Exploring the Complexity of Deep Neural Networks through Functional Equivalence

Guohao Shen [1]

Abstract

We investigate the complexity of deep neural networks through the lens of functional equivalence, which posits that different parameterizations can yield the same network function. Leveraging the equivalence property, we present a novel bound on the covering number for deep neural networks, which reveals that the complexity of neural networks can be reduced. Additionally, we demonstrate that functional equivalence benefits optimization, as overparameterized networks tend to be easier to train since increasing network width leads to a diminishing volume of the effective parameter space. These findings can offer valuable insights into the phenomenon of overparameterization and have implications for understanding generalization and optimization in deep learning.

1. Introduction

Artificial neural networks, particularly deep and wide ones, have shown remarkable success in various applications widely in machine learning and artificial intelligence. However, one of the major challenges in understanding the success is to explain their generalization ability when they are very large and prone to overfitting data (Neyshabur et al., 2014; 2017a; Razin & Cohen, 2020).

Theoretical studies have suggested that the generalization error can be related to the complexity, approximation power, and optimization of deep neural networks. Larger neural networks are proved to possess better approximation power (Yarotsky, 2017; Lu et al., 2021a; Zhou, 2020b), but may exhibit larger complexity and generalization gaps (Bartlett et al., 2017; Mohri et al., 2018; Bartlett et al., 2019), and can be more challenging to optimize (Glorot et al., 2011). However, some aspects of deep learning initially appeared to contradict common sense: overparameterized networks tend to be easier to train (Frankle & Carbin, 2019; Allen-Zhu et al., 2019; Du et al., 2019) and exhibit better generalization (Belkin et al., 2019; Neyshabur et al., 2019; Novak et al., 2018). Although the model class's capacity was immense, deep networks did not tend to overfit (Zhang et al., 2017).

Recent studies have highlighted that the functional form of neural networks may be less complex than their parametric form (Bui Thi Mai & Lampert, 2020; Stock & Gribonval, 2022; Grigsby et al., 2022b), as networks with different parameters may implement the same function. This insight provides us with a fresh perspective for reconsidering how overparameterization truly affects the generalization.

In this work, we quantitatively characterize the redundancy in the parameterization of deep neural networks and derive a complexity measure for these networks based on functional equivalence. We analyze the results to gain insights into generalization and optimization in deep learning.

1.1. Related work

The issue of redundancy or identification of parameterization of neural networks has been noted since 1990 in Hecht-Nielsen (1990). Subsequent studies for neural networks with Tanh and sigmoid activations (Chen et al., 1993; Fefferman & Markel, 1993; Kůrkov'a & Kainen, 1994) have proved that given the input-output mapping of a Tanh neural network, its architecture can be determined and weights are identified up to permutations and sign flips. Recently, the identifiability of parameterization in deep neural networks, particularly ReLU networks, has received considerable attention (Elbrächter et al., 2019; Bui Thi Mai & Lampert, 2020; Bona-Pellissier et al., 2021; Dereich & Kassing, 2022; Stock & Gribonval, 2022; Grigsby et al., 2022b;a). Most recently, (Bui Thi Mai & Lampert, 2020) demonstrated that ReLU networks with non-increasing widths are identifiable up to permutation and scaling of weight matrices. With redundant parameterization, the weight space of deep neural networks can exhibit symmetric structures, which leads to implications for optimization (Neyshabur et al., 2015a; Badrinarayanan et al., 2015a; Stock et al., 2019). These studies suggest that naive loss gradient is

[1]Department of Applied Mathematics, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong SAR, China. Correspondence to: Guohao Shen <guohao.shen@polyu.edu.hk>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

sensitive to reparameterization by scaling, and proposed alternative, scaling-invariant optimization procedures. In addition, the redundancy or identification properties of neural networks are closely related to the study of inverse stability (Elbrächter et al., 2019; Rolnick & Kording, 2020; Bona-Pellissier et al., 2021; Petersen et al., 2021; Stock & Gribonval, 2022), which investigates the possibility for one to recover the parameters (weights and biases) of a neural network.

The complexity of neural networks in terms of their parameterization redundancy has received limited attention. Among the few relevant studies, (Grigsby et al., 2022a) and (Grigsby et al., 2022b) are worth mentioning. In (Grigsby et al., 2022a), the authors investigated local and global notions of topological complexity for fully-connected feedforward ReLU neural network functions. On the other hand, (Grigsby et al., 2022b) defined the functional dimension of ReLU neural networks based on perturbations in parameter space. They explored functional redundancy and conditions under which the functional dimension reaches its theoretical maximum. However, it should be noted that these results on functional dimension do not directly translate into generalization error bounds for deep learning algorithms.

The complexity of a class of functions is closely related to generalization error, with larger complexities often leading to larger generalization error (Bartlett et al., 2017; Mohri et al., 2018). Various complexity upper bounds have been studied for deep neural networks using different measurements, such as Rademacher complexity (Neyshabur et al., 2015b; Golowich et al., 2018; Li et al., 2018), VC-dimension and Pseudo dimension (Baum & Haussler, 1988; Goldberg & Jerrum, 1993; Anthony et al., 1999; Bartlett et al., 2019), and covering number (Anthony et al., 1999; Neyshabur et al., 2017b; Bartlett et al., 2017; Lin & Zhang, 2019). These measurements characterize the complexity of the class of neural networks and are influenced by hyperparameters like network depth, width, number of weights and bias vectors, and corresponding norm bounds. While these bounds are not directly comparable in magnitude, they are closely related and can be converted to facilitate comparisons (Anthony et al., 1999; Mohri et al., 2018).

1.2. Our contributions

We summarize our contributions as follows:

1. We make use of the permutation equivalence property to firstly obtain a tighter upper bound on the covering number of neural networks, which improves existing results by factorials of the network widths and provides unprecedented insights into the intricate relationship between network complexity and layer width.
2. We improve existing covering number bounds in the sense that our results hold for neural networks with bias vectors and general activation functions. Since bias terms are indispensable for the approximation power of neural networks, our results are useful in both theory and practice. Additionally, we express our bound explicitly in terms of the network's width, depth, size, and the norm of the parameters.
3. We discuss the implications of our findings for understanding generalization and optimization. In particular, we found that overparameterized networks tend to be easier to train in the sense that increasing the width of neural networks leads to a vanishing volume of the effective parameter space.

The remainder of the paper is organized as follows. In section 2, we introduce the concept of functional equivalence and investigate the permutation invariance property of general feedforward neural networks. In section 3, we derive novel covering number bounds for shallow and deep neural networks by exploiting the permutation invariance property and compare our results with existing ones. In section 4, we discuss the extension to convolutional, residual and attention-based networks. In section 5, we demonstrate the theoretical implication of permutation invariance on the optimization complexity in deep learning and discuss the implications of our results on generalization. Finally, we discuss the limitations of this study and future research directions in section 6. All technical proofs are included in the Appendix.

2. Functionally equivalent Neural Networks

A feedforward neural network is a fully connected artificial neural network consisting of multiple layers of interconnected neurons. The network's architecture can be expressed as a composition of linear maps and activations. The functional form of an L-layer feedforward neural network is determined by its weight matrices, bias vectors, and activation functions:

$$f(x;\theta) = \mathcal{A}_{L+1} \circ \sigma_L \circ \mathcal{A}_L \circ \cdots \circ \sigma_2 \circ \mathcal{A}_2 \circ \sigma_1 \circ \mathcal{A}_1(x). \quad (1)$$

Here, $\mathcal{A}_l(x) = W^{(l)}x + b^{(l)}$ is the linear transformation for layer l, where $W^{(l)}$ and $b^{(l)}$ are the weight matrix and bias vector respectively. The activation function σ_l is applied element-wise to the output of $\mathcal{A}_l$, and can be different across layers. The collection of weight matrices and bias vectors is denoted by $\theta = (W^{(1)}, b^{(1)}, \ldots, W^{(L+1)}, b^{(L+1)})$. The input x is propagated through each layer of the network to produce the output $f(x;\theta)$.

The parameterization of a neural network can be redundant, with different parameter sets producing identical function outputs. This redundancy arises from the non-identifiability of weight matrices or activation functions.

Table 1. A comparison of recent results on the complexity of feedforward neural networks.

PAPER	COMPLEXITY	EXPLICIT	BIAS VECTORS	GENERAL ACTIVATIONS	PERMUTATION INVARIANCE
(BARTLETT ET AL., 2017)	$B_x^2(\bar{\rho}\bar{s})^2\mathcal{U}\log(W)/\epsilon^2$	✗	✗	✗	✗
(NEYSHABUR ET AL., 2017B)	$B_x^2(\bar{\rho}\bar{s})^2\mathcal{S}L^2\log(WL)/\epsilon^2$	✗	✗	✗	✗
(LIN & ZHANG, 2019)	$B_x(\bar{\rho}\bar{s})\mathcal{S}^2L/\epsilon$	✗	✗	✗	✗
(BARTLETT ET AL., 2019)	$L\mathcal{S}\log(\mathcal{S})\log(\bar{\rho}\bar{s}B_x/\epsilon)$	✓	✓	✓	✗
THIS PAPER	$L\mathcal{S}\log(\bar{\rho}\bar{s}B_x^{1/L}/((d_1!\cdots d_L!)^{1/\mathcal{S}}\epsilon)^{1/L})$	✓	✓	✓	✓

Notations: $\mathcal{S}$ number of parameters; $\mathcal{U}$ number of hidden neurons; L number of hidden layers; W maximum hidden layers width; B_x, L2 norm of input; $\bar{\rho} = \Pi_{j=1}^{L}\rho_j$, products of Lipschitz constants of activations; $\bar{s} = \Pi_{j=1}^{L}s_j$, products of spectral norms of hidden layer weight matrices; ϵ, radius for covering number.

Definition 2.1 (Functionally-Equivalent Neural Networks)**.** Two neural networks $f(x;\theta_1)$ and $f(x;\theta_2)$ are said to be functionally-equivalent on $\mathcal{X}$ if they produce the same input-output function for all possible inputs, i.e.,

$$f_1(x;\theta_1) = f_2(x;\theta_2) \quad \forall x \in \mathcal{X}, \tag{2}$$

where $\mathcal{X}$ is the input space and θ_1 and θ_2 denote the sets of parameters of the two networks, respectively.

Neural networks with a fixed architecture can have functionally-equivalent versions through weight scaling, sign flips, and permutations. This can even occur across networks with different architectures. In this paper, we focus on the complexity of a specific class of neural networks with fixed architecture but varying parameterizations. We provide examples of functionally-equivalent shallow neural networks to illustrate this concept.

Example 2.2 (Scaling). Consider two shallow neural networks parameterized by $\theta_1 = (W_1^{(1)}, b_1^{(1)}, W_1^{(2)}, b_1^{(2)})$ and $\theta_2 = (W_2^{(1)}, b_2^{(1)}, W_2^{(2)}, b_2^{(2)})$, defined as:

$$f(x;\theta_1) = W_1^{(2)}\sigma(W_1^{(1)}x + b_1^{(1)}) + b_1^{(2)},$$
$$f(x;\theta_2) = W_2^{(2)}\sigma(W_2^{(1)}x + b_2^{(1)}) + b_2^{(2)}$$

respectively, where $x \in \mathbb{R}^n$ is the input to the network and σ satisfies $\sigma(\lambda x) = \lambda\sigma(x)$ for all $x \in \mathbb{R}^n$ and $\lambda > 0$. If there exists a scalar value $\alpha > 0$ such that:

$$(W_2^{(1)}, b_2^{(1)}) = (\alpha W_1^{(1)}, \alpha b_1^{(1)}) \quad \text{and} \quad W_2^{(2)} = \frac{1}{\alpha}W_1^{(2)},$$

then $f(\cdot;\theta_1)$ and $f(\cdot;\theta_2)$ are functionally equivalent.

Scaling invariance property is applicable to ReLU, Leaky ReLU, and piecewise-linear activated neural networks. Specifically, for all $x \in \mathbb{R}^n$ and $\lambda \geq 0$, we have $\sigma(\lambda x) = \lambda\sigma(x)$ for σ being the ReLU or Leaky ReLU function. It is worth noting that the above example is presented for shallow neural networks, but the scaling invariance property can happen in deep networks across any two consecutive layers.

Example 2.3 (Sign Flipping). Consider two shallow neural networks $f(\cdot;\theta_1)$ and $f(\cdot;\theta_2)$ defined in Example 2.2 with σ being an odd function, that is $\sigma(-x) = -\sigma(x)$ for all $x \in \mathbb{R}^n$. If

$$(W_2^{(1)}, b_2^{(1)}) = (-W_1^{(1)}, -b_1^{(1)}) \quad \text{and} \quad W_2^{(2)} = -W_1^{(2)},$$

then $f(x;\theta_1)$ and $f(x;\theta_2)$ are functionally equivalent.

Sign flipping invariance property can happen for neural networks activated by Tanh, Sin and odd functions. It is worth noting that Sigmoid does not have a straightforward Sign flipping invariance. While Sigmoid is an odd function up to a constant 0.5, it can be Sign-flipping invariant up-to a constant and the constant can be mitigated by using a bias(Martinelli et al., 2023). The sign flipping invariance property can also be generalized to deep neural networks across any two consecutive layers.

Example 2.4 (Permutation). Consider two shallow neural networks $f(\cdot;\theta_1)$ and $f(\cdot;\theta_2)$ defined in Example 2.2 with σ being a general activation function. Let the dimension of the hidden layer of $f(x;\theta_1)$ and $f(x;\theta_2)$ be denoted by m. If there exists an $m \times m$ permutation matrix P such that

$$(PW_2^{(1)}, Pb_2^{(1)}) = (W_1^{(1)}, b_1^{(1)}) \quad \text{and} \quad W_2^{(2)}P = W_1^{(2)},$$

then $f(x;\theta_1)$ and $f(x;\theta_2)$ are functionally equivalent.

The feedforward neural networks are built of linear transformations and activations, and it is intuitive that simply re-indexing neurons in a hidden layer and the corresponding rows of the weights matrix and bias vector will lead to a functionally equivalent network. The permutation invariance is the most basic type of equivalence for neural

networks since it does not rely on any specific properties of activation functions, while scaling and sign flipping invariance are activation-dependent properties. A comparison on functional equivalence properties on neural network with commonly-used activation functions is presented in Table 2.

Next, we derive sufficient conditions fo feed-forward neural networks (FNNs) to be permutation-equivalent.

