Journal of Machine Learning Research 27 (2026) 1-50 Submitted 10/24; Revised 2/26; Published 8/26

Neural Scaling Laws of Deep ReLU and Deep Operator Network: A Theoretical Study

Hao Liu[1] HAOLIU@HKBU.EDU.HK
Zecheng Zhang[2*] ZECHENG.ZHANG.MATH@GMAIL.COM
Wenjing Liao[3] WLIAO60@GATECH.EDU
Hayden Schaeffer[4] HAYDEN@MATH.UCLA.EDU
[*] *Corresponding author*
[1] *Department of Mathematics, Hong Kong Baptist University, Hong Kong, China*
[2] *Department of Applied and Computational Mathematics and Statistics, University of Notre Dame, Notre Dame IN 46556*
[3] *School of Mathematics, Georgia Institute of Technology, Atlanta, GA 30332*
[4] *Department of Mathematics, UCLA, Los Angeles, CA 90095*

Editor: Weijie Su

Abstract

Neural scaling laws play a pivotal role in the performance of deep neural networks and have been observed in a wide range of tasks. However, a complete theoretical framework for understanding these scaling laws remains underdeveloped. In this paper, we explore the neural scaling laws for deep operator networks, which involve learning mappings between function spaces, with a focus on the Chen and Chen style architecture. These approaches, which include the popular Deep Operator Network (DeepONet), approximate the output functions using a linear combination of learnable basis functions and coefficients that depend on the input functions. We establish a theoretical framework to quantify the neural scaling laws by analyzing its approximation and generalization errors. We articulate the relationship between the approximation and generalization errors of deep operator networks and key factors such as network model size and training data size. Moreover, we address practical cases where input functions exhibit low-dimensional structures, allowing us to derive tighter error bounds. These results also hold for deep ReLU networks and other similar network structures. Our results offer a partial explanation of the neural scaling laws in operator learning and provide a theoretical foundation for their applications.

Keywords: deep operator learning, neural scaling law, approximation theory, generalization theory

1. Introduction

Deep neural networks have demonstrated remarkable performance in a wide range of applications, such as computer vision (He et al., 2016; Creswell et al., 2018), natural language processing (Graves et al., 2013), speech recognition (Hinton et al., 2012), scientific computing (Han et al., 2018; Khoo et al., 2021; Zhang et al., 2023b; Bao et al., 2024), etc. In many of these applications, the core problem is to learn an operator between function spaces. For example, in Bhattacharya et al. (2021); Li et al. (2021), deep neural networks are used to represent a solution map of Partial Differential Equations (PDEs), in which the net-

©2026 Hao Liu, Zecheng Zhang, Wenjing Liao, Hayden Schaeffer.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v27/24-1862.html.

work maps the initial/boundary conditions to PDE solutions. In Ronneberger et al. (2015), deep neural networks are used for image segmentation, in which the network represents an operator from any given image to its segmented counterpart.

In literature, many network architectures have been proposed to learn operators between function spaces, such as Chen and Chen neural operators (Chen and Chen, 1995, 1993), Principal Component Analysis-based Networks (PCANet) (Bhattacharya et al., 2021), Fourier Neural Operator (FNO) (Li et al., 2021), Deep Operator Network (DeepONet) (Lu et al., 2021b), Autoencoder-based Networks (AENet) (Kontolati et al., 2023; Liu et al., 2025) and Basis Enhanced Learning (BelNet) (Zhang et al., 2023b). Since directly learning an operator is difficult due to the curse of dimensionality, a popular strategy is to use an encoder-decoder framework, i.e., one encodes infinite-dimensional functions into finite-dimensional latent features and then learns a map in the latent space. FNO (Li et al., 2021) uses the Fourier transform to convert computation to the frequency domain and then the map is learned in the frequency domain. PCANet (Bhattacharya et al., 2021) uses Principal Component Analysis (PCA) for encoding and decoding. DeepONet (Lu et al., 2021b; Lin et al., 2023) uses a branch net to convert input functions to a set of coefficients, and a trunk net to learn a set of basis functions in the output space. The resulting neural operator in DeepONet is a linear combination of the bases weighted by the coefficients. A novel training strategy of DeepONet is recently proposed in Lee and Shin (2024).

In the training of neural networks, neural scaling laws are observed in regard to the scaling between the generalization error and the data size/model size/running time (Kaplan et al., 2020). Neural scaling laws between the generalization error and the data size/model size are also empirically observed for operator learning (Lu et al., 2021b; de Hoop et al., 2022; Li et al., 2020; Subramanian et al., 2024). For example, Lu et al. (2021b) reported an exponential convergence of the DeepONet test error as the training data size increases for small training datasets, and a polynomial convergence for moderate and large training datasets. de Hoop et al. (2022) reported a power law between the test error and the training data size on various examples of learning PDE solutions. In the multi-operator learning foundation model for PDE (Liu et al., 2023; Sun et al., 2024), the authors observed a heuristic scaling law of the testing error as the number of distinct families of operators increase (Sun et al., 2024); similar results were noted when scaling up the dataset diversity in climate models (Bodnar et al., 2024), where the authors additionally reported a power scaling law with increasing model size. The difficulty with PDE foundation scaling laws is that they dependent on increasing the dataset heterogeneity, since the data sequences cannot be i.i.d. due to temporal dependencies (Liu et al., 2024d).

Neural scaling laws are often used to quantify the performance of neural networks with respect to the data size/model size/running time. A theoretical understanding of neural scaling laws is of crucial importance, which allows one to analyze and quantify the generalization error in deep learning, and predicts how much the network performance can be improved by increasing the data size, model size, and running time (Hestness et al., 2017; Kaplan et al., 2020). A theoretical understanding of model/data scaling laws (scaling between the generalization error and model/data size) can be related to neural network representation and generalization theory. When feedforward ReLU networks are used for function approximation, the representation theory in Yarotsky (2017); Lu et al. (2021a) quantifies the network approximation error with respect to the model size, which partially

explains model scaling laws. Data scaling laws can be justified through the generalization error bound in terms of the data size. It was shown that when feedforward neural networks (Schmidt-Hieber, 2020) and convolutional neural networks (Oono and Suzuki, 2019; Yang et al., 2024) are used for the regression of s-Hölder functions in $\mathbb{R}^D$, the squared generalization error converges on the order of $n^{-\frac{2s}{2s+D}}$ where n denotes the training data size. Similar error bounds are also established for piecewise smooth functions in Petersen and Voigtlaender (2018); Imaizumi and Fukumizu (2019); Liu et al. (2024a) Due to the curse of dimensionality, this rate converges slowly when the data dimension is high (D is large). One way to mitigate the curse of data dimension and improve the rate is by incorporating low-dimensional data structures (Tenenbaum et al., 2000; Pope et al., 2021). Under a manifold hypothesis, one can achieve the same approximation error with a much smaller network size (Chen et al., 2019; Liu et al., 2021), and the squared generalization error is improved to the order of $n^{-\frac{2s}{2s+d}}$ where d is the intrinsic dimension of data (Nakada and Imaizumi, 2020; Dahal et al., 2022; Chen et al., 2022; Liu et al., 2024b).

Compared to regression, theoretical analysis of neural scaling laws for operator learning is less studied. An approximation result for PCANet was established in Bhattacharya et al. (2021) . A thorough study on the approximation error of PCANet was conducted in Lanthaler (2023), which derived both the upper and lower complexity bounds. The generalization error of an encoder-decoder framework for operator learning was studied in Liu et al. (2024c). This encoder-decoder framework assumes that the encoders and decoders are either given or estimated from data, and a network is used to learn the mapping between latent spaces. This encoder-decoder framework includes PCANet as a special case. The generalization error derived in Liu et al. (2024c) consists of a network estimation error and an encoding error. The squared network estimation error for Lipschitz operators is on the order of $n^{-\frac{2}{2+d_U}}$ where d_U is the dimension of the input latent space. Furthermore, if the input functions exhibit a single-chart manifold structure with intrinsic dimension d_U and the latent variables are learned by Autoencoder, Liu et al. (2025) provided a generalization error analysis where the squared generalization error is on the order of $n^{-\frac{1}{2+d_U}}$.

Regarding Chen and Chen (1995, 1993) style neural operators such as the popular DeepONet (Lu et al., 2021a), the first universal approximation theory was established in Chen and Chen (1995, 1993). The authors showed that DeepONet (Lu et al., 2021a,b) can approximate continuous operators with arbitrary accuracy, the authors in Zhang et al. (2023b,a) later extended the theorem to be invariant to the discretization. However, the network size was not specified in Chen and Chen (1995); Lu et al. (2021a) and therefore this theory cannot quantify model scaling laws. A more comprehensive analysis of DeepONet was conducted in Lanthaler et al. (2022), which studied the approximation error of each component in DeepONet with an estimation on the network size. These results were applied to study several concrete problems on the solution operator of differential equations. A generalization error was also studied in Lanthaler et al. (2022), which focused on the stochastic error (variance). The bias-variance trade-off was not addressed and the neural scaling law is not explicitly provided.

In this paper, we study the neural scaling laws of Chen-Chen style neural operators. Specifically, let U and V be two functions sets with domain dimensions d_1 and d_2 respectively, and $G : U \to V$ be a Lipschitz operator between U and V. We consider learning

	Approximation Error	Squared Generalization Error
General Case	$\left(\frac{\log N_{\#}}{\log\log N_{\#}}\right)^{-\frac{1}{d_1}}$	$\left(\frac{\log n}{\log\log n}\right)^{-\frac{2}{d_1}}$
U Expanded by b_U Bases	$N_{\#}^{-\frac{1}{b_U+d_2}}$	$n^{-\frac{2}{2+b_U+d_2}}$

Table 1: Summary of the orders of our approximation and generalization error bounds of DeepONet for Lipschitz operators. $N_{\#}$ denotes the network model size, n is the number of input-output function pairs in the training data. U is the input set. d_1 and d_2 are the dimension of input domain Ω_U and output domain Ω_V, respectively.

