.

Proof of Theorem 2.4. For any $\mathcal{D}_{\text{train}} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ and transformation $T \in \mathcal{G}_{\mathcal{X}}$, let $\mathcal{D}_{\text{train}}^T$ be the transformed dataset $\{(T(\boldsymbol{x}_i), y_i)\}_{i=1}^n$, and \mathcal{A}_K be the kernel regression algorithm w.r.t. K. For any \boldsymbol{x} , we have that

$$\mathcal{A}(T(\mathcal{D}_{\text{train}}))(T(\boldsymbol{x})) = \sum_{i=1}^{n} K(T(\boldsymbol{x}), T(\boldsymbol{x}_i)) \lambda_i(\mathcal{D}_{\text{train}}^T) = \sum_{i=1}^{n} K(\boldsymbol{x}, \boldsymbol{x}_i) \lambda_i(\mathcal{D}_{\text{train}}) = \mathcal{A}(\mathcal{D}_{\text{train}})(\boldsymbol{x}),$$

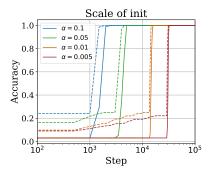
where the second equality follows from the equivariance of the kernel K w.r.t. $\mathcal{G}_{\mathcal{X}}$.

3 Regression Task

We will first present our theoretical analysis of the grokking phenomenon on modular arithmetic with two-layer quadratic networks in the regression setting, where we learn a function from (e_a, e_b, e_c) to $p\mathbf{1}(a+b=c\pmod{p})$. Although this is perhaps a less natural way to model modular arithmetic than the classification task, it admits some useful theoretical tools.

In this section, we show that networks in the kernel regime will provably fail to generalize as long as they do not have access to nearly all the points in the dataset (Theorem 3.4), although they can achieve zero training error (Theorem 3.1). However, the network eventually leave the kernel regime

⁴Our definition of kernel methods does not cover learning algorithms that choose the kernel based on the training data. *Any* learning algorithm could be framed as a kernel method with a data-dependent kernel $K(x, x') = \Psi(x)\Psi(x')$.



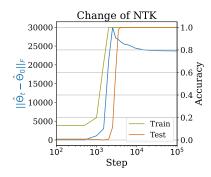


Figure 2: Empirical evidence for kernel regime in early training. **Left:** Train (dashed) and test (solid) accuracy while training in the regression setting with various initialization scales α and a fixed p=47. **Shrinking the scale of initialization can mitigate grokking** in the regression task, eventually eliminating of the gap between train and test accuracies (at the cost of slower improvement in each). **Right:** eNTK continues to significantly change after overfitting.

because of the weak ℓ_{∞} norm regularization. We further establish that if the network manages to achieve zero training error with small $\|\theta\|_{\infty}$, it will generalize with only a $\frac{n}{N} = \tilde{\omega}(1/p)$ portion of the overall dataset, as long as the width of the network is larger than 4p (Theorem 3.6). We also prove that such networks exist (Proposition 3.8), and demonstrate that gradient descent can find them with a small amount of explicit regularization (Figure 3).

3.1 Kernel Regime

A recent line of work on the *neural tangent kernel* (NTK) framework (Jacot et al., 2018; Arora et al., 2019b; Lee et al., 2019; Chizat et al., 2019; Yang and Hu, 2021) has shown that with typical initialization schemes, gradient descent in over-parameterized neural networks locally approximates the behavior of a kernel model using the empirical neural tangent kernel (eNTK) $K_{\theta}(x, x') \triangleq \nabla g(\theta; x) \nabla g(\theta; x')^{\mathsf{T}}$. In the "kernel regime," the change in θ over the course of gradient descent does not substantially change the eNTK K_{θ} , and hence the neural network behaves similarly to a kernel predictor trained with K_{θ_0} . With square loss, as here, these kernel predictors follow a particularly simple optimization path for which a closed form (corresponding to kernel regression) is available.

For networks of finite width (and in certain infinite-width cases), the eNTK will stay roughly constant and the network will closely track the kernel model for the first phase of optimization, but the tiny regularization will eventually lead it to depart the kernel regime. Thus, bounds on kernel models are informative about deep networks in the first part of optimization.

In the first phase of grokking, the model overfits to the training data, achieving very low training loss but retaining high loss on test points. We establish that both phenomena occur in the kernel regime.

3.1.1 Empirical Neural Tangent Kernels Can Achieve Zero Training Error

Our first result shows that kernel regression with the empirical neural tangent kernel of our quadratic network achieves zero training loss if the network is mildly wide, for instance, $n = \tilde{\Theta}(p^2)$ and width $h = \tilde{\Theta}(p)$. This implies, for example, that in a "lazy training" setting (Chizat et al., 2019), or alternatively when the first layer is initialized with huge weights and the second layer with tiny ones, gradient descent can achieve zero training loss. Informally, this also strongly suggests that networks

with more typical initializations can achieve very small training error without needing to leave the "kernel regime."

Theorem 3.1. Initialize the network of Section 2 with any values for V, and entries of W all independently $\operatorname{Uniform}([-s_W,s_W])$ for some $s_W>0$. Let the most frequent value of $c\in[p]$ appear ρ_c times. Then, if $h>36\rho_c\log(n/\delta)$, it holds with probability at least $1-\delta$ over the random initialization of W that kernel regression using the network's empirical neural tangent kernel can achieve zero training loss for any target labels such that if $x_i=x_j$ then $y_i=y_j$. Conversely, if h< n/(3p), there exist target labels (with $y_i=y_j$ when $x_i=x_j$) for which this method cannot achieve zero training loss.

Uniform initialization is the default behavior e.g. in PyTorch (Ansel et al., 2024). Note that ρ_c is always between n/p and n. Moreover, in the usual setting with randomly selected $\mathcal{D}_{\text{train}}$ and $n = \Omega(p \log p)$ – recall that full knowledge of a single (a,b) pair requires n=p in the regression setting – we have that $\rho_c = \Theta(n/p)$ with high probability (Raab and Steger, 1998, Theorem 1). Thus in this usual setting, the threshold for interpolation is at a network width $h = \tilde{\Theta}(n/p)$.

Proof sketch of Theorem 3.1. Kernel regression can achieve all possible target labels iff its kernel matrix is strictly positive definite (Lemmas B.1 and B.2). Evaluating the expected neural tangent kernel, we can see it is strictly positive definite under only mild assumptions on the weights (Proposition B.3). With bounded weights, a matrix Chernoff bound controls the convergence of the lowest eigenvalue of the kernel matrix to that of the expected kernel matrix (Proposition B.4). The lower bound follows from the rank of the kernel matrix (Proposition B.6). □

3.1.2 Permutation-Equivariant Kernel Methods Cannot Generalize

We now show that for any permutation-equivariant kernel method (Definition 2.3) (and hence for networks trained by gradient descent close enough to initialization), generalization is possible only when training on $n = \Omega(p^3)$ samples, i.e. the portion of all possible data points is $\frac{n}{N} = \Omega(1)$.

The key component of our analysis is the permutation equivariance of learning modular addition in this setting. We first define the permutation group on the modular addition data:

Definition 3.2. Let \mathbb{S}_p denote the set of all permutations on [p]. We define the *permutation group* $\mathcal{G}_{\mathcal{X}}$ on \mathcal{X} as the group

$$\left\{ (e_a, e_b, e_c) \mapsto (e_{\sigma_1(a)}, e_{\sigma_2(b)}, e_{\sigma_3(c)}) : \sigma_1, \sigma_2, \sigma_3 \in \mathbb{S}_p \right\},\,$$

with the group operation being composition of each permutation.

The following theorem establishes that our learning algorithm is equivariant (Definition 2.2) under this permutation group, which we call *permutation-equivariant* for short. Note that this result applies to the actual process of gradient descent on our neural network, not only to its NTK approximation. (This result is not particularly specific to the architecture defined in Section 2; it holds broadly.)

Theorem 3.3 (Permuation Equivariance in Regression). Training a two-layer neural network with quadratic activations (such as $g(\theta;x)$) initialized with random i.i.d. parameters using gradient-based update rules (such as gradient descent) on the modular addition problem in the regression setting is an equivariant learning algorithm (as defined in Definition 2.2) with respect to the permutation group of Definition 3.2 applied on the input data.

More details and the full proof are in Appendix C.

Roughly speaking, this indicates that learning the modular addition task on this setup is exactly the same difficulty as learning any permuted version of the dataset. Following the same argument, we can show that the kernel method corresponding to the neural nets in the early phase of training is also permutation-equivariant.

Further, as the distribution \mathcal{D} is uniform and thus invariant under permutation, Theorem 3.3 shows that the difficulty of learning the original ground-truth is the same as simultaneously learning all the permuted versions of the ground-truths, which turns out to be difficult for any kernel method. The following theorem formalizes this idea, establishing that any such kernel predictor needs at least $n = \Omega(p^3)$, or equivalently $\frac{n}{N} = \Omega(1)$, to generalize on the modular addition task; otherwise it cannot do substantially better than the trivial all-zero predictor.

Theorem 3.4 (Lower Bound). There exists a constant C > 0 such that for any $p \ge 2$, training data size $n < Cp^3$, and any permutation-equivariant kernel method A, it holds that

$$\mathbb{E}_{\left(\boldsymbol{x}_{i},y_{i}\right)_{i=1}^{n}\sim\mathcal{D}^{n}}\mathbb{E}\,\mathcal{L}_{\ell_{2}}\left(\mathcal{A}\left(\left\{\left(\boldsymbol{x}_{i},y_{i}\right)\right\}_{i=1}^{n}\right)\right)\geq\frac{w}{2}\,\,\mathcal{L}_{\ell_{2}}\left(\Psi_{\mathbf{0}}\right)=\frac{p}{2},$$

where $\mathbb{E}_{\mathcal{A}}$ takes the mean over the randomness in algorithm \mathcal{A} .

A full proof of this result is deferred to Appendix D.

Theorem 3.4 is in fact a special case of a more general theorem that lower bounds the population ℓ_2 loss of learning modular addition with m-summands using permutation-equivariant kernels, showing that poor generalization is inevitable. We present an informal version of this result below and refer the reader to Corollary D.8 for the formal version.

Theorem 3.5 (Informal). Consider the problem of learning modular addition over m-summands with a training set of size n using a permutation equivariant kernel method A. For any $p \geq 2$ and training data size $n < Cp^m$ the expected population ℓ_2 loss is at least $\Omega(p)$, which is of the same magnitude as the trivial all-zero predictor.

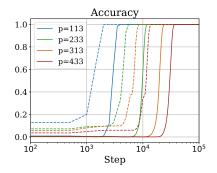
Poor population loss is thus inevitable for models which are well-approximated by a permutationequivariant kernel; since our quadratic network can also achieve zero training error in the kernel regime, this strongly suggests drastic overfitting in early training. Fortunately, however, regularized gradient descent on finite networks will eventually leave the kernel regime and begin learning features.

3.2 Rich Regime

While the transient behavior of gradient descent after leaving the kernel regime may be complicated, it is often the case that the limiting behavior can be better-understood based on analyses of implicit bias. Motivated by this, we present a generalization analysis of networks that achieve zero training error (as we expect in the long-term optimization limit) with bounded $\|\theta\|_{\infty}$. This assumption is motivated by recent work of Xie and Li (2024), who show that full-batch AdamW can only converge to KKT points of the ℓ_{∞} norm-constrained optimization problem, $\min_{\|\theta\|_{\infty} \leq \frac{1}{\lambda}} \mathcal{L}(\Psi, \theta, \mathcal{D}_{train})$, where λ is the weight decay coefficient. We will discuss the feasibility of this assumption more afterwards.

Theorem 3.6 (Upper Bound). For any width $h \geq 8p$, with probability at least $1 - \delta$ over the randomness of training dataset \mathcal{D}_{train} of size n, define the set of interpolating solutions as $\tilde{\Theta}^* = \{\theta \mid \mathcal{L}_{\ell_2}(\Psi, \theta, \mathcal{D}_{train}) = 0\}$. For any interpolating solution $\theta^* \in \tilde{\Theta}^*$ with small ℓ_{∞} norm, i.e., satisfying that $\|\theta^*\|_{\infty} = \mathcal{O}(\min_{\theta \in \tilde{\Theta}^*} \|\theta\|_{\infty})$, it holds that

$$\mathcal{L}_{\ell_2}(g, \theta^*) = \mathcal{O}\left(\frac{p^2}{n} \left(\log^3 n + \frac{1}{p} \log \frac{1}{\delta}\right)\right) \cdot \mathcal{L}_{\ell_2}\left(\Psi_{\mathbf{0}}\right).$$



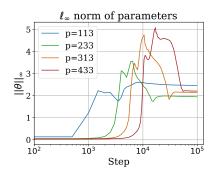


Figure 3: Empirical verification of our theoretical explanation for generalization. We train a network of width h=4p with gradient descent on ℓ_2 loss and 10^{-4} ℓ_∞ -regularization on $2\times p^{2.25}$ training samples (out of p^3) in the regression setting. Left: Generalization happens when the number of samples are more than $\Omega(p^2)$, as predicted by Proposition 3.7. The dashed and solid lines indicate train and test set statistics respectively. Right: ℓ_∞ norm of parameters after grokking remains the same for different problem dimensions p, as predicted by

Comparing Theorem 3.6 to Theorem 3.4, we see that when $n=\tilde{\omega}(p^2)$ and $h=\Omega(p)$, there is a strict separation between generalization in the kernel and rich regimes. Note that the classifier $\phi(e_a,e_b) \triangleq \arg\max_c f_c(\theta^*;(e_a,e_b))$ has population error rate at most $\frac{2}{p}\mathcal{L}_{\ell_2}(\phi)$ (as shown in Proposition E.7) and thus, its population error goes to zero when $n=\tilde{\omega}(p^2)$. The proof of Theorem 3.6 consists of showing all networks with small training error and small ℓ_∞ norms generalize (Proposition 3.7), and at least one such network exists (Proposition 3.8).

Proposition 3.7. For any $R > 0, \delta \in (0,1)$, \mathcal{D}_{train} of size n, and $\theta^* \in \{\theta = (W,V) : \mathcal{L}_{\ell_2}(g,\theta,\mathcal{D}_{train}) = 0 \wedge \|\theta\|_{\infty} \leq R\}$, there exists a positive constant C > 0 such that with probability at least $1 - \delta$ over the randomness of \mathcal{D}_{train} ,

$$\mathcal{L}_{\ell_2}(g, \theta^*) \le \frac{CR^6h^2}{n} \left(p\log^3 n + \log\frac{1}{\delta}\right).$$

Proof sketch of Proposition 3.7. We bound the Rademacher complexity of the set of networks with small ℓ_{∞} weights, and then apply Theorem E.6, due to Srebro et al. (2010), which gives an "optimistic" bound on the excess risk of smooth loss functions. Details in Appendix E.

Proposition 3.8. Let the set of models with zero population loss be $\Theta^* \triangleq \{\theta \mid \mathcal{L}_{\ell_2}(g,\theta) = 0\}$. For any $p \geq 2$ and $h \geq 8p$, Θ^* is nonempty and $\min_{\theta \in \Theta^*} \|\theta\|_{\infty} \leq \left\lfloor \frac{h}{8p} \right\rfloor^{-\frac{1}{3}}$.

Proof sketch of Proposition 3.8. The main difficulty is a manual Fourier-based construction of a zero-loss solution with h=8p and ℓ_{∞} norm at most one; this is inspired by similar constructions of Gromov (2023) and results of Nanda et al. (2023). In a concurrent work, Morwani et al. (2024) presented a similar construction. Once we have that, because our model is 3-homogeneous in its parameters, we can easily reduce the ℓ_{∞} norm by duplicating neurons, without changing the input-output function. Details in Appendix F.

We know that an interpolating solution with small ℓ_{∞} norm exist, and that any such solution will generalize. Does gradient descent find such a solution? In Figure 3, we empirically evaluate the ℓ_{∞} norm of weights learned by running gradient descent on the regression task, with varying p, and a

very small ℓ_{∞} regularizer.⁵ Consistent with our manual construction of weights, the ℓ_{∞} norm of the network weights does not grow with the problem dimension p. This supports the applicability of Theorem 3.6 and a far better sample complexity compared to the kernel regime.

Empirical Evidence that Kernel Lower Bounds Lead to Overfitting: One way to mitigate the grokking effect in learning modular addition is by preventing the network from overfitting the training set in the kernel regime. To do so, one can reduce the *scale of initialization* (e.g. smaller variance scales for weight initializations in the scheme of He et al. (2015)). Roughly speaking, networks initialized with a very small scale of parameters need to undergo a significant growth in the norm of parameters to be able to fit the training data. This norm growth will eventually lead the network to leave the kernel regime and begin learning features before overifting the data. Figure 2 confirms that in our setting: using a very small weight initialization can substantially mitigate the grokking effect by preventing the network from overfitting in the kernel regime, incidacting that grokking is indeed caused by a delayed transition from kernel (overiftting) to rich (generalizing) regime. However, this comes at a cost: training becomes increasingly more difficult as the scale of initialization decreases (Chizat et al., 2019; Moroshko et al., 2020; Telgarsky, 2022).

In Figure 2, we also evaluate the change of the empirical NTK during training, by computing the difference in eNTK matrices through training. To make this empirical investigation computationally feasible, we evaluated the eNTK approximation of Mohamadi et al. (2023) on 20,000 random data points, similar to previous schemes (Fort et al., 2020; Wei et al., 2019; Mohamadi et al., 2023). We see that the change in empirical NTK is orders of magnitude larger after overfitting the training set, implying that most feature learning happens only later, supporting our hypothesis that the initial overfitting occurs roughly in the kernel regime.

4 Classification Task

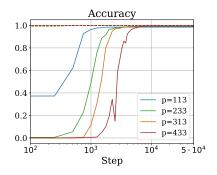
We now move onto the multi-class classification setting established in Section 2, where we train with cross-entropy loss. Similar to the regression problem, we first analyze the early stage of training where the network operates like a kernel machine, and then move onto the rich regime and focus on the weights learned through margin-maximization implicit bias of gradient descent. We prove that a sample complexity gap between kernel (Theorem 4.3) and rich (Theorem 4.6) regime exists when learning a two-layer neural network with gradient descent on modular addition modeled as a multi-class classification task. Our results imply that as long as the max-margin implicit bias of gradient descent on exponential-type loss functions drives the net to leave the kernel regime and $\hat{\Omega}(p^{5/3})$ samples are used for training, generalization to the whole population is guaranteed.

4.1 Kernel Regime

In the multi-class setting, the output is p-dimensional, and thus the eNTK for each pair of points is a $p \times p$ psd matrix (see Álvarez et al., 2012). Our notion of kernel methods (Definition 4.1) slightly changes due to account for multi-output functions, and the main lower bound result is similar to that in the regression case: kernel methods must see a constant fraction of data before learning.

Definition 4.1. For $\mathcal{Y} \subseteq \mathbb{R}^p$, we say a learning algorithm \mathcal{A} is a *kernel method* if before seeing the data it first picks a (potentially random) kernel K on \mathcal{X} such that for any $x, x' \in \mathcal{X}$, K(x, x') is a p

 $^{^{5}}$ We used a regularization weight of 10^{-4} . Without explicit regularization, a small number of network weights do grow with p, but we believe this phenomenon is not important to the overall behavior of gradient descent on this task.



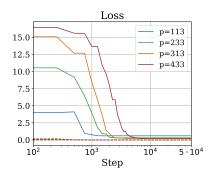


Figure 4: Empirical investigation of grokking in the classification setting with tiny ℓ_{∞} regularization on different problem dimensions p. Networks are trained with normalized gradient descent⁸ on cross-entropy loss and an ℓ_{∞} regularization of 10^{-20} strength. The dashed lines in the indicate train set statistics, and the solid lines correspond to the test set.

by p positive semi-definite matrix, and then outputs some hypothesis h based on $\mathcal{D}_{\text{train}}$ such that there exist $\{\lambda_i\}_{i=1}^n \in \mathbb{R}^p$ for which $h(\cdot) = \sum_{i=1}^n K(\cdot, \boldsymbol{x}_i) \lambda_i$.

We next establish the permutation-equivariance of gradient-based learning algorithms in the classification setting. They key difference is that in the multi-output setting, gradient-based learning algorithms are equivariant to permutations on *both input data and output labels*. More details are available in Appendix C.

Proposition 4.2 (Permuation Equivariance in Classification). We define the permutation group $\mathcal{G}_{\mathcal{X},\mathcal{Y}}$ on $\mathcal{X} \times \mathcal{Y}$ (defined in Section 2) as the group

$$\{(e_a, e_b), e_c \mapsto (e_{\sigma_1(a)}, e_{\sigma_2(b)}), e_{\sigma_3(c)} : \sigma_1, \sigma_2, \sigma_3 \in \mathbb{S}_p\}.$$

Training a two-layer neural network with quadratic activations (such as $f(\theta;x)$) initialized with random i.i.d. parameters using gradient-based update rules (as defined in Definition C.1)⁶ on the modular addition problem in the classification setting is an equivariant learning algorithm (as defined in Definition 2.2) with respect to $\mathcal{G}_{X,Y}$.

We say a learning algorithm is input-output permutation-equivariant it is equivariant with respect to $\mathcal{G}_{\mathcal{X},\mathcal{Y}}$. The following theorem establishes that any such kernel learning algorithm needs at least $n=\Omega(p^2)$, or equivalently $\frac{n}{N}=\Omega(1)$, to generalize on the modular addition task in the classification setting.

Theorem 4.3 (Kernel Lower Bound). There exists a constant C > 0 such that for any training data size $n < Cp^2$ and any kernel method \mathcal{A} which is input-output permutation-equivariant (with respect to $\mathcal{G}_{\mathcal{X},\mathcal{Y}}$), it holds that

$$\mathop{\mathbb{E}}_{\left(\boldsymbol{x}_{i},y_{i}\right)_{i=1}^{n}\sim\mathcal{D}^{n}}\mathop{\mathbb{E}}_{\mathcal{A}}\mathcal{L}_{\ell_{2}}\left(\mathcal{A}\left(\left\{\boldsymbol{x}_{i},y_{i}\right\}_{i=1}^{n}\right)\right)\geq\frac{1}{2}\;\mathcal{L}_{\ell_{2}}\left(\Psi_{\boldsymbol{0}}\right),$$

where $\mathbb{E}_{\mathcal{A}}$ takes expectation over the randomness in algorithm \mathcal{A} .