Proposition 2.5 (Permutation equivalence for deep FNNs)**.** *Consider two neural networks $f(x;\theta_1)$ and $f(x;\theta_2)$ with the same activations $\sigma_1,\dots,\sigma_L$ and architecture*

$$f(x;\theta) = W^{(L+1)}\sigma_L(\cdots\sigma_1(W_1^{(1)}x+b_1^{(1)})\cdots)+b_1^{(L)})+b^{(L+1)}$$

but parameterized by different parameters

$$\theta_j = (W_j^{(1)}, b_j^{(1)}, \dots, W_j^{(L+1)}, b_j^{(L+1)}), \quad j=1,2$$

respectively, where $x \in \mathbb{R}^n$ is the input to the network. Let $P^\top$ denote the transpose of matrix P. If there exists permutation matrices $P_1,\dots,P_L$ such that

$$\begin{aligned} &W_1^{(1)} = P_1 W_2^{(1)}, && b_1^{(1)} = P_1 b_2^{(1)}, \\ &W_1^{(l)} = P_l W_2^{(l)} P_{l-1}^\top, && b_1^{(l)} = P_l b_2^{(1)}, \quad l=2,\dots,L \\ &W_1^{(L+1)} = W_2^{(L+1)} P_L^\top, && b_1^{(L)} = b_2^{(L)}, \end{aligned}$$

then $f(x;\theta_1)$ and $f(x;\theta_2)$ are functionally equivalent.

Proposition 2.5 describes the relationship between the parameters of two permutation-equivalent deep feedforward neural networks. This relationship can be used to create functionally equivalent networks given fixed architectures. It's important to note that although permutation invariance is sufficient for functional equivalence of feedforward neural networks, it's not always necessary. (Petzka et al., 2020) gave a complete characterization for fully-connected networks with two layers. While for general networks, certain restrictions on the architecture and activation function are required to fully characterize (Sussmann, 1992; Kůrkov'a & Kainen, 1994; Bui Thi Mai & Lampert, 2020) and recover the parameters of a network (Martinelli et al., 2023). This study focuses only on utilizing permutation invariance to investigate neural network complexity.

3. Complexity of Deep Neural Networks

In this section, we analyze the complexity of a class of feedforward neural networks by examining the redundancy that arises from permutation invariance. Specifically, we study the covering number of real-valued, deep feedforward neural networks that share the same architecture but have different parameterization.

Let the vector $(d_0, d_1, \dots, d_L)$ represent the dimensions of the layers of the neural network $f(x;\theta)$ defined in (1), where $d_{L+1} = 1$ as the output is real-valued. Note that the bias vectors in hidden layers contain $\mathcal{U} := \sum_{i=1}^L d_i$ entries, and the weight matrices together with bias vectors contain $\mathcal{S} := \sum_{i=0}^L d_i \times d_{i+1} + d_{i+1}$ entries in total. We define the parameter space of θ as $\Theta = [-B,B]^{\mathcal{S}}$ for some $B \geq 1$, which is closed for permutation operations and ensures the absolute value of the weight matrix and bias vector entries are bounded by B. We set $\Theta = [-B,B]^{\mathcal{S}}$ for some $B \geq 1$. The setting is in line with complexity studies as in (Neyshabur et al., 2015b; Bartlett et al., 2017; Golowich et al., 2018) with norm controls. The setting of bounded parameter space can also correspond to the observed implicit regularization phenomena in SGD-based optimization algorithms (Neyshabur et al., 2014; Gunasekar et al., 2018a;b), which can lead to minimum-norm solutions (e.g., for least square problems). We do not specify any activation functions $\sigma_1,\dots,\sigma_L$ since we consider general deep feedforward neural networks. Finally, the class of feedforward neural networks we consider is denoted as

$$\begin{aligned} &\mathcal{F}(L, d_0, d_1, \dots, d_L, B) \qquad (3) \\ &= \{f(\cdot;\theta) : \mathbb{R}^{d_0} \to \mathbb{R} \text{ is defined in (1)} : \theta \in [-B,B]^{\mathcal{S}}\}. \end{aligned}$$

3.1. Shallow Feed-Forward Neural Networks

It is well-known that a shallow neural network with a single hidden layer has universal approximation properties (Cybenko, 1989; Hornik, 1991) and is sufficient for many learning tasks (Hanin & Rolnick, 2019; Hertrich et al., 2021). We begin our investigation by considering shallow neural networks of the form

$$\begin{aligned} &\mathcal{F}(1, d_0, d_1, B) \qquad (4) \\ &= \{f(x;\theta) = W^{(2)}\sigma_1(W^{(1)}x + b^{(1)}) + b^{(2)} : \theta \in [-B,B]^{\mathcal{S}}\} \end{aligned}$$

where the total number of parameters is given by $\mathcal{S} = (d_0+2)\times d_1 + 1$. By Theorem 2.5, for any $\theta = (W^{(1)}, b^{(1)}, W^{(2)}, b^{(2)}) \in \Theta$ and any permutation matrix P, $\tilde{\theta} = (PW^{(1)}, Pb^{(1)}, W^{(2)}P^\top, b^{(2)}) \in \Theta$ will produce the same input-out function. Actually, permutation invariance leads to equivalence classes of parameters that yield the same realization, and we can obtain a set of representatives from equivalence classes. A canonical choice is

$$\Theta_0 := \{\theta \in [-B,B]^{\mathcal{S}} : b_1^{(1)} \geq b_2^{(1)} \geq \cdots \geq b_{d_1}^{(1)}\},$$

where the set of representatives is by restricting the bias vector $b^{(1)} = (b_1^{(1)}, \dots, b_{d_1}^{(1)})^\top$ to have descending components. Alternatively, we can sort the first component of the rows of $W^{(1)}$ to obtain a set of representatives. It is worth mentioning that Θ_0 may not be the minimal set of representatives since there may be other symmetries within Θ_0. We did not further utilize these additional symmetries to reduce Θ_0 to be smaller since other symmetries can depend on the specific properties of the activation functions in the neural networks. In this work, we specifically employ permutation invariance (which holds for networks with any activations) to obtain a representative set Θ_0.

The set of representatives Θ_0 has two important properties

Table 2. Functional equivalence property for networks with different activation functions.

ACTIVATION	FORMULA	SIGN FLIPPING	SCALING	PERMUTATION
SIGMOID	$[1+\exp(-x)]^{-1}$	✗	✗	✓
TANH	$[1-\exp(-2x)]/[1+\exp(-2x)]$	✓	✗	✓
RELU	$\max\{0,x\}$	✗	✓	✓
LEAKY RELU	$\max\{ax,x\}$ FOR $a>0$	✗	✓	✓

- Neural networks $\{f(\cdot;\theta):\theta\in\Theta_0\}$ parameterized by Θ_0 contains all the functions in $\{f(\cdot;\theta):\theta\in\Theta\}$, i.e.,

$$\{f(\cdot;\theta):\theta\in\Theta_0\}=\{f(\cdot;\theta):\theta\in\Theta\}.$$

- The volume (in terms of Lebesgue measure) of the set of representatives Θ_0 is $(1/d_1!)$ times smaller that that of the parameter space Θ, i.e.,

$$\text{Volume}(\Theta)=d_1!\times\text{Volume}(\Theta_0).$$

The first property holds naturally since any parameter $\theta\in\Theta$ has a permuted version in Θ_0. Regarding the second property, we note that both Θ and Θ_0 belong to the Euclidean space $\mathbb{R}^{\mathcal{S}}$ (where $\mathcal{S}$ denotes the number of parameters), then their volumes in terms of Lebesgue measure can be calculated. For any parameter θ with distinct components in the bias vector $b^{(1)}=(b_1^{(1)},\dots,b_{d_1}^{(1)})^\top$, permutation of the bias vector and corresponding weights can lead to $(d_1!)$ distinct equivalent parameters $\theta_1,\dots,\theta_{d_1!}$. Then this implies a $(d_1!)$ times smaller volume of representative set Θ_0 compared to that of Θ.

Subsequently, the two properties suggest that Θ_0 can be a representative parameterization of neural networks when they are viewed only as input-output functions. Based on these observations, we derive improved complexities of the class of neural networks in terms of its covering number.

Definition 3.1 (Covering Number)**.** Let $\mathcal{F}=f:\mathcal{X}\to\mathbb{R}$ be a class of functions. We define the supremum norm of $f\in\mathcal{F}$ as $\|f\|_\infty:=\sup_{x\in\mathcal{X}}|f(x)|$. For a given $\epsilon>0$, we define the covering number of $\mathcal{F}$ with radius ϵ under the norm $\|\cdot\|_\infty$ as the least cardinality of a subset $\mathcal{G}\subseteq\mathcal{F}$ satisfying

$$\sup_{f\in\mathcal{F}}\min_{g\in\mathcal{G}}\|f-g\|_\infty\le\epsilon.$$

Denoted by $\mathcal{N}(\mathcal{F},\epsilon,\|\cdot\|_\infty)$, the covering number measures the minimum number of functions in $\mathcal{F}$ needed to cover the set of functions within a distance of ϵ under the supremum norm.

The covering number $\mathcal{N}(\mathcal{F},\epsilon,\|\cdot\|_\infty)$ provides a quantitative measure of the complexity of the class of functions $\mathcal{F}$ under the supremum norm, with smaller values indicating simpler classes. Covering numbers, along with Rademacher complexity, VC dimension, and Pseudo dimension, are essential complexity measures in the analysis of learning theories and in estimating generalization errors. Although these measures are different, they are correlated with each other, and we introduce the detailed correlations in Appendix.

Remark 3.2. We define the covering number of a class of functions in the uniform sense. This is an extension of the canonical definition of covering numbers, which was originally developed for subsets in Euclidean space. While most existing studies of covering numbers for function spaces consider the image of the functions on a finite sample (Anthony et al., 1999; Bartlett et al., 2017), our definition is formulated directly in terms of the function space itself, without requiring a finite sample or any other auxiliary construction.

Theorem 3.3 (Covering number of shallow neural networks)**.** *Consider the class of single hidden layer neural networks $\mathcal{F}:=\mathcal{F}(1,d_0,d_1,B)$ defined in (4) parameterized by $\theta\in\Theta=[-B,B]^{\mathcal{S}}$. Suppose the radius of the domain $\mathcal{X}$ of $f\in\mathcal{F}$ is bounded by some $B_x>0$, and the activation σ_1 is continuous. Then for any $\epsilon>0$, the covering number*

$$\mathcal{N}(\mathcal{F},\epsilon,\|\cdot\|_\infty)\le(16B^2(B_x+1)\sqrt{d_0}d_1/\epsilon)^{\mathcal{S}}\times\rho^{\mathcal{S}_h}/d_1!, \tag{5}$$

where ρ denotes the Lipschitz constant of σ_1 on the range of the hidden layer (i.e., $[-\sqrt{d_0}B(B_x)+1),\sqrt{d_0}B(B_x+1)]$), and $\mathcal{S}_h=d_0d_1+d_1$ is the total number of parameters in the linear transformation from input to the hidden layer, and $\mathcal{S}=d_0\times d_1+2d_1+1$ is the total number of parameters.

Our upper bound on the covering number firstly takes advantage of permutation invariance, resulting in a reduced complexity (by a factorial term $d_1!$ in the denominator) compared to existing studies (Neyshabur et al., 2015b; Bartlett et al., 2017; Neyshabur et al., 2017b; Neyshabur, 2017; Lin & Zhang, 2019). The factorial reduction $d_!!$ can be significant. For instance, for a shallow ReLU network with a hidden dimension of $d_1=128$, the factorial $128!\approx10^{215}$, which is far larger than 10^{82}, the upper estimate on the number of atoms in the known universe. This reduction can be substantial and can enhance theoretical analysis and results that rely on covering numbers. In addition, it's worth noting that bounds in Theorem 3.3 holds true for networks with bias vectors. This is also an improvement over existing

studies that didn't consider bias vectors in neural networks, since bias terms are crucial for the approximation power of neural networks (Yarotsky, 2017; Lu et al., 2021b; Shen et al., 2022).

Stirling's formula can be used to approximate the factorial term as $\sqrt{2\pi d_1}(d_1/e)^{d_1}\exp(1/(12d_1+1)) < d_1! < \sqrt{2\pi d_1}(d_1/e)^{d_1}\exp(1/12d_1)$ when $d_1 \geq 1$. This reduces the covering number approximately by a factor of $(d_1/e)^{d_1}$ and the bound in (5) is $(C_{B,B_x,d_0,\rho}/\epsilon)^{\mathcal{S}} \times d_1^{\mathcal{S}-\mathcal{U}}$, where $\mathcal{U} = d_1$ denotes the number of hidden neurons and $C_{B,B_x,d_0,\rho} > 0$ is a constant depending only on B, B_x, d_0 and ρ. Notably, $\mathcal{S} - \mathcal{U}$ is the number of weights (excluding bias), and the reduced bound is basically that for a single neural network without bias vectors and considering permutation invariance. Lastly, we note that increasing the number of neurons in a shallow neural network enlarges its approximation power (Lu et al., 2017; Ongie et al., 2019), but at a smaller increase in complexity according to our results.

Remark 3.4. Theorem 3.3 applies to any activation function that is locally Lipschitz on bounded sets (range of the hidden layer), which does not require any specific choice such as Hinge or ReLU in (Neyshabur et al., 2015b; 2017b), and does not require universal Lipschitz and $\sigma(0) = 0$ as in (Bartlett et al., 2017; Lin & Zhang, 2019). In the case of the ReLU or Leaky ReLU activation, our bound simplifies to $\rho = 1$ without any condition, leading to the disappearance of the term $\rho^{\mathcal{S}_h}$ in our bound.

3.2. Deep Feed-Forward Neural Networks

For deep neural networks, we can also analyze its effective parameter space based on permutation invariance properties. By Theorem 2.5, a set of representatives $\Theta_0 = \Theta_0^{(1)} \times \Theta_0^{(2)} \times \cdots \times \Theta_0^{(L)} \times \Theta_0^{(L+1)}$ can be constructed where

$$\Theta_0^{(l)} = \Big\{(W^{(l)}, b^{(l)}) \in [-B,B]^{\mathcal{S}_l} : b_1^{(l)} \geq b_2^{(l)} \geq \cdots \geq b_{d_l}^{(l)}\}$$
$$\text{for } l = 1,\ldots,L, \Theta_0^{(L+1)} = \{(W^{(L+1)}, b^{(L+1)}) \in [-B,B]^{\mathcal{S}_{L+1}}\Big\}.$$

Then we can obtain an upper bound of the covering number of deep feedforward neural networks.