Lipschitz operators by DeepONet and analyze its approximation error and generalization error. Our main results are summarized as follows and in Table 1:

1. We show that if the network architecture is properly set, DeepONet can approximate Lipschitz operators with arbitrary accuracy. In particular, if we denote the number of network parameters by $N_{\#}$, the approximation error of DeepONet for Lipschitz operators is on the order of $\left(\frac{\log N_{\#}}{\log\log N_{\#}}\right)^{-\frac{1}{d_1}}$.

2. We prove that the squared generalization error of DeepONet for learning Lipschitz operators is on the order of $\left(\frac{\log n}{\log\log n}\right)^{-\frac{2}{d_1}}$, where n is the number of input-output function pairs in the training data.

3. Furthermore, we incorporate low-dimensional structures of input functions into our analysis and improve the power law in $\log N_{\#}$ and $\log n$ above to a power law in $N_{\#}$ and n respectively. Specifically, when all functions in U can be represented by b_U orthogonal bases, the approximation error of DeepONet for Lipschitz operators is on the order of $N_{\#}^{-\frac{1}{b_U+d_2}}$, and the squared generalization error is on the order of $n^{-\frac{2}{2+b_U+d_2}}$ up to some logarithmic factor.

Our results establish novel approximation and generalization error bounds of a class of neural operators originated from Chen and Chen (1995); Lu et al. (2021a), which provide a theoretical justification of neural scaling laws. The slow convergence rate given by the power law in $\log N_{\#}$ and $\log n$ in the general case demonstrates the difficulty of learning general Lipschitz operators without additional data structures. This difficulty is also discussed in Mhaskar and Hahm (1997); Lanthaler and Stuart (2023). By utilizing low-dimensional data structures, the neural scaling law is significantly improved to a power law in $N_{\#}$ (model size) and n (data size), which partially explains the observed power scaling laws in many existing works (de Hoop et al., 2022; Lu et al., 2021b).

This paper is organized as follows: We introduce related concepts and notations in Section 2. The problem setup and DeepONet structure are presented in Section 3. We present our main results in Section 4: Section 4.2 for the approximation theory and 4.3 for the generalization theory of learning general Lipschitz operators, and Section 4.4 for an error analysis incorporating low-dimensional data structures. Our main results are proved

in Section 5. We conclude this paper in Section 6. All proofs of auxiliary lemmata and theorems are deferred to the appendix.

2. Preliminary

2.1 Neural Network

In this paper, we define a feedforward ReLU network $q : \mathbb{R}^{d_1} \to \mathbb{R}$ as

$$q(\mathbf{x}) = W_L \cdot \mathrm{ReLU}\left(W_{L-1} \cdots \mathrm{ReLU}(W_1 x + b_1) + \cdots + b_{L-1}\right) + b_L, \tag{1}$$

where W_l's are weight matrices, b_l's are bias vectors, $\mathrm{ReLU}(a) = \max\{a, 0\}$ is the rectified linear unit activation (ReLU) applied element-wise, and Ω is the domain. We define the network class $\mathcal{F}_{\mathrm{NN}} : \mathbb{R}^{d_1} \to \mathbb{R}^{d_2}$:

$$\begin{aligned}
&\mathcal{F}_{\mathrm{NN}}(d_1, d_2, L, p, K, \kappa, R) \\
&\quad = \{[q_1, q_2, ..., q_{d_2}]^\intercal \in \mathbb{R}^{d_2} : \text{ for each } k = 1, ..., d_2, \\
&\qquad q_k : \mathbb{R}^{d_1} \to \mathbb{R} \text{ is in the form of (1) with } L \text{ layers, width bounded by } p, \\
&\qquad \|q_k\|_{L^\infty} \le R,\ \|W_l\|_{\infty,\infty} \le \kappa,\ \|b_l\|_\infty \le \kappa,\ \sum_{l=1}^{L} \|W_l\|_0 + \|b_l\|_0 \le K,\ \forall l\},
\end{aligned} \tag{2}$$

where $\|q\|_{L^\infty(\Omega)} = \sup_{\mathbf{x}\in\Omega} |q(\mathbf{x})|$, $\|W_l\|_{\infty,\infty} = \max_{i,j} |W_{i,j}|$, $\|b\|_\infty = \max_i |b_i|$, and $\|\cdot\|_0$ denotes the number of nonzero elements of its argument. The network class above has input dimension d_1, output dimension d_2, L layers, width p, the number of nonzero parameters no larger than K. All parameters are bounded by κ and each element in the output is bounded by R.

2.2 Cover and Partition of Unity

We define the cover of a set as follows:

Definition 1 (Cover) *A collection of sets $\{S_k\}_{k=1}^{C_S}$ is a cover of Ω if $\Omega \subset \bigcup_{k=1}^{C_S} S_k$.*

The following lemma shows that, for a compact smooth manifold $\mathcal{M}$ and any given cover of $\mathcal{M}$, there exists a C^∞ partition of unity of $\mathcal{M}$ that subordinates to the given cover.

Lemma 2 (Theorem 13.7(ii) of Tu (2011)) *Let $\{\Omega_k\}_{k=1}^M$ be an open cover of a compact smooth manifold $\mathcal{M}$. There exists a C^∞ partition of unity $\{\omega_k\}_{k=1}^M$ that subordinates to $\{\Omega_k\}_{k=1}^M$ such that $\mathrm{supp}(\omega_k) \subset \Omega_k$ for any k.*

2.3 Lipschitz Functional

A Lipschitz functional is defined as follows:

Definition 3 (Lipschitz functional) *Given a function set U with domain Ω_U such that $U \subset L^2(\Omega_U)$, we say a functional $f : U \to \mathbb{R}$ is Lipschitz with Lipschitz constant L_f if*

$$|f(u_1) - f(u_2)| \le L_f \|u_1 - u_2\|_{L^2(\Omega_U)},\ \forall u_1, u_2 \in U.$$

2.4 Clipping Operation

For a function $f : \mathbb{R} \to \mathbb{R}$, we define the clipping operation:

$$\mathrm{CL}_a(f) = \min\{\max\{f, -a\}, a\}$$

for some $a \geq 0$. This clipping operation can be realized by a two-layer ReLU network

$$\mathrm{CL}_a(f) = -\mathrm{ReLU}(-\mathrm{ReLU}(f + a) + 2a) + a. \tag{3}$$

2.5 Notation

In this paper, we use normal lowercase letters to denote scalars, and bold lowercase letters to denote vectors. Matrices, sets and operators are denoted by upper case letters. We use U to denote the input function set with domain Ω_V, and V to denote the output function set with domain Ω_V. We denote the operator to be learned which maps functions in U to functions in V by G. Express a d-dimensional vector $\mathbf{x}$ as $\mathbf{x} = [x_1, ..., x_d]^\top$. The ℓ^∞ and ℓ^2 norm of a vector $\mathbf{x}$ is defined $\|\mathbf{x}\|_\infty = \max_k |x_k|$ and $\|\mathbf{x}\|_2 = \sqrt{\sum_k^d x_k^2}$, respectively. We denote the Euclidean ball with center $\mathbf{c}$ and radius δ by $\mathcal{B}_\delta(\mathbf{c})$. The L^∞ and L^2 norm of a function over domain Ω_U is defined as $\|u\|_{L^\infty(\Omega_U)} = \sup_{\mathbf{x}\in\Omega_U} |u(\mathbf{x})|$ and $\|u\|_{L^2(\Omega_U)} = \sqrt{\int_{\Omega_U} [u(\mathbf{x})]^2 d\mathbf{x}}$, respectively. We define the $\|\cdot\|_{\infty,\infty}$ norm of an operator $G : U \to V$ by $\|G\|_{\infty,\infty} = \sup_{\mathbf{y}\in\Omega_V} \sup_{u\in U} |G(u)(\mathbf{y})|$.

3. Problem Setup and Deep Operator Learning

3.1 Problem Setup and Examples

This paper studies the operator learning problem where the goal is to learn an unknown Lipschitz operator $G : U \to V$ between two function sets U and V from n training samples $\{(u_i, v_i)\}_{i=1}^n$, where $u_i \in U$ and

$$v_i = G(u_i) + \zeta_i \tag{4}$$

with ζ_i representing noise. We consider Lipschitz operators in the following sense:

Assumption 1 *Let Ω_U and Ω_V be the domain of functions in U and V respectively, and $U \subset L^2(\Omega_U)$, $V \subset L^\infty(\Omega_V)$. Assume $G : U \to V$ is a Lipschitz operator: there exists a constant $L_G > 0$ such that*

$$\|G(u_1) - G(u_2)\|_{L^\infty(\Omega_V)} \leq L_G \|u_1 - u_2\|_{L^2(\Omega_U)},$$

for any $u_1, u_2 \in U$.

In Assumption 1, the function distance in the output space is measured by the L^∞ norm, and the function distance in the input space is measured by the L^2 norm. This condition is needed in our network construction to derive an error bound for the branch net. Assumption 1 is satisfied for the solution operator of many differential equations. We provide some examples below.