Intuitively, the classification setting is not very different from the regression setting. They share the same goal and the amount of knowledge contained in each data in the classification setting is exactly equal to p data in the regression setting, where these p data share the same first two coordinates and only differ in the last coordinate.

⁶GD is an example of such algorithm.

⁷A similar result holds for adaptive algorithms like Adam as well, as discussed in Corollary C.7.

⁸We refer the reader to Appendix A for the definition of normalized GD.

More formally, we can define the following correspondence: given one data point in classification $(x,y) \in \mathbb{R}^{2p} \times \mathbb{R}^p$, we can view it as p data points in regression, $\{((x,e_i),y_i)\}_{i=1}^p$, denoted by F(x,y). Similarly, any function Ψ mapping from $\mathbf{x} = \{(e_i,e_j)\}$ to \mathbb{R}^p can be viewed as a function mapping $\{(e_i,e_j,e_k)\}$ to \mathbb{R} by defining $\Psi'(\mathbf{x},e_k) = [\Psi(\mathbf{x})]_k$. Moreover, these two functions Ψ and Ψ' share the same population ℓ_2 loss. Under this view, matrix-valued kernel learning with n classification data points is exactly the same as scalar-valued kernel learning with np regression data points.

The only obstacle to directly applying Theorem 3.4 is that the data distribution is different. For the regression setting, each data is sampled independently and uniformly, and the number of data points can be any integer; in the classification setting, the data points are sampled in independent groups and the number of data must be a multiple of p. However, it is easy to see this new distribution is still invariant under permutations as in Definition 3.2. Thus, we can directly apply Theorem 3.4 on this new distribution to get Theorem 4.3. A formal version of this argument is available in Appendix D.3.

Therefore, for networks operating in the kernel regime, generalization to unseen data in ℓ_2 loss is impossible unless $n/N=\Omega(1)$, i.e. we have observed a constant fraction of all the $N=p^2$ possible data points. It is worth emphasizing, however, that a large ℓ_2 loss does not necessarily imply a large classification error. It remains open to prove a classification error lower bound for permutation-equivariant kernel methods.

4.2 Rich Regime

To analyze the behaviour of the network in the rich regime, we first introduce the notion of margin and elaborate on the margin-maximization implicit bias of gradient descent. For a multi-class classification problem with p classes and a fixed network f, the margin of a data point (x, y) is

$$q(\theta; \boldsymbol{x}, y) \triangleq f_y(\theta; \boldsymbol{x}) - \max_{y' \neq y} f_{y'}(\theta; \boldsymbol{x}).$$

The margin for a dataset \mathcal{D} is defined as the minimum margin of all points on the dataset:

$$q_{\min}(\theta; \mathcal{D}) \triangleq \min_{(\boldsymbol{x}, y) \in \mathcal{D}} q(\theta; \boldsymbol{x}, y).$$

When the network f is homogeneous with respect to its parameters (as is the case in our setup), one can observe that as long as θ linearly separates the dataset \mathcal{D} , i.e., $q_{\min}(\theta;\mathcal{D})>0$, it is possible to arbitrary scale the minimum margin, through scaling the parameters of the network. Hence, in such homogeneous networks, one is usually concerned with a **normalized margin** $q_{\min}(\theta/\|\theta\|;\mathcal{D})$ according to some norm.

Lyu and Li (2020) proved that gradient descent on homogeneous models with the cross-entropy (or similar) losses on dataset \mathcal{D} , in the absence of explicit regularization, maximizes the normalized margin. Specifically, although $\|\theta\|_2 \to \infty$ as $t \to \infty$, $\theta/\|\theta\|_2$ converges to a solution (or more generally, a KKT point) of the following problem when one exists:

$$\min \frac{1}{2} \|\theta\|_2^2 \quad \text{s.t.} \quad q_{\min}(\theta; \mathcal{D}) \ge 1. \tag{4.1}$$

To establish our results in the classification setting, we borrow the following theorem from Wei et al. (2019), who prove that when the strength of the regularization used in (2.2) is small enough, the maximum normalized margin of the regularized loss converges to that of the unregularized loss.

Proposition 4.4 (Wei et al., 2019, Theorem 4.1). Consider a positively homogeneous function f with respect to parameters θ and a dataset $\mathcal{D}_{\text{train}}$ separable with f. Let $\|\cdot\|$ be any norm. Let γ^*

be the maximum normalized margin of the unregularized loss and γ^{λ} be normalized margin of the minimizer of the regularized loss with strength λ . As $\lambda \to 0$, $\gamma^{\lambda} \to \gamma^*$.

That is, if we use a small enough regularization weight λ with any norm in (2.2), we will obtain approximately the same solution as the unregularized problem. This confirms that training our two-layer network using gradient descent with a small enough ℓ_{∞} regularizer can lead to weights close to the solution of the max ℓ_{∞} -normalized margin problem. Next we present the main result of this subsection, a upper bound on test error, showing that all parameters whose ℓ_{∞} -normalized margin is close enough to the max ℓ_{∞} -normalized margin solution will generalize with a sample complexity of $\tilde{\mathcal{O}}(p^{5/3})$

Theorem 4.5 (Upper Bound). Let $\delta \in (0,1)$ be positive constants such that $8p \leq h = O(p)$ independent of p and n. For any n representing the size of the training set \mathcal{D}_{train} it holds with probability at least $1 - \delta$ over randomness of \mathcal{D}_{train} , for all interpolating solutions θ with normalized ℓ_{∞} -margin $\Omega(p)$, it holds that

$$L_0(f, \theta, \mathcal{D}) \le \tilde{\mathcal{O}}\left(\sqrt{\frac{p^{5/3}}{n}}\right).$$
 (4.2)

In other words, any interpolating solution θ with approximately maximal normalized ℓ_{∞} -margin, i.e., $q_{\min}(\theta/\|\theta\|_{\infty}; \mathcal{D}_{\mathrm{train}}) \geq \Omega(\max_{\|\theta'\|_{\infty} \leq 1} q_{\min}(\theta'; \mathcal{D}_{\mathrm{train}}))$, has a sample complexity of $\tilde{\mathcal{O}}(p^{5/3})$, since Proposition 3.8 shows that the maximal normalized ℓ_{∞} -margin is at least $\Omega(p)$.

Our proof of Theorem 4.5 is based on the PAC-Bayesian framework (McAllester, 2003), specifically using Lemma 1 of Neyshabur et al. (2018). This result provides a margin-based high probability generalization bound for any predictor based on the *margin loss*, which counts a prediction as correct only if it predicts with a margin at least γ :

$$L_{\gamma}(f, \theta, \mathcal{D}) \triangleq \Pr_{(\boldsymbol{x}, y) \sim \mathcal{D}} \left[q(\theta, \boldsymbol{x}, y) > \gamma \right]. \tag{4.3}$$

 L_0 is simply the misclassification rate, or 0-1 error. The following statement uses this result to prove an upper bound on the population 0-1 error of networks using margin loss on the training dataset.

Theorem 4.6. For any $p \geq 2$, training set size $n \geq 1, \delta \in (0,1)$, norm r > 0, width $h' > 4 \log \frac{2}{\delta}$ and normalized margin $\gamma/r^3 = \tilde{\Omega}(\sqrt{h})$ it holds with probability at least $1 - \delta$ over the randomness of the training set $\mathcal{D}_{\text{train}}$ that for any θ' of width h' with $\|\theta'\|_{\infty} \leq r$

$$L_0(f, \theta', \mathcal{D}) \le L_{\gamma}(f, \theta', \mathcal{D}_{\text{train}}) + \tilde{\mathcal{O}}\left(\sqrt{\frac{p}{n}} \cdot \sqrt[3]{\frac{h^2}{\gamma/r^3}}\right)$$
 (4.4)

where \mathcal{D} denotes the population and f, L are defined in Equations (2.1) and (4.3) respectively.

Proof Sketch of Theorem 4.6. Through a series of concentration inequalities, we show that as long as the assumptions of Theorem 4.6 are met, the output logits under gaussian perturbation on parameters, $f(\theta' + \tilde{\theta}'; \cdot)$, where $\tilde{\theta}'$ is entry-wise Gaussian noise of mean zero and variance σ^2 , are close to that of $f(\theta'; \cdot)$ up to an absolute difference of $\tilde{\mathcal{O}}(\sqrt{h}\sigma^3)$. The proof is completed by applying a PAC-bayesian generalization bound, e.g., using Lemma 1 of Neyshabur et al. (2018) and setting σ based on the value of ℓ_{∞} -normalized margin γ/r^3 and width h. The full proof is available in Appendix G.

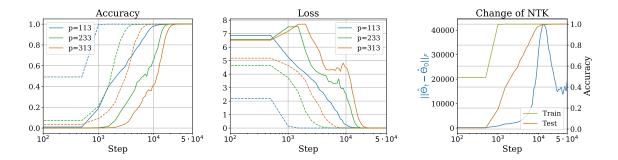


Figure 5: Grokking in transformers happens after a delayed transition from kernel to rich regime. A one-layer transformer is trained with gradient descent using cross-entropy loss and a tiny ℓ_{∞} regularization of 10^{-20} strength on $2 \times p^{5/3}$ training samples from the modular addition problem with various ps. Change of eNTK up to the point of fitting the training set is negligible. The eNTK has a drastic change only after fitting the whole training set, implying minimal feature learning until past overfitting. The dashed lines in the middle and left figures indicate train set statistics, and the solid lines correspond to the test set.

Now we show how to derive Theorem 4.5 using Theorem 4.6. Because Theorem 4.6 only relies on the normalized margin and our model is 3-homogeneous, without loss of generality, we can fix norm bound r=1. Then it suffices to set 8p < h = O(p) and $\gamma = \Omega(p)$ in Equation (4.4), where the first term becomes 0 because Theorem 4.5 assumes θ has normalized margin $\Omega(p)$ and the second term becomes $\tilde{O}(\sqrt{\frac{p^{5/3}}{n}})$.

Theorem 4.5 confirms and explains previous observations (e.g. Power et al., 2022; Gromov, 2023; Nanda et al., 2023; Liu et al., 2023) on the minimum threshold for the fraction of data used to achieve generalization. In combination with Theorem 4.2 of Lyu and Li (2020), this shows that with enough training data, gradient descent will eventually find a generalizing solution for this setting of the modular addition problem.

Empirical Verification of Our Theory: In terms of whether gradient descent finds a model satisfying these conditions: Figure 4 empirically evaluates the ℓ_{∞} norm of the learned weights through gradient descent on the classification task, with tiny ℓ_{∞} regularization (weight of 10^{-20}), across different values of p. Again, it can be seen that the ℓ_{∞} norm of the weights of the network do not grow with the problem dimension. These experiment use n=2 $p^{5/3}$ data points for each p, with a learning rate of 10.

Similar to the regression setting, Figure 1 presents empirical evidence on the impact of kernel regime on the poor generalization capabilities in the early phase of training by showing the minimal change of empirical NTK until after overfitting in the classification setting.

Through changing the *scale of initialization*, in Figure 1 we demonstrate that by decreasing the scale of initialization in the classification task one can mitigate grokking such that the gap between overfitting and generalization becomes larger or smaller. This further supports that as long as the network lies in the kernel regime, generalization without having access to the a constant fraction of the dataset is impossible.

Grokking Modular Addition in Transformers: Figure 5 suggests that, similarly to the two-layer network, grokking in the original Transformer studied by Power et al. (2022) might be explainable through the same mechanism. In fact, Theorem 3.3, albeit with small modifications to incorporate the shared embeddings in the transformer architecture, applies to transformers as well.

5 Additional Related Works

Limitations imposed by Equivariance of Learning Algorithms. Abbe and Boix-Adsera (2022) study the impact of equivariance of the training algorithms on the efficiency of learning different functions on different networks. In particular, they consider two main setups: a) learning with FCNs using noisy GD with clipped gradients throughout the training, and b) learning a specific instance of the modular addition task (p=2 with noisy inputs) with FCNs using SGD. Although their approach in studying lower bounds for efficient learning shares some high level similarity with ours in using equivariance of the training algorithm, the settings considered are significantly differs from ours. Moreover, in Appendix D.4 we present a novel abstract framework for analyzing lower bounds on population ℓ_2 loss for general function classes. Malach and Shalev-Shwartz (2022) present another technique in analysis of population loss lower bounds, which shares some high-level similarity with our framework, albeit their analysis is more restrictive on function classes. Ng (2004) also discusses rotational equivariance of many learning algorithms and presents a general lower bound on the 0-1 population error of such algorithms in the general case.

Margin Maximization as the Late Phase Implicit Bias. Morwani et al. (2024) present an analytical solution for the max-margin solution of learning modular addition in a classification setting similar to Section 4 when using all of the dataset in training the network. Similar to our analysis of the rich regime in this setting, they face difficulties in proving results under the assumption of bounded ℓ_2 norm for weights of the network, and assume an $\ell_{2,3}$ bound instead.

6 Conclusion

In this work, we studied the phenomenon of grokking in learning modular addition with gradient descent on two-layer networks, modeled as regression or classification. We showed that learning modular addition as presented is fundamentally a difficult task for kernel models (for example neural networks in kernel regime) due to the inherent symmetry and permutation-equivariance of the task. We theoretically established this difficulty by presenting sample complexity lower bounds of order of constant fraction of the whole dataset. We further showed that networks satisfying certain conditions generalize far better than those in the kernel regime, that such networks exist, and showed empirical evidence that simple regularized gradient descent can eventually find them, once it escapes the kernel regime. These results, in combination, attempt to address *why* grokking is observed when learning modular addition. We provide strong evidence to the hypothesis (Lyu et al., 2024; Kumar et al., 2024) that, on this important problem, it is indeed due to a separation between kernel and non-kernel behavior of gradient descent.

Future Work. We have guaranteed large ℓ_2 population loss for networks in kernel regime on both regression and classification settings. It is possible, however, for networks to have arbitrarily high ℓ_2 loss while having perfect classification accuracy. We conjecture that impossibility results for accuracy-based generalization may also be possible based on permutation equivariance in the early phase of training, but leave it as future work. Moreover, we only study the cause of grokking in these settings, but do not analyze possible training techniques to enable quick generalization on this task. Although we are able to eliminate grokking through changing the scale of initialization, doing so actually slows down the time to final generalization; finding practical methods to enable quick generalization would be more useful.

Acknowledgments

We would like to thank Kaifeng Lyu and Wei Hu for helpful discussions. This work was enabled in part by support provided by the Natural Sciences and Engineering Research Council of Canada, the Canada CIFAR AI Chairs program, Advanced Research Computing at the University of British Columbia, Calcul Québec, the BC DRI Group, and the Digital Research Alliance of Canada.

References

- Emmanuel Abbe and Enric Boix-Adsera (2022). "On the non-universality of deep learning: quantifying the cost of symmetry." *NeurIPS*. arXiv: 2208.03113.
- Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, Geeta Chauhan, Anjali Chourdia, Will Constable, Alban Desmaison, Zachary DeVito, Elias Ellison, Will Feng, Jiong Gong, Michael Gschwind, Brian Hirsh, Sherlock Huang, Kshiteej Kalambarkar, Laurent Kirsch, Michael Lazos, Mario Lezcano, Yanbo Liang, Jason Liang, Yinghai Lu, C. K. Luk, Bert Maher, Yunjie Pan, Christian Puhrsch, Matthias Reso, Mark Saroufim, Marcos Yukio Siraichi, Helen Suk, Shunting Zhang, Michael Suo, Phil Tillet, Xu Zhao, Eikan Wang, Keren Zhou, Richard Zou, Xiaodong Wang, Ajit Mathews, William Wen, Gregory Chanan, Peng Wu, and Soumith Chintala (2024). "PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation." *Architectural Support for Programming Languages and Operating Systems*.
- Sanjeev Arora, Nadav Cohen, Wei Hu, and Yuping Luo (2019a). "Implicit Regularization in Deep Matrix Factorization." *NeurIPS*. arXiv: 1905.13655.
- Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Russ R Salakhutdinov, and Ruosong Wang (2019b). "On exact computation with an infinitely wide neural net." *NeurIPS*. arXiv: 1904.11955.
- Boaz Barak, Benjamin Edelman, Surbhi Goel, Sham Kakade, Eran Malach, and Cyril Zhang (2022). "Hidden progress in deep learning: SGD learns parities near the computational limit." *NeurIPS*. arXiv: 2207.08799.
- Satwik Bhattamishra, Arkil Patel, Varun Kanade, and Phil Blunsom (2023). "Simplicity Bias in Transformers and their Ability to Learn Sparse Boolean Functions." *ACL*. arXiv: 2211.12316.
- Guy Blanc, Neha Gupta, Gregory Valiant, and Paul Valiant (2020). "Implicit regularization for deep neural networks driven by an Ornstein-Uhlenbeck like process." arXiv: 1904.09080.
- James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang (2018). "JAX: composable transformations of Python+NumPy programs." Version 0.3.13. URL: http://github.com/google/jax.
- François Charton (2024). "Learning the greatest common divisor: explaining transformer predictions." *ICLR*. arXiv: 2308.15594.
- Lenaic Chizat, Edouard Oyallon, and Francis Bach (2019). "On Lazy Training in Differentiable Programming." *NeurIPS*. arXiv: 1812.07956.
- Richard Courant and David Hilbert (1953). *Methods of Mathematical Physics*. Vol. 1. Interscience Publishers.
- Alex Damian, Tengyu Ma, and Jason D. Lee (2021). "Label Noise SGD Provably Prefers Flat Global Minimizers." *NeurIPS*. arXiv: 2106.06530.
- Stanislav Fort, Gintare Karolina Dziugaite, Mansheej Paul, Sepideh Kharaghani, Daniel M. Roy, and Surya Ganguli (2020). "Deep learning versus kernel learning: an empirical study of loss landscape geometry and the time evolution of the Neural Tangent Kernel." *NeurIPS*. arXiv: 2010.15110.

- Mario Geiger, Stefano Spigler, Arthur Jacot, and Matthieu Wyart (Nov. 2020). "Disentangling feature and lazy training in deep neural networks." *Journal of Statistical Mechanics: Theory and Experiment* 2020.11, p. 113301. arXiv: 1906.08034.
- Andrey Gromov (2023). "Grokking modular arithmetic." arXiv: 2301.02679.
- Suriya Gunasekar, Jason Lee, Daniel Soudry, and Nathan Srebro (2018). "Characterizing Implicit Bias in Terms of Optimization Geometry." *ICML*. arXiv: 1802.08246.
- Suriya Gunasekar, Blake Woodworth, Srinadh Bhojanapalli, Behnam Neyshabur, and Nathan Srebro (2017). "Implicit Regularization in Matrix Factorization." *NeurIPS*. arXiv: 1705.09280.
- Jeff Z. HaoChen, Colin Wei, Jason D. Lee, and Tengyu Ma (2020). "Shape Matters: Understanding the Implicit Bias of the Noise Covariance." *COLT*. arXiv: 2006.08680.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2015). "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification." *ICCV*. arXiv: 1502.01852.
- Arthur Jacot, Franck Gabriel, and Clément Hongler (2018). "Neural tangent kernel: Convergence and generalization in neural networks." *NeurIPS*. arXiv: 1806.07572.
- Tanishq Kumar, Blake Bordelon, Samuel J. Gershman, and Cengiz Pehlevan (2024). "Grokking as the Transition from Lazy to Rich Training Dynamics." *ICLR*. arXiv: 2310.06110.
- Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington (2019). "Wide neural networks of any depth evolve as linear models under gradient descent." *NeurIPS*. arXiv: 1902.06720.
- Noam Levi, Alon Beck, and Yohai Bar-Sinai (2024). "Grokking in Linear Estimators A Solvable Model that Groks without Understanding." *ICLR*. arXiv: 2310.16441.
- Zhiyuan Li, Tianhao Wang, and Sanjeev Arora (2022). "What Happens after SGD Reaches Zero Loss? –A Mathematical Framework." *ICLR*. arXiv: 2110.06914.
- Zhiyuan Li, Yi Zhang, and Sanjeev Arora (2021). "Why are convolutional nets more sample-efficient than fully-connected nets?" *ICLR*. arXiv: 2010.08515.
- Ziming Liu, Ouail Kitouni, Niklas S Nolte, Eric Michaud, Max Tegmark, and Mike Williams (2022). "Towards understanding grokking: An effective theory of representation learning." *NeurIPS*. arXiv: 2205.10343.
- Ziming Liu, Eric J Michaud, and Max Tegmark (2023). "Omnigrok: Grokking beyond algorithmic data." *ICLR*. arXiv: 2210.01117.
- Kaifeng Lyu, Jikai Jin, Zhiyuan Li, Simon S. Du, Jason D. Lee, and Wei Hu (2024). "Dichotomy of Early and Late Phase Implicit Biases Can Provably Induce Grokking." *ICLR*. arXiv: 2311.18817.
- Kaifeng Lyu and Jian Li (2020). "Gradient descent maximizes the margin of homogeneous neural networks." *ICLR*. arXiv: 1906.05890.
- Eran Malach and Shai Shalev-Shwartz (2022). "When Hardness of Approximation Meets Hardness of Learning." *Journal of Machine Learning Research* 23.91, pp. 1–24. arXiv: 2008.08059.
- David A. McAllester (2003). "Simplified PAC-Bayesian Margin Bounds." COLT.
- Mohamad Amin Mohamadi, Wonho Bae, and Danica J Sutherland (2023). "A fast, well-founded approximation to the empirical neural tangent kernel." *International Conference on Machine Learning*. PMLR, pp. 25061–25081.
- Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talkwalkar (2018). Foundations of Machine Learning. 2nd ed. MIT Press. url: https://cs.nyu.edu/~mohri/mlbook/.
- Edward Moroshko, Suriya Gunasekar, Blake Woodworth, Jason D. Lee, Nathan Srebro, and Daniel Soudry (2020). "Implicit Bias in Deep Linear Classification: Initialization Scale vs Training Accuracy." *NeurIPS*. arXiv: 2007.06738.
- Depen Morwani, Benjamin L. Edelman, Costin-Andrei Oncescu, Rosie Zhao, and Sham Kakade (2024). "Feature emergence via margin maximization: case studies in algebraic tasks." *ICLR*. arXiv: 2311.07568.