Theorem 3.5 (Covering number of deep neural networks)**.** *Consider the class of deep neural networks $\mathcal{F} := \mathcal{F}(1, d_0, d_1, \ldots, d_L, B)$ defined in (3) parameterized by $\theta \in \Theta = [-B,B]^{\mathcal{S}}$. Suppose the radius of the domain $\mathcal{X}$ of $f \in \mathcal{F}$ is bounded by B_x for some $B_x > 0$, and the activations $\sigma_1, \ldots, \sigma_L$ are locally Lipschitz. Then for any $\epsilon > 0$, the covering number $\mathcal{N}(\mathcal{F}, \epsilon, \|\cdot\|_\infty)$ is bounded by*

$$\frac{\Big(4(L+1)(B_x+1)(2B)^{L+2}(\Pi_{j=1}^{L}\rho_j)(\Pi_{j=0}^{L}d_j)\cdot\epsilon^{-1}\Big)^{\mathcal{S}}}{d_1!\times d_2!\times\cdots\times d_L!},$$

where $\mathcal{S} = \sum_{i=0}^{L} d_i d_{i+1} + d_{i+1}$ and ρ_i denotes the Lipschitz constant of σ_i on the range of $(i-1)$-th hidden layer, especially the range of $(i-1)$-th hidden layer is bounded by $[-B^{(i)}, B^{(i)}]$ with $B^{(i)} \leq (2B)^i \Pi_{j=1}^{i-1}\rho_j d_j$ for $i = 1, \ldots, L$.

Theorem 3.5 provides a novel upper bound for the covering number of deep neural networks based on permutation invariance, which reduces the complexity compared to previous results (Neyshabur et al., 2015b; 2017b; Bartlett et al., 2017; Lin & Zhang, 2019) by approximately a factor of $(d_1!d_2!\cdots d_L!)$. According to Theorem 3.5, increasing the depth of a neural network increases its complexity. However, it is interesting to note that the increased hidden layer l will have a $(d_l!)$ discount on the complexity. If the hidden layers have equal width ($d = d_1 = \cdots = d_L$), the bound reduces to $(C_{B,B_x,d_0,\rho}/\epsilon)^{\mathcal{S}} \times d^{\mathcal{S}-\mathcal{U}}$, where $\mathcal{U} = Ld$ denotes the number of hidden neurons and $C_{B,B_x,d_0,\rho} > 0$ is a constant depending only on B, B_x, d_0 and $\rho_i, i = 1, \ldots, L$. As with shallow neural networks, the improved rate $(\mathcal{S} - \mathcal{U})$ denotes the number of weights but excluding biases, which assures the approximation power of neural networks, but grows its complexity in a rate free of number of bias.

Remark 3.6. As discussed in Remark 3.4, our results take into account the permutation invariance and have looser requirements on activation functions compared to existing results (Neyshabur et al., 2015b; 2017b; Bartlett et al., 2017; Lin & Zhang, 2019). In addition, our upper bound is explicitly expressed in parameters which are known and can be specified in practice, e.g., network depth L, width $(d_0, d_1, \ldots, d_L)$, size $\mathcal{S}$ and uniform bound B for weights and biases. While most existing bounds in (Neyshabur et al., 2015b; 2017b; Bartlett et al., 2017; Lin & Zhang, 2019) are in terms of the spectral norm of weight matrices and some measurement on externally introduced reference matrices, which are usually unknown in practice.

3.3. Comparing to existing results

The complexity upper bounds in terms of covering number on the class of deep neural networks have been studied in (Anthony et al., 1999; Neyshabur et al., 2017b; Bartlett et al., 2017; Lin & Zhang, 2019). These results are proved by using similar approaches of mathematical induction (e.g. Lemma A.7 in (Bartlett et al., 2017), Lemma 2 in (Neyshabur et al., 2017b) and Lemma 14 of (Lin & Zhang, 2019)). Compared with these results, we improve upon their results in three ways.

- First, we consider the generally defined neural networks where the bias vector is allowed to appear. Bias terms are indispensable for the approximation power of neural networks (Yarotsky, 2017; Lu et al., 2021b; Shen et al., 2022). Neural networks without bias vectors may be disqualified in theory and practice.
- Second, we explicitly express our bound in terms of the width, depth, and size of the neural network, as

well as the infinity norm of the parameters.

- Third, we utilize the permutation equivalence property to obtain a tighter upper bound on the covering number. Notably, this bound improved existing results by a factorial of the network widths, providing unprecedented insights into the intricate relationship between network complexity and layer width.

Various complexity upper bounds for deep neural networks in other measurements have also been studied, including Rademacher complexity (Neyshabur et al., 2015b; Golowich et al., 2018; Li et al., 2018), VC-dimension, and Pseudo dimension (Baum & Haussler, 1988; Goldberg & Jerrum, 1993; Anthony et al., 1999; Bartlett et al., 2019). Our bounds in terms of covering number are not directly comparable with these measurements. To make a comparison with these other measurements, we convert the upper bounds into metric entropy, which is equivalent to the logarithm of the covering number $\log(\mathcal{N}(\mathcal{F}, \epsilon, \|\cdot\|_\infty))$. Specially, let $\bar{\rho} = \Pi_{j=1}^{L} \rho_j$ products of Lipschitz constants of activation functions and $\bar{s} = \Pi_{j=1}^{L} s_j$ products of spectral norms of hidden layer weight matrices (Bartlett et al., 2017) derived an spectral-norm based bound of metric entropy $B_x^2(\bar{\rho}\bar{s})^2\mathcal{U}\log(W)/\epsilon^2$, following which (Neyshabur et al., 2017b) obtained $B_x^2(\bar{\rho}\bar{s})^2\mathcal{S}L^2\log(WL)/\epsilon^2$ and (Lin & Zhang, 2019) obtained $B_x(\bar{\rho}\bar{s})\mathcal{S}^2L/\epsilon$. Based on Theorem 12.2 in (Anthony et al., 1999), the Pseudo dimension bound in (Bartlett et al., 2017) leads to $L\mathcal{S}\log(\mathcal{S})\log(\bar{\rho}\bar{s}B_x/\epsilon)$. Lastly, our results in Theorem 3.5 can be presented as $L\mathcal{S}\log(\bar{\rho}\bar{s}B_x^{1/L}/((d_1!\cdots d_L!)^{\mathcal{S}}\epsilon)^{1/L})$ by letting $s_i := B\sqrt{d_i d_{i-1}}$ given in our setting. It is important to note that the quantity $B\sqrt{d_i d_{i-1}}$ can provide an upper bound for the spectral norm s_i, but the reverse is not necessarily true. We cannot ensure Theorem 3.5 to still be true by directly substituting $B\sqrt{d_i d_{i-1}}$ with s_i in the theorem. In other words, if the covering number bounds are derived in terms of the spectral norm, the reduction factor $d_1!\cdots d_L!$ may not be obtained, and additional parameters like matrix norm bounds would appear in the upper bound(Bartlett et al., 2017). We would also mention that even our results improved over the existing ones, all these bounds scale with number of parameters $\mathcal{S}$ and can still result vacuous bounds in error analysis with extremely over-parametrized settings. We present a detailed comparison of the results in Table 1.

4. Extension to other neural networks

Functional equivalence can manifest ubiquitously across various types of neural networks, with a specific emphasis on the presence of permutation equivalence within neural networks featuring linear transformation layers. This section delves into the exploration of feasible extensions aimed at harnessing the power of functional equivalence within convolutional neural networks, residual networks, and attention-based networks.

4.1. Convolutional neural networks

Convolutional neural networks (CNNs) are featured by the utilization of convolution and pooling layers. In a convolution layer, the input is convolved with parameter-carrying filters, resembling a linear layer with a sparse weight matrix. A pooling layer is commonly employed for downsampling and summarizing input feature information. It partitions the input into non-overlapping regions and applies a pooling operation to each region. The most prevalent types of pooling operations include max/min/avg pooling, which retain the maximum, minimum, or average value within each region.

As demonstrated in Example 2.2-2.4, scaling, sign flip, and permutation equivalence directly apply to convolution layers (linear layer with sparse weight matrix). We also extend the permutation equivalence within pooling regions as follows.

Example 4.1 (Permutation within Pooling Regions). Consider two shallow CNNs defined by $f(x;\theta_1) = Pool(W_1x + b_1)$ and $f(x;\theta_2) = Pool(W_2x + b_2)$ respectively where "$Pool$" is a pooling operator. Let $\mathcal{I}_1, \ldots, \mathcal{I}_K$ be the non-overlapping index sets (correspond to the pooling operator) of rows of $W_1x + b_1$ and $W_2x + b_2$. Then $f(\cdot;\theta_1)$ and $f(\cdot;\theta_2)$ are functional equivalent if there exists a permutation matrix P such that $\forall k \in \{1, \ldots, K\}$

$$(PW_2)_{\mathcal{I}_k} \cong (W_1)_{\mathcal{I}_k} \text{ and } (Pb_2)_{\mathcal{I}_k} \cong (b_1)_{\mathcal{I}_k},$$

where $A_{\mathcal{I}_k}$ denotes the $\mathcal{I}_k$ rows of A and $A \cong B$ denotes that A equals to B up to row permutations.

Permutation equivalence within non-overlapping regions in CNNs preserves max/min/avg values, eliminating the need for cancel-off operations in subsequent layers. This further allows for the derivation of complexity bounds of CNNs by identifying their effective parameter space, similar to Theorem 3.5. In addition, the results on the connections between CNNs and feed-forward networks can be naturally utilized to improve the covering number bounds and capacity estimates (Fang et al., 2020; Mao et al., 2021), e.g., (Zhou, 2020b;a) showed that fully connected deep ReLU networks can be realized by deep CNNs with the same order of network parameter numbers,.

4.2. Residual Networks

Residual Networks are a type of deep CNN architecture that has a significant impact on computer visions (He et al., 2016). The key feature of ResNet is the use of skip connections that enable networks to learn residual mappings and bypass layers, leading to very deep but trainable networks. Mathematically, a residual layer $f(x;\theta) = x + \mathrm{F}(x;\theta)$ outputs the summation of the input x and its transformation

F$(x;\theta)$. The $F(\cdot;\theta)$ can be any transformation that maps x to the space of itself, and it defines the residual layer. Then the equivalence of F implies that of the residual layer.

Example 4.2 (Equivalence of Residual Layer). Consider two residual layers $f(x;\theta_1) = x + F(x;\theta_1)$ and $f(x;\theta_2) = x + F(x;\theta_2)$. Then $f(\cdot;\theta_1)$ and $f(\cdot;\theta_2)$ are functionally equivalent if and only if $F(\cdot;\theta_1)$ and $F(\cdot;\theta_2)$ are functionally equivalent.

4.3. Attention-based Networks

Attention-based models, well known for BERT, GPT and many others, have been successful in natural language processing and computer vision tasks (Vaswani et al., 2017; Devlin et al., 2018; Radford et al., 2018). It utilizes an attention mechanism to focus on relevant parts of the input data. Here we focus on the self-attention module due to its effectiveness. Let $X_{n\times d}$ denote the input of a n-sequence of d-dimensional embeddings, and let $W^Q_{d\times d_q}, W^K_{d\times d_k}$ and $W^V_{d\times d_v}$ be the weight matrices where $d_q = d_k$. Then the self-attention map outputs

$$Softmax\left(\frac{XW^Q(W^K)'X'}{\sqrt{d_k}}\right)XW^V$$

where the $Softmax(\cdot)$ is applied to each row of its input and A' denotes the transpose of a matrix A.

Example 4.3 (Permutation within Attention map). Consider two attention maps $f(x;\theta_1)$ and $f(x;\theta_2)$ with $f(x;\theta_i) = Softmax(XW^Q_i(W^K_i)'X'/\sqrt{d_k})XW^V_i$ for $i = 1, 2$. Then $f(\cdot;\theta_1)$ and $f(\cdot;\theta_2)$ are functionally equivalent if there exists $d_k \times d_k$ permutation matrix P such that

$$W^Q_2 P = W^Q_1 \quad \text{and} \quad W^K_2 P = W^K_1.$$

For the attention module, there is no activation function between the key and query matrices. The relevant symmetry can be considered for any equivalent linear maps $W^Q(W^K)'$. In addition, the output of $Softmax$ operator is invariant to the row shift of its input, which also leaves the possibility to further reduce the complexity of attention modules to understand the overparameterization of large language models.

5. Implications to generalization and optimization

In this section, we introduce the relevance and highlight the usefulness of our study to both generalization and optimization via empirical risk minimization (ERM) framework.

The goal of ERM is to find the target function f_0, which represents the true relationship between the inputs and outputs, and is typically defined as the minimizer (can be unbounded) of some risk $\mathcal{R}(\cdot)$, i.e., $f_0 := \arg\min_f \mathcal{R}(f)$. However, since the target function is unknown, we can only approximate it using a predefined hypothesis space $\mathcal{F}$, such as the class of neural networks parameterized by θ in deep learning, i.e., $\mathcal{F}(\Theta) = \{f_\theta(\cdot) = f(\cdot;\theta) : \theta \in \Theta\}$. Then the "best in class" estimator is defined by $f_{\theta^*} = \arg\min_{f\in\mathcal{F}_\Theta} \mathcal{R}(f)$. It's worth noting that the risk function $\mathcal{R}$ is defined with respect to the distribution of the data, which is unknown in practice. Instead, only a sample with size n is available, and the empirical risk $\mathcal{R}_n$ can be defined and minimized to obtain an empirical risk minimizer (ERM), i.e., $f_{\theta_n} \in \arg\min_{f\in\mathcal{F}_\Theta} \mathcal{R}_n(f)$. Finally, optimization algorithms such as SGD and ADAM, lead us to the estimator obtained in practice, i.e., $f_{\hat{\theta}_n}$. The generalization error of $f_{\hat{\theta}_n}$ can be defined and decomposed as (Mohri et al., 2018):

$$\underbrace{\mathcal{R}(f_{\hat{\theta}_{n,opt}}) - \mathcal{R}(f_0)}_{\text{generalization error}} = \underbrace{\mathcal{R}(f_{\hat{\theta}_n}) - \mathcal{R}(f_{\theta_n})}_{\text{optimization error}} + \underbrace{\mathcal{R}(f_{\theta_n}) - \mathcal{R}(f_{\theta^*})}_{\text{estimation error}} + \underbrace{\mathcal{R}(f_{\theta^*}) - \mathcal{R}(f_0)}_{\text{approximation error}}.$$

The estimation error is closely related to the complexity of the function class $\mathcal{F}(\Theta)$ and the sample size n. pecifically, for a wide range of problems such as regression and classification, the estimation error is $\mathcal{O}((\log\{\mathcal{N}(\mathcal{F}(\Theta), 1/n, \|\cdot\|_\infty)\}/n)^k)$ for $k = 1/2$ or 1 (Bartlett et al., 2019; Kohler & Langer, 2021; Shen et al., 2022; Jiao et al., 2023). Our results improve the estimation error by subtracting at least $\log(d_1! \cdots d_L!)$ from the numerator $(\cdot/n)^k$ compared to existing results.

The approximation error depends on the expressive power of networks $\mathcal{F}(\Theta)$ and the features of the target f_0, such as its input dimension d and smoothness β. Typical results for the bounds of the approximation error are $\mathcal{O}((L\mathcal{W})^{-\beta/d})$ (Yarotsky, 2017; 2018; Petersen & Voigtlaender, 2018; Lu et al., 2021b) where L and $\mathcal{W}$ denote the depth and width of the neural network. However, it is unclear how our reduced covering number bounds will improve the approximation error based on current theories.