The first example is a nonlinear ODE system known as gravity pendulum with external force, which is studied in Lu et al. (2021b); Lanthaler et al. (2022); Reid and King (2009).

Example 1 *Consider the following ODE system*

$$\begin{cases} \frac{dv_1}{dt} = v_2, \\ \frac{dv_2}{dt} = -\gamma \sin(v_1) + u(t) \end{cases} \tag{5}$$

with initial condition $v_1(0) = v_2(0) = 0$, and $\gamma > 0$ is a parameter. In (5), v_1, v_2 represent the angle and angular velocity of the pendulum, γ is the frequency parameter and $u(t)$ is an external force controlling the dynamics of the pendulum. For this ODE, we consider the operator: $G : u(t) \to (v_1(t), v_2(t))$. Let $T > 0$ the ending time. For any $u_1, u_2 \in L^2([0,T])$, there exists a constant L_G such that

$$\|G(u_1) - G(u_2)\|_{L^\infty([0,T])} \leq L_G \|u_1 - u_2\|_{L^2([0,T])} \tag{6}$$

which is proved in Lanthaler et al. (2022, Proof of Lemma 4.1).

In the second example, we consider a transport equation.

Example 2 *Let $\Omega \subset \mathbb{R}^d$ be a hyper-cube. Consider the transport equation on $\Omega \times [0,T]$*

$$\begin{cases} v_t = \mathbf{c} \cdot \nabla v & \text{on } \Omega \times [0,T] \\ v(\mathbf{x}, 0) = u(\mathbf{x}) & \text{on } \Omega \end{cases}$$

equipped with periodic boundary condition where $\mathbf{c} \in \mathbb{R}^d$ is the velocity. Let G be the solution operator from the initial condition u to the solution $v(\mathbf{x}, T)$ at time $T > 0$. We set $\Omega_U = \Omega_V = \Omega$. Let $\{w_j\}_{j=1}^J$ be a set of Fourier basis for some positive integer $J > 0$, and

$$U = \left\{ \sum_{j=1}^{J} a_j w_j : \max_j |a_j| \leq C \right\}$$

for some $C > 0$. Then Assumption 1 is satisfied with $L_G = \sqrt{J}$ (see Section A.1 for a proof).

3.2 Deep Operator Learning

We study the DeepONet (Chen and Chen, 1995; Lu et al., 2021b) architecture which consists of a branch net and a trunk net. The branch net encodes the input function and produces a set of coefficients. The trunk net learns a set of basis functions for the output space. A DeepONet takes an input function together with points in the output function domain. It outputs a scalar which is the output function evaluated at the given points.

Let $\mathcal{F}_1 = \mathcal{F}_{\text{NN}}(d_1, 1, L_1, p_1, K_1, \kappa_1, R_1)$ be the network class for the branch net and $\mathcal{F}_2 = \mathcal{F}_{\text{NN}}(d_2, 1, L_2, p_2, K_2, \kappa_2, R_2)$ be the network class for the trunk net. We define the network class of DeepONet as

$$\mathcal{G}_{\text{NN}} = \left\{ G_{\text{NN}}(\mathbf{u})(\mathbf{y}) = \text{CL}_{\beta_V} \left(\sum_{k=1}^{N} \widetilde{a}_k(\mathbf{u}) \widetilde{q}_k(\mathbf{y}) \right) : \widetilde{q}_k \in \mathcal{F}_1, \widetilde{a}_k \in \mathcal{F}_2 \text{ for } k = 1, .., N \right\}, \tag{7}$$

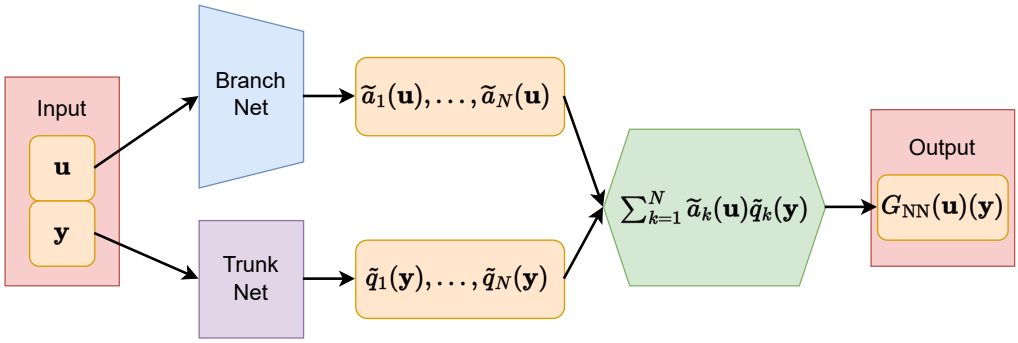


Figure 1: Illustration of the DeepONet architecture. Here $\mathbf{u}$ is the discretization of $u \in U$, and $\mathbf{y} \in \Omega_V$.

where $\mathbf{u}$ is an input vector, which can be thought as a discretization of the input function, and $\mathbf{y}$ is a point in the domain of output functions. The network architecture is illustrated in Figure 1. A DeepONet takes the discretized function $\mathbf{u}$ and a point $\mathbf{y} \in \Omega_V$ as input, where $\mathbf{u}$ is passed to the branch net $\widetilde{a}_k$'s to compute a set of coefficients, and $\mathbf{y}$ is passed to the trunk net to evaluate each basis function $\widetilde{q}_k$ at $\mathbf{y}$. The output $G_{\mathrm{NN}}(\mathbf{u})(\mathbf{y})$ is the sum of the $\widetilde{q}_k$'s value weighted by the coefficients $\widetilde{a}_k$'s from the branch net. Each network in $\mathcal{G}_{\mathrm{NN}}$ is clipped so that its output is bounded by β_V. This step does not increase the error and is important for our theory.

4. Main Results

4.1 Assumptions

In this section, we make some assumptions on the function sets U and V.

Assumption 2 (Input space U) *Suppose U is a function set such that*

(i) Any function $u \in U$ is defined on $\Omega_U = [-\gamma_1, \gamma_1]^{d_1}$ for some $\gamma_1 > 0$.

(ii) Any function $u \in U$ is Lipschitz with a Lipschitz constant no more than $L_U > 0$:

$$|u(\mathbf{x}_1) - u(\mathbf{x}_2)| \leq L_U \|\mathbf{x}_1 - \mathbf{x}_2\|_2$$

for any $\mathbf{x}_1, \mathbf{x}_2 \in \Omega_U$.

(iii) Any function $u \in U$ satisfies $\|u\|_{L^\infty(\Omega_U)} \leq \beta_U$ for some $\beta_U > 0$.

The following assumption is made on the output function set V.

Assumption 3 (Output space V) *Suppose V is a function space such that*

(i) Any function in V is defined on $\Omega_V = [-\gamma_2, \gamma_2]^{d_2}$ for some $\gamma_2 > 0$,

(ii) Any function $v \in V$ is Lipschitz with a Lipschitz constant no more than $L_V > 0$:

$$|v(\boldsymbol{y}_1) - v(\boldsymbol{y}_2)| \leq L_V \|\boldsymbol{y}_1 - \boldsymbol{y}_2\|_2$$

for any $\boldsymbol{y}_1, \boldsymbol{y}_2 \in \Omega_V$.

(iii) Any function $v \in V$ satisfies $\|v\|_{L^\infty(\Omega_V)} \leq \beta_V$ for some $\beta_V > 0$.

Assumption 2 and 3 are mild conditions on U and V and are usually satisfied in applications.

4.2 DeepONet Approximation Error and Model Scaling Law

Our first result is on the approximation error of DeepONet for the representation of Lipschitz operators.

Theorem 4 *Let $d_1, d_2 > 0$ be integers, $\gamma_1, \gamma_2, \beta_U, \beta_V, L_U, L_V, L_G > 0$, and U, V be function sets satisfying Assumption 2 and 3 respectively. There exist constants C depending on d_2, L_V, γ_2, C_F depending on $\gamma_1, \beta_1, d_1, L_G$ and C_δ depending on $\gamma_1, \gamma_2, d_1, L_f, L_U$ such that the following holds: For any $\varepsilon > 0$, set $\delta = C_\delta \varepsilon$ and $N = C\varepsilon^{-d_2}$. Choose $\{\mathbf{c}_m\}_{m=1}^{c_U} \subset \Omega_U$ so that $\{\mathcal{B}_\delta(\mathbf{c}_m)\}_{m=1}^{c_U}$ is a cover of Ω_U. Then there exist two network architectures: $\mathcal{F}_1 = \mathcal{F}_{\mathrm{NN}}(d_2, 1, L_1, p_1, K_1, \kappa_1, R_1)$ with*

$$L_1 = O\left(\log(\varepsilon^{-1})\right),\ p_1 = O(1),\ K_1 = O\left(\log(\varepsilon^{-1})\right),\ \kappa_1 = O(\varepsilon^{-d_2-1}),\ R_1 = 1$$

and $\mathcal{F}_2 = \mathcal{F}_{\mathrm{NN}}(c_U, 1, L_2, p_2, K_2, \kappa_2, R_2)$ with

$$\begin{aligned}
&L_2 = O\left(c_U^2 \log c_U + c_U^2 \log(\varepsilon^{-1})\right),\ p_2 = O((C_F\sqrt{c_U})^{c_U}\varepsilon^{-c_U}),\\
&K_2 = O\left(((C_F\sqrt{c_U})^{c_U}\varepsilon^{-c_U})(c_U^2 \log c_U + c_U^2 \log(\varepsilon^{-1}))\right),\\
&\kappa_2 = O(c_U^{c_U/2+1}\varepsilon^{-c_U-1}),\ R = \beta_V.
\end{aligned}$$

such that, for any operator $G : U \to V$ satisfying Assumption 1, there are $\{\widetilde{q}_k\}_{k=1}^N$ with $\widetilde{q}_k \in \mathcal{F}_1$ and $\{\widetilde{a}_k\}_{k=1}^N$ with $\widetilde{a}_k \subset \mathcal{F}_2$ such that

$$\sup_{u\in U}\sup_{\mathbf{y}\in\Omega_V}\left|G(u)(\mathbf{y}) - \sum_{k=1}^N \widetilde{a}_k(\mathbf{u})\widetilde{q}_k(\mathbf{y})\right| \leq \varepsilon,$$

where $\mathbf{u} = [u(\mathbf{c}_1), u(\mathbf{c}_2), ..., u(\mathbf{c}_{c_U})]^\top$ is a discretization of u. The constant hidden in O depends on $\gamma_1, \gamma_2, \beta_U, \beta_V, d_1, d_2, L_G, L_U, L_V$.