- Mor Shpigel Nacson, Suriya Gunasekar, Jason D. Lee, Nathan Srebro, and Daniel Soudry (2019). "Lexicographic and Depth-Sensitive Margins in Homogeneous and Non-Homogeneous Deep Models." *ICML*. arXiv: 1905.07325.
- Neel Nanda, Lawrence Chan, Tom Liberum, Jess Smith, and Jacob Steinhardt (2023). "Progress measures for grokking via mechanistic interpretability." *ICLR*. arXiv: 2301.05217.
- Behnam Neyshabur, Srinadh Bhojanapalli, and Nathan Srebro (2018). "A PAC-Bayesian Approach to Spectrally-Normalized Margin Bounds for Neural Networks." *ICLR*. arXiv: 1707.09564.
- Andrew Y Ng (2004). "Feature selection, L_1 vs. L_2 regularization, and rotational invariance." *ICML*. Pascal Jr. Tikeng Notsawo, Hattie Zhou, Mohammad Pezeshki, Irina Rish, and Guillaume Dumas (2023). "Predicting Grokking Long Before it Happens: A look into the loss landscape of models which grok." arXiv: 2306.13253.
- Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra (2022). "Grokking: Generalization Beyond Overfitting on Small Algorithmic Datasets." arXiv: 2201.02177.
- Martin Raab and Angelika Steger (1998). "Balls into Bins' A Simple and Tight Analysis." *Randomization and Approximation Techniques in Computer Science*.
- Noa Rubin, Inbar Seroussi, and Zohar Ringel (2024). "Grokking as a First Order Phase Transition in Two Layer Networks." *ICLR*. arXiv: 2310.03789.
- Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro (2018). "The Implicit Bias of Gradient Descent on Separable Data." *JMLR* 19.70. arXiv: 1710.10345.
- Nathan Srebro, Karthik Sridharan, and Ambuj Tewari (2010). "Smoothness, low noise and fast rates." arXiv: 1009.3896.
- Matus Telgarsky (2022). "Feature selection with gradient descent on two-layer networks in low-rotation regimes." arXiv: 2208.02789.
- Vimal Thilak, Etai Littwin, Shuangfei Zhai, Omid Saremi, Roni Paiss, and Joshua Susskind (2022). "The slingshot mechanism: An empirical study of adaptive optimizers and the grokking phenomenon." arXiv: 2206.04817.
- Joel A. Tropp (2015). "An Introduction to Matrix Concentration Inequalities." *Foundations and Trends® in Machine Learning* 8.1-2, pp. 1–230. arXiv: 1501.01571.
- Vikrant Varma, Rohin Shah, Zachary Kenton, János Kramár, and Ramana Kumar (2023). "Explaining grokking through circuit efficiency." arXiv: 2309.02390.
- Martin J. Wainwright (2019). *High-Dimensional Statistics: A Non-Asymptotic Viewpoint*. Cambridge University Press.
- Colin Wei, Jason D. Lee, Qiang Liu, and Tengyu Ma (2019). "Regularization Matters: Generalization and Optimization of Neural Nets v.s. their Induced Kernel." *NeurIPS*. arXiv: 1810.05369.
- Shuo Xie and Zhiyuan Li (2024). "Implicit Bias of AdamW: ℓ_{∞} Norm Constrained Optimization." *ICML*. arXiv: 2404.04454.
- Zhiwei Xu, Yutong Wang, Spencer Frei, Gal Vardi, and Wei Hu (2024). "Benign Overfitting and Grokking in ReLU Networks for XOR Cluster Data." *ICLR*. arXiv: 2310.02541.
- Greg Yang and Edward J. Hu (2021). "Tensor Programs IV: Feature Learning in Infinite-Width Neural Networks." *ICML*. arXiv: 2011.14522.
- Chenyang Zhang, Difan Zou, and Yuan Cao (2024). "The Implicit Bias of Adam on Separable Data." arXiv: 2406.10650.
- Mauricio A Álvarez, Lorenzo Rosasco, and Neil D Lawrence (2012). "Kernels for vector-valued functions: A review." *Foundations and Trends® in Machine Learning* 4.3, pp. 195–266. arXiv: 1106.6251.

A Experimental Setup

In this section, we briefly explain the setup used for our experimental evaluations.

A.1 Regression

We use vanilla gradient descent with squared loss and tiny ℓ_{∞} regularization for 50,000, 100,000 or 200,000 steps for each experiment. In regression, our learning rate has been fixed to 1, and the regularization strength has been set to 10^{-4} . The network has been initialized according to He et al. (2015). The amount of data used for training in regression task has been set to $2 \times p^{2.25}$.

A.2 Classification

We use vanilla gradient descent with cross-entropy loss and tiny ℓ_{∞} regularization, for up to 100,000 steps. The learning rate in the presented experiments was set to 10 and was kept constant during the training. The regularization strength of ℓ_{∞} regularizer has been set to 10^{-20} . To accelerate the training with cross-entropy loss, we use the "normalized" GD trick, where the learning rate of each step is scaled by the inverse of the norm of the gradient:

$$\theta_{t+1} = \theta_t - \eta \frac{\nabla_{\theta} \ell(\theta_t)}{\|\nabla_{\theta} \ell(\theta_t)\|_2}$$
(A.1)

where ℓ denotes the loss function and η denotes the learning rate. The network has been initialized according to He et al. (2015). The amount of data used for training in regression task has been set to $2 \times p^{5/3}$.

A.3 Transformer

To train the one-layer transformer we have used full-batch gradient descent with a learning rate of $\eta=0.25$ and a tiny ℓ_{∞} regularization with the strength of 1.0e-20. The network has been initialized according to default PyTorch initialization (migrated ot JAX). We have used $2\times p^{5/3}$ of the data for training.

A.4 Logistics

We used the JAX framework (Bradbury et al., 2018) to implement and run the experiments on machines using NVIDIA V100 or A100 GPUs.

B Ability of Neural Tangent Kernel Models to Interpolate

In this section, we prove Theorem 3.1, that neural tangent kernel models for our one-hidden-layer quadratic network are able to exactly interpolate their training data, in the regression setting.

If the training set contains duplicate x, say $x_i = x_j$ for $i \neq j$, then K cannot be full rank – the ith and jth rows are necessarily identical. Of course, there also exist unattainable targets; just set $y_i \neq y_j$. Call a labeling *consistent* if for all i and j such that $x_i = x_j$, $y_i = y_j$. A learning algorithm can achieve any consistent labeling if and only if that algorithm applied to the largest distinct subset (e.g. removing j but keeping i) can achieve any labeling. Thus, we assume without loss of generality in the remainder of this section that the training set contains no duplicates.

B.1 Full-Rank Kernels are Expressive

Recalling that neural tangent kernel models correspond to "ridgeless" kernel regression (e.g. Jacot et al., 2018; Lee et al., 2019), we first notice that this method can interpolate *any* set of training labels when the kernel is strictly positive definite.

Specifically, empirical neural tangent kernel regression corresponds to ridgeless regression with a prior mean corresponding to the network at initialization f_0 ,

$$\underset{f:\forall i, f(\boldsymbol{x}_i)=y_i}{\operatorname{argmin}} \|f - f_0\|_K = f_0 + \underset{f:\forall i, f(\boldsymbol{x}_i)=y_i - f_0}{\operatorname{argmin}} \|f\|_K = f_0 + \sum_{i=1}^n K(\cdot, \boldsymbol{x}_i) \lambda_i \text{ for } \lambda = \mathbf{K}^{\dagger} \mathbf{y}',$$

where \mathbf{K}^{\dagger} is the Moore-Penrose pseudoinverse of the $n \times n$ kernel matrix $[K(\boldsymbol{x}_i, \boldsymbol{x}_j)]_{ij}$, and \mathbf{y}' has ith entry $y_i - f_0(\boldsymbol{x}_i)$. By changing variables between \mathbf{y}' and \mathbf{y} , we need not explicitly consider f_0 in the following two results.

Lemma B.1. Let $\mathcal{D}_{train} = \{(x_i, y_i)\}_{i \in [n]}$, and let K be a kernel such that the kernel matrix $\mathbf{K} = [K(x_i, x_j)]_{ij}$ is strictly positive definite. Then kernel ridgeless regression achieves zero training error on \mathcal{D}_{train} .

Proof. This well-known result follows from the fact that
$$[\hat{f}(\mathbf{x}_i)]_i = \mathbf{K}\mathbf{K}^{\dagger}\mathbf{y}$$
. If **K** is strictly positive definite, $\mathbf{K}^{\dagger} = \mathbf{K}^{-1}$, and so $[\hat{f}(x_i)]_i = \mathbf{y}$.

The following result is a partial converse.

Lemma B.2. Let $\{x_i\}_{i\in[n]}$ and K be a kernel such that the kernel matrix $\mathbf{K} = [K(x_i, x_j)]_{ij}$ is singular. Then there exists an assignment of $y_i \in \mathbb{R}$ such that kernel ridgeless regression achieves nonzero training error on $\mathcal{D}_{\text{train}} = \{(x_i, y_i)\}_{i\in[n]}$.

Proof. Let y be any nonzero vector in the null space of K; such vectors exist since K is singular. Then y is also in the null space of K^{\dagger} , and the training set predictions are $KK^{\dagger}y = 0 \neq y$.

B.2 Expected NTK is Full-Rank

We next show that, in the regression setting, the *expected* neural tangent kernel is strictly positive definite. While perhaps of interest of its own accord, this will be a key component in our analysis of finite-width networks that follows.

Proposition B.3. Let the entries of W follow a distribution \mathcal{P}_W , all mutually independent. Assume \mathcal{P}_W has mean zero, variance $\sigma_W^2 > 0$, skewness zero, and kurtosis κ_W such that $\mathbb{E}_{w \sim \mathcal{P}_W} w^4 = \kappa_W \sigma_W^4$. The entries of V can be arbitrary. Then the expected neural tangent kernel for the regression network with any finite $h \geq 1$ is strictly positive definite on any set of distinct inputs $\{x_i : i \in [n]\}$, with minimum eigenvalue at least $4h\sigma_W^4$.

We first note that if $\mathcal{P}_W = \mathcal{N}(0, \sigma_W^2)$, $\kappa_V = 3$. If $\mathcal{P}_W = \mathrm{Unif}([-s_W, s_W])$, $\sigma_W^2 = \frac{1}{3}s_W^2$ and $\kappa_W = \frac{9}{5}$. These two distributions cover the vast majority of initialization schemes used in practice.

Proof. It will be more convenient in this proof to give different names to the sub-matrix of the parameter vector W that act on the a inputs and the b inputs; we will write $W = \begin{bmatrix} Q & R \end{bmatrix}$, where Q, R are each $h \times p$ matrices. Then we can write the full model as

$$g(\theta; (e_a, e_b, e_c)) = e_c^{\mathsf{T}} V(Qe_a + Re_b)^{\odot 2} = \sum_{k=1}^h V_{ck} (Q_{ka} + R_{kb})^2.$$

The empirical neural tangent kernel between two inputs $x = (e_a, e_b, e_c)$ and $x' = (e_{a'}, e_{b'}, e_{c'})$ is then given by

$$K_{\theta}(x, x') = \sum_{k=1}^{h} \left[\sum_{c''=1}^{p} \frac{\partial g(\theta; x)}{\partial V_{c''k}} \frac{\partial g(\theta; x')}{\partial V_{c''k}} + \sum_{a''=1}^{p} \frac{\partial g(\theta; x)}{\partial Q_{ka''}} \frac{\partial g(\theta; x')}{\partial Q_{ka''}} + \sum_{b''=1}^{p} \frac{\partial g(\theta; x)}{\partial R_{kb''}} \frac{\partial g(\theta; x')}{\partial R_{kb''}} \right].$$
(B.1)

Evaluating the V derivatives,

$$\frac{\partial g(\theta;x)}{\partial V_{c''k}} = \begin{cases} (Q_{ka} + R_{kb})^2 & \text{if } c = c'' \\ 0 & \text{otherwise} \end{cases}$$

$$\sum_{c''=1}^p \frac{\partial g(\theta;x)}{\partial V_{c''k}} \frac{\partial g(\theta;x')}{\partial V_{c''k}} = \begin{cases} (Q_{ka} + R_{kb})^2 (Q_{ka'} + R_{kb'})^2 & \text{if } c = c' \\ 0 & \text{otherwise.} \end{cases}$$

While it would not be difficult to analyze the Q and R derivatives as well, their exact form will not be important. Instead, we only need to write

$$K_{\theta}(x, x') = J_{\theta}(x, x') + \sum_{k=1}^{h} L_{\theta_k}(x, x')$$
 (B.2)

$$J_{\theta}(x, x') = \sum_{k=1}^{h} \sum_{a''=1}^{p} \frac{\partial g(\theta; x)}{\partial Q_{ka''}} \frac{\partial g(\theta; x')}{\partial Q_{ka''}} + \sum_{b''=1}^{p} \frac{\partial g(\theta; x)}{\partial R_{kb''}} \frac{\partial g(\theta; x')}{\partial R_{kb''}}$$

$$L_{\theta_{k}}(x, x') = (Q_{ka} + R_{kb})^{2} (Q_{ka'} + R_{kb'})^{2} \mathbf{1}(c = c'). \tag{B.3}$$

The function J_{θ} has an explicit feature map corresponding to the relevant gradients; thus, for any set of inputs $\{x_i: i \in [n]\}$ and any value of θ , the kernel matrix $\mathbf{J}_{\theta} = [J_{\theta}(x_i, x_j)]_{ij}$ is positive semi-definite. This implies, e.g. via Weyl's inequality, that $\mathbb{E}_{\theta} \mathbf{J}_{\theta}$ is also positive semi-definite.

For any fixed set of inputs $\{x_i : i \in [n]\}$, the kernel matrices $\mathbf{L}_{\theta_k} = [L_{\theta_k}(\mathbf{x}_i, \mathbf{x}_j)]_{ij}$ are independent and identically distributed, and in particular have the same mean $\mathbb{E} \mathbf{L}_{\theta_k}$ for any arbitrary choice of $k \in [p]$. Thus the expected neural tangent kernel matrix can be written as

$$[\mathbb{E} K_{\theta}(x_i, x_j)]_{ij} = \mathbb{E} \mathbf{J}_{\theta} + h \,\mathbb{E} \mathbf{L}_{\theta_k}.$$

We will show that $\mathbb{E} \mathbf{L}_{\theta_k}$ has minimum eigenvalue at least $4\sigma_W^4$, from which the result follows. To do so, we will evaluate

$$\mathbb{E} L_{\theta_k}(x, x') = \mathbb{E} \Big[(Q_{ka} + R_{kb})^2 (Q_{ka'} + R_{kb'})^2 \Big] \mathbf{1}(c = c')$$

for arbitrary nonrandom $\mathbf{x} = (e_a, e_b, e_c)$ and $\mathbf{x}' = (e_{a'}, e_{b'}, e_{c'})$, where all expectations will be over the relevant parameters $\{Q_{ka} : a \in [p]\} \cup \{R_{kb} : b \in [p]\}$.

If a = a' and b = b',

$$\mathbb{E}\left[\left(Q_{ka}+R_{kb}\right)^{4}\right] = \underbrace{\mathbb{E}Q_{ka}^{4}}_{\kappa_{W}\sigma_{W}^{4}} + 4\underbrace{\mathbb{E}Q_{ka}^{3}}_{0}\underbrace{\mathbb{E}R_{kb}}_{0} + 6\underbrace{\mathbb{E}Q_{ka}^{2}}_{\sigma_{W}^{2}}\underbrace{\mathbb{E}R_{kb}^{2}}_{\sigma_{W}^{2}} + 4\underbrace{\mathbb{E}Q_{ka}}_{0}\underbrace{\mathbb{E}R_{kb}^{3}}_{0} + \underbrace{\mathbb{E}R_{kb}^{4}}_{\kappa_{W}\sigma_{W}^{4}}$$
$$= (2\kappa_{W}+6)\sigma_{W}^{4}.$$

If instead a = a' but $b \neq b'$, then

$$\mathbb{E}\left[(Q_{ka}+R_{kb})^{2}(Q_{ka}+R_{kb'})^{2}\right] = \mathbb{E}\left[(Q_{ka}^{2}+2Q_{ka}R_{kb}+R_{kb}^{2})(Q_{ka}^{2}+2Q_{ka}R_{kb'}+R_{kb'}^{2})\right]$$

$$= \underbrace{\mathbb{E}Q_{ka}^{4}}_{\kappa_{W}} + 2\underbrace{\mathbb{E}Q_{ka}^{3}}_{0}\underbrace{\mathbb{E}R_{kb'}}_{0} + \underbrace{\mathbb{E}Q_{ka}^{2}}_{\sigma_{W}^{2}}\underbrace{\mathbb{E}R_{kb'}^{2}}_{\sigma_{W}^{2}}$$

$$+ 2\underbrace{\mathbb{E}Q_{ka}^{3}}_{0}\underbrace{\mathbb{E}R_{kb}}_{0} + 4\underbrace{\mathbb{E}Q_{ka}^{2}}_{\sigma_{W}^{2}}\underbrace{\mathbb{E}R_{kb}}_{0}\underbrace{\mathbb{E}R_{kb'}}_{0} + 2\underbrace{\mathbb{E}Q_{ka}}_{0}\underbrace{\mathbb{E}R_{kb}}_{0}\underbrace{\mathbb{E}R_{kb'}}_{0}$$

$$+ \underbrace{\mathbb{E}R_{kb}^{2}}_{\sigma_{W}^{2}}\underbrace{\mathbb{E}Q_{ka}^{2}}_{\sigma_{W}^{2}} + 2\underbrace{\mathbb{E}R_{kb}^{2}}_{0}\underbrace{\mathbb{E}Q_{ka}}_{0}\underbrace{\mathbb{E}R_{kb'}}_{0} + \underbrace{\mathbb{E}R_{kb}^{2}}_{\sigma_{W}^{2}}\underbrace{\mathbb{E}R_{kb'}^{2}}_{\sigma_{W}^{2}}$$

$$= (\kappa_{W} + 3) \sigma_{W}^{4};$$

the case where $a \neq a'$ but b = b' is the same by symmetry.

Finally, when $a \neq a'$ and $b \neq b'$, we have by independence that

$$\mathbb{E}[(Q_{ka} + R_{kb})^{2}(Q_{ka'} + R_{kb'})^{2}] = \mathbb{E}[(Q_{ka} + R_{kb})^{2}] \mathbb{E}[(Q_{ka'} + R_{kb'})^{2}] = \left(\mathbb{E}[(Q_{ka} + R_{kb})^{2}]\right)^{2}$$

$$= \left(\mathbb{E}\frac{Q_{ka}^{2}}{\sigma_{W}^{2}} + 2\mathbb{E}\frac{Q_{ka}}{0}\mathbb{E}\frac{R_{kb}}{0} + \mathbb{E}\frac{R_{kb}^{2}}{\sigma_{W}^{2}}\right)^{2} = (2\sigma_{W}^{2})^{2} = 4\sigma_{W}^{4}.$$

Combining the cases, it holds in general that

$$\mathbb{E} L_{\theta_k}(x, x') = \sigma_W^4 \mathbf{1}(c = c') \begin{cases} 4 & \text{if } a \neq a', b \neq b' \\ \kappa_W + 3 & \text{if } a = a', b \neq b' \\ \kappa_W + 3 & \text{if } a \neq a', b = b' \\ 2\kappa_W + 6 & \text{if } a = a', b = b' \end{cases}$$
$$= 4\sigma_W^4 \mathbf{1}(c = c') + (\kappa_W - 1)\sigma_W^4 \mathbf{1}(a = a', c = c') + (\kappa_W - 1)\sigma_W^4 \mathbf{1}(b = b', c = c') + 4\sigma_W^4 \mathbf{1}(a = a', b = b', c = c').$$

The kurtosis of any probability distribution is at least 1 by Jensen's inequality, so all the coefficients in this last form are nonnegative. We can then use this to construct an explicit feature map for each term in $\mathbb{E} L_{\theta_k}$, because if $\gamma \geq 0$,

$$\gamma \mathbf{1}(c = c') = \begin{bmatrix} \sqrt{\gamma} \mathbf{1}(c = 1) \\ \sqrt{\gamma} \mathbf{1}(c = 2) \\ \vdots \\ \sqrt{\gamma} \mathbf{1}(c = p) \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} \sqrt{\gamma} \mathbf{1}(c' = 1) \\ \sqrt{\gamma} \mathbf{1}(c' = 2) \\ \vdots \\ \sqrt{\gamma} \mathbf{1}(c' = p) \end{bmatrix}$$
(B.4)

corresponds to explicit features in \mathbb{R}^p . The other indicator functions can be implemented in the same way, in \mathbb{R}^{p^2} or \mathbb{R}^{p^3} ; their sum can then be obtained by concatenating the individual features together. This construction makes it clear that the function

$$M(x, x') = 4\sigma_W^4 \mathbf{1}(c = c') + (\kappa_W - 1)\sigma_W^4 \mathbf{1}(a = a', c = c') + (\kappa_W - 1)\sigma_W^4 \mathbf{1}(b = b', c = c')$$

is a positive semi-definite kernel, so $\mathbf{M} = [M(x_i, x_j)]_{ij}$ is a positive semi-definite matrix for any set of inputs $\{x_i\}$.

It remains to show that the final component of the kernel is strictly positive definite. Noticing that $\mathbf{1}(a=a',b=b',c=c')=\mathbf{1}(x=x')$, if the $\{x_i:i\in[n]\}$ are distinct, the expected kernel matrix is $\mathbb{E}\mathbf{L}_{\theta_k}=\mathbf{M}+4\sigma_W^4\mathbf{I}$. Since \mathbf{M} is positive semi-definite, this has minimum eigenvalue at least $4\sigma_W^4>0$, as desired.

B.3 Empirical NTKs are Likely Full-Rank

Now, for *bounded* initialization schemes, we use matrix concentration inequalities to show that the empirical neural tangent kernel is also likely to be full-rank when h is large enough.