Regarding the optimization error, due to the high nonconvexity and complexity of deep learning problems, quantitative analysis based on current theories is limited. Even proving convergence (to stationary points) of existing methods is a difficult task (Sun, 2020). However, we found that the symmetric structure of the parameter space can facilitate optimization. To be specific, our Theorem 5.1 in the following indicates that considering the symmetry structure of the deep network parameter space can make the probability of achieving zero (or some level of) optimization error $(d_1! \cdots d_L!)$ times larger.

For a deep neural network in (3), we say two rows in the parameters $\theta^{(l)} := (W^{(l)}; b^{(l)})$ in the lth hidden layer are identical if the two rows of the concatenated matrix $(W^{(l)}; b^{(l)})$ are identical. Here we concatenate the weight matrix $W^{(l)}$ and bias vector $b^{(l)}$ by $(W^{(l)}; b^{(l)})$ due to the one-one correspondence of the rows in $W^{(l)}$ and $b^{(l)}$. Specifically, if the

lth layer of the network is activated by σ, then the ith row of the output vector $\sigma(W^{(l)}x+b^{(l)})$ is given by $\sigma(W_i^{(l)}x+b_i^{(l)})$ where $W_i^{(l)}$ and $b_i^{(l)}$ denote the ith row of $W^{(l)}$ and $b^{(l)}$ respectively. We let d_l^* denote the number of distinct permutations of rows in $\theta^{(l)}$, and let $(d_1^*, \dots, d_L^*)$ collect the number of distinct permutations in the hidden layers of the network parameterization $\theta = (\theta^{(1)}, \dots, \theta^{(L)})$. We let $\Delta_{\min}(\theta)$ and $\Delta_{\max}(\theta)$ denote the minimum and maximum of the L_∞ norm of distinct rows in $\theta^{(l)}$ over $l \in 1, \dots, L$ (see Definition (8) and (9) in Appendix for details). Then we have the following result.

Theorem 5.1. *Suppose we have an ERM $f_{\theta_n}(\cdot) = f(\cdot; \theta_n)$ with parameter θ_n having $(d_1^*, \dots, d_L^*)$ distinct permutations and $\Delta_{\min}(\theta_n) = \delta$. For any optimization algorithm $\mathcal{A}$, if it guarantees producing a convergent solution of θ_n when its initialization $\theta_n^{(0)}$ satisfies $\Delta_{\max}(\theta_n^{(0)} - \theta_n) \leq \delta/2$, then any initialization scheme that uses identical random distributions for the entries of weights and biases within a layer will produce a convergent solution with probability at least $d_1^* \times \cdots \times d_L^* \times \mathbb{P}(\Delta_{\max}(\theta^{(0)} - \theta_n) \leq \delta/2)$. Here, $\theta^{(0)}$ denotes the random initialization, and $\mathbb{P}(\cdot)$ is with respect to the randomness from initialization.*

Theorem 5.1 can be understood straightforwardly. By Theorem 2.5, if f_{θ_n} parameterized by θ_n is a solution, then $f_{\tilde{\theta}_n}$ parameterized by $\tilde{\theta}_n$ (the permuted θ_n) is also a solution. The conditions in Theorem 2.5 ensure that the convergent regions for permutation-implemented solutions are disjoint. Thus, the probability fro convergence can be multiplied by the number of distinct permutations. These conditions can hold true under specific scenarios. For instance, when the loss is locally (strongly) convex within the $(\delta/2)$-neighborhood under $\Delta_{\max}$ of a global solution θ_n, then (stochastic) gradient descent algorithms $\mathcal{A}$ can guarantee convergence to the solution if the initialization $\theta_n^{(0)}$ falls within its $(\delta/2)$-neighborhood. It is also worth noting that when the parameter space is $\Theta = [-B, B]^{\mathcal{S}}$, the optimization problem is equivalent when restricted to the effective parameter space Θ_0, as defined in section 3.2. The volume of Θ_0 is $(2B)^{\mathcal{S}}/(d_1! \cdots d_L!)$. Specifically, when B is fixed, $(2B)^{\mathcal{S}}/(d_1! \cdots d_L!)$ approaches zero when $d_l \to \infty$ for any $l = 1, \dots, L$. Remarkably, increasing the width of neural networks leads to the effective parameter space's volume tending towards zero. As a result, this may explain the observations in (Frankle & Carbin, 2019; Allen-Zhu et al., 2019; Du et al., 2019) where overparameterized networks tend to be easier to train. In (Simsek et al., 2021), the geometry (in terms of manifold and connected affine subspace) of sets of minima and critical points in deep learning was described, which also indicates that overparameterized networks bear more minima solutions, thereby facilitating optimization.

The landscape of the loss surface in deep learning has been studied by considering the symmetry of the parameter space in several works. Specifically, (Brea et al., 2019) discovered that permutation critical points are embedded in high-dimensional flat plateaus and proved that all permutation points in a given layer are connected with equal-loss paths. (Entezari et al., 2021) conjectured that SGD solutions will likely have no barrier in the linear interpolation between them if the permutation invariance of neural networks is taken into account. (Ainsworth et al., 2022) further explored the role of permutation symmetries in the linear mode connectivity of SGD solutions, and argued that neural network loss landscapes often contain (nearly) a single basin after accounting for all possible permutation symmetries of hidden units. Subsequently, (Jordan et al., 2022) proposed methods to mitigate the variance collapse phenomenon that occurs in the interpolated networks, and to improve their empirical performance. Additionally, optimization algorithms for deep learning have been proposed to enhance training based on the symmetry of network parameter space (Badrinarayanan et al., 2015b; Cho & Lee, 2017; Meng et al., 2018; Navon et al., 2023).

Remark 5.2. The popular initialization schemes, including the Xavier and He's methods, use normal random numbers to initialize the entries of weight matrices and bias vectors identically within a layer (Glorot & Bengio, 2010; He et al., 2015; Shang et al., 2016; Reddi et al., 2019). By Theorem 5.1, these initializations reduce the optimization difficulty due to the permutation invariance property.

6. Conclusion

In this work, we quantitatively characterized the redundancy in the parameterization of deep neural networks based on functional equivalence, and derived a tighter complexity upper bound of the covering number, which is explicit and holds for networks with bias vectors and general activations. We also explored functional equivalence in convolutional, residual and attention-based networks. We discussed the implications for understanding generalization and optimization. Specifically, we found that permutation equivalence can indicate a reduced theoretical complexity of both estimation and optimizations in deep learning.

A limitation of our work is that we only considered permutation invariance, neglecting sign flip and scaling invariance, which may be relevant for specific activations. Furthermore, functional equivalence in practice may be limited to a finite sample, potentially resulting in reduced complexity. Future research could explore the effects of sign flip and scaling invariance and investigate advanced optimization algorithms or designs for deep learning. We also acknowledge that the importance of deriving the lower bound of the covering numbers. We intend to pursue these areas of study to enhance our understanding in the future.

Acknowledgements

The author thanks the anonymous reviewers for their valuable comments and suggestions that have improved the quality of this paper. Guohao Shen's research is (partially) supported by the Hong Kong Research Grants Council (Grant No. 15305523) and research grants from The Hong Kong Polytechnic University.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Ainsworth, S. K., Hayase, J., and Srinivasa, S. Git re-basin: Merging models modulo permutation symmetries. *arXiv preprint arXiv:2209.04836*, 2022.

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

Anthony, M., Bartlett, P. L., Bartlett, P. L., et al. *Neural network learning: Theoretical foundations*, volume 9. cambridge university press Cambridge, 1999.

Badrinarayanan, V., Mishra, B., and Cipolla, R. Symmetry-invariant optimization in deep networks. *arXiv preprint arXiv:1511.01754*, 2015a.

Badrinarayanan, V., Mishra, B., and Cipolla, R. Understanding symmetries in deep networks. *arXiv preprint arXiv:1511.01029*, 2015b.

Bartlett, P. L., Foster, D. J., and Telgarsky, M. J. Spectrally-normalized margin bounds for neural networks. *Advances in neural information processing systems*, 30, 2017.

Bartlett, P. L., Harvey, N., Liaw, C., and Mehrabian, A. Nearly-tight vc-dimension and pseudodimension bounds for piecewise linear neural networks. *The Journal of Machine Learning Research*, 20(1):2285–2301, 2019.

Baum, E. and Haussler, D. What size net gives valid generalization? *Advances in neural information processing systems*, 1, 1988.

Belkin, M., Hsu, D., Ma, S., and Mandal, S. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019.

Bona-Pellissier, J., Bachoc, F., and Malgouyres, F. Parameter identifiability of a deep feedforward relu neural network. *arXiv preprint arXiv:2112.12982*, 2021.

Brea, J., Simsek, B., Illing, B., and Gerstner, W. Weight-space symmetry in deep networks gives rise to permutation saddles, connected by equal-loss valleys across the loss landscape. *arXiv preprint arXiv:1907.02911*, 2019.

Bui Thi Mai, P. and Lampert, C. Functional vs. parametric equivalence of relu networks. In *8th International Conference on Learning Representations*, 2020.

Chen, A. M., Lu, H.-m., and Hecht-Nielsen, R. On the geometry of feedforward neural network error surfaces. *Neural computation*, 5(6):910–927, 1993.

Cho, M. and Lee, J. Riemannian approach to batch normalization. *Advances in Neural Information Processing Systems*, 30, 2017.

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

Dereich, S. and Kassing, S. On minimal representations of shallow relu networks. *Neural Networks*, 148:121–128, 2022.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Du, S., Lee, J., Li, H., Wang, L., and Zhai, X. Gradient descent finds global minima of deep neural networks. In *International conference on machine learning*, pp. 1675–1685. PMLR, 2019.

Dudley, R. M. The sizes of compact subsets of hilbert space and continuity of gaussian processes. *Journal of Functional Analysis*, 1(3):290–330, 1967.

Dudley, R. M. Universal donsker classes and metric entropy. In *Selected Works of RM Dudley*, pp. 345–365. Springer, 2010.

Elbrächter, D. M., Berner, J., and Grohs, P. How degenerate is the parametrization of neural networks with the relu activation function? *Advances in Neural Information Processing Systems*, 32, 2019.

Entezari, R., Sedghi, H., Saukh, O., and Neyshabur, B. The role of permutation invariance in linear mode connectivity of neural networks. *arXiv preprint arXiv:2110.06296*, 2021.

Fang, Z., Feng, H., Huang, S., and Zhou, D.-X. Theory of deep convolutional neural networks ii: Spherical analysis. *Neural Networks*, 131:154–162, 2020.

Fefferman, C. and Markel, S. Recovering a feed-forward net from its output. *Advances in neural information processing systems*, 6, 1993.

Frankle, J. and Carbin, M. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJl-b3RcF7.

Glorot, X. and Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256. JMLR Workshop and Conference Proceedings, 2010.

Glorot, X., Bordes, A., and Bengio, Y. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 315–323. JMLR Workshop and Conference Proceedings, 2011.

Godfrey, C., Brown, D., Emerson, T., and Kvinge, H. On the symmetries of deep learning models and their internal representations. *Advances in Neural Information Processing Systems*, 35:11893–11905, 2022.

Goldberg, P. and Jerrum, M. Bounding the vapnik-chervonenkis dimension of concept classes parameterized by real numbers. In *Proceedings of the sixth annual conference on Computational learning theory*, pp. 361–369, 1993.

Golowich, N., Rakhlin, A., and Shamir, O. Size-independent sample complexity of neural networks. In *Conference On Learning Theory*, pp. 297–299. PMLR, 2018.

Grigsby, J. E., Lindsey, K., and Masden, M. Local and global topological complexity measures of relu neural network functions. *arXiv preprint arXiv:2204.06062*, 2022a.

Grigsby, J. E., Lindsey, K., Meyerhoff, R., and Wu, C. Functional dimension of feedforward relu neural networks. *arXiv preprint arXiv:2209.04036*, 2022b.

Gunasekar, S., Lee, J., Soudry, D., and Srebro, N. Characterizing implicit bias in terms of optimization geometry. In *International Conference on Machine Learning*, pp. 1832–1841. PMLR, 2018a.

Gunasekar, S., Lee, J. D., Soudry, D., and Srebro, N. Implicit bias of gradient descent on linear convolutional networks. *Advances in neural information processing systems*, 31, 2018b.

Hanin, B. and Rolnick, D. Deep relu networks have surprisingly few activation patterns. *Advances in neural information processing systems*, 32, 2019.

Haussler, D. Sphere packing numbers for subsets of the boolean n-cube with bounded vapnik-chervonenkis dimension. *Journal of Combinatorial Theory, Series A*, 69 (2):217–232, 1995.

He, K., Zhang, X., Ren, S., and Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

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

Hecht-Nielsen, R. On the algebraic structure of feedforward network weight spaces. In *Advanced Neural Computers*, pp. 129–135. Elsevier, 1990.

Hertrich, C., Basu, A., Di Summa, M., and Skutella, M. Towards lower bounds on the depth of relu neural networks. *Advances in Neural Information Processing Systems*, 34: 3336–3348, 2021.

Hornik, K. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257, 1991.

Jiao, Y., Shen, G., Lin, Y., and Huang, J. Deep nonparametric regression on approximate manifolds: Nonasymptotic error bounds with polynomial prefactors. *The Annals of Statistics*, 51(2):691–716, 2023.

Jordan, K., Sedghi, H., Saukh, O., Entezari, R., and Neyshabur, B. Repair: Renormalizing permuted activations for interpolation repair. *arXiv preprint arXiv:2211.08403*, 2022.

Kohler, M. and Langer, S. On the rate of convergence of fully connected deep neural network regression estimates. *The Annals of Statistics*, 49(4):2231–2249, 2021.

Kůrkov'a, V. and Kainen, P. C. Functionally equivalent feedforward neural networks. *Neural Computation*, 6(3): 543–558, 1994.

Li, X., Lu, J., Wang, Z., Haupt, J., and Zhao, T. On tighter generalization bound for deep neural networks: Cnns, resnets, and beyond. *arXiv preprint arXiv:1806.05159*, 2018.

Lin, S. and Zhang, J. Generalization bounds for convolutional neural networks. *arXiv preprint arXiv:1910.01487*, 2019.

Lu, J., Shen, Z., Yang, H., and Zhang, S. Deep network approximation for smooth functions. *SIAM Journal on Mathematical Analysis*, 53(5):5465–5506, 2021a.

Lu, J., Shen, Z., Yang, H., and Zhang, S. Deep network approximation for smooth functions. *SIAM Journal on Mathematical Analysis*, 53(5):5465–5506, 2021b.

Lu, Z., Pu, H., Wang, F., Hu, Z., and Wang, L. The expressive power of neural networks: A view from the width. *Advances in neural information processing systems*, 30, 2017.

Mao, T., Shi, Z., and Zhou, D.-X. Theory of deep convolutional neural networks iii: Approximating radial functions. *Neural Networks*, 144:778–790, 2021.