Theorem 4 is proved in Section 5.1. Theorem 4 is a general result that holds for any $\{\mathcal{B}_\delta(\mathbf{c}_m)\}_{m=1}^{c_U}$ covering Ω_U. The following lemma (see a proof in Section B.5) shows that for any bounded hypercube, there always exists a cover with Euclidean balls, and an upper bound on the covering number is provided.

Lemma 5 *Let $\Omega = [-\gamma, \gamma]^d$ for some $\gamma > 0$. For any $\delta > 0$, there exists a cover $\{\mathcal{B}_\delta(\mathbf{c}_m)\}_{m=1}^M$ of Ω with*

$$M \leq C\delta^{-d} \tag{8}$$

where C is a constant depending on γ and d.

Combining Theorem 4 and Lemma 5 yields the following corollary, which quantifies c_U in terms of ε and d_1.

Corollary 6 *Let $d_1, d_2 > 0$ be integers, $\gamma_1, \gamma_2, \beta_U, \beta_V, L_U, L_V > 0$, and U, V be function sets satisfying Assumption 2 and 3 respectively. There exist constants C depending on d_2, L_V, γ_2, C_F depending on $\gamma_1, \beta_1, d_1, L_G$ and C_δ, C_1 depending on γ_1, d_1, L_f, L_U, such that the following hold: For any $\varepsilon > 0$, set $\delta = C_\delta \varepsilon$, $c_U = C_1 \varepsilon^{-d_1}$ and $N = C\varepsilon^{-d_2}$. There exist $\{\mathbf{c}_m\}_{m=1}^{c_U} \subset \Omega_U$ so that $\{\mathcal{B}_\delta(\mathbf{c}_m)\}_{m=1}^{c_U}$ covers Ω_U, and two network architectures: $\mathcal{F}_1 = \mathcal{F}_{\mathrm{NN}}(d_2, 1, L_1, p_1, K_1, \kappa_1, R_1)$ and $\mathcal{F}_2 = \mathcal{F}_{\mathrm{NN}}(c_U, 1, L_2, p_2, K_2, \kappa_2, R_2)$ with*

$$L_1 = O\left(\log(\varepsilon^{-1})\right),\ p_1 = O(1),\ K_1 = O\left(\log(\varepsilon^{-1})\right),\ \kappa_1 = O(\varepsilon^{-d_2-1}),\ R_1 = 1$$

and

$$\begin{aligned}
&L_2 = O\left(\varepsilon^{-2d_1}\log\varepsilon^{-1}\right),\ p_2 = O\left(C_F^{\varepsilon^{-d_1}}\varepsilon^{-(d_1/2+C_1)\varepsilon^{-d_1}}\right),\\
&K_2 = O\left(C_F^{\varepsilon^{-d_1}}\varepsilon^{-(d_1/2+C_1)\varepsilon^{-d_1}-2d_1}\log\varepsilon^{-1}\right),\\
&\kappa_2 = O\left((C_1\varepsilon)^{-(C_1 d_1/2+C_1)\varepsilon^{-d_1}-d_1-1}\right),\ R = \beta_V,
\end{aligned}$$

such that, for any operator $G : U \to V$ satisfying Assumption 1, there are $\{\widetilde{q}_k\}_{k=1}^N$ with $\widetilde{q}_k \in \mathcal{F}_1$ and $\{\widetilde{a}_k\}_{k=1}^N$ with $\widetilde{a}_k \in \mathcal{F}_2$ such that

$$\sup_{u\in U}\sup_{\mathbf{y}\in\Omega_V}|G(u)(\mathbf{y}) - \sum_{k=1}^N \widetilde{a}_k(\mathbf{u})\widetilde{q}_k(\mathbf{y})| \le \varepsilon,$$

where $\mathbf{u} = [u(\mathbf{c}_1), u(\mathbf{c}_2), ..., u(\mathbf{c}_{c_U})]^\top$ is a discretization of u. The constant hidden in O depends on $\gamma_1, \gamma_2, \beta_U, \beta_V, d_1, d_2, L_G, L_U, L_V$.

Corollary 6 can be proved by replacing Theorem 11 by Corollary 12 in the proof of Theorem 4. Theorem 4 and Corollary 6 have the following implications:

- **Model scaling law.** Theorem 4 and Corollary 6 show that if the network architecture is properly set, DeepONet can approximate any Lipschitz operator to arbitrary accuracy. To achieve an accuracy ε, the network size is on the order of $NK_2 = C_F^{\varepsilon^{-d_1}}\varepsilon^{-(d_1/2+C_1)\varepsilon^{-d_1}-2d_1-d_2}\log\varepsilon^{-1}$. In other words, if we denote the total number of network parameters by $N_\# = NK_2$, then the network approximation error is in the order of $\left(\frac{\log N_\#}{\log\log N_\#}\right)^{-1/d_1}$, which critically depends on the input domain dimension d_1. This is because the network size is dominated by $\varepsilon^{-\varepsilon^{-d_1}}$ in NK_2. This dependence is consistent with existing analysis of curse of dimensionality for learning functions, for which the curse is caused by the input dimension, not the output dimension. This result gives a theoretical estimation of the model scaling law, which depicts the relation between the network approximation error and the network size. Without making additional assumptions on the low-dimensional structures of the input function set, the network approximation error scales poorly (converges at an extremely slow rate) as the model size increases. In Section 4.4, we will show that this scaling law can be improved by utilizing low-dimensional structures.

- **Optimality.** In the proof of Theorem 4, an important ingredient is to approximate Lipschitz functionals, which is given in Theorem 11 and Corollary 12. The network size in Theorem 4 is comparable to that in Corollary 12. As discussed in Remark 13, our network size for approximating Lipschitz functionals is optimal up to a logarithmic factor. Since approximating a Lipschitz operator is more difficult than approximating a Lipschitz functional, we expect the network size in Theorem 4 to be close to the optimal one. Notably, a lower bound of the network complexity of approximating r-times Fréchet differentiable operators is analyzed in Lanthaler and Stuart (2023) for several popular network architectures, including DeepONet. The lower bound of the DeepONet size given in Lanthaler and Stuart (2023, Proposition 2.21) for the approximation of Lipschitz functionals is on the order of $\exp(c_1\varepsilon^{-1/(\alpha+1+\delta)})$ where α is a parameter depending on d_1 and δ is a positive number.

- **Connection to existing works.** Approximation theory of DeepONet has been studied in Chen and Chen (1995) and Lanthaler et al. (2022). The network size in Chen and Chen (1995) was not explicitly specified, which cannot explain model scaling laws. Lanthaler et al. (2022) conducted an in-depth study of DeepONet, in which a DeepONet is decomposed into three components: an encoder, an approximator and a reconstructor. Lanthaler et al. (2022) analyzed the network structure of each component on several concrete examples. Our settings and results are different from those in Lanthaler et al. (2022) in the following aspects: (i) Our approximation error is measured by the L^∞ norm, while Lanthaler et al. (2022) studied the L^2 error. (ii) Lanthaler et al. (2022) decomposed the DeepONet approximation error into an encoder error, an approximator error and a reconstructor error, and analyzed each of them. The encoder error and reconstructor error are expressed in terms of the eigenvalues of the covariate operator of the input and output function distributions. An explicit relation between the network size and DeepONet approximation error for general operators was not given. In our paper, we analyze the DeepONet approximation error for general Lipschitz operators and explicitly quantify how the error scales with respect to the network size.

4.3 Generalization Error and Data Scaling Law

Let $n > 0$ be a positive integer. Assume we are given the data set $\mathcal{S} = \{u_i, v_i\}_{i=1}^n$ where u_i's are i.i.d. samples following a distribution ρ_u, and v_i is given by (4). Our setting is summarized below.

Setting 1 *Let $\{\mathbf{x}_j\}_{j=1}^{n_x} \subset \Omega_U$ (independent of i) be a fixed grid in Ω_U, where n_x is the number of grid points in Ω_U which is to be specified later. For $i = 1, ..., n$, let $\{\mathbf{y}_{i,j}\}_{j=1}^{n_y} \subset \Omega_V$ be i.i.d. samples following a distribution ρ_y on Ω_V. Assume we are given a set of paired samples $\{\mathbf{u}_i, \mathbf{v}_i\}_{i=1}^n$ with*

$$\mathbf{u}_i = [u_i(\mathbf{x}_1), ..., u_i(\mathbf{x}_{n_x})]^\top, \quad \mathbf{v}_i = [G(u_i)(\mathbf{y}_{i,1}) + \xi_{i,1}, ..., G(u_i)(\mathbf{y}_{i,n_y}) + \xi_{i,n_y}]^\top, \tag{9}$$

where u_i's are i.i.d. samples from the distribution ρ_u in U, and $\{\xi_{i,j}\}$ follows i.i.d sub-Gaussian distribution with variance proxy σ^2. We denote $\boldsymbol{\xi}_i = [\zeta(\mathbf{y}_{i,1}), ..., \zeta(\mathbf{y}_{i,n_y})]$. Suppose U, V satisfy Assumption 2 and 3, respectively.