Proposition B.4. In the setting of Proposition B.3, further assume that $\Pr_{w \sim \mathcal{P}_W}(|w| \leq s_W) = 1$. Let the set of inputs $\{x_i : i \in [n]\}$ be distinct and such that the most common c value is seen ρ_c times. Then the empirical neural tangent kernel for the regression network is strictly positive definite with probability at least $1 - \delta$ over the choice of random parameters θ as long as

$$h > \frac{4s_W^4}{\sigma_W^4} \rho_c \log \frac{n}{\delta}.$$

If \mathcal{P}_W is uniform on $[-s_W, s_W]$ for $s_W > 0$, this condition is equivalent to

$$h > 36\rho_c \log \frac{n}{\delta}.$$

Proof. Recall the decomposition of $K_{\theta}(\boldsymbol{x}, \boldsymbol{x}') = J_{\theta}(\boldsymbol{x}, \boldsymbol{x}') + \sum_{k=1}^{h} L_{\theta_{k}}(\boldsymbol{x}, \boldsymbol{x}')$ from (B.2), and the corresponding $n \times n$ matrices $\mathbf{K}_{\theta} = [K_{\theta}(\boldsymbol{x}_{i}, \boldsymbol{x}_{j})]_{ij}$, $\mathbf{J}_{\theta} = [J_{\theta}(\boldsymbol{x}_{i}, \boldsymbol{x}_{j})]_{ij}$, and $\mathbf{L}_{\theta_{k}} = [L_{\theta_{k}}(\boldsymbol{x}_{i}, \boldsymbol{x}_{j})]_{ij}$, so that $\mathbf{K}_{\theta} = \mathbf{J}_{\theta} + \sum_{k=1}^{h} \mathbf{L}_{\theta_{k}}$. As shown in the proof of Proposition B.3, \mathbf{J}_{θ} is positive semi-definite; we now wish to show that the sum of h iid matrices $\mathbf{L}_{\theta_{k}}$, whose mean $h \mathbb{E} \mathbf{L}_{\theta_{k}}$ has minimum eigenvalue $\mu_{\min} := 4h\sigma_{W}^{4}$, is likely to be full-rank. We can do so by applying Lemma B.5, a direct corollary of matrix Chernoff bounds, if we additionally have an almost sure upper bound on the operator norm $\|\mathbf{L}_{\theta_{k}}\|$.

Notate the input x_i as $(e_{a_i}, e_{b_i}, e_{c_i})$ for $a_i, b_i, c_i \in [p]$. Using (B.3), we can write

$$\mathbf{L}_{\theta_k} = \operatorname{diag}(\mathbf{w}_{\theta_k}) \mathbf{C} \operatorname{diag}(\mathbf{w}_{\theta_k}), \tag{B.5}$$

where diag : $\mathbb{R}^n \to \mathbb{R}^{n \times n}$ constructs a diagonal matrix with the given vector on its diagonal and

$$(\mathbf{C})_{ij} = \mathbf{1}(c_i = c_j)$$
$$(\mathbf{w}_{\theta_k})_i = (Q_{ka_i} + R_{kb_i})^2.$$

Using $||AB|| \le ||A|| ||B||$ and $||\operatorname{diag}(v)|| = \max_i |v_i|$, we obtain that

$$\|\mathbf{L}_{\theta_k}\| \le (2s_W)^2 \|\mathbf{C}\| (2s_W)^2 = 16s_W^4 \|\mathbf{C}\|.$$

Using (B.4), we can write $\mathbf{C} = \Phi_c \Phi_c^\mathsf{T}$ so that $\|\mathbf{C}\| = \|\Phi_c\|^2$, where $\Phi_c \in \mathbb{R}^{n \times p}$ is given by

$$\Phi_c = \begin{bmatrix} \mathbf{1}(c_1 = 1) & \cdots & \mathbf{1}(c_1 = p) \\ \vdots & \ddots & \vdots \\ \mathbf{1}(c_n = 1) & \cdots & \mathbf{1}(c_n = p) \end{bmatrix}.$$

We can evaluate this operator norm with

$$(\Phi_{c}x)_{i} = \sum_{\ell=1}^{p} \mathbf{1}(c_{i} = \ell)x_{\ell} = x_{c_{i}}$$

$$\|\Phi_{c}x\|^{2} = \sum_{i=1}^{n} x_{c_{i}}^{2} = \sum_{\ell=1}^{p} x_{\ell}^{2} \left(\sum_{i=1}^{n} \mathbf{1}(c_{i} = \ell)\right)$$

$$\|\Phi_{c}\|^{2} = \sup_{\|x\|=1} \sum_{\ell=1}^{p} x_{\ell}^{2} \left(\sum_{i=1}^{n} \mathbf{1}(c_{i} = \ell)\right) = \max_{\ell \in [p]} \sum_{i=1}^{n} \mathbf{1}(c_{i} = \ell) = \rho_{c}.$$

Thus it holds almost surely that

$$\|\mathbf{L}_{\theta_k}\| \le 16s_W^4 \rho_c =: L.$$

Plugging in Lemma B.5, we have shown that

$$\Pr\left(\lambda_{\min}\left(\mathbf{K}_{\theta}\right) > 0\right) \ge 1 - n \exp\left(-\frac{4h\sigma_W^4}{16s_W^4\rho_c}\right),$$

and so \mathbf{K}_{θ} is full rank with probability at least $1 - \delta$ as long as

$$h > \frac{4s_W^4}{\sigma_W^4} \rho_c \log \frac{n}{\delta}.$$

If the entries of W are iid $\operatorname{Unif}([-s_w, s_w])$ for $s_w > 0$, $\sigma_W^2 = \frac{1}{3}s_W^2$, giving the condition

$$h > 36\rho_c \log \frac{n}{\delta}.$$

Lemma B.5. Consider a finite sequence of independent positive semi-definite $d \times d$ real random matrices $\mathbf{X}_1, \dots, \mathbf{X}_m$. Assume that $\|\mathbf{X}_k\| \leq L$ almost surely, and that $\mu_{\min} = \lambda_{\min} \left(\sum_{k=1}^m \mathbb{E} \mathbf{X}_k \right) > 0$. Then

$$\Pr\left(\lambda_{\min}\left(\sum_{k=1}^{m} \mathbf{X}_{k}\right) > 0\right) \ge 1 - d\exp\left(-\frac{\mu_{\min}}{L}\right).$$

Proof. A matrix Chernoff bound (Tropp, 2015, Theorem 5.1.1) in gives that for each $\epsilon \in [0, 1)$,

$$\Pr\left(\lambda_{\min}\left(\sum_{k=1}^{h} \mathbf{X}_{k}\right) > (1 - \epsilon)\mu_{\min}\right) \ge 1 - d\left(\frac{e^{-\epsilon}}{(1 - \epsilon)^{1 - \epsilon}}\right)^{\mu_{\min}/L}.$$
 (B.6)

If the smallest eigenvalue is strictly positive, it must exceed some $\delta > 0$, which corresponds to some $\epsilon < 1$. Thus the event of being full-rank is the limit of the events of exceeding each ϵ , and since the right-hand side of (B.6) is continuous for $\epsilon < 1$, we can take the limit as $\epsilon \nearrow 1$; since $e^{-\epsilon}/(1-\epsilon)^{1-\epsilon} \to 1/e$, this gives the desired result.

This threshold is indeed tight up to logarithmic factors, in the sense that there are some labels which kernel ridgeless regression cannot achieve when h = o(n/p).

Proposition B.6. In the setting of Proposition B.3, the empirical neural tangent kernel of the regression network \mathbf{K}_{θ} is guaranteed to be singular when h < n/(3p).

Proof. Expanding on (B.1), we will additionally need to evaluate the Q and R derivatives:

$$\frac{\partial g(\theta; \boldsymbol{x})}{\partial Q_{ka''}} = \begin{cases} 2V_{ck}(Q_{ka} + R_{kb}) & \text{if } a = a'' \\ 0 & \text{otherwise} \end{cases}$$

$$\sum_{a''=1}^{p} \frac{\partial g(\theta; \boldsymbol{x})}{\partial Q_{ka''}} \frac{\partial g(\boldsymbol{x}')}{\partial Q_{ka''}} = \begin{cases} 4V_{ck}V_{c'k}(Q_{ka} + R_{kb})(Q_{ka} + R_{kb'}) & \text{if } a = a' \\ 0 & \text{otherwise} \end{cases}$$

$$\frac{\partial g(\theta; \boldsymbol{x})}{\partial R_{kb''}} = \begin{cases} 2V_{ck}(Q_{ka} + R_{kb}) & \text{if } b = b'' \\ 0 & \text{otherwise} \end{cases}$$

$$\sum_{b''=1}^{p} \frac{\partial g(\boldsymbol{x})}{\partial R_{kb''}} \frac{\partial g(\boldsymbol{x}')}{\partial R_{kb''}} = \begin{cases} 4V_{ck}V_{c'k}(Q_{ka} + R_{kb})(Q_{ka'} + R_{kb}) & \text{if } b = b' \\ 0 & \text{otherwise}. \end{cases}$$

Writing as in (B.2) and (B.5), we have that

$$\mathbf{K}_{\theta} = \sum_{k=1}^{h} \operatorname{diag}(\mathbf{w}_{\theta_k}) \mathbf{C} \operatorname{diag}(\mathbf{w}_{\theta_k}) + \operatorname{diag}(\mathbf{v}_{\theta_k}) \mathbf{A} \operatorname{diag}(\mathbf{v}_{\theta_k}) + \operatorname{diag}(\mathbf{v}_{\theta_k}) \mathbf{B} \operatorname{diag}(\mathbf{v}_{\theta_k}),$$

where $(\mathbf{v}_{\theta_k})_i = 2V_{ck}(Q_{ka} + R_{kb})$, $\mathbf{A}_{ij} = \mathbf{1}(a_i = a_j)$, and $\mathbf{B}_{ij} = \mathbf{1}(b_i = b_j)$. By (B.4), each of the matrices \mathbf{A} , \mathbf{B} , and \mathbf{C} have rank at most p; thus $\mathrm{rank}(\mathbf{K}_{\theta}) \leq 3ph$. If 3ph < n, this $n \times n$ matrix cannot be full-rank.

C Permutation-Equivariance of Gradient-Based Training

We first define the notion of permutation equivariance. Towards this, we borrow the following proof from Appendix C of Li et al. (2021):

Definition C.1 (Gradient-Based Algorithm \mathcal{A}). We borrow the definition of Algorithm 1 in Li et al. (2021) with the slight modification of restricting the update rule $F(U, \mathcal{M}, \mathcal{D}_{train})$ where U denotes the parameters and $\mathcal{M}: U \to (\mathcal{X} \to \mathbb{R})$ denotes the model mapping parameters to a function. We restrict the update rule to only allow gradient-based update rules, such that

$$F(U, \mathcal{M}, \mathcal{D}_{train}) = U - \eta \nabla_U \mathcal{L}(\mathcal{M}(U), \mathcal{D}_{train})$$

for some $\eta \in \mathbb{R}$ and loss function \mathcal{L} mapping a function and a dataset to a loss $(\mathcal{X} \to \mathbb{R}) \times \mathcal{D} \to \mathbb{R}$.

Theorem C.2. Suppose $\mathcal{G}_{\mathcal{X}}$ is a group acting on \mathcal{X} . The gradient-based iterative algorithm \mathcal{A} (defined in Definition C.1) is $\mathcal{G}_{\mathcal{X}}$ -equivariant if:

- 1. There exists a group \mathcal{G}_{θ} acting on parameters U and a group isomorphism $\tau: \mathcal{G}_{\mathcal{X}} \to \mathcal{G}_{\theta}$ such that for all $x \in \mathcal{X}, T \in \mathcal{G}_{\mathcal{X}}, U$ we have that $g(U; x) = g(\tau(T)(U); T(x))$.
- 2. The gradient update rule is invariant under joint group action $(T, \tau(T))$ for all $T \in \mathcal{G}_{\mathcal{X}}$: $\tau(T)(\nabla_U q(U;x)) = \nabla_U q(\tau(T)(U);T(x))$.
- 3. The initialization distribution P_{init} is invariant under $\mathcal{G}_{\mathcal{X}}$.

We now present our permutation-equivariance results.

Definition C.3. We define \mathcal{G}_{θ} as a group of actions to be applied on U = (W, V) as

$$\mathcal{G}_{\theta} \triangleq \{ \pi_{\sigma_1, \sigma_2, \sigma_3} \mid \sigma_1, \sigma_2, \sigma_3 \in \mathbb{S}_p \}$$

where $\pi_{\sigma_1,\sigma_2,\sigma_3}$ is defined as $\pi_{\sigma_1,\sigma_2,\sigma_3}(W,V) \triangleq (W',V')$, where $\forall i \in [h], j \in [p], W'_{ij} = W_{i,\sigma_1(j)}, W'_{i(j+p)} = W_{i,\sigma_2(j)}, V'_{\sigma_3(j),i} = V_{j,i}$. This means the concatenation of the row is permuted in the same way as the data does. Furthermore, the rows of V also get permuted according to σ_3 .

Remark C.4. There is a one-to-one mapping τ between \mathcal{G}_{θ} and $\mathcal{G}_{\mathcal{X}}$, which is $\tau(\sigma_1, \sigma_2, \sigma_3) = \pi_{\sigma_1, \sigma_2, \sigma_3}$.

Lemma C.5. For every x = (i, j, k) where $i, j, k \in [p]$, $T \in \mathcal{G}_{\mathcal{X}}$ we have

$$g(U;x) = g(\tau(T)(U);T(x)).$$

Proof. For each $x=(i,j,k)\in\mathcal{X}; \sigma_1,\sigma_2,\sigma_3\in\mathbb{S}_p; U=(W,V)$ we have that

$$g(\tau(\sigma_{1}, \sigma_{2}, \sigma_{3})(U); (\sigma_{1}, \sigma_{2}, \sigma_{3})(x)) = \left\langle e_{\sigma_{3}(k)}, V'_{\sigma_{3}} \begin{pmatrix} \begin{bmatrix} W'_{1}^{\top}(\sigma_{1}(i), \sigma_{2}(j)) \\ W'_{2}^{\top}(\sigma_{1}(i), \sigma_{2}(j)) \\ \vdots \\ W'_{h}^{\top}(\sigma_{1}(i), \sigma_{2}(j)) \end{bmatrix} \right\rangle^{\odot 2}$$

$$= \left\langle e_{k}, V \begin{pmatrix} \begin{bmatrix} W'_{1,\sigma_{1}(i)} + W'_{1,\sigma_{2}(j)+p} \\ W'_{2,\sigma_{1}(i)} + W'_{2,\sigma_{2}(j)+p} \\ \vdots \\ W'_{h,\sigma_{1}(i)} + W'_{h,\sigma_{2}(j)+p} \end{bmatrix} \right\rangle^{\odot 2}$$

$$= \left\langle e_{k}, V \begin{pmatrix} \begin{bmatrix} W_{1,i} + W_{1,j+p} \\ W_{2,i} + W_{2,j+p} \\ \vdots \\ W_{h,i} + W_{h,j+p} \end{bmatrix} \right\rangle^{\odot 2}$$

$$= \left\langle e_{k}, V(Wx)^{\odot 2} \right\rangle$$

$$= g(U; x).$$

Lemma C.6. For every x = (i, j, k) where $i, j, k \in [p]$, $T \in \mathcal{G}_{\mathcal{X}}$ we have

$$\tau(T)(\nabla_U q(U;x)) = \nabla_U q(\tau(T)(U);T(x)).$$

Proof. We first consider the gradient of the second layer. For all $a, b \in [p]$ and $\sigma_1, \sigma_2, \sigma_3 \in \mathbb{S}_p$:

$$\tau^{-1}(\pi_{\sigma_{1},\sigma_{2},\sigma_{3}}) \left(\nabla_{V_{b}} g(\tau(\sigma_{1},\sigma_{2},\sigma_{3})(U); (\sigma_{1},\sigma_{2},\sigma_{3})(x)) \right)$$

$$= I(\sigma_{3}(k) = \sigma_{3}(b)) \nabla_{V_{b}} g\left(\tau(\sigma_{1},\sigma_{2},\sigma_{3})(U); (\sigma_{1},\sigma_{2},\sigma_{3})(x)\right)$$

$$= I(k = b)(W'(\sigma_{1},\sigma_{2})(x))^{\odot 2}$$

$$= I(k = b)(Wx)^{\odot 2}$$

$$= \nabla_{V_{b}} g(U; x).$$

For the gradients of the first layer we have:

$$\begin{split} &\tau^{-1}(\pi_{\sigma_{1},\sigma_{2},\sigma_{3}})\left(\nabla_{W}g(\tau(\sigma_{1},\sigma_{2},\sigma_{3})(U);(\sigma_{1},\sigma_{2},\sigma_{3})(x))\right)\\ &=\tau^{-1}(\pi_{\sigma_{1},\sigma_{2},\sigma_{3}})\left(2V_{k}\odot(W'_{(\sigma_{1},\sigma_{2})}(e_{\sigma_{1}(i)},e_{\sigma_{2}(j)})^{\top})\left(e_{\sigma_{1}(i)},e_{\sigma_{2}(j)}\right)\right)\\ &=\tau^{-1}(\sigma_{1},\sigma_{2},\sigma_{3})\left(2V_{k}\odot(W(e_{i},e_{j}))\left(e_{\sigma_{1}(i)},e_{\sigma_{2}(j)}\right)^{\top}\right)\\ &=2V_{k}\odot(Wx)\left(\sigma_{1},\sigma_{2}\right)^{-1}\left(\left(e_{\sigma_{1}(i)},e_{\sigma_{2}(j)}\right)\right)\\ &=2V_{k}\odot(W(e_{i},e_{j})^{\top})\left(e_{i},e_{j}\right)\\ &=\nabla_{W}g(U;x). \end{split}$$

Theorem 3.3 (Permuation Equivariance in Regression). *Training a two-layer neural network with quadratic activations (such as* $g(\theta; x)$) *initialized with random i.i.d. parameters using gradient-based update rules (such as gradient descent) on the modular addition problem in the regression setting is an equivariant learning algorithm (as defined in Definition 2.2) with respect to the permutation group of Definition 3.2 applied on the input data.*

Proof. We show that training a neural network with gradient descent in our setting satisfies the three conditions proposed in Theorem C.2 and thus, this theorem applies to the training process. Equivariance of the forward pass and the backward pass (update rule) are settled through Lemmas C.5 and C.6. Finally, it is straightforward to see that the initialization is invariant under \mathcal{G}_{θ} defined in Definition C.3. Since the distribution of initialization is symmetric and each parameter is initialized independently and identically from the same distribution, the action of swapping rows or columns doesn't change the distribution.

Proposition 4.2 (Permuation Equivariance in Classification). We define the permutation group $\mathcal{G}_{\mathcal{X},\mathcal{Y}}$ on $\mathcal{X} \times \mathcal{Y}$ (defined in Section 2) as the group

$$\left\{(e_a,e_b),e_c\mapsto (e_{\sigma_1(a)},e_{\sigma_2(b)}),e_{\sigma_3(c)}:\sigma_1,\sigma_2,\sigma_3\in\mathbb{S}_p\right\}.$$

Training a two-layer neural network with quadratic activations (such as $f(\theta;x)$) initialized with random i.i.d. parameters using gradient-based update rules (as defined in Definition C.1)⁸ on the modular addition problem in the classification setting is an equivariant learning algorithm (as defined in Definition 2.2) with respect to $\mathcal{G}_{X,Y}$.

Proof Sketch. Consider the permutation group \mathcal{G}_{θ} defined in Definition C.3. Note that there is a one-to-one mapping κ mapping $\mathcal{G}_{\mathcal{X},\mathcal{Y}}$ defined in Definition 3.2 to $\mathcal{G}_{\mathcal{X}}$ and another one-to-one mapping τ mapping $\mathcal{G}_{\mathcal{X}}$ to \mathcal{G}_{θ} where $\tau(\sigma_1, \sigma_2, \sigma_3) = \pi_{\sigma_1, \sigma_2, \sigma_3}$. Since f and g share the parameters, it's clear that the distribution of initialization for f is equivariant under \mathcal{G}_{θ} . To see the equivariance of forward and backward passes on $f(U; \cdot)$ it suffices to see that Lemmas C.5 and C.6 hold for $(\mathcal{G}_{\mathcal{X},\mathcal{Y}}, \mathcal{G}_{\theta})$ and function f under the isomorphism $\tau \circ \kappa$ since for any parameters U and inputs $i, j, k \in [p]$ it holds that $f_k(U; (i, j)) = g(U; (i, j, k))$.

Corollary C.7 (Equivariance of Adam). *Theorem 3.3* (and similarly Proposition 4.2) applies to other gradient-based training algorithms that need memory, such as Adam.

⁸GD is an example of such algorithm.

⁹A similar result holds for adaptive algorithms like Adam as well, as discussed in Corollary C.7.

Proof Sketch for Corollary C.7. To prove that other gradient-based algorithms, particularly Adam, are also equivariant to the permutation group $\mathcal{G}_{\mathcal{X}}$ we just need to ensure that the update rule of these algorithms is equivariant under the joint group action $(T, \tau(T))$. First, note that linear operations (such as weight decay) on gradients and parameters equivariant. To track the momentum at different steps of the algorithm we can apply an induction on equivariance of these variables on the step number. At t=0 they're both zero. m_t is a linear combination m_{t-1} and g_t which both are equivarint under the joint action $(T,\tau(T))$. Since the gradient is equivariant, the coordinate-wise squared gradient is also equivariant and the linear combination of it with v_{t_1} is also equivariant. This settles the equivariance of the update rule of Adam and other similar gradient-based algorithms that need memory and buffers.

D Lower Bound of Population Loss for Kernel Methods

In this section we present the formal version of Theorem 3.4 alongside a proof of it. Note that a kernel-based predictor h on a training data $\{(\boldsymbol{x}_i,y_i)\}_{i=1}^n$ can be expressed as $h(x) = \sum_{i=1}^n \lambda_i K(\boldsymbol{x}_i,x)$ where $\lambda_i; i \in [n]$ are constants. Assuming that the kernel's feature maps are of dimension d, the predictions are linear combinations of d-dimensional feature maps. We first present a general proof that every permutation invariant kernel requires $\Omega(p^2)$ training points to outperform the null predictor in terms of ℓ_2 loss, and then show that this theorem applies to the distribution of empirical NTKs at initialization.

D.1 Notation

We use [p] to denote the set $\{1,\ldots,p\}$. We use \mathbb{S}_p to denote the permutation groups over p elements, \mathbb{S}_m as permutation group over m elements and id is the identity mapping. For any nonempty set \mathcal{X} , a symmetric function $K:\mathcal{X}\times\mathcal{X}\to\mathbb{R}$ is called a positive semi-definite kernel (p.s.d) kernel on \mathcal{X} if for any $n\in\mathbb{N}$, any $\mathbf{x}_1,\ldots,\mathbf{x}_n$ and $\lambda_1,\ldots,\lambda_n\in\mathbb{R}$, it holds that $\sum_{i=1}^n\sum_{j=1}^n\lambda_i\lambda_jK(\mathbf{x}_i,\mathbf{x}_j)\geq 0$. For a subspace V of \mathbb{R}^n and vector $x\in\mathbb{R}^n$, we define $\mathrm{dist}(x,V)\triangleq \min_{v\in V}\|x-v\|_2$.