Martinelli, F., Simsek, B., Brea, J., and Gerstner, W. Expand-and-cluster: exact parameter recovery of neural networks. *arXiv preprint arXiv:2304.12794*, 2023.

Meng, Q., Zheng, S., Zhang, H., Chen, W., Ye, Q., Ma, Z.-M., Yu, N., and Liu, T.-Y. G-sgd: Optimizing relu neural networks in its positively scale-invariant space. In *International Conference on Learning Representations*, 2018.

Mohri, M., Rostamizadeh, A., and Talwalkar, A. *Foundations of machine learning*. MIT press, 2018.

Navon, A., Shamsian, A., Achituve, I., Fetaya, E., Chechik, G., and Maron, H. Equivariant architectures for learning in deep weight spaces. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 25790–25816. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/navon23a.html.

Neyshabur, B. Implicit regularization in deep learning. *arXiv preprint arXiv:1709.01953*, 2017.

Neyshabur, B., Tomioka, R., and Srebro, N. In search of the real inductive bias: On the role of implicit regularization in deep learning. *arXiv preprint arXiv:1412.6614*, 2014.

Neyshabur, B., Salakhutdinov, R. R., and Srebro, N. Path-sgd: Path-normalized optimization in deep neural networks. *Advances in neural information processing systems*, 28, 2015a.

Neyshabur, B., Tomioka, R., and Srebro, N. Norm-based capacity control in neural networks. In *Conference on learning theory*, pp. 1376–1401. PMLR, 2015b.

Neyshabur, B., Bhojanapalli, S., McAllester, D., and Srebro, N. Exploring generalization in deep learning. *Advances in neural information processing systems*, 30, 2017a.

Neyshabur, B., Bhojanapalli, S., and Srebro, N. A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. *arXiv preprint arXiv:1707.09564*, 2017b.

Neyshabur, B., Li, Z., Bhojanapalli, S., LeCun, Y., and Srebro, N. The role of over-parametrization in generalization of neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=BygfghAcYX.

Novak, R., Bahri, Y., Abolafia, D. A., Pennington, J., and Sohl-Dickstein, J. Sensitivity and generalization in neural networks: an empirical study. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=HJC2SzZCW.

Ongie, G., Willett, R., Soudry, D., and Srebro, N. A function space view of bounded norm infinite width relu nets: The multivariate case. *arXiv preprint arXiv:1910.01635*, 2019.

Petersen, P. and Voigtlaender, F. Optimal approximation of piecewise smooth functions using deep relu neural networks. *Neural Networks*, 108:296–330, 2018.

Petersen, P., Raslan, M., and Voigtlaender, F. Topological properties of the set of functions generated by neural networks of fixed size. *Foundations of computational mathematics*, 21:375–444, 2021.

Petzka, H., Trimmel, M., and Sminchisescu, C. Notes on the symmetries of 2-layer relu-networks. In *Proceedings of the northern lights deep learning workshop*, volume 1, pp. 6–6, 2020.

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. Improving language understanding by generative pre-training. 2018.

Razin, N. and Cohen, N. Implicit regularization in deep learning may not be explainable by norms. *Advances in neural information processing systems*, 33:21174–21187, 2020.

Reddi, S. J., Kale, S., and Kumar, S. On the convergence of adam and beyond. *arXiv preprint arXiv:1904.09237*, 2019.

Rolnick, D. and Kording, K. Reverse-engineering deep relu networks. In *International Conference on Machine Learning*, pp. 8178–8187. PMLR, 2020.

Shang, W., Sohn, K., Almeida, D., and Lee, H. Understanding and improving convolutional neural networks via concatenated rectified linear units. In *international conference on machine learning*, pp. 2217–2225. PMLR, 2016.

Shen, G., Jiao, Y., Lin, Y., and Huang, J. Approximation with cnns in sobolev space: with applications to classification. *Advances in Neural Information Processing Systems*, 35:2876–2888, 2022.

Simsek, B., Ged, F., Jacot, A., Spadaro, F., Hongler, C., Gerstner, W., and Brea, J. Geometry of the loss landscape in overparameterized neural networks: Symmetries and invariances. In *International Conference on Machine Learning*, pp. 9722–9732. PMLR, 2021.

Stock, P. and Gribonval, R. An embedding of relu networks and an analysis of their identifiability. *Constructive Approximation*, pp. 1–47, 2022.

Stock, P., Graham, B., Gribonval, R., and Jégou, H. Equi-normalization of neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=r1gEqiC9FX.

Sun, R.-Y. Optimization for deep learning: An overview. *Journal of the Operations Research Society of China*, 8 (2):249–294, 2020.

Sussmann, H. J. Uniqueness of the weights for minimal feedforward nets with a given input-output map. *Neural networks*, 5(4):589–593, 1992.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Yarotsky, D. Error bounds for approximations with deep relu networks. *Neural Networks*, 94:103–114, 2017.

Yarotsky, D. Optimal approximation of continuous functions by very deep relu networks. In *Conference on Learning Theory*, pp. 639–649. PMLR, 2018.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Sy8gdB9xx.

Zhou, D.-X. Theory of deep convolutional neural networks: Downsampling. *Neural Networks*, 124:319–327, 2020a.

Zhou, D.-X. Universality of deep convolutional neural networks. *Appl. Comput. Harmon. Anal.*, 48(2):787–794, 2020b.

In this Appendix, we present the technical details of the proof of theorems and provide supporting definition and lemmas.

A. Proof of Theorems

In the proofs, we adopt the following notation. We introduce the L^∞-norm of a collection of parameters $\theta = (W^{(1)}, b^{(1)}, \dots, W^{(L)}, b^{(L)}, W^{(L+1)}, b^{(L+1)})$. We define the infinity norm of the collection of parameters by $\|\theta\|_\infty = \|(W^{(1)}, b^{(1)}, \dots, W^{(L)}, b^{(L)}, W^{(L+1)}, b^{(L+1)})\|_\infty = \max\{\max_{l=1,\dots,L} \|W^{(l)}\|_\infty, \max_{l=1,\dots,L} \|b^{(l)}\|_\infty\}$. Here, $\|\cdot\|_\infty$ denotes the maximum absolute value of a vector or matrix. We let $\|\cdot\|_2$ denote the L^2 norm of a vector. For any matrix $A \in \mathbb{R}^{m\times n}$, the spectral norm of A is denoted by $\|A\|_2 = \max_{x\neq 0} \|Ax\|_2/\|x\|_2$, defined as the largest singular value of A or the square root of the largest eigenvalue of the matrix $A^\top A$. Also we have $\|A\|_2 \leq \sqrt{mn}\|A\|_\infty$.

A.1. Proof of Theorem 2.5

Proof. The proof is straight forward based on the properties of permutation matrix and element-wise activation functions. Firstly, for any $n \times n$ permutation matrix P, it is true that $PP^\top = P^\top P = I_n$ where I_n is the $n \times n$ identity matrix. Secondly, for any element-wise activation function σ, any n-dimensional vector $x \in \mathbb{R}^n$ and any $n \times n$ permutation matrix P, it is easy to check

$$\sigma(Px) = P\sigma(x).$$

Then for any deep neural network

$$f(x;\theta) = W^{(L+1)}\sigma_L(W^{(L)} \cdots \sigma_1(W_1^{(1)}x + b_1^{(1)}) \cdots) + b_1^{(L)}) + b^{(L+1)},$$

and any permutation matrices $P_1, \dots, P_L$, it is easy to check

$$\begin{aligned} &W^{(L+1)}P_L^\top\sigma_L(P_L W^{(L)} P_{L-1}^\top \cdots \sigma_1(P_1 W^{(1)}x + P_1 b^{(1)}) \cdots) + P_L b^{(L)}) + b^{(L+1)} \\ &= W^{(L+1)}\sigma_L(W^{(L)} \cdots \sigma_1(W^{(1)}x + b^{(1)}) \cdots) + b^{(L)}) + b^{(L+1)}, \end{aligned}$$

which completes the proof. □

A.2. Proof of Theorem 3.3

Proof. Firstly, by the property of permutation invariance, we know that the neural networks $\{f(x;\theta) = W^{(2)}\sigma_1(W^{(1)}x + b^{(1)}) + b^{(2)} : \theta \in \Theta_0\}$ parameterized by Θ_0 contains all the functions in $\{f(x;\theta) = W^{(2)}\sigma_1(W^{(1)}x + b^{(1)}) + b^{(2)} : \theta \in \Theta\}$. The covering numbers of these two class of functions are the same. Then we can consider the covering number of $\{f(x;\theta) = W^{(2)}\sigma_1(W^{(1)}x + b^{(1)}) + b^{(2)} : \theta \in \Theta_0\}$ where

$$\Theta_0 := \{\theta \in [-B, B]^{\mathcal{S}} : b_1^{(1)} \geq b_2^{(1)} \geq \cdots \geq b_{d_1}^{(1)}\}.$$

Recall that for a single hidden layer neural network $f(\cdot;\theta)$ parameterized by $\theta = (W^{(1)}, b^{(1)}, W^{(2)}, b^{(2)})$, the parameters $W^{(1)} \in \mathbb{R}^{d_1\times d_0}$, $b^{(1)} \in \mathbb{R}^{d_1}$, $W^{(2)} \in \mathbb{R}^{1\times d_1}$, and $b^{(2)} \in \mathbb{R}^1$ have components bounded by B. We start by considering the covering number of the activated linear transformations

$$\mathcal{H} := \{\sigma_1 \circ \mathcal{A}_1 : (W^{(1)}, b^{(1)}) \in \Theta_0^{(1)}\},$$

where $\Theta_0^{(1)} = \{[-B, B]^{d_1\times d_0 + d_1}, b_1^{(1)} \geq b_2^{(1)} \geq \cdots \geq b_{d_1}^{(1)}\}$, σ_1 is a ρ-Lipschitz activation function on $[-BB_x - B, BB_x + B]$, and $\mathcal{A}_1(x) = W^{(1)}x + b^{(1)}$. Here $\mathcal{A}_1$ output d_1-dimensional vectors, and we define $\|\mathcal{A}_1\|_\infty := \sup_{x\in\mathcal{X}} \|\sigma_1 \circ \mathcal{A}_1(x)\|_2$ for vector-valued functions.

Let $\epsilon_1 > 0$ be a real number and $\Theta_{0,\epsilon_1}^{(1)} = \{(W_j^{(1)}, b_j^{(1)})\}_{j=1}^{N_1}$ be a minimal ϵ_1-covering of $\Theta_0^{(1)}$ under $\|\cdot\|_\infty$ norm with covering number $\mathcal{N}_1 = \mathcal{N}(\Theta_0^{(1)}, \epsilon_1, \|\cdot\|_\infty)$. Then for any $(W^{(1)}, b^{(1)}) \in \Theta_0^{(1)}$, there exist a $(W_j^{(1)}, b_j^{(1)})$ such that $\|(W_j^{(1)} - W^{(1)}, b_j^{(1)} - b^{(1)})\|_\infty \leq \epsilon_1$. For any $x \in \mathcal{X}$ and $(W^{(1)}, b^{(1)}) \in \Theta_0^{(1)}$, it is not hard to check $\|W^{(1)}x + b^{(1)}\|_2 \leq$

$B(\sqrt{d_0 d_1}B_x + 1)$. Then we have

$$\begin{aligned}\|\sigma_1(W_i^{(1)}x + b_i^{(1)}) - \sigma_1(W_j^{(1)}x + b_j^{(1)})\|_2 \leq & \rho\|(W^{(1)}x + b^{(1)}) - (W_j^{(1)}x + b_j^{(1)})\|_2 \\ \leq & \rho\|(W^{(1)} - W_j^{(1)})x\|_2 + \rho\|(b^{(1)} - b_j^{(1)})\|_2 \\ \leq & \rho\sqrt{d_0 d_1}\|W^{(1)} - W_j^{(1)}\|_\infty \|x\|_2 + \rho\sqrt{d_1}\|(b^{(1)} - b_j^{(1)})\|_\infty \\ \leq & \rho\epsilon_1(\sqrt{d_0 d_1}B_x + \sqrt{d_1}).\end{aligned}$$

This implies that

$$\mathcal{H}_1 = \{\sigma_1 \circ \mathcal{A}_1 : (W^{(1)}.b^{(1)}) \in \Theta_{0,\epsilon_1}^{(1)}\}$$

is a set with no more than $\mathcal{N}_1$ elements and it covers $\mathcal{H}$ under $\|\cdot\|_\infty$ norm with radius $\epsilon_1\rho(\sqrt{d_0 d_1}B_x + \sqrt{d_1})$. By Lemma B.4, the covering number $\mathcal{N}_1 = \mathcal{N}(\Theta_0^{(1)}, \epsilon, \|\cdot\|_\infty) \leq \text{Volume}(\Theta_0^{(1)}) \times (2/\epsilon_1)^{d_1 \times d_0 + d_1} = (4B/\epsilon_1)^{d_1 \times d_0 + d_1}/d_1!$.