In Setting 1, $\{\mathbf{y}_{i,j}\}_{j=1}^{n_y}$ is the set of discretization grids in Ω_V for the output function v_i. This setting allows output functions in the data set to have different discretization grids in Ω_V.

We consider training DeepONet by minimizing $\frac{1}{nn_y}\sum_{i=1}^{n}\sum_{j=1}^{n_y}(G_{\mathrm{NN}}(\mathbf{u}_i)(\mathbf{y}_{i,j})-v_{i,j})^2$ over $G_{\mathrm{NN}}\in\mathcal{G}_{\mathrm{NN}}$ to obtain the following minimizer:

$$\widehat{G}=\underset{G_{\mathrm{NN}}\in\mathcal{G}_{\mathrm{NN}}}{\operatorname{argmin}}\frac{1}{nn_y}\sum_{i=1}^{n}\sum_{j=1}^{n_y}(G_{\mathrm{NN}}(\mathbf{u}_i)(\mathbf{y}_{i,j})-v_{i,j})^2, \tag{10}$$

where $v_{i,j}=(\mathbf{v}_i)_j$ an $\mathcal{G}_{\mathrm{NN}}$ denotes the DeepONet network class given in (7). In this paper, we study the squared generalization error of DeepONet given by:

$$\text{Squared Generalization Error}:=\mathbb{E}_{\mathcal{S}}\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}(\widehat{G}(\mathbf{u})(\mathbf{y}_j)-G(u)(\mathbf{y}_j))^2\right],$$

where $\boldsymbol{u}=[u(\mathbf{x}_1),...,u(\mathbf{x}_{n_x})]^\top$. The following theorem gives an upper bound of the generalization error of DeepONet for learning Lipschitz operators.

Theorem 7 *Let $d_1,d_2,n_y,n>0$ be integers, $\gamma_1,\gamma_2,\beta_U,\beta_V,L_U,L_V,L_G>0$. Suppose $G:U\to V$ satisfy Assumption 1 and consider Setting 1. There exist constants C depending on d_2,L_V, C_F depending on γ_1,β_1,d_1,L_G and γ_2, C_1 depending on γ_1,d_1,L_f,L_U, C_2 depending on $\gamma_1,\gamma_2,\beta_U,\beta_V,d_1,d_2,L_G,L_U,L_V$, and C_δ depending on γ_1,d_1,L_f,L_U, such that the following holds: Let $\varepsilon\in(0,1)$, $\delta=C_\delta\varepsilon$ and $N=C\varepsilon^{-d_2}$. Set $n_x=C_1\varepsilon^{-d_1}$, and then there exist $\{\mathbf{x}_j\}_{j=1}^{n_x}$ such that $\{\mathcal{B}_\delta(\mathbf{x}_j)\}_{j=1}^{n_x}$ is a cover of Ω_U. Consider the DeepONet network (7) with the network architecture $\mathcal{F}_1=\mathcal{F}_{\mathrm{NN}}(d_2,1,L_1,p_1,K_1,\kappa_1,R_1)$ and $\mathcal{F}_2=\mathcal{F}_{\mathrm{NN}}(C\varepsilon^{-C_1\varepsilon^{-d_1}},1,L_2,p_2,K_2,\kappa_2,R_2)$ with*

$$L_1=O\left(\log(\varepsilon^{-1})\right),\ p_1=O(1),\ K_1=O\left(\log(\varepsilon^{-1})\right),\ \kappa_1=O(\varepsilon^{-d_2-1}),\ R_1=1 \tag{11}$$

and

$$\begin{aligned}&L_2=O\left(\varepsilon^{-2d_1}\log\varepsilon^{-1}\right),\ p_2=O\left(C_F^{\varepsilon^{-d_1}}\varepsilon^{-(d_1/2+C_1)\varepsilon^{-d_1}}\right),\\&K_2=O\left(C_F^{\varepsilon^{-d_1}}\varepsilon^{-(d_1/2+C_1)\varepsilon^{-d_1}-2d_1}\log\varepsilon^{-1}\right),\ \kappa_2=O\left((C_1\varepsilon)^{-(C_1d_1/2+C_1)\varepsilon^{-d_1}/2-d_1-1}\right),\ R=\beta_V,\end{aligned} \tag{12}$$

where the constant hidden in O depends on $\sigma,\gamma_1,\gamma_2,\beta_U,\beta_V,d_1,d_2,L_G,L_U,L_V$. Let $\widehat{G}$ be the minimizer in (10). Then we have

$$\begin{aligned}&\mathbb{E}_{\mathcal{S}}\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}(\widehat{G}(\boldsymbol{u})(\mathbf{y}_j)-G(u)(\mathbf{y}_j))^2\right]\\&\qquad\le C_2\left(\varepsilon^2+\left(\frac{\sigma^2}{n_y}+1\right)\frac{1}{n}C_F^{\varepsilon^{-d_1}}\varepsilon^{-(d_1/2+C_1)\varepsilon^{-d_1}-5d_1-d_2}\right)\log^3\frac{1}{\varepsilon}.\end{aligned} \tag{13}$$

In particular, setting $\varepsilon = \left(\frac{d_1}{d_1+2C_1}\frac{\log n}{\log\log n}\right)^{-\frac{1}{d_1}}$ *gives rise to*

$$\mathbb{E}_{\mathcal{S}}\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}(\widehat{G}(\mathbf{u})(\mathbf{y}_j)-G(\mathbf{u})(\mathbf{y}_j))^2\right] \le C_3\left(\frac{\log n}{\log\log n}\right)^{-\frac{2}{d_1}}. \tag{14}$$

for some C_3 *depending on* $\sigma,\gamma_1,\gamma_2,\beta_U,\beta_V,d_1,d_2,L_G,L_U,L_V$.

Theorem 7 is proved in Section 5.2. To prove Theorem 7, we need to carefully perform a bias-variance trade off. The bias term is related with the approximation error of DeepONet. In practice, one has to train the network to learn the operator from a given data set. The variance term captures the difference between the trained network and the network used in the approximation theory. The network architecture suggested in Theorem 7 is a trade-off by balancing the two error terms.

We have the following discussions:

- **Data scaling law.** Theorem 7 shows that to learn Lipschitz operators, the generalization error of DeepONet decays in a power law of $\log n$. This rate is slower than the empirical observations in Lu et al. (2021b); de Hoop et al. (2022), which suggest a power data scaling law on the order of $n^{-\alpha}$ for some $\alpha > 0$. This slow decay in our theory comes from the intrinsic difficulty of operator learning in infinite-dimensional spaces. Due to the curse of dimensionality, learning an operator is much more difficult than learning a finite-dimensional function, leading to a slower decay of the generalization error. In the next subsection, we will show that when the input function set U has some low-dimensional structures, we can derive an upper bound that matches the empirical power law observed in Lu et al. (2021b); de Hoop et al. (2022).

- **Effects of n and n_y.** The upper bound in Theorem 7 critically depends on n. As indicated by equation (13), increasing n_y reduces the impact of noise.

- **Connection with existing works.** A similar rate was derived in Liu et al. (2024c, Section 4.3) for the encoder-decoder framework of operator learning. In Liu et al. (2024c, Section 4.3), one assumed the input and output functions are C^s functions and Legendre polynomials were used as encoders and decoders. With a fixed grid in Ω_U and Ω_V, it was shown that the squared generalization error decays on the order of $(\log n)^{-s/d_1}$, where n is the number of training samples. The generalization error of DeepONet was also analyzed in Lanthaler et al. (2022), which studied the variance part and did not address the bias-variance trade-off.

4.4 Utilizing low-dimensional structures

Corollary 6 and Theorem 7 give rise to a slow rate of convergence of DeepONet due to the curse of dimensionality. In this subsection, we incorporate low-dimensional structures of input functions and prove a power law convergence which is consistent with empirical observations in Lu et al. (2021b); de Hoop et al. (2022). Specifically, we consider the following assumption on low-dimensional structures of U:

Assumption 4 *Suppose the function set* U *satisfies*

(i) There exists a finite orthonormal basis functions $\{\omega_k\}_{k=1}^{b_U}$ *so that any* $u \in U$ *can be expressed as*

$$u = \sum_{k=1}^{b_U} \alpha_k \omega_k, \quad \text{with} \quad \alpha_k = \int_{\Omega_U} u(\mathbf{x})\omega_k(\mathbf{x})d\mathbf{x}. \tag{15}$$

(ii) The discretization grid $\{\mathbf{x}_j\}_{j=1}^{n_x}$ *satisfies that: there is a matrix* $A \in \mathbb{R}^{b_U \times n_x}$ *such that for any* $u \in U$*, we have*

$$A\mathbf{u} = [\alpha_1, ..., \alpha_{b_U}]^\top, \tag{16}$$

where $\mathbf{u} = [u(\mathbf{x}_1), \ldots, u(\mathbf{x}_{n_x})]^T \in \mathbb{R}^{n_x}$ *and the* α_k*'s are the coefficients of* u *in (15). We denote* $C_A = \|A\|_{\infty,\infty}$.