For any p^m -dimensional vector $v \in \mathbb{R}^{p^m}$ we denote by v(x) the vector v indexed by an m-dimensional index vector $x \in [p]^m$. We define the vector $s_{i,a} \in \mathbb{R}^{p^m}$ whose entries are

$$s_{i,a}(x) \triangleq \begin{cases} 1, & x[i] = a; \\ 0, & \text{otherwise.} \end{cases}$$
 (D.1)

which helps us denote the all-once slices in this vector space, $V_s \triangleq \operatorname{span}\{s_{i,a}\}_{i \in [m], a \in [p]}$. We further define $\Delta(x, x')$ where x, x' are two index vectors of size m as the number of equal indices between them, formally:

$$\Delta(x, x') = \sum_{i=1}^{m} \mathbf{1}_{x[i] = x'[i]}.$$

We also define $\mathcal{X}_{a,b}$ as the set of all $x, x' \in [p]^b$ index vector pairs such that $\operatorname{dist}(x, x') = a$, formally:

$$\mathcal{X}_{a,b} \triangleq \{(x, x') : x, x' \in [p]^b \land \Delta(x, x') = a\}.$$

When b=m, we drop the second index and write \mathcal{X}_a (instead of $\mathcal{X}_{a,m}$) for simplicity. It's clear that the collection of all \mathcal{X}_d for $0 \le d \le m$ is a partitioning of the set of all pairs of index vectors. Unless stated otherwise, x refers to the m-dimensional index vector. Finally, we define $\mathbb{U}_m \triangleq [\mathrm{Unif}(\mathbb{S}_p)]^m$ as the product of m Uniform distribution on \mathbb{S}_p .

D.2 Loss Lower Bound: Regression Setting

For convenience of notation, we will use the (i,j,k) and $e_i+e_{j+p}+e_{k+2p}$ interchangebly for $i,j,k\in[p]$. We denote the function $K(\boldsymbol{x}_t,\cdot):[p]\times[p]\times[p]\times[p]\to\mathbb{R}$ as a tensor on $\mathbb{R}^{p\times p\times p}$ by $v_t(\cdot)$ for each $t\in[n]$. We also define function $\Psi_{\sigma_1,\sigma_2,\sigma_3}(i,j,k)\triangleq\mathbf{1}\bigg(\sigma_1(i)+\sigma_2(j)\equiv\sigma_3(k)\pmod{p}\bigg)$. We can view a function mapping from $[p]\times[p]\times[p]\to\mathbb{R}$ as a vector of size p^3 and define inner products and dist on the function space, i.e., $\langle\Psi,\Psi'\rangle\triangleq\sum_{i,j,k\in[p]}\Psi(i,j,k)\Psi'(i,j,k)$ and $\|h\|_2^2\triangleq\langle h,h\rangle$.

Theorem D.1. For any integers $n \ge 1$, $p \ge 2$ and kernel $K : ([p] \times [p] \times [p]) \times ([p] \times [p]) \to \mathbb{R}$, for any $\mathbf{x}_t = (i_t, j_t, k_t) \in [p]^3$ for each $t \in [n]$, it holds that

$$\min_{\boldsymbol{x}_{1},\dots,\boldsymbol{x}_{n}} \mathbb{E}_{\sigma_{1},\sigma_{2},\sigma_{3} \sim \text{Unif}(\mathbb{S}_{p})} \inf_{\lambda_{1},\dots,\lambda_{n} \in \mathbb{R}} \left\| \sum_{t=1}^{n} \lambda_{t} K(\boldsymbol{x}_{t},\cdot) - \Psi_{\sigma_{1},\sigma_{2},\sigma_{3}}(\cdot) \right\|_{2}^{2} \geq p^{2} \left(1 - \frac{1}{p} - \frac{n}{p^{3}} \exp\left(\frac{2}{p-1}\right)\right) \tag{D.2}$$

In other words, if $n \leq (1 - \Omega(1))p^3$, then the expected population ℓ_2 loss is at least $\Omega(p^2)$, which is of the same magnitude as the trivial all-zero predictor.

Proof of Theorem D.1. This Theorem is a direct result of Corollary D.8 for the case where m=3.

Theorem 3.4 (Lower Bound). There exists a constant C > 0 such that for any $p \ge 2$, training data size $n < Cp^3$, and any permutation-equivariant kernel method A, it holds that

$$\mathbb{E}_{\left(\boldsymbol{x}_{i},y_{i}\right)_{i=1}^{n}\sim\mathcal{D}^{n}}\mathbb{E}\,\mathcal{L}_{\ell_{2}}\left(\mathcal{A}\left(\left\{\left(\boldsymbol{x}_{i},y_{i}\right)\right\}_{i=1}^{n}\right)\right)\geq\frac{w}{2}\,\,\mathcal{L}_{\ell_{2}}\left(\Psi_{\mathbf{0}}\right)=\frac{p}{2},$$

where $\mathbb{E}_{\mathcal{A}}$ takes the mean over the randomness in algorithm \mathcal{A} .

Proof of Theorem 3.4. Because A is permutation-equivariant, we have that

$$\begin{split} & \underset{(\boldsymbol{x}_{i},y_{i})_{i=1}^{n} \sim \mathcal{D}^{n}}{\mathbb{E}} \, \mathcal{L}_{\ell_{2}} \left(\mathcal{A} \left(\left\{ \left(\boldsymbol{x}_{i},y_{i} \right) \right\}_{i=1}^{n} \right) \right) \\ &= \underset{(\boldsymbol{x}_{i},y_{i})_{i=1}^{n} \sim \mathcal{D}^{n}}{\mathbb{E}} \, \mathbb{E} \, \| \mathcal{A} \left(\left\{ \left(\boldsymbol{x}_{i},y_{i} \right) \right\}_{i=1}^{n} \right) - p \cdot \Psi_{\mathsf{id},\mathsf{id},\mathsf{id}} \|^{2} / p^{3} \\ &= \underset{(\boldsymbol{x}_{i},y_{i})_{i=1}^{n} \sim \mathcal{D}^{n}}{\mathbb{E}} \, \mathbb{E} \, \| \mathcal{A} \left(\left\{ \left(\boldsymbol{x}_{i},y_{i} \right) \right\}_{i=1}^{n} \right) / p - \Psi_{\mathsf{id},\mathsf{id},\mathsf{id}} \|^{2} / p \\ &= \underset{(\boldsymbol{x}_{i},y_{i})_{i=1}^{n} \sim \mathcal{D}^{n}}{\mathbb{E}} \, \underset{\sigma_{1},\sigma_{2},\sigma_{3} \sim \mathsf{Unif}(\mathbb{S}_{p})}{\mathbb{E}} \, \mathcal{A} \, \left(\left\{ \left(\boldsymbol{x}_{i},y_{i} \right) \right\}_{i=1}^{n} \right) / p - \Psi_{\sigma_{1},\sigma_{2},\sigma_{3}} \|^{2} / p. \end{split}$$

Because A is a kernel method, we have for any $(x_i, y_i)_{i=1}^n$ and $\sigma_1, \sigma_2, \sigma_3 \in \mathbb{S}_p$

$$\left\|\mathcal{A}\left(\left\{\left(\boldsymbol{x}_{i},y_{i}\right)\right\}_{i=1}^{n}\right)/p - \Psi_{\sigma_{1},\sigma_{2},\sigma_{3}}\right\|^{2} \geq \inf_{\lambda_{1},\dots,\lambda_{n}\in\mathbb{R}} \left\|\sum_{t=1}^{n} \lambda_{t}K(\boldsymbol{x}_{t},\cdot) - \Psi_{\sigma_{1},\sigma_{2},\sigma_{3}}(\cdot)\right\|_{2}^{2}.$$

Applying Theorem D.1 completes the proof.

D.3 Loss Lower Bound: Classification Setting

For this subsection, we define the modular addition function $\Psi:[p]\times[p]\to[p]$ as $\left[\Psi^{\sigma_3}_{\sigma_1,\sigma_2}(i,j)\right]_k\triangleq\mathbf{1}\left(\sigma_1(i)+\sigma_2(j)\equiv\sigma_3(k)\pmod{p}\right)$ for all $k\in[p],\sigma_1,\sigma_2,\sigma_3\in\mathbb{S}_p$.

Theorem D.2. For any integers n > 1, $p \ge 2$ and input-output permutation equivariant kernel $K: ([p] \times [p]) \times ([p] \times [p]) \to \mathbb{R}^{p \times p}$ (according to Proposition 4.2), suppose $x_t = (i_t, j_t) \stackrel{i.i.d.}{\sim} \text{Unif}([p] \times [p])$ for each $t \in [n]$ it holds that

$$\mathbb{E}_{x_1,\dots,x_n \sigma_1,\sigma_2\sigma_3 \sim \text{Unif}(\mathbb{S}_p)} \mathbb{E}_{\lambda_1,\dots,\lambda_n \in \mathbb{R}^p} \inf_{x \in [p] \times [p]} \left\| \sum_{t=1}^n K(x_t,x) \lambda_t - \Psi_{\sigma_1,\sigma_2}^{\sigma_3}(x) \right\|_2^2 \ge p^2 \left(1 - \frac{1}{p} - \frac{n}{p^2} \exp\left(\frac{2}{p-1}\right) \right).$$

In other words, if $n \leq (1 - \Omega(1))p^2$, then the expected population ℓ_2 loss is at least $\Omega(p^2)$, which is of the same magnitude as the trivial all-zero predictor.

Proof. Note that

$$\mathbb{E}_{x_{1},\dots,x_{n}} \mathbb{E}_{\sigma_{1},\sigma_{2},\sigma_{3} \sim \text{Unif}(\mathbb{S}_{p})} \inf_{\lambda \in \mathbb{R}^{n \times p}} \sum_{x \in [p] \times [p]} \left\| \sum_{i=1}^{n} K(x_{i},x) \lambda_{i} - \Psi_{\sigma_{1},\sigma_{2}}^{\sigma_{3}}(x) \right\|_{2}^{2}$$

$$= \mathbb{E}_{x_{1},\dots,x_{n}} \mathbb{E}_{\sigma_{1},\sigma_{2} \sim \text{Unif}(\mathbb{S}_{p})} \inf_{\lambda \in \mathbb{R}^{n \times p}} \sum_{x \in [p] \times [p]} \sum_{j=1}^{p} \left(\sum_{i=1}^{n} \left\langle [K(x_{i},x)]_{j,:}, \lambda_{i} \right\rangle - \left[\Psi_{\sigma_{1},\sigma_{2}}^{\sigma_{3}}(x)\right]_{j} \right)^{2}$$

$$= \mathbb{E}_{x_{1},\dots,x_{n}} \mathbb{E}_{\sigma_{1},\sigma_{2},\sigma_{3} \sim \text{Unif}(\mathbb{S}_{p})} \inf_{\lambda \in \mathbb{R}^{n \times p}} \sum_{x \in [p] \times [p]} \sum_{j=1}^{p} \left(\sum_{i=1}^{n} \sum_{k=1}^{p} \lambda_{i,k} \left[K(x_{i},x) \right]_{j,k} - \left[\Psi_{\sigma_{1},\sigma_{2}}^{\sigma_{3}}(x)\right]_{j} \right)^{2}.$$
(D.3)

We define

$$K'((i,j,k),(i',j',k')) \triangleq K((i,j),(i',j'))_{k,k'},$$

$$\Psi'_{\sigma_{1},\sigma_{2},\sigma_{3}}((i,j,k)) \triangleq \left[\Psi^{\sigma_{3}}_{\sigma_{1},\sigma_{2}}((i,j))\right]_{k},$$

$$\lambda'_{t} \triangleq \lambda_{\lfloor t/p \rfloor, t \bmod p},$$

$$x'_{t} \triangleq (i_{\lfloor t/p \rfloor}, j_{\lfloor t/p \rfloor}, t \bmod p)$$
(D.4)

for any $i, j, k, i', j', k' \in [p]$, $\sigma_1, \sigma_2, \sigma_3 \in \mathbb{S}_p$, $t \in [np]$, $\lambda \in \mathbb{R}^{n \times p}$ and $\lambda' \in \mathbb{R}^{np}$. It can be seen that

$$\mathbb{E}_{\substack{x_1, \dots, x_n \\ \sigma_1, \sigma_2, \sigma_3 \sim \text{Unif}(\mathbb{S}_p)}} \inf_{\lambda \in \mathbb{R}^{n \times p}} \sum_{x \in [p] \times [p]} \sum_{j=1}^{p} \left(\sum_{i=1}^{n} \sum_{k=1}^{p} \lambda_{i,k} \left[K(x_i, x) \right]_{j,k} - \left[\Psi_{\sigma_1, \sigma_2}^{\sigma_3}(x) \right]_{j} \right)^{2}$$

$$\mathbb{E}_{\substack{x'_1, \dots, x'_{np} \\ \sigma_1, \sigma_2, \sigma_3 \sim \text{Unif}(\mathbb{S}_p)}} \inf_{\lambda' \in \mathbb{R}^{np}} \sum_{x \in [p] \times [p]} \sum_{j=1}^{p} \left(\sum_{i=1}^{np} \lambda'_i K'(x'_i, (x[0], x[1], i \mod p)) - \Psi'_{\sigma_1, \sigma_2, \sigma_3}(x) \right)^{2}.$$
(D.5)

This is similar to Theorem D.1 except that the underlying data is generated through sampling from independent groups each having a size of p (we sample i.i.d from $\mathrm{Unif}([p] \times [p])$ and for each sample we consider the set of all possible responses). Applying the result of Theorem D.1 yields

$$\mathbb{E}_{\substack{x'_1, \dots, x'_{np} \\ \sigma_1, \sigma_2, \sigma_3 \sim \text{Unif}(\mathbb{S}_p)}} \inf_{\lambda' \in \mathbb{R}^{np}} \sum_{x \in [p] \times [p]} \sum_{j=1}^{p} \left(\sum_{i=1}^{np} \lambda'_i K'(x'_i, (x[0], x[1], i \mod p)) - \Psi'_{\sigma_1, \sigma_2, \sigma_3}(x) \right)^2$$

$$\geq p^2 \left(1 - \frac{1}{p} - \frac{n}{p^2} \exp\left(\frac{2}{p-1}\right) \right) \tag{D.6}$$

which completes the proof.

Theorem 4.3 (Kernel Lower Bound). There exists a constant C > 0 such that for any training data size $n < Cp^2$ and any kernel method A which is input-output permutation-equivariant (with respect to $\mathcal{G}_{\mathcal{X},\mathcal{Y}}$), it holds that

$$\mathbb{E}_{\left(\boldsymbol{x}_{i},y_{i}\right)_{i=1}^{n} \sim \mathcal{D}^{n}} \mathbb{E}_{\mathcal{A}} \mathcal{L}_{\ell_{2}}\left(\mathcal{A}\left(\left\{\boldsymbol{x}_{i},y_{i}\right\}_{i=1}^{n}\right)\right) \geq \frac{1}{2} \mathcal{L}_{\ell_{2}}\left(\Psi_{\mathbf{0}}\right),$$

where $\mathbb{E}_{\mathcal{A}}$ takes expectation over the randomness in algorithm \mathcal{A} .

Proof of Theorem 4.3. Because \mathcal{A} is input-output permutation-equivariant, we have that

$$\begin{split} & \underset{(\boldsymbol{x}_{i},y_{i})_{i=1}^{n} \sim \mathcal{D}^{n}}{\mathbb{E}} \, \mathcal{L}_{\ell_{2}} \left(\mathcal{A} \big(\{ (\boldsymbol{x}_{i},y_{i}) \}_{i=1}^{n} \big) \right) \\ & = \underset{(\boldsymbol{x}_{i},y_{i})_{i=1}^{n} \sim \mathcal{D}^{n}}{\mathbb{E}} \, \underset{\mathcal{A}}{\mathbb{E}} \, \sum_{x \in [p] \times [p]} \left\| \mathcal{A} \big(\{ (\boldsymbol{x}_{i},y_{i}) \}_{i=1}^{n} \big) (x) - p \cdot \Psi_{\mathsf{id},\mathsf{id}}^{\mathsf{id}} (x) \right\|_{2}^{2} / p^{2} \\ & = \underset{(\boldsymbol{x}_{i},y_{i})_{i=1}^{n} \sim \mathcal{D}^{n}}{\mathbb{E}} \, \underset{\mathcal{A}}{\mathbb{E}} \, \sum_{x \in [p] \times [p]} \left\| \mathcal{A} \big(\{ (\boldsymbol{x}_{i},y_{i}) \}_{i=1}^{n} \big) (x) / p - \Psi_{\mathsf{id},\mathsf{id}}^{\mathsf{id}} (x) \right\|_{2}^{2} \\ & = \underset{(\boldsymbol{x}_{i},y_{i})_{i=1}^{n} \sim \mathcal{D}^{n}}{\mathbb{E}} \, \underset{\sigma_{1},\sigma_{2},\sigma_{3} \sim \mathsf{Unif}(\mathbb{S}_{p})}{\mathbb{E}} \, \underset{\mathcal{A}}{\mathbb{E}} \, \sum_{x \in [p] \times [p]} \left\| \mathcal{A} \big(\{ (\boldsymbol{x}_{i},y_{i}) \}_{i=1}^{n} \big) (x) / p - \Psi_{\sigma_{1},\sigma_{2}}^{\sigma_{3}} (x) \right\|_{2}^{2}. \end{split}$$

Because A is a kernel method, we have for any $(x_i, y_i)_{i=1}^n$ and $\sigma_1, \sigma_2, \sigma_3 \in \mathbb{S}_p$

$$\sum_{x \in [p] \times [p]} \left\| \mathcal{A} \left(\{ (\boldsymbol{x}_i, y_i) \}_{i=1}^n \right) (x) / p - \Psi_{\sigma_1, \sigma_2}^{\sigma_3}(x) \right\|_2^2 \ge \inf_{\lambda_1, \dots, \lambda_n \in \mathbb{R}} \sum_{x \in [p] \times [p]} \left\| \sum_{t=1}^n \lambda_t K(\boldsymbol{x}_t, x) - \Psi_{\sigma_1, \sigma_2}^{\sigma_3}(x) \right\|_2^2.$$

Applying Theorem D.2 completes the proof.

D.4 Loss Lower Bound: General Theorem For Arbitrary Functions

Lemma D.3. For any subspace V of \mathbb{R}^n and vector $x \in \mathbb{R}^n$, let $\{v_i\}_{i=1}^m$ be an orthonormal basis of V. It holds that $\operatorname{dist}^2(x,V) = \|x\|_2^2 - \sum_{i=1}^m \langle x, v_i \rangle^2$.

The proof of Lemma D.3 is straightforward and thus omitted.

Lemma D.4. For any distribution \mathcal{D} over functions mapping from $\mathcal{X} \to \mathbb{R}$ and n functions $\{k_i\}_{i=1}^n$ where $k_i : \mathcal{X} \to \mathbb{R}$ for each $i \in [n]$ it holds that

$$\mathbb{E}_{h \sim \mathcal{D}} \left[\min_{\alpha \in \mathbb{R}^n} \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} \left(\Psi(x) - \sum_{i=1}^n \alpha_i k_i(x) \right)^2 \right] \ge \frac{1}{|\mathcal{X}|} \sum_{i=n+1}^{|\mathcal{X}|} \lambda_i(\Sigma).$$

where $\Sigma_{\mathcal{D}}(x, x') \triangleq \mathbb{E}_{\Psi \sim \mathcal{D}} [\Psi(x) \Psi(x')]$ and λ_i denotes the *i'th* largest eigenvalue function. For notational convenience, we also view $\Sigma_{\mathcal{D}}$ as a $\mathbb{R}^{|\mathcal{X}| \times |\mathcal{X}|}$ matrix.

Proof of Lemma D.4. For each Ψ , we define $r_{\Psi} \in \mathbb{R}^{|\mathcal{X}|}$ whose entries are realization of the function Ψ on inputs $x \in \mathcal{X}$. It suffices to show that

$$\underset{h \sim \mathcal{D}}{\mathbb{E}} \left[\mathsf{dist}^2(r_{\Psi}, V) \right] \geq \underset{i=n+1}{\overset{|\mathcal{X}|}{\sum}} \lambda_i(\Sigma_{\mathcal{D}})$$

for any subspace $V = \mathrm{span}\{v_1, v_2, \cdots, v_n\} \subset \mathbb{R}^{|\mathcal{X}| \times |\mathcal{X}|}$ where v_i for $i \in [n]$ are orthonormal vectors. Note that

$$\mathbb{E}_{\Psi \sim \mathcal{D}} \left[\mathsf{dist}^{2}(r_{\Psi}, V) \right] = \mathbb{E}_{\Psi \sim \mathcal{D}} \left[\| r_{\Psi} \|_{2}^{2} - \sum_{t=0}^{n} \langle v_{t}, r_{\Psi} \rangle^{2} \right]$$

$$= \operatorname{Tr} \left(\mathbb{E}_{\Psi \sim \mathcal{D}} \left[\Psi \Psi^{\top} \right] \right) - \sum_{t=1}^{n} v_{t}^{\top} \left(\mathbb{E}_{\Psi \sim \mathcal{D}} \left[\Psi \Psi^{\top} \right] \right) v_{t}$$

$$\geq \operatorname{Tr} \left(\Sigma_{\mathcal{D}} \right) - \sum_{i=1}^{n} \lambda_{i} \left(\Sigma_{\mathcal{D}} \right)$$

$$= \sum_{i=n+1}^{|\mathcal{X}|} \lambda_{i} (\Sigma_{\mathcal{D}})$$
(D.7)

where the inequality in the second to last line is due to the min-max theorem (also called Courant–Fischer–Weyl min-max principle) (Courant and Hilbert, 1953). This completes the proof.

The following Corollary D.5 is a direct consequence of Lemma D.4, noting that $\mathbb{E}_{\Psi \sim \mathcal{D}} \left[\frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} \Psi(x)^2 \right] = \text{Tr}(\Sigma_{\mathcal{D}})$.

Corollary D.5. For any distribution \mathcal{D} over functions mapping from $\mathcal{X} \to \mathbb{R}$ and n functions $\{k_i\}_{i=1}^n$ where $k_i : \mathcal{X} \to \mathbb{R}$ for each $i \in [n]$, if $\sum_{i=1}^n \lambda_i(\Sigma_{\mathcal{D}}) \leq \frac{1}{2} \operatorname{Tr}(\Sigma_{\mathcal{D}})$ then it is guaranteed that

$$\mathbb{E}_{\Psi \sim \mathcal{D}} \left[\min_{\alpha \in \mathbb{R}^n} \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} \left(\Psi(x) - \sum_{i=1}^n \alpha_i k_i(x) \right)^2 \right] \ge \frac{1}{2} \mathbb{E}_{\Psi \sim \mathcal{D}} \left[\frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} \Psi(x)^2 \right]$$

where the right-hand side denotes the expected loss of the all-0 predictor.