Now, let $\epsilon_2 > 0$ be a real number and $\Theta_{0,\epsilon_2}^{(2)} = \{(W_j^{(2)}, b_j^{(2)})\}_{j=1}^{N_2}$ be a minimal ϵ_2-covering of $\Theta_0^{(2)}$ under $\|\cdot\|_\infty$ norm with $\mathcal{N}_2 = \mathcal{N}(\Theta_0^{(2)}, \epsilon_2, \|\cdot\|_\infty) \leq \text{Volume}(\Theta_0^{(2)}) \times (2/\epsilon_2)^{d_1 \times 1 + 1} = (4B/\epsilon_2)^{d_1+1}$. Also we construct a class of functions by

$$\mathcal{H}_2 = \{\mathcal{A}_2 \circ h : h \in \mathcal{H}_1, (W^{(2)}, b^{(2)}) \in \Theta_{0,\epsilon_2}^{(2)}\},$$

where $\mathcal{A}_2(x) = W^{(2)}x + b^{(2)}$. Now for any $f = \mathcal{A}_2 \circ \sigma_1 \circ \mathcal{A}_1$ parameterized by $\theta = (W^{(1)}, b^{(1)}, W^{(2)}, b^{(2)}) \in \Theta_0$, by the definition of covering, there exists $h_j \in \mathcal{H}_1$ such that $\|h_j - \sigma_1 \circ \mathcal{A}_1\|_\infty \leq \rho\epsilon_1(\sqrt{d_0 d_1}B_x + \sqrt{d_1})$ and there exists $(W_k^{(2)}, b_k^{(2)}) \in \Theta_{0,\epsilon_2}^{(2)}$ such that $\|(W_k^{(2)} - W^{(2)}, b_k^{(2)} - b^{(2)})\|_\infty \leq \epsilon_2$. Then fro any $x \in \mathcal{X}$

$$\begin{aligned}&\|f(x) - W_k^{(2)}h_j(x) + b_k^{(2)}\|_2 \\ =&\|W^{(2)}\sigma_1 \circ \mathcal{A}_1(x) + b^{(2)} - W_k^{(2)}h_j(x) - b_k^{(2)}\|_2 \\ \leq&\|W^{(2)}\sigma_1 \circ \mathcal{A}_1(x) - W_k^{(2)}h_j(x)\|_2 + \|b^{(2)} - b_k^{(2)}\|_2 \\ \leq&\|W^{(2)}\sigma_1 \circ \mathcal{A}_1(x) - W^{(2)}h_j(x)\|_2 + \|W^{(2)}h_j(x) - W_k^{(2)}h_j(x)\|_2 + \sqrt{d_2}\epsilon_2 \\ \leq&\sqrt{d_1}B\rho\epsilon_1(\sqrt{d_0 d_1}B_x + \sqrt{d_1}) + \epsilon_2\sqrt{d_1}B(\sqrt{d_0 d_1}B_x + \sqrt{d_1}) + \epsilon_2 \\ \leq&2\sqrt{d_0}d_1 B(B_x + 1)[\rho\epsilon_1 + \epsilon_2],\end{aligned}$$

which implies that $\mathcal{H}_2$ is a $(2\sqrt{d_0}d_1 B(B_x + 1)[\rho\epsilon_1 + \epsilon_2])$-covering of the neural networks $\mathcal{F}(1, d_0, d_1, B)$, where there are at most $(4B/\epsilon_1)^{d_1 \times d_0 + d_1}/d_1! \times (4B/\epsilon_2)^{d_1+1}$ elements in $\mathcal{H}_2$. Given $\epsilon > 0$, we take $\epsilon_1 = \epsilon/(4\rho\sqrt{d_0}d_1 B(B_x + 1))$ and $\epsilon_2 = \epsilon/(4\sqrt{d_0}d_1 B(B_x + 1))$, then this implies

$$\begin{aligned}&\mathcal{N}(\mathcal{F}(1, d_0, d_1, B), \epsilon, \|\cdot\|_\infty) \\ \leq&(4B/\epsilon_1)^{d_1 \times d_0 + d_1}/d_1! \times (4B/\epsilon_2)^{d_1+1} \\ =&(16B/\epsilon)^{d_0 \times d_1 + d_1 + d_1 + 1} \times (\rho\sqrt{d_0}d_1 B(B_x + 1))^{d_0 \times d_1 + d_1} \times (\sqrt{d_0}d_1 B(B_x + 1))^{d_1+1}/d_1! \\ =&(16B/\epsilon)^{d_0 \times d_1 + d_1 + d_1 + 1} \times (\sqrt{d_0}d_1 B(B_x + 1))^{d_0 \times d_1 + d_1 + d_1 + 1} \times \rho^{d_0 \times d_1 + d_1}/d_1! \\ =&(16B^2(B_x + 1)\sqrt{d_0}d_1/\epsilon)^{d_0 \times d_1 + d_1 + d_1 + 1} \times \rho^{d_0 \times d_1 + d_1}/d_1!, \\ =&(16B^2(B_x + 1)\sqrt{d_0}d_1/\epsilon)^{\mathcal{S}} \times \rho^{\mathcal{S}_1}/d_1!,\end{aligned}$$

where $\mathcal{S} = d_0 \times d_1 + d_1 + d_1 + 1$ is the total number of parameters in the network and $\mathcal{S}_1$ denotes the number of parameters in the linear transformation from the input to the hidden layer. □

A.3. Proof of Theorem 3.5

Proof. Our proof takes into account permutation equivalence and extends Theorem 3.3 by using mathematical induction. Similar approaches using induction can be found in Lemma A.7 of (Bartlett et al., 2017) and Lemma 14 of (Lin & Zhang, 2019). However, we improve upon their results in three ways. Firstly, we consider the generally defined neural networks

where the bias vector is allowed to appear. Secondly, we express our bound in terms of the width, depth, and size of the neural network as well as the infinity norm of the parameters, instead of the spectral norm of the weight matrices, which can be unknown in practice. Thirdly, we utilize permutation equivalence to derive tighter upper bounds on the covering number.

Step 1. We analyze the effective region of the parameter space

$$\Theta = [-B, B]^{\mathcal{S}},$$

where $\mathcal{S} = \sum_{i=0}^{L} d_i d_{i+1} + d_{i+1}$ is the total number of parameters in the deep neural network. We denote $\mathcal{S}_l = d_{l-1}d_l + d_l$ by the total number of parameters in $(W^{(l)}, B^{(l)})$ in the lth layer, and denote $\Theta^{(l)} = \{(W^{(l)}, b^{l}) : (W^{(l)}, b^{(l)}) \in [-B, B]^{\mathcal{S}_l}\}$ by the parameter space for $l = 1, \ldots, L+1$. By our Theorem 2.5, for any given neural network $f(\cdot; \theta)$ parameterized by

$$\theta = (W^{(1)}, b^{(1)}, \ldots, W^{(L+1)}, b^{(L+1)}) \in \Theta,$$

there exists $P_1, \ldots, P_L$ such that $f(\cdot; \tilde{\theta})$ parameterized by

$$\tilde{\theta} = (P_1 W^{(1)}, P_1 b^{(1)}, \ldots, P_l W^{(l)} P_{l-1}^{\top}, P_l b^{(1)}, \ldots, W^{(L+1)} P_L^{\top}, b^{(L+1)})$$

implements the same input-output function. This implies that there exists a subset Θ_0 of Θ such that neural networks $\{f(\cdot; \theta) : \theta \in \Theta_0\}$ parameterized by Θ_0 contains all the functions in $\{f(\cdot; \theta) : \theta \in \Theta\}$. The covering numbers of these two class of functions are the same. To be specific, the effective parameter space

$$\Theta_0 = \Theta_0^{(1)} \times \Theta_0^{(2)} \times \cdots \times \Theta_0^{(L)} \times \Theta_0^{(L+1)},$$

where

$$\begin{aligned}
\Theta_0^{(1)} =& \{(W^{(1)}, b^{(1)}) \in [-B, B]^{\mathcal{S}_1} : b_1^{(1)} \geq b_2^{(1)} \geq \cdots \geq b_{d_1}^{(1)}\}, \\
\Theta_0^{(l)} =& \{(W^{(l)}, b^{(l)}) \in [-B, B]^{\mathcal{S}_l} : b_1^{(l)} \geq b_2^{(l)} \geq \cdots \geq b_{d_l}^{(l)}\} \text{ for } l = 2, \ldots, L, \\
\Theta_0^{(L+1)} =& \{(W^{(L+1)}, b^{(L+1)}) \in [-B, B]^{\mathcal{S}_{L+1}}\}.
\end{aligned}$$

In the following, we focus on considering the covering number of $\{f(\cdot; \theta) : \theta \in \Theta_0\}$.

Step 2. We start by bounding the covering number for the first activated hidden layer. Let $\mathcal{H}_1 = \{\sigma_1 \circ \mathcal{A}_1 : (W^{(1)}x + b^{(1)}) \in \Theta_0^{(1)}\}$ where $\mathcal{A}_1(x) = W^{(1)}x + b^{(1)}$ is the linear transformation from the input to the first hidden layer. Given any $\epsilon_1 > 0$, in the proof of Theorem 3.3, we have shown that

$$\mathcal{N}_1 := \mathcal{N}(\mathcal{H}_1, \epsilon_1 \rho_1 \sqrt{d_0 d_1}(B_x + 1), \|\cdot\|_\infty) \leq \text{Volume}(\Theta_0^{(1)}) \times (2/\epsilon_1)^{\mathcal{S}_1},$$

and $\|h^{(1)}\|_\infty \leq \rho_1 \sqrt{d_0 d_1} B(B_x + 1)$ for $h^{(1)} \in \mathcal{H}_1$.

Step 3. We use induction to proceed the proof for $l = 1, \ldots, L$. Let $\mathcal{H}_l = \{\sigma_l \circ \mathcal{A}_l \circ \cdots \circ \sigma_1 \circ \mathcal{A}_1 : (W^{(k)}x + b^{(k)}) \in \Theta_0^{(k)}, k = 1, \ldots, l\}$ where $\mathcal{A}_k(x) = W^{(k)}x + b^{(k)}$ is the linear transformation from the $(k-1)$th layer to the kth layer for $k = 1, \ldots, l$. Let $B^{(l)}$ denotes the infinity norm of functions $h^{(l)} \in \mathcal{H}_l$ for $k = 1, \ldots, L$. For any $e_l > 0$, let

$$\tilde{\mathcal{H}}_l = \{h_j^{(l)}\}_{j=1}^{\mathcal{N}(\mathcal{H}_l, e_l, \|\cdot\|_\infty)}$$

be a e_l-covering of $\mathcal{H}_l$ under the $\|\cdot\|_\infty$ norm. For any $\epsilon_{l+1} > 0$, let

$$\Theta_{0,\epsilon_{l+1}}^{(l+1)} = \{(W_t^{(l+1)}, b_t^{(l+1)})\}_{t=1}^{\mathcal{N}(\Theta_0^{(l+1)}, \epsilon_{l+1}, \|\cdot\|_\infty)}$$

be a ϵ_{l+1}-covering of $\Theta_0^{(l+1)}$. Then for any $h^{(l+1)} = \sigma_{l+1} \circ \mathcal{A}_{l+1} \circ h^{(l)} \in \mathcal{H}_{l+1}$ where $\mathcal{A}_{l+1}(x) = W^{(l+1)}x + b^{(l+1)}$, there exists $h_j^{(l)} \in \tilde{\mathcal{H}}_l$ and $(W_t^{(l+1)}, b_t^{(l+1)}) \in \Theta_{0,\epsilon_{l+1}}^{(l+1)}$ such that

$$\|h^{(l)} - h_j^{(l)}\|_\infty \leq e_l$$

and

$$\|(W_t^{(l+1)} - W^{(l+1)}, b_t^{(l+1)} - b^{(l+1)})\|_\infty \le \epsilon_{l+1}.$$

Then for any $x \in \mathcal{X}$, we have

$$\begin{aligned}
&\|h^{(l+1)}(x) - \sigma_{l+1}(W_t^{(l+1)} h_j^{(l)}(x) + b_t^{(l+1)})\|_2 \\
=&\|\sigma_{l+1}(W^{(l+1)} h^{(l)}(x) + b^{(l+1)}) - \sigma_{l+1}(W_s^{(l+1)} h_j^{(l)}(x) - b_s^{(l+1)})\|_2 \\
\le&\rho_{l+1}\|(W^{(l+1)} h^{(l)}(x) + b^{(l+1)}) - (W_s^{(l+1)} h_j^{(l)}(x) - b_s^{(l+1)})\|_2 \\
\le&\rho_{l+1}\|W^{(l+1)} h^{(l)}(x) - W_s^{(l+1)} h_j^{(l)}(x)\|_2 + \rho_{l+1}\sqrt{d_{l+1}}\|b^{(l+1)} - b_s^{(l+1)}\|_\infty \\
\le&\rho_{l+1}\|W^{(l+1)} h^{(l)}(x) - W^{(l+1)} h_j^{(l)}(x)\|_2 + \rho_{l+1}\|W^{(l+1)} h_j^{(l)}(x) - W_s^{(l+1)} h_j^{(l)}(x)\|_2 + \rho_{l+1}\sqrt{d_{l+1}}\epsilon_{l+1} \\
\le&\rho_{l+1}\sqrt{d_l} B e_l + \rho_{l+1}\epsilon_{l+1}\sqrt{d_l} B^{(l)} + \rho_{l+1}\sqrt{d_{l+1}}\epsilon_{l+1} \\
=&\rho_{l+1}(\sqrt{d_l} B e_l + (B^{(l)} + \sqrt{d_{l+1}})\epsilon_{l+1}).
\end{aligned}$$

Then it is proved that the covering number $\mathcal{N}(\mathcal{H}^{(l+1)}, e_{l+1}, \|\cdot\|_\infty)$ with $e_{l+1} = \rho_{l+1}(\sqrt{d_l} B e_l + (B^{(l)} + \sqrt{d_{l+1}})\epsilon_{l+1})$ satisfying

$$\mathcal{N}(\mathcal{H}^{(l+1)}, e_{l+1}, \|\cdot\|_\infty) \le \mathcal{N}(\mathcal{H}^{(l)}, e_l, \|\cdot\|_\infty) \times \mathcal{N}(\Theta_0^{(l+1)}, \epsilon_{l+1}, \|\cdot\|_\infty)$$

for $l = 1, \ldots, L-1$. Recall that in the proof of Theorem 3.3, we have proved that

$$\mathcal{N}(\mathcal{H}^{(1)}, e_1, \|\cdot\|_\infty) \le \mathcal{N}(\Theta_0^{(1)}, \epsilon_1, \|\cdot\|_\infty),$$

where $e_1 = \rho_1\epsilon_1\sqrt{d_0 d_1}(B_x + 1)$, which leads to

$$\mathcal{N}(\mathcal{H}^{(l)}, e_l, \|\cdot\|_\infty) \le \Pi_{i=1}^l \mathcal{N}(\Theta_0^{(l)}, \epsilon_l, \|\cdot\|_\infty),$$

for $l = 1, \ldots, L$.