Assumption 4(i) assumes that the input functions in U live in a b_U-dimensional linear subspace. This assumption is commonly used in numerical PDEs. In particular, for some popular bases, such as Legendre polynomials or Fourier bases, Assumption 4(ii) is satisfied by properly choosing $\{\mathbf{x}_j\}_{j=1}^{n_x}$:

- **Legendre polynomials.** Let $\{\omega_k\}_{k=1}^{b_U}$ consist of Legendre polynomials up to degree μ along each dimension. Then $u\omega_k$ is a polynomial with a degree no larger than 2μ along each dimension. By choosing $\{\mathbf{x}_j\}_{j=1}^{n_x}$ so that along each dimension, the points are quadrature points corresponding to Legendre polynomials of degree 2μ, the integral for α_k in (15) can be exactly computed by quadrature rules:

$$\alpha_k = \sum_{j=1}^{n_x} \beta_j u(\mathbf{x}_j)\omega_k(\mathbf{x}_j),$$

  where $\{\beta_j\}_{j=1}^{n_x}$ are quadrature weights. Assumption 4(ii) is satisfied by setting $A = [\mathbf{a}_1, ..., \mathbf{a}_{b_U}]$ with $\mathbf{a}_k = [\beta_1\omega_k(\mathbf{x}_1), ..., \beta_{n_x}\omega_k(\mathbf{x}_{n_x})]^\top$.

- **Fourier bases.** Let $\{\omega_k\}_{k=1}^{b_U}$ be Fourier bases so that ω_k has period $2/N_k$ along each dimension for some integer $N_k \geq 1$, i.e., $\omega_k = \prod_{j=1}^{d_1} \omega_{k,j}(x_j)$ with $\omega_{k,j} = \sin\left(\frac{N_k\pi}{\gamma_1}x_j\right)$ or $\omega_{k,j} = \cos\left(\frac{N_k\pi}{\gamma_1}x_j\right)$. One can choose $\{\mathbf{x}_j\}_{j=1}^{n_x}$ as uniform grids so that the number of grids along each dimension is $\max_k N_k$. We set $A = [\mathbf{a}_1, ..., \mathbf{a}_{b_U}]$ with $\mathbf{a}_k = [\omega_k(\mathbf{x}_1), ..., \omega_k(\mathbf{x}_{n_x})]^\top$. Note A is independent of u as β_j are the quadrature weights.

The following theorem provides an approximation theory of DeepONet under the low dimensional structure in Assumption 4.

Theorem 8 *Let* $d_1, d_2, b_U, n_x > 0$ *be integers,* $\gamma_1, \gamma_2, \beta_U, \beta_V, L_U, L_V, C_A > 0$*, U satisfy Assumption 2 and 4(i), V satisfy Assumption 3, and the discretization grids* $\{\mathbf{x}_j\}_{j=1}^{n_x}$ *in* Ω_U *satisfy Assumption 4(ii). There exist constants* C_δ *depending on* γ_1, d_1, L_f, L_U *and* C

depending on d_2, L_V, γ_2 *such that the following holds: For any* $\varepsilon > 0$*, set* $N = C\varepsilon^{-d_2}$*. There exist two network architectures:* $\mathcal{F}_1 = \mathcal{F}_{\rm NN}(d_2, 1, L_1, p_1, K_1, \kappa_1, R_1)$ *with*

$$L_1 = O\left(\log(\varepsilon^{-1})\right),\ p_1 = O(1),\ K_1 = O\left(\log(\varepsilon^{-1})\right),\ \kappa_1 = O(\varepsilon^{-1}),\ R_1 = 1$$

and $\mathcal{F}_2 = \mathcal{F}_{\rm NN}(n_x, 1, L_2, p_2, K_2, \kappa_2, R_2)$ *with*

$$L_2 = O\left(\log(\varepsilon^{-1})\right),\ p_2 = O(\varepsilon^{-b_U}),\ K_2 = O\left(\varepsilon^{-b_U}(\log(\varepsilon^{-1}) + n_x)\right),$$
$$\kappa_2 = O(\varepsilon^{-b_U - 1}),\ R = \beta_V.$$

such that, for any operator $G : U \to V$ *satisfying Assumption 1, there are* $\{\widetilde{q}_k\}_{k=1}^N$ *with* $\widetilde{q}_k \in \mathcal{F}_1$ *and* $\{\widetilde{a}_k\}_{k=1}^N$ *with* $\widetilde{a}_k \subset \mathcal{F}_2$ *such that*

$$\sup_{u\in U}\sup_{\mathbf{y}\in\Omega_V}\left|G(u)(\mathbf{y}) - \sum_{k=1}^N \widetilde{a}_k(\mathbf{u})\widetilde{q}_k(\mathbf{y})\right| \le \varepsilon.$$

The constant hidden in O *depends on* $\gamma_1, \gamma_2, \beta_U, \beta_V, d_1, d_2, L_G, L_U, L_V, b_U, C_A$*.*

Theorem 8 is proved in Section 5.3. Importantly, Theorem 8 implies a power-law convergence of the DeepONet approximation error.

- **Model scaling law.** Compared to Corollary 6, Theorem 8 has a significant improvement on the network size: the number of nonzero parameters is improved from the order of $C_F^{\varepsilon^{-d_1}}\varepsilon^{-(d_1/2+C_1)\varepsilon^{-d_1}-2d_1-d_2}\log\varepsilon^{-1}$ in Corollary 6 to the order of $\varepsilon^{-b_U-d_2}$ in Theorem 8 for the C_1 defined in Corollary 6 up to logarithmic factors. If we denote the number of network parameters in Theorem 8 by $N_\#$, the approximation error is on the order of $N_\#^{-\frac{1}{b_U+d_2}}$, demonstrating a power model scaling law.
- **Extension to nonlinear low-dimensional structures.** Assumption 4 is a type of linear low-dimensional structures of input functions. An interesting extension is to consider nonlinear structures, such as the input functions locate on a manifold. To characterize this structure one can utilize the concept of Minkowski dimension, as used in Liu et al. (2025). Proof techniques of Theorem 8 cannot be directly extended to study the case of nonlinear low-dimensional structures. The proof of Theorem 8 uses one network layer to extract the linear latent features. Nonlinear low-dimensional structures are more complicated. Extra efforts are needed to analyze how DeepONet learns and extracts these structures. We leave this direction as our future work.

Based on Assumption 4, we consider the following setting for learning Lipschitz operators under low-dimensional structures of U:

Setting 2 *Let* $\{\mathbf{x}_j\}_{j=1}^{n_x} \subset \Omega_U$ *be a fixed discretization gird in* Ω_U*, where* n_x *is the number of grid points in* Ω_U *which is to be specified later. For* $i = 1, ..., n$*, let* $\{\mathbf{y}_{i,j}\}_{j=1}^{n_y} \subset \Omega_V$ *be i.i.d. samples following a distribution* ρ_y *on* Ω_V*. Assume we are given a set of paired samples* $\{\mathbf{u}_i, \mathbf{v}_i\}_{i=1}^n$ *with*

$$\mathbf{u}_i = [u_i(\mathbf{x}_1), ..., u_i(\mathbf{x}_{n_x})]^\top, \quad \mathbf{v}_i = [G(u_i)(\mathbf{y}_{i,1}) + \xi_{i,1}, ..., G(u_i)(\mathbf{y}_{i,n_y}) + \xi_{i,n_y}]^\top, \tag{17}$$

where u_i's are i.i.d. samples from the distribution ρ_u in U, and $\{\xi_{i,j}\}$ follows i.i.d sub-Gaussian distribution with variance proxy σ^2. We denote $\boldsymbol{\xi}_i = [\zeta(\mathbf{y}_{i,1}), ..., \zeta(\mathbf{y}_{i,n_y})]$. Suppose U satisfies Assumption 2 and 4(i), V satisfies Assumption 3, and $\{\mathbf{x}_j\}_{j=1}^{n_x}$ satisfies Assumption 4(ii).

The generalization error of DeepONet under Setting 2 is given in the following theorem:

Theorem 9 *In Setting 2, let $d_1, d_2, n_x, n_y, n, b_U > 0$ be integers, $\gamma_1, \gamma_2, \beta_U, \beta_V, L_U, L_V, L_G, C_A > 0$. Suppose $G : U \to V$ satisfies Assumption 1, and consider Setting 2. There exist constants C depending on d_2, L_V, γ_2 and C_1 depending on $\sigma, \gamma_1, \gamma_2, \beta_U, \beta_V, d_1, d_2, L_G, L_U, L_V$, such that the following holds: Consider the DeepONet network (7) with $N = Cn^{\frac{d_2}{2+b_U+d_2}}$, $\mathcal{F}_1 = \mathcal{F}_{\rm NN}(d_2, 1, L_1, p_1, K_1, \kappa_1, R_1)$ and $\mathcal{F}_2 = \mathcal{F}_{\rm NN}(n_x, 1, L_2, p_2, K_2, \kappa_2, R_2)$ where*

$$L_1 = O(\log n),\ p_1 = O(1),\ K_1 = O(\log n),\ \kappa_1 = O\left(n^{\frac{1}{2+b_U+d_2}}\right),\ R_1 = 1, \tag{18}$$

and

$$\begin{aligned} &L_2 = O(\log n),\ p_2 = O\left(n^{\frac{b_U}{2+b_U+d_2}}\right),\ K_2 = O\left(\varepsilon^{\frac{b_U}{2+b_U+d_2}}\log n + n_x\right), \\ &\kappa_2 = O\left(n^{\frac{b_U+1}{2+b_U+d_2}}\right),\ R_2 = \beta_V. \end{aligned} \tag{19}$$