Lemma D.6. For any matrix $\Sigma \in \mathbb{R}^{d \times d}$, subspace V and projection matrix $P_V \in \mathbb{R}^{d \times d}$ corresponding to V, it holds that

$$\sum_{i=1}^{n} \lambda_i(\Sigma) \le n \cdot \lambda_1 \bigg((I - P_V) \Sigma (I - P_V) \bigg) + \text{Tr} \left(P_V \Sigma P_V \right).$$

Proof. Let $V_n = \operatorname{span}\{\alpha_i\}_{i=1}^n$ where $\alpha_i \in \mathbb{R}^d$ is the *i*'th eigenvector of Σ and let P_{V_n} be its corresponding projection matrix. Let P_{V_n+V} be the projection onto the sum of two subspaces V_n+V , it holds that

$$\sum_{i=1}^{n} \lambda_{i}(\Sigma) = \operatorname{Tr}(P_{V_{n}} \Sigma P_{V_{n}})$$

$$\leq \operatorname{Tr}(P_{V_{n}+V} \Sigma P_{V_{n}+V})$$

$$= \operatorname{Tr}(P_{V} \Sigma P_{V}) + \operatorname{Tr}\left((P_{V_{n}+V} - P_{V}) \Sigma (P_{V_{n}+V} - P_{V})\right)$$

$$\leq \operatorname{Tr}(P_{V} \Sigma P_{V}) + n \cdot \lambda_{1} \left((P_{V_{n}+V} - P_{V}) \Sigma (P_{V_{n}+V} - P_{V})\right)$$

$$\leq \operatorname{Tr}(P_{V} \Sigma P_{V}) + n \cdot \lambda_{1} \left((I - P_{V}) \Sigma (I - P_{V})\right). \tag{D.8}$$

Here the second to last inequality is because $P_{V_n+V}-P_V$ is at most rank-n and so is $(P_{V_n+V}-P_V)\Sigma(P_{V_n+V}-P_V)$. The last inequality is because $P_{V_n+V}\leq I$ and thus $(P_{V_n+V}-P_V)\Sigma(P_{V_n+V}-P_V)\leq (I-P_V)\Sigma(I-P_V)$.

D.5 Loss Lower Bound: Modular Addition with m Summands

Lemma D.7. Let \mathcal{D} be the uniform distribution over

$$\mathcal{H} \triangleq \left\{ \Psi(x) = \mathbf{1} \left(\sum_{i=1}^{m} \sigma_i(x_i) \equiv 0 \pmod{p} \right) \middle| \sigma_i \in \mathbb{S}_p \text{ for all } i \in [m] \right\}$$
 (D.9)

and $\Sigma_{\mathcal{D}}(x, x') = \mathbb{E}_{\Psi \sim \mathcal{D}} [\Psi(x) \Psi(x')]$. It holds that

$$\sum_{i=1}^{n} \lambda_i(\Sigma) \le p^{m-2} + \frac{n}{p} \exp\left(\frac{m-1}{p-1}\right).$$

Proof of Lemma D.7. Consider the vector space V_s defined in Equation (D.1) and the projection matrix $P_{V_s} \in \mathbb{R}^{|\mathcal{X}| \times |\mathcal{X}|}$ onto V_S . To find $\operatorname{Tr} \left(P_{V_s} \Sigma P_{V_s} \right)$ we can first derive $\operatorname{Tr} \left(P_{V_s} \Psi \Psi^\top P_{V_s} \right)$ where $\Psi \in \mathbb{R}^{|\mathcal{X}|}$ is a sample from \mathcal{D} . Note that since $(I - P_{V_s})h$ should be orthogonal to V_s (the sum of each slice of the projected vector should be zero) we have that

$$[(I - P_{V_s})\Psi](x) = \begin{cases} -\frac{1}{p} & \Psi(x) = 0\\ \frac{p-1}{p} & \Psi(x) = 1 \end{cases}$$

Hence, $P_{V_s}\Psi(x)=\frac{1}{p}$ for all $x\in\mathcal{X}$. Thus,

$$\operatorname{Tr}\left(P_{V_s}\Sigma P_{V_s}\right) = \underset{\Psi\sim\mathcal{D}}{\mathbb{E}}\left[\operatorname{Tr}\left(P_{V_s}\Psi\Psi^{\top}P_{V_s}\right)\right]$$

$$= \underset{\Psi\sim\mathcal{D}}{\mathbb{E}}\left[\operatorname{Tr}\left(\frac{1}{p^2}\mathbf{1}_{|\mathcal{X}|\times|\mathcal{X}|}\right)\right]$$

$$= \frac{|\mathcal{X}|}{p^2}$$

$$= p^{m-2} \tag{D.10}$$

where $\mathbf{1}_{|\mathcal{X}|\times|\mathcal{X}|}$ denotes the all-one square matrix of size $|\mathcal{X}|$. Moreover, by Lemma D.9 we have that

$$n \cdot \lambda_1 \left((I - P_{V_s}) \Sigma (I - P_{V_s}) \right) = \sup_{\|v\|_2 \le 1, v \perp V_s} v^\top \Sigma_{\mathcal{D}} v \le \frac{n}{p} \exp\left(\frac{m - 1}{p - 1}\right). \tag{D.11}$$

Combining the two equations above, we can see that

$$\sum_{i=1}^{n} \lambda_i(\Sigma) \le p^{m-2} + \frac{n}{p} \exp\left(\frac{m-1}{p-1}\right).$$

This concludes the proof.

Corollary D.8. Consider the function class \mathcal{H} defined in Equation (D.9) and the uniform distribution over it denoted by \mathcal{D} . For any integers $n \geq 1, p \geq 2, 1 \leq m < p$, permutation-equivariant kernel $K: [p]^m \times [p]^m \to \mathbb{R}$ it holds that

$$\min_{x_1, x_2, \dots x_n} \mathbb{E} \inf_{\Psi \sim \mathcal{D}} \inf_{\alpha \in \mathbb{R}^n} \left\| \sum_{t=1}^n \lambda_t K(x_t, \cdot) - \Psi \right\|_2^2 \ge p^{m-1} \left(1 - \frac{1}{p} - \frac{n}{p^m} \exp\left(\frac{m-1}{p-1}\right) \right).$$

for any $x_1, x_2, \dots, x_n \in [p]^m$. In other words, if $n < (1 - \Omega(1))p^m$, then the expected population ℓ_2 loss is at least $\Omega(p^{m-1})$, which is of the same magnitude as the trivial all-zero predictor.

Proof of Corollary D.8. It suffices to show that for any n-dimensional subspace $V \subset \mathbb{R}^{p^m}$ it holds that

$$\underset{\Psi \sim \mathcal{D}}{\mathbb{E}} \operatorname{dist}^2(V, \Psi) \geq p^{m-1} \bigg(1 - \frac{1}{p} - \frac{n}{p^m - 1} \exp \bigg(\frac{m - 1}{p - 1} \bigg) \bigg).$$

Combining Lemma D.7 and Corollary D.5 yields this statement.

Lemma D.9. For any $v \in \mathbb{R}^{p^m}$ such that ||v|| = 1 and $v \perp V_s$ (defined in Equation (D.1)), it holds that

$$\underset{\sigma \sim \mathbb{U}_m}{\mathbb{E}} \left[\langle \Psi_{\sigma}, v \rangle^2 \right] \leq \frac{1}{p} \exp \left(\frac{m-1}{p-1} \right).$$

The main implication of this Lemma D.9 is that for any n-dimensional space $V \subset \mathbb{R}^{p^m}$ can not "cover" the vector space of all Ψ_{σ} functions for different permutations $\sigma \in \mathbb{S}_p$. To prove this Lemma, we decompose the inner product $\langle v, h_{\sigma} \rangle$ to d+1 sums using the Multinomial Theorem. This decomposition is enabled through observing the fact that there are d+1 equivalence groups in the possible set of indices of v. Based on Lemma D.10, we can use the Binomial Theorem to decompose the expectation of the inner product as follows

$$\mathbb{E}_{\sigma \sim \mathbb{U}_m} \left[\langle v, \Psi_{\sigma} \rangle^2 \right] = \mathbb{E}_{\sigma \sim \mathbb{U}_m} \left[\left(\sum_{x} v(x) \Psi_{\sigma}(x) \right)^2 \right] \\
= \mathbb{E}_{\sigma \sim \mathbb{U}_m} \left[\sum_{d=0}^m \sum_{\substack{x, x' \\ \text{dist}(x, x') = d}} \Psi_{\sigma}(x) \Psi_{\sigma}(x') v(x) v(x') \right] \\
= \sum_{d=0}^m C_d \sum_{\substack{x, x' \\ \text{dist}(x, x') = d}} v(x) v(x') \tag{D.12}$$

where $C_d \triangleq \mathbb{E}_{\sigma \sim \mathbb{U}_m} \left[\Psi_{\sigma}(x) \Psi_{\sigma}(x') \right]$ for any $(x, x') \in \mathcal{X}_d$.

To complete the proof, we now have to bound two terms. First, we need to show that the expectation $\mathbb{E}_{\sigma \sim \mathbb{U}_m} \left[\Psi_{\sigma}(x) \Psi_{\sigma}(x') \right]$ for each (x,x') in the same equivalence group is bounded. Next, we need to show that for each set \mathcal{X}_d , the sum $\sum_{(x,x')\in\mathcal{X}_d} v(x)v(x')$ is also bounded. These are correspondingly shown in Lemmas D.10 and D.11. Based on these two Lemmas, we can now present the proof of Lemma D.9.

Proof of Lemma D.9.

$$\mathbb{E}_{\sigma \sim \mathbb{U}_m} \left[\langle v, \Psi_{\sigma} \rangle^2 \right] = \sum_{d=0}^m C_d \sum_{\substack{x, x' \\ \text{dist}(x, x') = d}} v(x) v(x')$$

$$= \sum_{d=0}^m \frac{1}{p^2} \left(1 - \frac{1}{(1-p)^{d-1}} \right) (-1)^d \binom{m}{d}$$

$$= \frac{1}{p} \left(1 + \frac{1}{p-1} \right)^{m-1}$$

$$\leq \frac{1}{p} \exp\left(\frac{m-1}{p-1} \right).$$
(D.13)

where the second to last step is due to the binomial Theorem and the last step is due to the fact that for all $x \in \mathbb{R}$, $1 + x \le \exp(x)$.

Lemma D.10. For any index vector pair $(x, x') \in \mathcal{X}_d$ it holds that

$$\mathbb{E}_{\sigma \sim \mathbb{U}_m} \left[\Psi_{\sigma}(x) \Psi_{\sigma}(x') \right] = \frac{1}{p^2} \left(1 - \frac{1}{(1-p)^{d-1}} \right).$$

Proof of Lemma D.10. Let us re-iterate the definition of h:

$$\Psi_{\sigma}(x) = \mathbf{1} \bigg(\sum_{i=1}^{m} \sigma_i(x_i) \equiv 0 \pmod{p} \bigg).$$

For each pair (x', x') we define $I(x, x') \triangleq \{i : x[i] \neq x'[i]\}$ and $E(x, x') \triangleq \{i : x[i] = x'[i]\}$. We also define

$$C_d \triangleq \mathbb{E}_{\sigma \sim \mathbb{U}_m} \left[\Psi_{\sigma}(x) \Psi_{\sigma}(x') \right]$$

where for each $(x, x') \in \mathcal{X}_d$. Note that for d = 0, we have that

$$C_{0} = \underset{\sigma \sim \mathbb{U}_{m}}{\mathbb{E}} \left[\Psi_{\sigma}(x) \Psi_{\sigma}(x') \right]$$

$$= \underset{\sigma \sim \mathbb{U}_{m}}{\mathbb{E}} \left[\Psi_{\sigma}(x) \right]$$

$$= \underset{\sigma \sim \mathbb{U}_{m}}{\mathbb{E}} \left[\mathbf{1} \left(\sum_{i=1}^{m} \sigma_{i}(x_{i}) \equiv 0 \pmod{p} \right) \right]$$

$$= \underset{a \sim \mathrm{Unif}([p])}{\mathbb{E}} \left[a \equiv 0 \pmod{p} \right]$$

$$= 1/p. \tag{D.14}$$

For any $2 \le d \le m$ it holds that

$$C_{d} = \underset{\sigma \sim \mathbb{U}_{m}}{\mathbb{E}} \left[\mathbf{1} \left(\sum_{i=1}^{m} \sigma_{i}(x_{i}) \equiv \sum_{i=1}^{m} \sigma_{i}(x'_{i}) \equiv 0 \pmod{p} \right) \right]$$

$$= \underset{\sigma \sim \mathbb{U}_{m}}{\mathbb{E}} \left[\mathbf{1} \left(\sum_{i \in I(x, x')} \sigma_{i}(x_{i}) + \sum_{i \in E(x, x')} \sigma_{i}(x_{i}) \equiv \sum_{i \in I(x, x')} \sigma_{i}(x'_{i}) + \sum_{i \in E(x, x')} \sigma_{i}(x_{i}) \equiv 0 \pmod{p} \right) \right]$$

$$= \underset{\sigma \sim \mathbb{U}_{m}}{\mathbb{E}} \left[\mathbf{1} \left(\sum_{i \in I(x, x')} \sigma_{i}(x_{i}) \equiv \sum_{i \in I(x, x')} \sigma_{i}(x'_{i}) \equiv -\sum_{i \in E(x, x')} \sigma_{i}(x_{i}) \pmod{p} \right) \right]$$

$$= \underset{\zeta \sim \text{Unif}([p])}{\mathbb{E}} \left[\mathbf{1} \left(\sum_{i \in I(x, x')} \sigma_{i}(x_{i}) \equiv \sum_{i \in I(x, x')} \sigma_{i}(x'_{i}) \equiv \zeta \pmod{p} \right) \right]$$

$$= \underset{\zeta \sim \text{Unif}([p])}{\mathbb{E}} \left[\mathbf{1} \left(\sum_{i \in I(x, x')} \sigma_{i}(x_{i}) \equiv \sum_{i \in I(x, x')} \sigma_{i}(x'_{i}) \equiv \zeta \pmod{p} \right) \right]$$

$$(D.15)$$

where in the second to last line, we replaced $\mathbb{E}_{\sigma \sim \mathbb{U}_m} \left[-\sum_{i \in E(x,x')} \sigma_i(x_i) \right]$ with $\mathbb{E}_{\zeta \sim \mathrm{Unif}([p])}[\zeta]$ (assuming $d \geq 1$). We aim to find the closed form formula for the general $d \geq 2$. As $\zeta \sim \mathrm{Unif}([p])$

is independent of the two sums, we can absorb it and write (from here until the rest of the proof we drop \pmod{p} from equivalences for ease of presentation)

$$Q_d \triangleq \Pr_{(y,y') \sim \text{Unif}(\mathcal{X}_{d,d})} \left[\mathbf{1} \left(\sum_{i=1}^d y_i \equiv \sum_{i=1}^d y_i' \right) \right] = p \cdot C_d.$$
 (D.16)

Note that for $d \geq 2$ we have that

$$Q_{d} = \Pr_{(y,y') \in \text{Unif}(\mathcal{X}_{d,d})} \left[\mathbf{1} \left(\sum_{i=1}^{d-1} y_{i} \equiv \sum_{i=1}^{d-1} y'_{i} \right) \cdot \mathbf{1}(y_{d} = y'_{d}) \right]$$

$$+ \Pr_{(y,y') \in \text{Unif}(\mathcal{X}_{d,d})} \left[\mathbf{1} \left(\sum_{i=1}^{d-1} y_{i} \not\equiv \sum_{i=1}^{d-1} y'_{i} \right) \cdot \mathbf{1} \left(y_{d} \equiv y'_{d} + \sum_{i=1}^{d-1} y'_{i} - \sum_{i=1}^{d-1} y_{i} \right) \right]$$

$$= (1 - Q_{d-1}) \cdot \frac{1}{p-1}.$$
(D.17)

Hence for $d \geq 2$ we have that

$$C_d = \frac{1}{p(p-1)} (1 - p \cdot C_{d-1})$$

$$= \frac{1}{p(p-1)} - \frac{C_{d-1}}{p-1}$$

$$= \frac{1}{p^2} \left(1 - \frac{1}{(1-p)^{d-1}} \right).$$
 (D.18)

This completes the proof.

Lemma D.11. For any \mathcal{X}_d where $d \in [m]$ and $v \in \mathbb{R}^{p^m}$ such that $||v||_2 = 1$ and $v \perp V_s$ (defined in Equation (D.1)) it holds that

$$\sum_{(x,x')\in\mathcal{X}_d} v(x)v(x') = (-1)^d \binom{m}{d}.$$

Proof of Lemma D.11. Let us define G(d) as

$$G(d) \triangleq \frac{1}{\binom{m}{d}} \sum_{(x,x')\in\mathcal{X}_d} v(x)v(x')$$

$$= \underset{\sigma\in\mathbb{S}_m}{\mathbb{E}} \sum_{y\in[p]^{m-d}} \sum_{(t,t')\in\mathcal{X}_{d,d}} v(\sigma(y||t))v(\sigma(y||t')). \tag{D.19}$$

Since $v \perp V_s$ we have that

$$\mathbb{E}_{\sigma \in \mathbb{S}_{m}} \left[\sum_{y \in [p]^{d}} \sum_{(t,t') \in \mathcal{X}_{d-1,d-1}} \left(\sum_{s \in [p]} v(\sigma(y||s||t)) \right) \left(\sum_{s \in [p]} v(\sigma(y||s||t')) \right) \right]$$

$$= \mathbb{E}_{\sigma \in \mathbb{S}_{m}} \left[\sum_{(y||s) \in [p]^{m-d+1}} \sum_{(t,t') \in \mathcal{X}_{d-1,d-1}} v(\sigma(y||s||t)) v(\sigma(y||s||t')) \right]$$

$$+ \mathbb{E}_{\sigma \in \mathbb{S}_{m}} \left[\sum_{y \in [p]^{m-d}} \sum_{(s||t,s'||t') \in \mathcal{X}_{d,d}} v(\sigma(y||s||t')) v(\sigma(y||s'||t')) \right]$$

$$= G(d-1) + G(d) = 0. \tag{D.20}$$

Note that $G(0) = \sum_{x} v(x)^2 = ||v||_2^2 = 1$. Hence, $G(d) = (-1)^d$. This completes the proof.

E Generalization Upper Bound for Regression

The original model is $f\left(\left(e_{i},e_{j}\right)^{\top}\right)=V(W\left(e_{i},e_{j}\right)^{\top})^{\odot2}$. We consider the model $g\left(\left(e_{i},e_{j},e_{k}\right)^{\top}\right)=\left\langle e_{k},f\left(\left(e_{i},e_{j}\right)^{\top}\right)\right\rangle$ and the function class \mathcal{H} is defined over g with different weights $\theta=(W,V)$ where $W\in\mathbb{R}^{h\times2p}$ and $V\in\mathbb{R}^{p\times h}$. For an input $x\in\mathbb{R}^{3p}$, we define two slices $x'\triangleq x$ [: 2p] and $x''\triangleq x$ [2p:]. We also define $\mathcal{W}_{h,r}\triangleq\{W\in\mathbb{R}^{h\times2p}:\|W\|_{\infty}\leq r\}$ and $\mathcal{V}_{h,r}\triangleq\{V\in\mathbb{R}^{p\times h}:\|V\|_{\infty}\leq r\}$ which we will use later to denote parameters of our function. We further define $\mathcal{D}_{n}=\{x_{1},x_{2},\cdots,x_{n}\}$ such that for all $a\in[n]$ we have $x_{a}=\left(e_{i},e_{j},e_{k}\right)^{\top}$ for some $i,j,k\in[p]$. For this section, we fix the set $\{x_{1},x_{2},\cdots,x_{n}\}\sim\mathrm{Unif}(\mathcal{X})$ and denote by $\mathcal{R}_{n}(\mathcal{H})$ the empirical Rademacher complexity of the function class \mathcal{H} defined as

$$\mathcal{R}_n(\mathcal{H}) \triangleq \underset{\sigma \sim \text{Unif}(\{\pm 1\}^n)}{\mathbb{E}} \left[\sup_{\Psi \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^n \Psi(x_i) \sigma_i \right]$$
 (E.1)

where \mathcal{H} maps $\mathcal{X} \to \mathbb{R}$ and $n \in \mathbb{N}$.

Lemma E.1. Consider the function classes

$$\mathcal{H}_{r,r'}^{w} \triangleq \left\{ \Psi : \mathbb{R}^{3p} \to \mathbb{R} \mid \exists \left[W^{\top} \in \mathcal{W}_{w,r} \land V \in \mathcal{V}_{w,r'} \right] \text{ s.t. } \Psi(x) = \left\langle x'', V \left\langle W, x' \right\rangle^{2} \right\rangle \right\}$$
 (E.2)

and

$$\mathcal{G}_r \triangleq \left\{ g : \mathbb{R}^{3p} \to \mathbb{R} \mid \exists \left[U \in \mathbb{R}^{4 \times 3p} \land \|U\|_{\infty} \le r \right] \text{ s.t. } g(x) = \sum_{i=1}^{4} \langle U_i^\top, x \rangle^3 \right\}$$
 (E.3)

where U_i denotes the *i*'th row of U. The function class $\mathcal{H}^1_{r,r'}$ is contained in $\mathcal{G}_{\max(r,r')}$ (and hence $\mathcal{R}_n(\mathcal{H}_{r,r'}) \leq \mathcal{R}_n(\mathcal{G}_{\max(r,r')})$).