Step 4. Next we give upper bounds of e_l and $B^{(l)}$ for $l = 1, \ldots, L$. Recall that $B^{(l)}$ denotes the infinity norm of functions $h^{(l)} \in \mathcal{H}_l$ for $k = 1, \ldots, L$. Then it is easy to see that $B^{(l)} \ge \rho_l B$ since we can always take the bias vectors to have components B or $-B$. In addition,

$$B^{(l+1)} = \|\sigma_{l+1} \circ \mathcal{A}_{l+1} \circ h_l\|_\infty \le \rho_{l+1} B(\sqrt{d_l} B^{(l)} + \sqrt{d_{l+1}}) \le 2\rho_{l+1}\sqrt{d_l d_{l+1}} B B^{(l)}.$$

As proved in Theorem 3.3, we know $B^{(1)} \le \rho_1 B\sqrt{d_0 d_1}(B_x + 1)$, then we can get

$$B^{(l)} \le (2B)^l (B_x + 1)\sqrt{d_0}\Pi_{i=1}^l d_i \rho_i / (\sqrt{d_l}).$$

Recall $e_{l+1} = \rho_{l+1}(\sqrt{d_l} B e_l + (B^{(l)} + \sqrt{d_{l+1}})\epsilon_{l+1})$ for $l = 1, \ldots, L-1$, and $e_1 = \rho_1\epsilon_1\sqrt{d_0}(B_x + 1)$, then by simple mathematics

$$\begin{aligned}
e_L &= B^{L-1}\Pi_{i=2}^L \rho_i \sqrt{d_{i-1}} e_1 + \sum_{i=2}^L (B^{(i-1)} + \sqrt{d_i})\epsilon_i \Pi_{j=i}^L \rho_j \sqrt{d_{j-1}} / \sqrt{d_L} \\
&\le B^{L-1}\Pi_{i=1}^L \rho_i \sqrt{d_{i-1}}(B_x + 1)\epsilon_1 + \sum_{i=2}^L (B_x + 1)(2B)^{i-1}\sqrt{d_0}(\Pi_{j=1}^L \rho_j d_j)\epsilon_i / (\sqrt{d_L}) \\
&\le 2\sum_{i=1}^L \sqrt{d_0}(B_x + 1)(2B)^{i-1}(\Pi_{j=1}^L \rho_j d_j)\epsilon_i / \sqrt{d_L}.
\end{aligned}$$

Step 5. Last we construct a covering of $\mathcal{F} = \mathcal{F}(1, d_0, d_1, B) = \{f(x;\theta) = W^{(2)}\sigma_1(W^{(1)}x + b^{(1)}) + b^{(2)} : \theta \in \Theta_0\}$. Let

$$\tilde{\mathcal{H}}_L = \{h_j^{(L)}\}_{j=1}^{\mathcal{N}(\mathcal{H}_L, e_L, \|\cdot\|_\infty)}$$

be a e_L-covering of $\mathcal{H}_L$ under the $\|\cdot\|_\infty$ norm. For any $\epsilon_{L+1} > 0$, let

$$\Theta_{0,\epsilon_{L+1}}^{(L+1)} = \{(W_t^{(L+1)}, b_t^{(L+1)})\}_{t=1}^{\mathcal{N}(\Theta_0^{(L+1)},\epsilon_{L+1},\|\cdot\|_\infty)}$$

be a ϵ_{L+1}-covering of $\Theta_0^{(L+1)}$. Then for any $f = \mathcal{A}_{L+1} \circ h^{(L)} \in \mathcal{F}$ where $\mathcal{A}_{L+1}(x) = W^{(L+1)}x + b^{(L+1)}$, there exists $h_j^{(L)} \in \tilde{\mathcal{H}}_L$ and $(W_t^{(L+1)}, b_t^{(L+1)}) \in \Theta_{0,\epsilon_{L+1}}^{(L+1)}$ such that

$$\|h^{(L)} - h_j^{(L)}\|_\infty \leq e_L$$

and

$$\|(W_t^{(L+1)} - W^{(L+1)}, b_t^{(L+1)} - b^{(L+1)})\|_\infty \leq \epsilon_{L+1}.$$

Then for any $x \in \mathcal{X}$, we have

$$\begin{aligned}
&\|f(x) - W_t^{(L+1)}h_j^{(L)}(x) + b_t^{(L+1)}\|_2 \\
=&\|W^{(L+1)}h^{(L)}(x) + b^{(L+1)} - W_s^{(L+1)}h_j^{(L)}(x) - b_s^{(L+1)}\|_2 \\
\leq&\|(W^{(L+1)}h^{(L)}(x) + b^{(L+1)}) - (W_s^{(L+1)}h_j^{(L)}(x) - b_s^{(L+1)})\|_2 \\
\leq&\|W^{(L+1)}h^{(L)}(x) - W_s^{(L+1)}h_j^{(L)}(x)\|_2 + \|b^{(L+1)} - b_s^{(L+1)}\|_2 \\
\leq&\|W^{(L+1)}h^{(L)}(x) - W^{(L+1)}h_j^{(L)}(x)\|_2 + \|W^{(L+1)}h_j^{(L)}(x) - W_s^{(L+1)}h_j^{(L)}(x)\|_2 + \epsilon_{L+1} \\
\leq&\sqrt{d_L}Be_L + \epsilon_{L+1}\sqrt{d_L}B^{(L)} + \epsilon_{L+1} \\
=&(\sqrt{d_L}Be_L + (\sqrt{d_L}B^{(L)} + 1)\epsilon_{L+1}) \\
\leq&2\sqrt{d_0}\sum_{i=1}^{L}(B_x+1)(2B)^i(\Pi_{j=1}^L\rho_j d_j)\epsilon_i + 2(2B)^L(B_x+1)\sqrt{d_0}(\Pi_{j=1}^L\rho_j d_j)\epsilon_{L+1} \\
\leq&2\sqrt{d_0}(B_x+1)(\Pi_{j=1}^L\rho_j d_j)\sum_{i=1}^{L+1}(2B)^i\epsilon_i.
\end{aligned}$$

Then we know that the covering number $\mathcal{N}(\mathcal{F}, e_{L+1}, \|\cdot\|_\infty)$ with radius

$$e_{L+1} = 2(B_x+1)\sqrt{d_0}(\Pi_{j=1}^L\rho_j d_j)\sum_{i=1}^{L+1}(2B)^{i-1}\epsilon_i$$

satisfies

$$\begin{aligned}
\mathcal{N}(\mathcal{F}, e_{L+1}, \|\cdot\|_\infty) &\leq \mathcal{N}(\mathcal{H}^{(L)}, e_L, \|\cdot\|_\infty) \times \mathcal{N}(\Theta_0^{(L+1)}, \epsilon_{L+1}, \|\cdot\|_\infty) \\
&\leq \Pi_{i=1}^{L+1}\mathcal{N}(\Theta_0^{(i)}, \epsilon_{L+1}, \|\cdot\|_\infty) \\
&\leq \Pi_{i=1}^{L+1}\text{Volume}(\Theta_0^{(i)}) \times (2/\epsilon_i)^{\mathcal{S}_i} \\
&= (4B/\epsilon_{L+1})^{\mathcal{S}_{L+1}} \times \Pi_{i=1}^L(4B/\epsilon_i)^{\mathcal{S}_i}/(d_i!).
\end{aligned}$$

Finally, setting $\epsilon_i = \{2(L+1)\sqrt{d_0}(B_x+1)(2B)^i(\Pi_{j=1}^L\rho_j d_j)\}^{-1}\epsilon$ for $i = 1, \ldots, L+1$ leads to an upper bound for the covering number $\mathcal{N}(\mathcal{F}, \epsilon, \|\cdot\|_\infty)$

$$\begin{aligned}
\mathcal{N}(\mathcal{F}, \epsilon, \|\cdot\|_\infty) \leq& (4(L+1)(B_x+1)\sqrt{d_0}(B_x+1)(2B)^{L+2}(\Pi_{j=1}^L\rho_j d_j)/\epsilon)^{\mathcal{S}_{L+1}} \\
&\times \Pi_{i=1}^L(4(L+1)(2B)^{i+1}(\Pi_{j=1}^L\rho_j\sqrt{d_j})/\epsilon)^{\mathcal{S}_i}/(d_i!) \\
\leq& \frac{(4(L+1)\sqrt{d_0}(B_x+1)(2B)^{L+2}(\Pi_{j=1}^L\rho_j d_j)/\epsilon)^{\mathcal{S}}}{d_1! \times d_2! \times \cdots \times d_L!}.
\end{aligned}$$

This completes the proof. □

A.4. Proof of Theorem 5.1

Proof. The proof is straightforward. And we present the proof in three steps. First, we show that if f_{θ_n} is an empirical risk minimizer, then there are at least $d_1^* \times d_2^* \times \cdots \times d_L^*$ empirical risk minimizers with distinct parameterization. Second, we prove that the $(\delta/2)$ neighborhood of these distinct parameterization are disjoint under the L_∞ norm. Lastly, we show that initialization schemes with identical random distribution for weights and bias within layers can indeed increase the probability of convergence for any appropriate optimization algorithms $\mathcal{A}$.

Step 1. Suppose f_{θ_n} is an empirical risk minimizer with parameterization

$$\theta_n = (W_n^{(1)}, b_n^{(1)}, W_n^{(2)}, b_n^{(2)}, \ldots, W_n^{(L)}, b_n^{(L)}, W_n^{(L+1)}, b_n^{(L+1)}).$$

By Theorem 2.5, for any permutation matrices $P_1, \ldots, P_L$,

Step 1. Suppose f_{θ_n} is an empirical risk minimizer with parameterization

$$\theta_n = (W_n^{(1)}, b_n^{(1)}, W_n^{(2)}, b_n^{(2)}, \ldots, W_n^{(L)}, b_n^{(L)}, W_n^{(L+1)}, b_n^{(L+1)}).$$

By Theorem 2.5, for any permutation matrices $P_1, \ldots, P_L$,

$$\tilde{\theta}_n = (P_1 W_n^{(1)}, P_1 b_n^{(1)}, P_2 W_n^{(2)} P_1^\top, P_2 b_n^{(2)}, \ldots, P_L W_n^{(L)}, P_L b_n^{(L)}, W_n^{(L+1)} P_L^\top, b_n^{(L+1)}), \tag{6}$$

will lead to an empirical risk minimizer $f_{\tilde{\theta}_n}$. However, the concatenated matrices $(W_n^{(l)}; b_n^{(l)})$ may have identical rows, and the permutation-implemented matrices $(P_l W_n^{(l)} P_{l-1}^\top; P_l; b_n^{(l)})$ may remain unchanged for some permutation matrices P_l. Let d_l^* denote the number of distinct permutations of the rows in $(W_n^{(l)}; b_n^{(l)})$. Then it is guaranteed that $\{(P_l W_n^{(l)} P_{l-1}^\top; P_l; b_n^{(l)}) : P_l, P_{l-1} \text{ are permutation matrices}\}$ has at least d_l^* distinct elements. Note that $1 \leq d_l^* \leq d_l!$ for $l = 1, \ldots, L$ where d_l is the dimension of the bias vector $b^{(l)}$ as well as the number of rows of $(W_n^{(l)}; b_n^{(l)})$. Specifically, $d_l^* = 1$ if and only if all the entries of $b_n^{(l)}$ are identical and all the rows of $W^{(l)}$ are identical. And $d_l^* = d_l!$ if and only if all the rows of concatenated matrix $(W_n^{(l)}, b_n^{(l)})$ are distinct. Moreover, the distinct elements in $\{(P_l W_n^{(l)} P_{l-1}^\top; P_l; b_n^{(l)}) : P_l, P_{l-1} \text{ are permutation matrices}\}$ can range from 1 to $d_{l-1}! d_l!$. It is 1 if and only if all the entries of $b_n^{(l)}$ are identical and all the entries of $W_n^{(l)}$ are identical; it is $d_{l-1}! d_l!$ if and only if all the entries of $b_n^{(l)}$ are distinct and all the entries of $W_n^{(l)}$ are distinct. Lastly, it is easy to see that $\theta_n \neq \tilde{\theta}_n$ if $(W_n^{(l)}; b_n^{(l)}) \neq (P_l W_n^{(l)} P_{l-1}^\top; P_l; b_n^{(l)})$ for any $l \in \{1, \ldots, L\}$. Then there are at least $d_1^* \times \cdots \times d_L^*$ distinct elements in

$$\tilde{\Theta}_n = \{\tilde{\theta}_n \text{ defined in (6)} : P_1, \ldots P_L \text{ are permutation matrices}\}. \tag{7}$$

Step 2.

Let $\theta = (W^{(1)}, b^{(1)}, W^{(2)}, b^{(2)}, \ldots, W^{(L)}, b^{(L)}, W^{(L+1)}, b^{(L+1)})$ be the collection of parameters of a network and let $\theta_i^{(l)}$ be the ith row of concatenated matrix $\theta^{(l)} = (W^{(l)}; b^{(l)})$. We define

$$\Delta_{\min}(\theta) := \min_{l \in \{1, \ldots, L\}} \left[\min_{i,j \in \{1, \ldots, d_l\}, \theta_i^{(l)} \neq \theta_j^{(l)}} \|\theta_i^{(l)} - \theta_j^{(l)}\|_\infty \right], \tag{8}$$

and

$$\Delta_{\max}(\theta) := \max_{l \in \{1, \ldots, L\}} \left[\max_{i,j \in \{1, \ldots, d_l\}} \|\theta_i^{(l)} - \theta_j^{(l)}\|_\infty \right]. \tag{9}$$

Recall that $\Delta_{\min}(\theta_n) = \delta$. This implies that for any two distinct permutation-implemented $\tilde{\theta}_{n,1}$ and $\tilde{\theta}_{n,2}$ in (7),

$$\Delta_{\max}(\tilde{\theta}_{n,1} - \tilde{\theta}_{n,2}) \geq \delta,$$

where $\Delta_{\max}(\theta_n)$ denote the maximum of the $\|\cdot\|_\infty$ norm of distinct rows in $\theta_n^{(l)}$ over $l \in 1, \ldots, L$. Then the neighborhoods $B_\infty(\tilde{\theta}_n, \delta/2) := \{\theta : \Delta_{\max}(\theta - \tilde{\theta}_n) \leq \delta/2\}$ in the collection

$$\mathcal{B}_{(\delta/2)} = \{B_\infty(\tilde{\theta}_n, \delta/2) : \tilde{\theta}_n \in \tilde{\Theta}_n\}$$

are pairwise disjoint. It worth mentioning that for $\tilde{\theta} = \mathcal{P}(\theta_n)$, we have

$$B_\infty(\tilde{\theta}_n, \delta/2) = \{\mathcal{P}(\theta) : \Delta_{\max}(\theta - \theta_n) \leq \delta/2\}, \tag{10}$$

by the symmetry of permutation and the definition of $\Delta_{\max}$.