The constant hidden in O depends on $\sigma, \gamma_1, \gamma_2, \beta_U, \beta_V, d_1, d_2, L_G, L_U, L_V, b_U, C_A$. Then $\widehat{G} \in \mathcal{G}_{NN}$ solving (10) satisfies the following generalization error bound

$$\mathbb{E}_{\mathcal{S}}\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}(\widehat{G}(\boldsymbol{u})(\mathbf{y}_j) - G(u)(\mathbf{y}_j))^2\right] \le C_1 n^{-\frac{2}{2+b_U+d_2}}\log^3 n. \tag{20}$$

Theorem 9 is proved in Section 5.4. We have the following discussion:

- **Data scaling law.** By exploiting low-dimensional structures of U, the squared generalization error is improved from the order of $(\log n/\log\log n)^{-2/d_1}$ in Theorem 7 to the order of $n^{-\frac{2}{2+b_U+d_2}}\log^3 n$ in Theorem 9. This power law decay is consistent with the empirical observations in Lu et al. (2021b); de Hoop et al. (2022), and our theory provides a rigorous justification of the data scaling law.
- **Adapting to low-dimensional data structures.** By incorporating the low-dimensional structure in Assumption 4, we can derive a faster rate of convergence in comparison with the general case. In our network construction, we do not need to explicitly know or learn the bases $\{\omega_l\}_{k=1}^{b_U}$ by neural networks. The bases are encoded in some functionals which are to be learned by neural networks. Our results show that deep neural networks are automatically adaptive to low-dimensional data structures.

5. Proof of main Results

5.1 Proof of Theorem 4

The proof of Theorem 4 relies on Theorem 10 and Theorem 11 below. Theorem 10 is an approximation result for Lipschitz functions by deep neural networks.

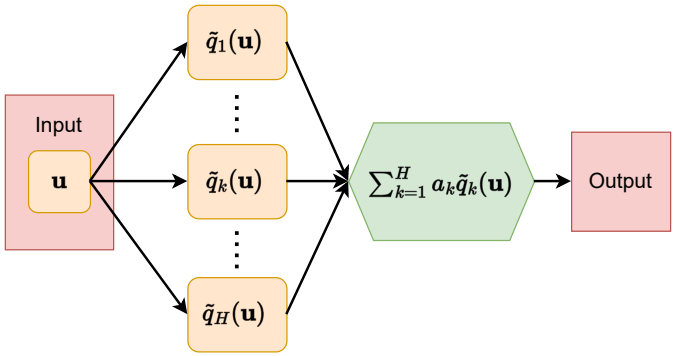


Figure 2: Illustration of the network architecture in Theorem 11. Here $\mathbf{u}$ is the discretization of $u \in U$.

Theorem 10 *Let $d_1 > 0$ be an integer, $\gamma_1, \beta_U, L_U > 0$ and U satisfy Assumption 2. There exists some constant C depending on γ_1, L_U such that the following holds: For any $\varepsilon > 0$, set $N = C\sqrt{d_1}\varepsilon^{-1}$. Let $\{\mathbf{c}_k\}_{k=1}^{N^{d_1}}$ be a uniform grid on Ω_U with spacing $2\gamma_1/N$ along each dimension. There exists a network architecture $\mathcal{F}_{\mathrm{NN}}(d_1, 1, L, p, K, \kappa, R)$ and networks $\{\widetilde{q}_k\}_{k=1}^{N^{d_1}}$ with $\widetilde{q}_k \in \mathcal{F}_{\mathrm{NN}}(d_1, 1, L, p, K, \kappa, R)$, $\widetilde{q}_k \geq 0$ for $k = 1, ..., N^{d_1}$, such that for any $u \in U$, we have*

$$\left\| u - \sum_{k=1}^{N^{d_1}} u(\mathbf{c}_k)\widetilde{q}_k \right\|_{L^\infty(\Omega_U)} \leq \varepsilon. \tag{21}$$

Such a network architecture has

$$L = O\left(d_1^2 \log d_1 + d_1^2 \log(\varepsilon^{-1})\right), p = O(1), K = O\left(d_1^2 \log d_1 + d_1^2 \log(\varepsilon^{-1})\right),$$
$$\kappa = O(d_1^{d_1/2+1}\varepsilon^{-d_1-1}), R = 1.$$

The constant hidden in O depends on L_U and γ_1.

Theorem 10 is proved in Section B.1. Theorem 11 below guarantees the approximation error for Lipschitz functionals.

Theorem 11 *Let $d_1 > 0$ be an integer, $\gamma_1, \beta_U, L_U, L_f, R_f > 0$, and U satisfy Assumption 2. There exist constants C depending on $\gamma_1, \beta_U, d_1, L_f$ and C_δ depending on γ_1, d_1, L_f, L_U such that the following holds: For any $\varepsilon > 0$, set $\delta = C_\delta \varepsilon$ and let $\{\mathbf{c}_m\}_{m=1}^{c_U} \subset \Omega_U$ so that $\{\mathcal{B}_\delta(\mathbf{c}_m)\}_{m=1}^{c_U}$ is a cover of Ω_U for some $c_U > 0$. Let $H = (C\sqrt{c_U})^{c_U}\varepsilon^{-c_U}$, and set the network $\mathcal{F}_{\mathrm{NN}}(c_U, 1, L, p, K, \kappa, R)$ with*

$$L = O\left(c_U^2 \log c_U + c_U^2 \log(\varepsilon^{-1})\right), \ p = O(1), \ K = O\left(c_U^2 \log c_U + c_U^2 \log(\varepsilon^{-1})\right),$$
$$\kappa = O(c_U^{c_U/2+1}\varepsilon^{-c_U-1}), \ R = 1.$$

Denote $\bar{U}_{\beta_U}$ as the set of functions defined on Ω_U and having range $[-\beta_U, \beta_U]$. There are $\{\widetilde{q}_k\}_{k=1}^H$ with $\widetilde{q}_k \in \mathcal{F}_{\rm NN}(c_U, 1, L, p, K, \kappa, R)$ for any k, such that for any Lipschitz functional f on $\bar{U}_{\beta_U}$ with Lipschitz constant L_f for and $\|f\|_{L^\infty(\bar{U}_{\beta_U})} \le R_f$, we have

$$\sup_{u\in U}\left|f(u) - \sum_{k=1}^H a_k \widetilde{q}_k(\mathbf{u})\right| \le \varepsilon, \tag{22}$$

where $\mathbf{u} = [u(\mathbf{c}_1), u(\mathbf{c}_2), ..., u(\mathbf{c}_{c_U})]^\top$, a_k's are coefficients depending on f and satisfying $|a_k| \le R_f$. The constant hidden in O depends on $\gamma_1, \beta_U, d_1, L_f, L_U$.

Theorem 11 is proved in Section B.2. The network architecture is illustrated in Figure 2. Theorem 11 expresses the functional network as a sum of H parallel branches, and the network architecture of each branch is quantified. In the following, we express the functional network as one large network and quantifies the network architecture of this large network. This network can be achieved by stacking the parallel branches together. It has depth similar to that of each $\widetilde{q}$ and width being H times of the width of each $\widetilde{q}_k$, as shown in Figure 2.

We set the network architecture $\mathcal{F}_{\rm NN}(c_U, 1, L, p, K, \kappa, R)$ as

$$\begin{aligned}
&L = O\left(c_U^2 \log c_U + c_U^2 \log(\varepsilon^{-1})\right),\ p = O((C\sqrt{c_U})^{c_U}\varepsilon^{-c_U}),\\
&K = O\left((C\sqrt{c_U})^{c_U}\varepsilon^{-c_U})(c_U^2 \log c_U + c_U^2 \log(\varepsilon^{-1}))\right),\ \kappa = O(c_U^{c_U/2+1}\varepsilon^{-c_U-1}),\ R = R_f.
\end{aligned} \tag{23}$$

For any Lipschitz functional f with Lipschitz constant L_f and $\|f\|_{L^\infty(U)} \le R_f$, there exists $\widetilde{f} \in \mathcal{F}_{\rm NN}(c_U, 1, L, p, K, \kappa, R)$ such that

$$\sup_{u\in U}|f(u) - \widetilde{f}(\mathbf{u})| \le \varepsilon. \tag{24}$$

The constant hidden in O in (23) depends on $\gamma_1, \beta_U, d_1, L_f, L_U$ and L_f.

The following corollary gives an estimation of c_U.