Proof of Lemma E.1. We prove this lemma by showing that for each pair of matrices W,V of $\Psi \in \mathcal{H}^1_{r,r'}$, we can construct a matrix U of $g \in \mathcal{G}_{\max(r,r')}$ such that for all $x \in \mathbb{R}^{3p}$, h(x) = g(x). Consider an arbitrary parameterization W,V of h such that $\Psi(x) = \left\langle x'',V\left\langle W,x'\right\rangle^2\right\rangle$. We can construct $U = \sqrt[3]{\frac{2}{9}} \left(Q_1,Q_2,Q_3,Q_4\right)^{\top}$ where

$$Q_1 = \begin{pmatrix} W \\ V \end{pmatrix}, \qquad Q_2 = \begin{pmatrix} -W \\ V \end{pmatrix}, \qquad Q_3 = \begin{pmatrix} -W/2 \\ V \end{pmatrix}, \qquad Q_4 = \begin{pmatrix} W/2 \\ -V \end{pmatrix}.$$
 (E.4)

Observe that

$$g(x) = \sum_{i=1}^{4} \left\langle U_{i}^{\top}, x \right\rangle^{3}$$

$$= \frac{2}{9} \left[\left\langle U_{1}, x \right\rangle^{3} + \left\langle U_{2}, x \right\rangle^{3} + \left\langle U_{3}, x \right\rangle^{3} + \left\langle U_{4}, x \right\rangle^{3} \right]$$

$$= \frac{2}{9} \left[\left(\left\langle W, x' \right\rangle + \left\langle V, x'' \right\rangle \right)^{3} - \left(\left\langle W, x' \right\rangle - \left\langle V, x'' \right\rangle \right)^{3} - \left(\frac{\left\langle W, x' \right\rangle}{2} + \left\langle V, x'' \right\rangle \right)^{3} + \left(\frac{\left\langle W, x' \right\rangle}{2} - \left\langle V, x'' \right\rangle \right)^{3} \right]$$

$$= \left\langle x'', V \left\langle W, x' \right\rangle^{2} \right\rangle$$

$$= \Psi(x). \tag{E.5}$$

It is straightforward to see that $\|U\|_{\infty} = \max(\|W\|_{\infty}, \|V\|_{\infty})$, which completes the proof. \square

Lemma E.2. Consider the function class \mathcal{G}_r defined in Equation (E.3). It holds that

$$\mathcal{R}_n(\mathcal{G}_r) \le \frac{324r^3\sqrt{p}}{\sqrt{n}}.$$

Proof of Lemma E.2. We can derive the Rademacher complexity of \mathcal{G}_r as

$$\mathcal{R}_{n}(\mathcal{G}_{r}) = \underset{\sigma \sim \text{Unif}(\{\pm 1\}^{n})}{\mathbb{E}} \left[\underset{U \in \mathbb{R}^{4 \times 3p}, \|U\|_{\infty} \leq r}{\sup} \frac{1}{n} \sum_{a=1}^{n} \sigma_{a} \sum_{c=1}^{4} \langle U_{c}^{\top}, x_{a} \rangle^{3} \right] \\
\leq \underset{\sigma \sim \text{Unif}(\{\pm 1\}^{n})}{\mathbb{E}} \left[\underset{U \in \mathbb{R}^{3p}, \|U\|_{\infty} \leq r}{\sup} \frac{4}{n} \sum_{a=1}^{n} \sigma_{a} \langle U, x_{a} \rangle^{3} \right] \\
\leq \underset{\sigma \sim \text{Unif}(\{\pm 1\}^{n})}{\mathbb{E}} \left[\underset{U \in \mathbb{R}^{3p}, \|U\|_{\infty} \leq r}{\sup} \frac{108r^{2}}{n} \sum_{a=1}^{n} \sigma_{a} \langle U, x_{a} \rangle \right]$$
(E.6)

where we used Talagrand's contraction lemma (Lemma E.3) using the fact that $f(x)=x^3$ is $27r^2$ -lipschitzness in [-3r,3r] and that $|\langle U,x_a\rangle|\leq 3r$ for all $\|U\|_{\infty}\leq r$, and we can continue:

$$\leq \underset{\sigma \sim \text{Unif}(\{\pm 1\}^n)}{\mathbb{E}} \left[\sup_{U \in \mathbb{R}^{3p}, \|U\|_{\infty} \leq r} \frac{108r^2}{n} \|U\|_2 \left\| \sum_{a=1}^n \sigma_a x_a \right\|_2 \right] \\
\leq \frac{108r^3 \sqrt{3p}}{n} \underset{\sigma \sim \text{Unif}(\{\pm 1\}^n)}{\mathbb{E}} \left[\left\| \sum_{a=1}^n \sigma_a x_a \right\|_2 \right] \\
\leq \frac{324r^3 \sqrt{p}}{\sqrt{n}}. \tag{E.7}$$

where the last step follows from Rademacher complexity of linear model and the fact that $\|x\|_2 = \sqrt{3}$ for all $x \in \mathcal{D}_n$ and $\mathbb{E}_{\sigma \sim \mathrm{Unif}(\{\pm 1\}^n)} \left[\left\| \sum_{a=1}^N \sigma_a \right\|_2 \right] \leq \sqrt{n}$.

Lemma E.3 (Talagrand's Contraction Lemma). Let \mathcal{H} be an arbitrary function class and g be an L-Lipschitz function. It holds that

$$\mathcal{R}_n(g \circ \mathcal{H}) \leq L \cdot \mathcal{R}_n(\mathcal{H})$$
.

A proof of this standard result can be found, for instance, as Lemma 5.7 of Mohri et al. (2018).

Remark E.4. Lemma E.2 directly implies the following Rademacher complexity bound for the function class $\mathcal{H}_{r,r'}^1$ defined in Equation (E.2):

$$\mathcal{R}_n(\mathcal{H}^1_{r,r'}) \le \frac{324\sqrt{p}}{\sqrt{n}} \max(r,r')^3.$$

Lemma E.5.

$$\mathcal{R}_n(\mathcal{H}_{r,r'}^h) \le \frac{324h\sqrt{p}}{\sqrt{n}} \max(r,r')^3.$$

Proof of Lemma E.5.

$$\mathcal{R}_{n}(\mathcal{H}_{r,r'}^{h}) = \underset{\sigma \sim \text{Unif}(\{\pm 1\}^{n})}{\mathbb{E}} \left[\sup_{\Psi \in \mathcal{H}_{r,r'}^{h}} \frac{1}{n} \sum_{i=1}^{n} \sigma_{i} \Psi(x_{i}) \right] \\
= \underset{\sigma \sim \text{Unif}(\{\pm 1\}^{n})}{\mathbb{E}} \left[\sup_{W \in \mathcal{W}_{h,r}, V \in \mathcal{V}_{h,r'}} \frac{1}{n} \sum_{a=1}^{n} \sigma_{a} \left\langle e_{k_{a}}, V\left(W\left(e_{i_{a}}\right)\right)^{\odot 2} \right\rangle \right] \\
= \underset{\sigma \sim \text{Unif}(\{\pm 1\}^{n})}{\mathbb{E}} \left[\sup_{W \in \mathcal{W}_{h,r}, V \in \mathcal{V}_{h,r'}} \frac{1}{n} \sum_{a=1}^{n} \sigma_{a} \left\langle e_{k_{a}}, \sum_{b=1}^{h} V_{:,b} \left(W_{b}\left(e_{i_{a}}\right)\right)^{\odot 2} \right\rangle \right] \\
\leq \underset{\sigma \sim \text{Unif}(\{\pm 1\}^{n})}{\mathbb{E}} \left[\sup_{W \in \mathcal{W}_{1,r}, V \in \mathcal{V}_{1,r'}} \frac{h}{n} \sum_{a=1}^{n} \sigma_{a} \left\langle e_{k_{a}}, V\left\langle W, \left(e_{i_{a}}\right)\right\rangle^{2} \right\rangle \right] \\
= h \mathcal{R}_{n}(\mathcal{H}_{r\,r'}^{1}) \tag{E.8}$$

where in the second to last line we used the fact that the Rademacher complexity of a two-layer NN with h hidden neurons is bounded by h times that of a single-hidden-neuron counterpart. Thus, applying Remark E.4 we can conclude that

$$\mathcal{R}_n(\mathcal{H}_{r,r'}^h) \le \frac{324h\sqrt{p}}{\sqrt{n}}\max(r,r')^3.$$

We are now ready to present the proof of the sample complexity upper bound for one-hidden-layer networks in the regression task. We first present the following Theorem from Srebro et al. (2010) on bounding the excess risk of H-smooth loss functions.

Theorem E.6 (Theorem 1 from Srebro et al. (2010)). For an H-smooth non-negative loss ℓ such that for all $x, y, \Psi, |\ell(\Psi(x), y)| \leq b$, for any $\delta > 0$ we have with probability at least $1 - \delta$ over a random sample size of n, for any $\Psi \in \mathcal{H}$,

$$L(\Psi) \leq \hat{L}(\Psi) + K\left(\sqrt{\hat{L}(\Psi)}\left(\sqrt{H}\log^{1.5} n\mathcal{R}_n(\mathcal{H}) + \sqrt{\frac{b\log(1/\delta)}{n}}\right) + H\log^3 n\mathcal{R}_n^2(\mathcal{H}) + \frac{b\log(1/\delta)}{n}\right)$$

where K is a positive constant, $L(\Psi)$ denotes the population loss of Ψ according to ℓ and $\hat{L}(\Psi)$ denotes the loss of Ψ on the mentioned sample of size n according to ℓ .

We now present the proof of Proposition 3.7.

Proposition 3.7. For any $R > 0, \delta \in (0,1)$, \mathcal{D}_{train} of size n, and $\theta^* \in \{\theta = (W,V) : \mathcal{L}_{\ell_2}(g,\theta,\mathcal{D}_{train}) = 0 \land \|\theta\|_{\infty} \leq R\}$, there exists a positive constant C > 0 such that with probability at least $1 - \delta$ over the randomness of \mathcal{D}_{train} ,

$$\mathcal{L}_{\ell_2}(g, \theta^*) \le \frac{CR^6h^2}{n} \left(p\log^3 n + \log\frac{1}{\delta}\right).$$

Proof. Consider the function class $\mathcal{H}_{R,R}^h$ for whom we have already proved a Rademacher complexity upper bound. As $\|\theta\|_{\infty} \leq R$, and all the inputs are one-hot, for all $x = (e_i, e_j, e_k)$ and $g \in \mathcal{H}_{R,R}^h$

it holds that $g(x) \le 4hR^3$. This boundedness accordingly implies smoothness of ℓ_2 loss on this function class with H=1. Hence, Theorem E.6 directly applies to our function class, yielding:

$$\mathcal{L}_{l_2}(g(\theta,\cdot)) \le \frac{CR^6h^2}{n} \left(p\log^3 n + \log(1/\delta)\right)$$

for some positive constant C independet of other values.

Finally, we remark that if the ℓ_2 loss is small enough, then the misclassification error is guaranteed to be zero.

Proposition E.7. Consider a predictor $g: \mathcal{X} \to \mathbb{R}$. The population misclassification error is upper-bounded by $2\mathcal{L}_{\ell_2}(g)/p$.

Proof. Note that each $(x,y) \in \mathcal{X} \times \mathcal{Y}$ that is misclassified induces an ℓ_2 loss of at least $p^2/2$. To see that why, for each pair (e_a, e_b) to be misclassified while attaining minimum possible ℓ_2 loss we need

$$g((e_a, e_b, e_c)) < \frac{p}{2}$$

$$g((e_a, e_b, e_d)) > \frac{p}{2}$$

$$g((e_a, e_b, e_k)) = 0 \quad \text{for all } k \notin \{c, d\}$$

where $c = a + b \pmod{p}$, $d \in [p] \neq c$ and $k \in [p]$. Hence, each of (e_a, e_b, e_c) and (e_a, e_b, e_d) introduce an ℓ_2 loss of at least $p^2/4$ in the regerssion task.

F Construction of Interpolating Solution with Small ℓ_{∞} Norm

In this section, we prove Proposition 3.8. We present a construction of weights that interpolates the dataset for h=8p. Then we generalize this result to any $h\geq 8p$ by duplicating the first 8p neurons $\left\lfloor \frac{h}{8p} \right\rfloor$ times, where each copy is $\left\lfloor \frac{h}{8p} \right\rfloor^{-\frac{1}{3}}$ times smaller in magnitude.

Proposition 3.8. Let the set of models with zero population loss be $\Theta^* \triangleq \{\theta \mid \mathcal{L}_{\ell_2}(g,\theta) = 0\}$. For any $p \geq 2$ and $h \geq 8p$, Θ^* is nonempty and $\min_{\theta \in \Theta^*} \|\theta\|_{\infty} \leq \left\lfloor \frac{h}{8p} \right\rfloor^{-\frac{1}{3}}$.

Proof of Proposition 3.8. We begin by constructing 8 matrices of size $p \times 2p$ denoted by $W^{(i)}$. For every $n, m \in [p]$, we have that

$$W_{k,n}^{(1)} = \cos\left(\frac{2\pi k}{p}n\right) \qquad W_{k,m+p}^{(1)} = \cos\left(\frac{2\pi k}{p}m\right)$$

$$W_{k,n}^{(2)} = \cos\left(\frac{2\pi k}{p}n\right) \qquad W_{k,m+p}^{(2)} = -\cos\left(\frac{2\pi k}{p}m\right)$$

$$W_{k,n}^{(3)} = \sin\left(\frac{2\pi k}{p}n\right) \qquad W_{k,m+p}^{(3)} = \sin\left(\frac{2\pi k}{p}m\right)$$

$$W_{k,n}^{(4)} = \sin\left(\frac{2\pi k}{p}n\right) \qquad W_{k,m+p}^{(4)} = -\sin\left(\frac{2\pi k}{p}m\right)$$

$$W_{k,n}^{(5)} = \sin\left(\frac{2\pi k}{p}n\right) \qquad W_{k,m+p}^{(5)} = \cos\left(\frac{2\pi k}{p}m\right)$$

$$W_{k,n}^{(6)} = \sin\left(\frac{2\pi k}{p}n\right) \qquad W_{k,m+p}^{(6)} = -\cos\left(\frac{2\pi k}{p}m\right)$$

$$W_{k,n}^{(7)} = \cos\left(\frac{2\pi k}{p}n\right) \qquad W_{k,m+p}^{(7)} = \sin\left(\frac{2\pi k}{p}m\right)$$

$$W_{k,n}^{(8)} = -\cos\left(\frac{2\pi k}{p}n\right) \qquad W_{k,m+p}^{(8)} = \sin\left(\frac{2\pi k}{p}m\right)$$

$$W_{k,n}^{(8)} = -\cos\left(\frac{2\pi k}{p}n\right) \qquad W_{k,m+p}^{(8)} = \sin\left(\frac{2\pi k}{p}m\right)$$

$$(F.2)$$

Each $W^{(i)}$ for $1 \leq i \leq 8$ is a $p \times 2p$ matrix, whose elements are given by the equations presented above. Hence, in each equation $k, n, m \in \{0, 1, \cdots, p\}$. The construction of the first layer is based on stacking $W^{(i)}$ for $1 \leq i \leq 8$ to construct $W \in \mathbb{R}^{8p \times 2p}$. The weights of the second layer are given in Equation (F.2), where $V_{q,\,8k:8(k+1)}$ presents a slice of the second layer and $q,k \in \{0,1,\cdots,p-1\}$.

To show that this construction solves the modular addition problem analytically, we will analytically perform the inference step for two arbitrary inputs n, m where $x = (e_n, e_m)$. We denote $h = (Wx)^{\odot 2} \in \mathbb{R}^{8p}$ as the post-activations of the first layer, which is given by

$$h_{8k:8(k+1)} = \begin{pmatrix} \cos\left(\frac{2\pi k}{p}n\right) + \cos\left(\frac{2\pi k}{p}m\right) \\ \cos\left(\frac{2\pi k}{p}n\right) - \cos\left(\frac{2\pi k}{p}m\right) \\ \sin\left(\frac{2\pi k}{p}n\right) + \sin\left(\frac{2\pi k}{p}m\right) \\ \sin\left(\frac{2\pi k}{p}n\right) - \sin\left(\frac{2\pi k}{p}m\right) \\ \sin\left(\frac{2\pi k}{p}n\right) + \cos\left(\frac{2\pi k}{p}m\right) \\ \sin\left(\frac{2\pi k}{p}n\right) - \cos\left(\frac{2\pi k}{p}m\right) \\ \cos\left(\frac{2\pi k}{p}n\right) + \sin\left(\frac{2\pi k}{p}m\right) \\ \cos\left(\frac{2\pi k}{p}n\right) - \sin\left(\frac{2\pi k}{p}m\right) \end{pmatrix}$$
(F.3)

Note that for each k, we have that (after dropping (e_n, e_m) for simplicity)

$$h_{8k} - h_{8k+1} = 2\cos\left(\frac{2\pi k}{p}(n+m)\right) + 2\cos\left(\frac{2\pi k}{p}(n-m)\right)$$
 (F.4)

and

$$h_{8k+2} - h_{8k+3} = 2\cos\left(\frac{2\pi k}{p}(n-m)\right) - 2\cos\left(\frac{2\pi k}{p}(n+m)\right)$$
 (F.5)

and

$$h_{8k+4} - h_{8k+5} = 2\sin\left(\frac{2\pi k}{p}(n+m)\right) + 2\sin\left(\frac{2\pi k}{p}(n-m)\right)$$
 (F.6)

and

$$h_{8k+6} - h_{8k+7} = 2\sin\left(\frac{2\pi k}{p}(n+m)\right) - 2\sin\left(\frac{2\pi k}{p}(n-m)\right).$$
 (F.7)

Hence,

$$h_{8k} - h_{8k+1} - h_{8k+2} + h_{8k+3} = 4\cos\left(\frac{2\pi k}{p}(n+m)\right)$$
 (F.8)

and

$$h_{8k+4} - h_{8k+5} + h_{8k+6} - h_{8k+7} = 4\sin\left(\frac{2\pi k}{p}(n+m)\right).$$
 (F.9)

Using the fact that $\cos(a-b) = \cos(a)\cos(b) - \sin(a)\sin(b)$, we can see that

$$[f(e_n, e_m)]_q = \langle V_{q,:}, h \rangle = 4 \sum_{k=0}^{p-1} \cos\left(\frac{2\pi k}{p}q\right) \cos\left(\frac{2\pi k}{p}(n+m)\right) - \sin\left(\frac{2\pi k}{p}q\right) \sin\left(\frac{2\pi k}{p}(n+m)\right)$$

$$= 4 \sum_{k=0}^{p-1} \cos\left(\frac{2\pi k}{p}(m+n-q)\right)$$

$$= 4p \mathbf{1}((m+n-q) \bmod p = 0)$$
(F.10)

where the last equality follows from Euler's identity and needs p to be odd.

Remark F.1. Assuming p is odd, we need at most 4p hidden neurons to interpolate the modular addition task.

Observing the fact that $\cos(2\pi - a) = \cos(a)$, we can see that

$$\sum_{k=0}^{p-1} \cos\left(\frac{2\pi k}{p}(m+n-q)\right) = 1 + 2\sum_{k=1}^{\frac{p-1}{2}} \cos\left(\frac{2\pi k}{p}(m+n-q)\right)$$
 (F.11)

where we replaced $\cos\left(\frac{2\pi 0}{p}(m+n-q)\right)$ with 1. Based on Equation (F.11), we can cut out half of the weights of the first and second layer, and only construct the frequencies up to $\frac{p-1}{2}$, which results in only needing 4p hidden neurons to construct the interpolating solution.

G Margin-Based Generalization Bound for Classification

We begin by providing some background and notation on sub-exponential random variables, which will be later used in the proof of our margin-based generalization bound.

G.1 Background on sub-exponential variables

The following proofs rely heavily on concentration inequalities for sub-exponential random variables; we will first review some background on these quantities.

A real-valued random variable X with mean μ is called *sub-exponential* (see e.g. Wainwright, 2019) if there are non-negative parameters (ν, α) such that

$$\mathbb{E}[e^{\lambda(X-\mu)}] \le e^{\frac{\nu^2 \lambda^2}{2}} \quad \text{for all } |\lambda| < \frac{1}{\alpha}. \tag{G.1}$$

We use $X \sim SE(\nu, \alpha)$ to denote that X is a sub-exponential random variable with parameters (ν, α) , but note that this is not a particular distribution.

One famous sub-exponential random variable is the product of the absolute value of two standard normal distributions, $z_i \sim \mathcal{N}(0,1)$, such that the two factors are either independent $(X_1 = |z_1||z_2| \sim SE(\nu_p,\alpha_p)$ with mean $2/\pi$) or the same $(X_2 = z^2 \sim SE(2,4)$ with mean 1). We now present a few lemmas regarding sub-exponential random variables that will come in handy in the later subsections of the appendix.

Lemma G.1. Assume X is sub-exponential with parameters (ν, α) . It holds that the random variable sX where $s \in \mathbb{R}^+$ is also sub-exponential, but with parameters $(s\nu, s\alpha)$.

Proof. Consider $X \sim SE(\nu, \alpha)$ and X' = sX with $\mathbb{E}[X'] = s\mathbb{E}[X]$. Based on the definition of sub-exponential random variables

$$\mathbb{E}\left[\exp\left(\lambda(X-\mu)\right)\right] \leq \exp\left(\frac{\nu^2\lambda^2}{2}\right) \quad \text{for all } |\lambda| < \frac{1}{\alpha}$$

$$\Longrightarrow \mathbb{E}\left[\exp\left(\frac{\lambda}{s}(sX-s\mu)\right)\right] \leq \exp\left(\frac{\nu^2s^2\frac{\lambda^2}{s^2}}{2}\right) \quad \text{for all } |\frac{\lambda}{s}| < \frac{1}{s\alpha}$$

$$\stackrel{\lambda'=\frac{\lambda}{s}}{\Longrightarrow} \mathbb{E}\left[\exp\left(\lambda'(X'-\mu')\right)\right] \leq \exp\left(\frac{\nu^2s^2\lambda'^2}{2}\right) \quad \text{for all } |\lambda'| < \frac{1}{s\alpha}$$
(G.2)

Defining $\alpha' = s\alpha$ and $\nu' = s\nu$ we see that $X' \sim SE(s\nu, s\alpha)$.

Proposition G.2. If all of the random variables X_i for $i \in [N]$ for $N \in \mathbb{N}^+$ are sub-exponential with parameters (ν_i, α_i) , and all of them are independent, then $\sum_{i=1}^N X_i \in SE(\sqrt{\sum_{i=1}^N \nu_i^2}, \max_i \alpha_i)$, and $\frac{1}{N} \sum_{i=1}^N X_i \sim SE\left(\frac{1}{\sqrt{N}}\sqrt{\frac{1}{N}\sum_{i=1}^N \nu_i^2}, \frac{1}{N}\max_i \alpha_i\right)$.

Proof. This is a simplification of the discussion prior to equation 2.18 of Wainwright (2019).