Step 3. For any given permutation matrices $P_1, \ldots, P_L$, we let $\mathcal{P} = \mathcal{P}(P_1, \ldots, P_L)$ denote the operator such that

$$\mathcal{P}(\theta) = \tilde{\theta}$$

for any $\theta \in \Theta$ where

$$\theta_n = (W_n^{(1)}, b_n^{(1)}, W_n^{(2)}, b_n^{(2)}, \ldots, W_n^{(L)}, b_n^{(L)}, W_n^{(L+1)}, b_n^{(L+1)}).$$

and

$$\tilde{\theta}_n = (P_1 W_n^{(1)}, P_1 b_n^{(1)}, P_2 W_n^{(2)} P_1^\top, P_2 b_n^{(2)}, \ldots, P_L W_n^{(L)}, P_L b_n^{(L)}, W_n^{(L+1)} P_L^\top, b_n^{(L+1)}).$$

Now if an optimization algorithm $\mathcal{A}$ guarantees to produce a convergent solution towards θ_n when $\theta^{(0)}$ is an initialization belong to the $(\delta/2)$-neighborhood of θ_n, then for any $\mathcal{P}$, the algorithm $\mathcal{A}$ should guarantee to produce a convergent solution towards $\mathcal{P}(\theta_n)$ when $\theta^{(0)}$ is an initialization belong to the $(\delta/2)$-neighborhood of $\mathcal{P}(\theta_n)$. The reason is that the loss surface of the empirical risk keeps the same structure on

$$B_\infty(\theta_n, \delta/2) = \{\theta^{(0)} : \Delta_{\max}(\theta^{(0)} - \theta_n) \leq \delta/2\}$$

and

$$B_\infty(\tilde{\theta}_n, \delta/2) = \{\mathcal{P}(\theta^{(0)}) : \Delta_{\max}(\theta^{(0)} - \theta_n) \leq \delta/2\}.$$

This implies that if the initialization $\theta^{(0)}$ belongs to $(\delta/2)$-neighborhood of any $\tilde{\theta}_n \in \tilde{\Theta}_n$, the algorithm $\mathcal{A}$ should guarantee to produce a convergent solution towards some $\tilde{\theta}_n \in \tilde{\Theta}_n$, which learns the an empirical risk minimizer $f_{\tilde{\theta}_n}$. The rest of the proof is to calculate the probability of an random initialization $\theta^{(0)}$ locates in the union of neighborhoods in $\mathcal{B}_{\delta/2}$, i.e, we targets for

$$\mathbb{P}(\theta^{(0)} \in \cup_{\tilde{\theta}_n \in \tilde{\Theta}_n} B_\infty(\tilde{\theta}_n, \delta/2)),$$

where the probability is with respect to the randomness of initialization. Firstly, for any random initialization scheme and permutation operator $\mathcal{P}$, we have

$$\mathbb{P}(\theta^{(0)} \in B_\infty(\theta_n, \delta/2)) = \mathbb{P}(\mathcal{P}(\theta^{(0)}) \in B_\infty(\mathcal{P}(\theta_n), \delta/2)), \tag{11}$$

by (10) and the definition of permutation. With a little bit abuse of notation, let

$$\theta^{(0)} = (W_{(0)}^{(1)}, b_{(0)}^{(1)}, \ldots, W_{(0)}^{(L+1)}, b_{(0)}^{(L+1)}).$$

If the initialization method uses identical random distributions for the entries of weights and biases within a layer, then we know that the entries of $W_{(0)}^{(l)}$ are independent and identically distributed, and entries of $b_{(0)}^{(l)}$ are independent and identically distributed. This further implies that for any permutation matrices $P_1, \ldots, P_L$,

$$\begin{aligned}
(W_{(0)}^{(1)}, b_{(0)}^{(1)}) &\overset{d}{=} (P_1 W_{(0)}^{(1)}, P_1 b_{(0)}^{(1)}), \\
(W_{(0)}^{(l)}, b_{(0)}^{(l)}) &\overset{d}{=} (P_l W_{(0)}^{(l)} P_{l-1}^\top, P_l b_{(0)}^{(l)}), \quad l = 2, \ldots, L \\
(W_{(0)}^{(L+1)}, b_{(0)}^{(L+1)}) &\overset{d}{=} (W_{(0)}^{(L+1)} P_L^\top, b_{(0)}^{(L+1)}),
\end{aligned}$$

where $\overset{d}{=}$ denotes the equivalence in distribution. Consequently, $\mathcal{P}(\theta^{(0)})$ has the same distribution of $\theta^{(0)}$. And we have

$$\mathbb{P}(\mathcal{P}(\theta^{(0)}) \in B_\infty(\mathcal{P}(\theta_n), \delta/2)) = \mathbb{P}(\theta^{(0)} \in B_\infty(\mathcal{P}(\theta_n), \delta/2)), \tag{12}$$

for any permutation $\mathcal{P}$.

Combining (11), (12) and the property in (7), we have

$$\begin{aligned}\mathbb{P}(\theta^{(0)} \in \cup_{\tilde{\theta}_n \in \tilde{\Theta}_n} B_\infty(\tilde{\theta}_n, \delta/2)) &= \Sigma_{\tilde{\theta} \in \tilde{\Theta}} \mathbb{P}(\theta^{(0)} \in B_\infty(\tilde{\theta}_n, \delta/2)) \\ &= \Sigma_{\tilde{\theta} \in \tilde{\Theta}} \mathbb{P}(\theta^{(0)} \in B_\infty(\theta_n, \delta/2)) \\ &= d_1^* \times \cdots \times d_L^* \times \mathbb{P}(\theta^{(0)} \in B_\infty(\theta_n, \delta/2)) \\ &= d_1^* \times \cdots \times d_L^* \times \mathbb{P}(\Delta_{\max}(\theta^{(0)} - \theta_n) \leq \delta/2).\end{aligned}$$

This completes the proof.

□

B. Supporting Lemmas

In this section, we give definitions of covering number, packing number of subsets in Euclidean space. We also present definitions of other complexity measures including VC-dimension, Pseudo-dimension, and Rademacher complexity. Lemmas regarding their properties and correlations are also given.

Definition B.1 (Covering number)**.** Let $(K, \|\cdot\|)$ be a metric space, let C be a subset of K, and let ϵ be a positive real number. Let $B_\epsilon(x)$ denote the ball of radius ϵ centered at x. Then C is called a ϵ-covering of K, if $K \subset \cup_{x \in C} B_\epsilon(x)$. The covering number of the metric space $(K, \|\cdot\|)$ with any radius $\epsilon > 0$ is the minimum cardinality of any ϵ-covering, which is defined by $\mathcal{N}(K, \epsilon, \|\cdot\|) = \min\{|C| : C \text{ is a } \epsilon\text{−covering of } K\}$.

Definition B.2 (Packing number)**.** Let $(K, \|\cdot\|)$ be a metric space, let P be a subset of K, and let ϵ be a positive real number. Let $B_\epsilon(x)$ denote the ball of radius ϵ centered at x. Then P is called a ϵ-packing of K, if $\{B_\epsilon(x)\}_{x \in P}$ is pairwise disjoint. The ϵ-packing number of the metric space $(K, \|\cdot\|)$ with any radius $\epsilon > 0$ is the maximum cardinality of any ϵ-packing, which is defined by $\mathcal{M}(K, \epsilon, \|\cdot\|) = \max\{|P| : P \text{ is a } \epsilon\text{−packing of } K\}$.

Lemma B.3. *Let $(K, \|\cdot\|)$ be a metric space, and for any $\epsilon > 0$, let $\mathcal{N}(K, \epsilon, \|\cdot\|)$ and $\mathcal{M}(K, \epsilon, \|\cdot\|)$ denote the ϵ-covering number and ϵ-packing number respectively, then*

$$\mathcal{M}(K, 2\epsilon, \|\cdot\|) \leq \mathcal{N}(K, \epsilon, \|\cdot\|) \leq \mathcal{M}(K, \epsilon/2, \|\cdot\|).$$

Proof. For simplicity, we write $\mathcal{N}_\epsilon = \mathcal{N}(K, \epsilon, \|\cdot\|)$ and $\mathcal{M}_\epsilon = \mathcal{M}(K, \epsilon, \|\cdot\|)$. We firstly proof $\mathcal{M}_{2\epsilon} \leq \mathcal{N}_\epsilon$ by contradiction. Let $P = \{p_1, \ldots, p_{\mathcal{M}_{2\epsilon}}\}$ be any maximal 2ϵ-packing of K and $C = \{c_1, \ldots, c_{\mathcal{N}_\epsilon}\}$ be any minimal ϵ-covering of K. If $\mathcal{M}_{2\epsilon} \geq \mathcal{N}_\epsilon + 1$, then we must have p_i and p_j belonging to the same ϵ-ball $B_\epsilon(c_k)$ for some $i \neq j$ and k. This means that the distance between p_i and p_j cannot be more than the diameter of the ball, i.e. $\|p_i - p_j\| \leq 2\epsilon$, which leads to a contradiction since $\|p_i - p_j\| > 2\epsilon$ by the definition of packing.

Secondly, we prove $\mathcal{N}_\epsilon \leq \mathcal{M}_{\epsilon/2}$ by showing that each maximal $(\epsilon/2)$-packing $P = \{p_1, \ldots, p_{\mathcal{M}_\epsilon}\}$ is also a ϵ-covering. Note that for any $x \in K \backslash P$, there exist a $p_i \in P$ such that $\|x - p_i\| \leq \epsilon$ since if this does not hold, then we can construct a bigger packing with $p_{\mathcal{M}_\epsilon + 1} = x$. Thus P is also a ϵ-covering and we have $\mathcal{N}_\epsilon \leq \mathcal{M}_\epsilon$ by the definition of covering. □

Lemma B.4. *Let S be a subset of $\mathbb{R}^d$ with volume V, and let $\epsilon > 0$ be a positive real number. Then, the covering number $\mathcal{N}(V, \epsilon, \|\cdot\|_\infty)$ and packing number $\mathcal{M}(V, \epsilon/2, \|\cdot\|_\infty)$ of S satisfies:*

$$\mathcal{N}(V, \epsilon, \|\cdot\|_\infty) \leq \mathcal{M}(V, \epsilon/2, \|\cdot\|_\infty) \leq V \times (\frac{2}{\epsilon})^d.$$

Proof. Consider a packing of S with non-overlapping hypercubes of side length $\epsilon/2$ under the L^∞ norm. The volume of each hypercube is $(\epsilon/2)^d$, and since the hypercubes do not overlap, the total volume of the hypercubes in the packing is at most the volume of S. Thus, we have:

$$\mathcal{M}(V, \epsilon/2, \|\cdot\|_\infty) \cdot (\epsilon/2)^d \leq V,$$

which implies:

$$\mathcal{M}(V, \epsilon/2, \|\cdot\|_\infty) \leq V \times (\frac{2}{\epsilon})^d.$$

Then by Lemma B.3, $\mathcal{N}(V, \epsilon, \|\cdot\|_\infty) \leq \mathcal{M}(V, \epsilon/2, \|\cdot\|_\infty)$, which completes the proof.

□

Definition B.5 (Shattering, Definition 11.4 in (Mohri et al., 2018))**.** Let $\mathcal{F}$ be a family of functions from a set $\mathcal{Z}$ to $\mathbb{R}$. A set $\{z_1, \ldots, Z_n\} \subset \mathcal{Z}$ is said to be shattered by $\mathcal{F}$, if there exists $t_1, \ldots, t_n \in \mathbb{R}$ such that

$$\left| \left\{ \left[\begin{array}{c} \operatorname{sgn}(f(z_1) - t_1) \\ \ldots \\ \operatorname{sgn}(f(z_n) - t_n) \end{array} \right] : f \in \mathcal{F} \right\} \right| = 2^n,$$

where $rmsgn$ is the sign function returns $+1$ or -1 and $|\cdot|$ denotes the cardinality of a set. When they exist, the threshold values $t_1, \ldots, t_n$ are said to witness the shattering.

Definition B.6 (Pseudo dimension, Definition 11.5 in (Mohri et al., 2018))**.** Let $\mathcal{F}$ be a family of functions mapping from $\mathcal{Z}$ to $\mathbb{R}$. Then, the pseudo dimension of $\mathcal{F}$, denoted by $\operatorname{Pdim}(\mathcal{F})$, is the size of the largest set shattered by $\mathcal{F}$.

Definition B.7 (VC dimension)**.** Let $\mathcal{F}$ be a family of functions mapping from $\mathcal{Z}$ to $\mathbb{R}$. Then, the Vapnik–Chervonenkis (VC) dimension of $\mathcal{F}$, denoted by $\operatorname{VCdim}(\mathcal{F})$, is the size of the largest set shattered by $\mathcal{F}$ with all threshold values being zero, i.e., $t_1 = \ldots, = t_n = 0$.

Definition B.8 (Empirical Rademacher Complexity, Definition 3.1 in (Mohri et al., 2018))**.** Let $\mathcal{F}$ be a family of functions mapping from $\mathcal{Z}$ to $[a, b]$ and $S = (z_1, \ldots, z_n)$ a fixed sample of size m with elements in $\mathcal{Z}$. Then, the empirical Rademacher complexity of $\mathcal{F}$ with respect to the sample S is defined as:

$$\hat{\mathcal{R}}_S(\mathcal{F}) = \mathbb{E}_\sigma \left[\sup_{f \in \mathcal{F}} \left| \frac{1}{n} \sum_{i=1}^{n} \sigma_i f(x_i) \right| \right],$$

where $\sigma = (\sigma_1, \ldots, \sigma_n)^\top$, with σ_is independent uniform random variables taking values in $\{+1, -1\}$. The random variables σ_i are called Rademacher variables.

Definition B.9 (Rademacher Complexity, Definition 3.2 in Mohri et al. (2018))**.** Let $\mathcal{D}$ denote the distribution according to which samples are drawn. For any integer $n \geq 1$, the Rademacher complexity of $\mathcal{F}$ is the expectation of the empirical Rademacher complexity over all samples of size n drawn according to $\mathcal{D}$:

$$\mathcal{R}_n(\mathcal{F}) = \mathbb{E}_{S \sim \mathcal{D}^n} \left[\hat{\mathcal{R}}_S(\mathcal{F}) \right].$$

Lemma B.10 (Dudley's Theorem, (Dudley, 1967))**.** *Let $\mathcal{F}$ be a set of functions $f : \mathcal{X} \to \mathbb{R}$. Then*

$$\mathcal{R}_n(\mathcal{F}) \leq 12 \int_0^\infty \sqrt{\frac{\log \mathcal{N}(\mathcal{F}, \epsilon, \|\cdot\|_\infty)}{n}} d\epsilon.$$

Dudley's Theorem gives a way to bound Rademacher complexity using covering number (Dudley, 1967; 2010). And the covering and packing numbers can be bounded in terms of the VC dimension or Pseudo dimension (Haussler, 1995; Anthony et al., 1999).

Lemma B.11 (Theorem 12.2 in (Anthony et al., 1999))**.** *Suppose that $\mathcal{F}$ is a class of functions from X to the bounded interval $[0, B] \subset \mathbb{R}$. Given a sequence $x = (x_1, \ldots, x_n) \in \mathcal{X}^n$, we let $\mathcal{F}|_x$ be the subset of $\mathbb{R}^n$ given by*

$$\mathcal{F}|_x = \{(f(x_1), \ldots, f(x_n)) : f \in \mathcal{F}\}.$$

we define the uniform covering number $\mathcal{N}_n(\mathcal{F}, \epsilon, \|\cdot\|_\infty)$ to be the maximum, over all $x \in \mathcal{X}^n$, of the covering number $\mathcal{N}(\mathcal{F}|_x, \epsilon, \|\cdot\|_\infty)$, that is

$$\mathcal{N}_n(\mathcal{F}, \epsilon, \|\cdot\|_\infty) = \max\{\mathcal{N}(\mathcal{F}|_x, \epsilon, \|\cdot\|_\infty) : x \in \mathcal{X}^n\}.$$

Let $\epsilon > 0$ and suppose the pseudo-dimension of $\mathcal{F}$ is d. Then

$$\mathcal{N}_n(\mathcal{F}, \epsilon, \|\cdot\|_\infty) \leq \sum_{i=1}^{d} \binom{n}{i} (\frac{B}{\epsilon})^i,$$

which is less that $(enB/(\epsilon d))^d$ for $n \geq d$.