Proposition G.3. For a random variable $X \sim SE(\nu, \alpha)$, the following concentration inequality holds:

$$\Pr(|X - \mu| \ge t) \le 2 \exp\left(-\min\left(\frac{t^2}{2\nu^2}, \frac{t}{2\alpha}\right)\right).$$

Proof. The proof is straightforward from multiplying the result derived in Equation 2.18 of Wainwright (2019) by a scalar. \Box

Corollary G.4. Consider $X \sim SE(\nu, \alpha)$, the following bound holds with probability at least $1 - \delta$:

$$|X - \mu| < \max\left(\nu\sqrt{2\log\frac{2}{\delta}}, 2\alpha\log\frac{2}{\delta}\right).$$

A sub-Gaussian random variable, $SG(\nu)$, is one which satisfies (G.1) for all λ , i.e. it is the limit of $SE(\nu, \alpha)$ as $\alpha \to 0$.

Proposition G.5 (Chernoff bound). If X is $SG(\nu)$, then with probability at least $1 - \delta$, $|X - \mu| \le \nu \sqrt{2 \log \frac{2}{\delta}}$.

Proposition G.6 (Hoeffding's inequality). If X_1, \ldots, X_n are independent variables with means μ_i and each $SG(\nu_i)$, then $|\sum_{i=1}^n X_i - \sum_{i=1}^n \mu_i| \le \sqrt{2\left(\sum_{i=1}^n \nu_i^2\right)\log\frac{2}{\delta}}$ with probability at least $1 - \delta$.

G.2 Generalization Bound

We are now ready to state the main theorem for proving an upper bound on the number of training points needed to generalize. We begin by defining some notations and operators that will be useful in the main proof. First, we define $\mathbf{0}_{a \times b}$ to be the zero matrix (or vector in case its one-dimensional) of shape $a \times b$.

Definition G.7. Assume $p \ge 2$ is an integer. We define $\Theta_r^{h,b}$ as the set of possible parameters of one-hidden layer quadratic networks of width h whose ℓ_{∞} norm is bounded by r and have b output logits. Formally,

$$\Theta_r^{h,b} \triangleq \left\{ (W,V) \mid W \in \mathbb{R}^{h \times 2p}, V \in \mathbb{R}^{b \times h}, \|V\|_{\infty} \le r, \|W\|_{\infty} \le r \right\}.$$

We also define $\mathcal{M}_r^h \triangleq \left\{ (W,V) \in \Theta_r^h \mid \forall i \in [p]; \sum_{j=1}^h V_{ij} = 0 \right\}$ as the parameters whose second layer weights' rows have a zero sum.

Definition G.8. We next define an operator for adding gaussian noise to input matrices or vectors:

$$\Lambda_{\sigma}(A) = A + \tilde{A}$$

where A is any matrix or vector and \tilde{A} has the same shape as A except its entries are i.i.d sampled from the Gaussian distribution $\mathcal{N}(0, \sigma^2)$.

Our proof relies on Lemma 1 from Neyshabur et al. (2018). For convenience, we re-iterate this lemma here.

Lemma G.9 (Lemma 1 from (Neyshabur et al., 2018)). Let $f(\theta;\cdot)$ be a predictor $\mathcal{X} \to \mathbb{R}^K$ for some integer K>0 with parameters θ and P be any distribution on parameters that is independent of the training data. For any $\gamma, \delta > 0$, it holds with probability at least $1-\delta$ over randomness of training for any θ and any distribution on parameters \mathcal{P}_{θ} such that $\Pr_{u\sim\mathcal{P}_{\theta}}\left[\max_{x\in\mathcal{X},k\in[K]}\left|f_k(\theta+u;x)-f_k(\theta;x)\right|<\gamma/4\right]>1/2$

$$L_0(f, \theta, \mathcal{D}) \le L_{\gamma}(f, \theta, \mathcal{D}_{\text{train}}) + 4\sqrt{\frac{\text{KL}(\theta + u||P) + \log \frac{6m}{\delta}}{m - 1}}$$

where \mathcal{D} denotes the population.

The following is main Theorem for this section.

Theorem 4.6. For any $p \geq 2$, training set size $n \geq 1$, $\delta \in (0,1)$, norm r > 0, width $h' > 4\log\frac{2}{\delta}$ and normalized margin $\gamma/r^3 = \tilde{\Omega}(\sqrt{h})$ it holds with probability at least $1 - \delta$ over the randomness of the training set $\mathcal{D}_{\text{train}}$ that for any θ' of width h' with $\|\theta'\|_{\infty} \leq r$

$$L_0(f, \theta', \mathcal{D}) \le L_{\gamma}(f, \theta', \mathcal{D}_{\text{train}}) + \tilde{\mathcal{O}}\left(\sqrt{\frac{p}{n}} \cdot \sqrt[3]{\frac{h^2}{\gamma/r^3}}\right)$$
 (4.4)

where \mathcal{D} denotes the population and f, L are defined in Equations (2.1) and (4.3) respectively.

Proof. Let us first construct $\theta = (W, V) \in \mathcal{M}_r^{h,p}$ where h = 2h' from $\theta' = (W', V')$ such that $W = \begin{bmatrix} W' \\ W' \end{bmatrix}$ and $V = \begin{bmatrix} V' & -V' \end{bmatrix}$. This network has the same outputs as the original one with parameters θ' , while each row in V has a zero sum (and hence $\theta \in \mathcal{M}_r^{h,p}$). Since the outputs of the network with parameters θ are the same as those of θ' , any generalization bound applying to parameters θ also applies to the parameters θ' . Note that

$$f_{c}\Big(\big(\Lambda_{\sigma}(V),\Lambda_{\sigma}(W)\big),(e_{a},e_{b})\Big) = (V_{c} + \tilde{V}_{c})^{\top}\Big(\big(W + \tilde{W}\big)(e_{a},e_{b})\Big)^{\odot 2}$$

$$= \underbrace{f_{c}(\theta,(e_{a},e_{b})) + V_{c}\Big(\tilde{Q}^{\odot 2} + 2Q\odot\tilde{Q}\Big)}_{\text{Lemma G.11}} + \underbrace{\tilde{V}_{c}(Q + \tilde{Q})^{\odot 2}}_{\text{Lemma G.13}}. \quad (G.3)$$

where we denoted $\tilde{V} = \Lambda_{\sigma}(V) - V$, $\tilde{W} = \Lambda_{\sigma}(W) - W$, $Q = (W_a + W_b)$ and $\tilde{Q} = (\tilde{W}_a + \tilde{W}_b)$. As noted in the inequality, we can apply Lemmas G.11 and G.13 to show that for any $\delta_1 \in (0,1)$ with probability at least $1 - \delta_1$ over the randomness of perturbation it holds that

$$\left| f_c \left(\left(\Lambda_{\sigma}(V), \Lambda_{\sigma}(W) \right), (e_a, e_b) \right) - f_c \left(\theta, (e_a, e_b) \right) \right|$$

$$\leq 16 \sqrt{2h \log \frac{2}{\delta_1}} \max \left(r^2 \sigma, r \sigma^2 \right) + 32 \sqrt{2h} \left(\log \frac{2(h+1)}{\delta_1} \right)^{3/2} \max(\sigma^3, r \sigma^2, r^2 \sigma)$$

$$\leq 64 \sqrt{2h} \left(\log \frac{2(h+1)}{\delta_1} \right)^{3/2} \max(\sigma^3, r \sigma^2, r^2 \sigma).$$

$$(G.4)$$

Apply a union bound on all different $c \in [p]$ to see that for any $\delta_2 \in (0,1)$ with probability at least $1 - \delta_2$ over randomness of perturbation

$$\max_{c \in [p]} \left| f_c \Big((\Lambda_{\sigma}(V), \Lambda_{\sigma}(W)), (e_a, e_b) \Big) - f_c \Big(\theta, (e_a, e_b) \Big) \right|$$
 (G.5)

$$\leq 64\sqrt{2h} \left(\log \frac{2p(h+1)}{\delta_2}\right)^{3/2} \max(\sigma^3, r\sigma^2, r^2\sigma). \tag{G.6}$$

Hence, we'd want

$$64\sqrt{2h} \left(\log \frac{2p(h+1)}{\delta_2}\right)^{3/2} \max\left((\sigma/r)^3, (\sigma/r)^2, \sigma/r\right) \le \frac{\gamma}{4r^3}$$

$$\Longrightarrow \max\left((\sigma/r)^3, (\sigma/r)^2, \sigma/r\right) \le \frac{\gamma/r^3}{256\sqrt{2h} \left(\log \frac{2(h+1)}{\delta_1}\right)^{3/2}}$$
(G.7)

$$\Rightarrow \sigma = \begin{cases} \left(\frac{\gamma/r^3}{256\sqrt{2h}\left(\log\frac{2(h+1)}{\delta_1}\right)^{3/2}}\right)^{1/3} r & (*) \\ \frac{\gamma/r^3}{256\sqrt{2h}\left(\log\frac{2(h+1)}{\delta_1}\right)^{3/2}} r & (**) \end{cases}$$
(G.8)

where $\frac{\gamma/r^3}{256\sqrt{2h}\left(\log\frac{2(h+1)}{\delta_1}\right)^{3/2}}>1$ decides the event (*) and $\frac{\gamma/r^3}{256\sqrt{2h}\left(\log\frac{2(h+1)}{\delta_1}\right)^{3/2}}\leq 1$ decides the event (**). Assuming we're in the regime where $\sigma>r$, we can choose $\delta_2<1/2$ to see that with

probability at least 1/2

$$\max_{c \in [p]} \left| f_c \Big(\big(\Lambda_{\sigma}(V), \Lambda_{\sigma}(W) \big), (e_a, e_b) \Big) - f_c \Big(\theta, (e_a, e_b) \Big) \right| \le \gamma/4.$$
 (G.9)

Note that $\mathrm{KL}(\Lambda_{\sigma}(A)\|\mathcal{N}(0,\sigma^2)) \leq \frac{\|A\|_F^2}{2\sigma^2}$ for any matrix A. Apply Lemma G.9 to see that with probability at least $1-\delta$ over randomness of $\mathcal{D}_{\mathrm{train}}$ of size n we have that

$$L_{0}(f, \theta, \mathcal{D}) \leq L_{\gamma}(f, \theta, \mathcal{D}_{\text{train}}) + 4\sqrt{\frac{\frac{3hp\log\frac{2(h+1)}{\delta_{1}}}{\left(\frac{\gamma/r^{3}}{256\sqrt{2h}}\right)^{2/3}} + \log\frac{6n}{\delta}}{n-1}}$$

$$\leq L_{\gamma}(f, \theta, \mathcal{D}_{\text{train}}) + \tilde{\mathcal{O}}\left(\sqrt{\frac{p}{n}} \cdot \sqrt[3]{\frac{h^{2}}{\gamma/r^{3}}}\right)$$
(G.10)

for any $\theta \in \mathcal{M}_r^{h,p}$.

Lemma G.10. Choose $\sigma, \delta > 0$ and integer $p \geq 2$. For any r > 0 and integers $h \geq 1, a, b \in [p]$ it holds with probability at least $1 - \delta$ that

$$\left| \Lambda_{\sigma}(V)^{\top} (W(e_a, e_b))^{\odot 2} - V^{\top} (W(e_a, e_b))^{\odot 2} \right| \leq 4r^2 \sigma \sqrt{h \log \frac{2}{\delta}}$$

where $(W, V) \in \Theta_r^{h,1}$.

Proof. Denote by $\tilde{V} = \Lambda_{\sigma}(V) - V$ and $Q = \left(W(e_a, e_b)\right)^{\odot 2}$. We can expand the target as $V^{\top}Q + \tilde{V}^{\top}Q$ where the first summand is constant and the second summand is distributed according to $\mathcal{N}\left(0, \sigma^2 \|Q\|_2^2\right)$. Note that since $\|W\|_{\infty} \leq r$, we have that $\|Q\|_2^2 \leq 16hr^4$. Applying Proposition G.5 on this Gaussian random variable, one can see that with probability at least $1 - \delta$ over randomness of perturbation

$$\left| \Lambda_{\sigma}(V)^{\top} \left(W(e_a, e_b) \right)^{\odot 2} - V^{\top} Q \right| \le 4r^2 \sigma \sqrt{h \log \frac{2}{\delta}}. \tag{G.11}$$

Lemma G.11. Choose $\sigma, \delta > 0$ and integer $p \geq 2$. For any r > 0 and integers $h \geq 8 \log \frac{2}{\delta}, a, b \in [p]$ it holds with probability at least $1 - \delta$ that

$$\left| V^{\top} \left(\Lambda_{\sigma}(W)(e_a, e_b) \right)^{\odot 2} - V^{\top} \left(W(e_a, e_b) \right)^{\odot 2} \right| \le 16 \sqrt{h \log \frac{2}{\delta} \max \left(r^2 \sigma, r \sigma^2 \right)}$$

where $(W, V) \in \mathcal{M}_r^{h,1}$.

Proof. Denote by $\tilde{W} = \Lambda_{\sigma}(W) - W$, $Q = (W(e_a, e_b))^{\odot 2}$ and $\tilde{Q} = (\tilde{W}(e_a, e_b))^{\odot 2}$. Note that each coordinate of \tilde{Q} is sub-exponential with parameters $SE(4\sigma^2, 8\sigma^2)$ and mean 2. We can expand

$$V^{\top} (\Lambda_{\sigma}(W)(e_a, e_b))^{\odot 2} = V^{\top} Q + V^{\top} \tilde{Q} + 2V^{\top} ((W_a + W_b) \odot (\tilde{W}_a + \tilde{W}_b)).$$

Note that $V^{\top}\tilde{Q}$ is sub-exponential with parameters $SE(4r\sigma^2\sqrt{h},8r\sigma^2)^{-10}$ and mean 0 (due to $\sum_{i=1}^h V_i=0$ and linearity of expectation). We can apply Corollary G.4 to see that with probability at least $1-\delta/2$

$$\left| V^{\top} \tilde{Q} \right| \leq 8r\sigma^2 \max \left(\sqrt{2h \log \frac{2}{\delta}}, 4\log \frac{2}{\delta} \right).$$

Moreover, Since $2V^{\top}\left((W_a+W_b)\odot(\tilde{W}_a+\tilde{W}_b)\right)$ is distributed according to $\mathcal{N}(0,8\sigma^2 \|V\odot(W_a+W_b)\|_2^2)$ and $\|V\odot(W_a+W_b)\|_2^2 \leq 4hr^4$, applying Proposition G.5 reveals that with probability at least $1-\delta/2$ over randomness of perturbation

$$\left| 2V^{\top} \left((W_a + W_b) \odot (\tilde{W}_a + \tilde{W}_b) \right) \right| \le 8r^2 \sigma \sqrt{h \log \frac{2}{\delta}}.$$

Combining the two equations above shows that with probability at least $1-\delta$ over randomness of perturbation it holds that

$$\left| V^{\top} \left(\Lambda_{\sigma}(W)(e_a, e_b) \right)^{\odot 2} - V^{\top} Q \right| \leq 16 \sqrt{h \log \frac{2}{\delta}} \max \left(r^2 \sigma, r \sigma^2 \right).$$

Lemma G.12. Choose $\sigma, \delta > 0$ and integer $p \geq 2$. For any r > 0 and integers $h \geq e\delta, a, b \in [p]$ it holds with probability at least $1 - \delta$ over randomness of perturbation that

$$\left| \Lambda_{\sigma}(\mathbf{0}_h)^{\top} \left(\Lambda_{\sigma}(\mathbf{0}_{h \times 2p})(e_a, e_b) \right)^{\odot 2} \right| \leq \left(2\sigma^2 + 8\sqrt{2}\sigma^2 \log \frac{2(h+1)}{\delta} \right) \sigma \sqrt{h \log \frac{2(h+1)}{\delta}}.$$

Proof. Denote by $\tilde{V} = \Lambda_{\sigma}(\mathbf{0}_h)$ and $\tilde{Q} = \left(\Lambda_{\sigma}(\mathbf{0}_{h\times 2p})(e_a,e_b)\right)^{\odot 2}$. It's easy to see that each coordinate of \tilde{Q} is sub-exponential with parameters $\mathrm{SE}(2\sigma^2\sqrt{2},4\sigma^2\sqrt{2})$ and mean $2\sigma^2$. To bound $\tilde{V}^\top \tilde{Q}$, we employ the following strategy: since each coordinate of \tilde{Q} is a sub-exponential random variable, we can use a union bound in combination with Corollary G.4 to derive a bound on the maximum value of them. Next, we pull this maximum value out of the sum, and apply Proposition G.6 to bound the sum of remaining independent Gaussians. Combining these two high probability events, we present a high probability bound on $\tilde{V}^\top \tilde{Q}$ being bounded. Formally, for arbitrary $\delta_1, \delta_2 > 0$:

$$\Pr\left[\tilde{Q}_{i} \leq 2\sigma^{2} + \sigma^{2} \max\left(4\sqrt{\log\frac{2h}{\delta_{1}}}, 8\sqrt{2}\log\frac{2h}{\delta_{1}}\right) \text{ for all } i \in [h]\right] \geq 1 - \delta_{1}$$

$$\Rightarrow \Pr\left[\left|\tilde{V}^{\top}\tilde{Q}\right| \leq \left(2\sigma^{2} + 8\sqrt{2}\sigma^{2}\log\frac{2h}{\delta_{1}}\right) \left|\sum_{i=1}^{h}\tilde{V}_{i}\right|\right] \geq 1 - \delta_{1}$$

$$\Rightarrow \Pr\left[\left|\tilde{V}^{\top}\tilde{Q}\right| \leq \left(2\sigma^{2} + 8\sqrt{2}\sigma^{2}\log\frac{2h}{\delta_{1}}\right)\sigma\sqrt{h\log\frac{2}{\delta_{2}}}\right] \geq 1 - \delta_{1} - \delta_{2}$$

$$\Rightarrow \Pr\left[\left|\tilde{V}^{\top}\tilde{Q}\right| \leq \left(2\sigma^{2} + 8\sqrt{2}\sigma^{2}\log\frac{2(h+1)}{\delta}\right)\sigma\sqrt{h\log\frac{2(h+1)}{\delta}}\right] \geq 1 - \delta \quad \text{(G.12)}$$

where for the last step to be correct we chose $\delta_1 = \frac{h}{h+1}\delta$ and $\delta_2 = \frac{1}{h+1}\delta$.

¹⁰Note that these parameters are not tight, but this doesn't affect the correctness of this argument. For example, a random variable that is SE(a, b) is also SE(2a, 2b), or if it's SG(a), then it is also SG(2a).

Lemma G.13. Choose $\sigma, \delta > 0$ and integer $p \geq 2$. For any r > 0 and integers $h \geq 8 \log \frac{2}{\delta}, a, b \in [p]$ it holds that

$$\left| \Lambda_{\sigma}(\mathbf{0}_h)^{\top} \left(\Lambda_{\sigma}(W)(e_a, e_b) \right)^{\odot 2} \right| \leq 32\sqrt{h} \left(\log \frac{2(h+1)}{\delta_4} \right)^{3/2} \max(\sigma^3, r\sigma^2, r^2\sigma, \sigma).$$

where $W \in \mathbb{R}^{h \times p}$ such that $\|W\|_{\infty} \leq r$.

Proof. Denote by $\tilde{V} = \Lambda_{\sigma}(\mathbf{0}_h)$, $\tilde{W} = \Lambda_{\sigma}(W) - W$, $Q = W(e_a, e_b)$ and $\tilde{Q} = \tilde{W}(e_a, e_b)$. We have that

$$\Lambda_{\sigma}(\mathbf{0}_{h})^{\top} \left(\Lambda_{\sigma}(W)(e_{a}, e_{b})\right)^{\odot 2} = \tilde{V}^{\top} \left(W_{a} + W_{b} + \tilde{W}_{a} + \tilde{W}_{b}\right)^{\odot 2}
= \tilde{V}^{\top} \left(Q^{\odot 2} + \tilde{Q}^{\odot 2} + 2Q \odot \tilde{Q}\right)$$
(G.13)

In Lemma G.10 we have already shown that for any $\delta_1 > 0$ with probability at least $1 - \delta_1$ over randomness of perturbation it holds that

$$\left| \tilde{V}^{\top} Q^{\odot 2} \right| \le 4r^2 \sigma \sqrt{h \log \frac{2}{\delta_1}}. \tag{G.14}$$

Denote $\xi = \tilde{V} \odot \tilde{Q}$. ξ , the vector of product of two independent Gaussians, is sub-exponential with parameters $\mathrm{SE}(2\sigma^2\sqrt{2},4\sigma^2\sqrt{2})$ and mean 0 (and hence sum of its coordinates is $\mathrm{SE}(2\sigma^2\sqrt{2h},4\sigma^2\sqrt{2})$). Since $\|W\|_{\infty} \leq r$, applying Corollary G.4 yields that for any $\delta_2 > 0$, it holds with probability at least $1-\delta_2$ that

$$\left| 2\tilde{V}^{\top} (Q \odot \tilde{Q}) \right| \leq 4\sqrt{2}r\sigma^{2} \max\left(\sqrt{2h\log\frac{2}{\delta_{2}}}, 4\log\frac{2}{\delta_{2}}\right) \\
\leq 16r\sigma^{2} \sqrt{h\log\frac{2}{\delta_{2}}}.$$
(G.15)

Finally, we employ Lemma G.12 to show that for any $\delta_3 > 0$ it holds with probability at least $1 - \delta_3$ that

$$\left| \tilde{V}^{\top} \tilde{Q}^{\odot 2} \right| \le \left(2\sigma^2 + 8\sqrt{2}\sigma^2 \log \frac{2(h+1)}{\delta_3} \right) \sigma \sqrt{h \log \frac{2(h+1)}{\delta_3}}. \tag{G.16}$$

Applying a union bound on Equations (G.14) to (G.16) and choosign $\delta_4 = \delta_1/3 = \delta_2/3 = \delta_3/(3h+3)$ reveals that with probability at least $1-\delta_4$

$$\left| \Lambda_{\sigma}(\mathbf{0}_{h})^{\top} \left(\Lambda_{\sigma}(W)(e_{a}, e_{b}) \right)^{\odot 2} \right| \leq 16 \sqrt{h \log \frac{h}{\delta_{4}}} \max \left(r\sigma^{2}, r^{2}\sigma \right) + 16\sqrt{2h}\sigma^{3} \left(\log \frac{2(h+1)}{\delta_{4}} \right)^{3/2}$$

$$\leq 32\sqrt{h} \left(\log \frac{2(h+1)}{\delta_{4}} \right)^{3/2} \max(\sigma^{3}, r\sigma^{2}, r^{2}\sigma). \tag{G.17}$$