Corollary 12 *Let $d_1 > 0$ be an integer, $\gamma_1, \beta_U, L_U, L_f, R_f > 0$, and U satisfy Assumption 2. There exist constants C depending on $\gamma_1, \beta_U, d_1, L_f$, and C_δ, C_1 depending on γ_1, d_1, L_f, L_U such that the following holds: For any $\varepsilon > 0$, set $\delta = C_\delta\varepsilon$, $c_U = C_1\varepsilon^{-d_1}$ and $H = (C\sqrt{c_U})^{c_U}\varepsilon^{-c_U}$. There exist $\{\mathbf{c}_m\}_{m=1}^{c_U}$ such that $\{\mathcal{B}_\delta(\mathbf{c}_m)\}_{m=1}^{c_U}$ is a cover of Ω_U. Set the network $\mathcal{F}_{\rm NN}(c_U, 1, L, p, K, \kappa, R)$ with*

$$\begin{aligned}
&L = O\left(c_U^2 \log c_U + c_U^2 \log(\varepsilon^{-1})\right),\ p = O(1),\ K = O\left(c_U^2 \log c_U + c_U^2 \log(\varepsilon^{-1})\right),\\
&\kappa = O(c_U^{c_U/2+1}\varepsilon^{-c_U-1}),\ R = 1.
\end{aligned}$$

For any Lipschitz functional f with Lipschitz constant L_f and $\|f\|_{L^\infty(U)} \le R_f$, there are $\{\widetilde{q}_k\}_{k=1}^H$ with $\widetilde{q}_k \in \mathcal{F}_{\rm NN}(c_U, 1, L, p, K, \kappa, R)$ such that

$$\sup_{u\in U}\left|f(u) - \sum_{k=1}^H a_k \widetilde{q}_k(\mathbf{u})\right| \le \varepsilon, \tag{25}$$

where $\mathbf{u} = [u(\mathbf{c}_1), u(\mathbf{c}_2), ..., u(\mathbf{c}_{c_U})]^\top$, a_k's are coefficients depending on f and satisfying $|a_k| \le R_f$. The constant hidden in O depends on $\gamma_1, \beta_U, d_1, L_f, R_f, L_U$.

Corollary 12 is proved in Section B.3.

Remark 13 *The approximation theory for functionals has been studied in Mhaskar and Hahm (1997). It was proved in Mhaskar and Hahm (1997, Theorem 2.2) that, when the activation function is infinitely smooth, the approximation error of a Lipschitz functional by a two-layer network is lower bounded by $O((\log N)^{-1/d_1})$ where N is the number of computational neurons. In Corollary 12, N is bounded by the total number of weight parameters, thus $N = O\left((\varepsilon^{-(d_1/2+1)C_1\varepsilon^{-d_1}})(\varepsilon^{-2d_1}\log\varepsilon^{-1})\right)$, which implies $\varepsilon = O\left(\left(\frac{\log N}{\log\log N}\right)^{-\frac{1}{d_1}}\right)$. Rewriting (25) gives*

$$\left\| f(u) - \sum_{k=1}^{H} a_k \widetilde{q}_k(\mathbf{u}) \right\|_{L^\infty(U)} \leq C_2 \left(\frac{\log N}{\log\log N}\right)^{-\frac{1}{d_1}}$$

for some C_2 depending on $\gamma_1, \beta_U, d_1, L_f, R_f, L_U$. Our result is consistent with the approximation rate in Song et al. (2023) and is optimal up to a $(\log\log N)^{1/d_1}$ factor according to Mhaskar and Hahm (1997).

Remark 14 *A simple set of $\{\mathbf{c}_m\}_{m=1}^{c_U}$ satisfying the condition in Corollary 12 is the uniform grid in Ω_U with grid spacing $\frac{\varepsilon}{4\sqrt{d_1}(2\gamma_1)^{d_1/2}L_f L_U}$.*

Now we are ready to prove Theorem 4.

Proof [Proof of Theorem 4] By Theorem 10, for any $\varepsilon_1 > 0$, there exists a constant $N = C\varepsilon_1^{-d_2}$ for some constant C depending on d_2, L_V and γ_2, and a network architecture $\mathcal{F}_1 = \mathcal{F}_{\mathrm{NN}}(d_2, 1, L_1, p_1, K_1, \kappa_1, R_1)$ and $\{\widetilde{q}_k\}_{k=1}^N$ with $\widetilde{q}_k \in \mathcal{F}_1$, $\widetilde{q}_k \geq 0$ and $\{\mathbf{c}_k\}_{k=1}^N \subset \Omega_V$ such that for any $u \in U$, we have

$$\sup_{\mathbf{y}\in\Omega_V} \left| G(u)(\mathbf{y}) - \sum_{k=1}^{N} G(u)(\mathbf{c}_k)\widetilde{q}_k(\mathbf{y}) \right| \leq \varepsilon_1. \tag{26}$$

Such a network has parameters

$$L_1 = O\left(\log(\varepsilon_1^{-1})\right),\ p_1 = O(1),\ K_1 = O\left(\log(\varepsilon_1^{-1})\right),\ \kappa_1 = O(\varepsilon_1^{-d_2-1}),\ R_1 = 1,$$

where the constant hidden in O depends on d_2, L_V and γ_2.

For each k, define the functional $f_k : V \to R$ such that

$$f_k(G(u)) = G(u)(\mathbf{c}_k). \tag{27}$$

For any $u_1, u_2 \in U$, we have $|f_k(G(u_1))| \leq \beta_V$, $|f_k(G(u_2))| \leq \beta_V$ and

$$\begin{aligned} |f_k(G(u_1)) - f_k(G(u_2))| =& |G(u_1)(\mathbf{c}_k) - G(u_2)(\mathbf{c}_k)| \\ \leq& \sup_{\mathbf{y}\in\Omega_V} |G(u_1)(\mathbf{y}) - G(u_2)(\mathbf{y})| \\ \leq& L_G \|u_1 - u_2\|_{L^2(\Omega_U)}, \end{aligned} \tag{28}$$

where the last inequality follows from Assumption 1.

By Theorem 11, let C_F be a constant depending on $\gamma_1, \beta_U, d_1, L_G$ and C_δ be a constant depending on γ_1, d_1, L_f, L_U. For any $\varepsilon_2 > 0$, set $\delta = C'_\delta \varepsilon_2$. There exists a network architecture $\mathcal{F}_2 = \mathcal{F}_{\text{NN}}(c_U, 1, L_2, p_2, K_2, \kappa_2, R_2)$ with

$$\begin{aligned} &L_2 = O\left(c_U^2 \log c_U + c_U^2 \log(\varepsilon_2^{-1})\right),\ p_2 = O((C_F\sqrt{c_U})^{c_U} \varepsilon_2^{-c_U}), \\ &K_2 = O\left(((C_F\sqrt{c_U})^{c_U} \varepsilon_2^{-c_U})(c_U^2 \log c_U + c_U^2 \log(\varepsilon^{-1}))\right),\ \kappa_2 = O(c_U^{c_U/2+1} \varepsilon_2^{-c_U-1}),\ R = \beta_V, \end{aligned} \tag{29}$$

such that, for every functional f_k defined in (27), this network architecture gives a network $\widetilde{f}_k$ satisfying

$$\sup_{u\in U} |f_k(G(u)) - \widetilde{f}_k(\mathbf{u})| \le \varepsilon_2.$$

The constant hidden in O of (29) depends on $\gamma_1, \beta_U, \beta_V, d_1, L_G, L_U$.

Since $0 \le \widetilde{q}_k(\mathbf{y}) \le 1$ for any $\mathbf{y} \in \Omega_V$, we deduce

$$\begin{aligned} &\sup_{\mathbf{y}\in\Omega_V} \left| \sum_{k=1}^N f_k(G(u))\widetilde{q}_k(\mathbf{y}) - \sum_{k=1}^N \widetilde{f}_k(\mathbf{u})\widetilde{q}_k(\mathbf{y}) \right| \\ =& \sup_{\mathbf{y}\in\Omega_V} \left| \sum_{k=1}^N \left(f_k(G(u)) - \widetilde{f}_k(\mathbf{u}) \right) \widetilde{q}_k(\mathbf{y}) \right| \\ \le& \sup_{\mathbf{y}\in\Omega_V} \sum_{k=1}^N \left| \left(f_k(G(u)) - \widetilde{f}_k(\mathbf{u}) \right) \right| \widetilde{q}_k(\mathbf{y}) \\ \le& \sum_{k=1}^N \left\| f_k(G(u)) - \widetilde{f}_k(\mathbf{u}) \right\|_{L^\infty(U)} \widetilde{q}_k(\mathbf{y}) \\ \le& \varepsilon_2 \sum_{k=1}^N \widetilde{q}_k(\mathbf{y}). \end{aligned} \tag{30}$$

According to Theorem 10, for $\{\widetilde{q}_k\}_{k=1}^N$, the error bound (21) holds for any v in V. Set $v = \gamma_2$, a constant function. We have

$$\sup_{\mathbf{y}\in\Omega_V} \left| \gamma_2 - \sum_{k=1}^N \gamma_2 \widetilde{q}_k(\mathbf{y}) \right| = \sup_{\mathbf{y}\in\Omega_V} \left| \gamma_2 - \gamma_2 \sum_{k=1}^N \widetilde{q}_k(\mathbf{y}) \right| = \gamma_2 \sup_{\mathbf{y}\in\Omega_V} \left| 1 - \sum_{k=1}^N \widetilde{q}_k(\mathbf{y}) \right| \le \varepsilon_1,$$

implying that

$$\sum_{k=1}^N \widetilde{q}_k(\mathbf{y}) \le 1 + \varepsilon_1/\gamma_2 \le 1 + 1/\gamma_2 \tag{31}$$

when $\varepsilon_1 < 1$. Substituting (31) into (30) gives rise to

$$\sup_{\mathbf{y}\in\Omega_V} \left| \sum_{k=1}^N f_k(G(u))\widetilde{q}_k(\mathbf{y}) - \sum_{k=1}^N \widetilde{f}_k(\mathbf{u})\widetilde{q}_k(\mathbf{y}) \right| \le (1 + 1/\gamma_2)\varepsilon_2. \tag{32}$$

Putting (26) and (30) together, we have

$$\begin{aligned}
&\sup_{u\in U}\sup_{\mathbf{y}\in\Omega_V}\left|G(u)(\mathbf{y})-\sum_{k=1}^{N}\widetilde{f}_k(\mathbf{u})\widetilde{q}_k(\mathbf{y})\right|\\
\leq&\sup_{u\in U}\sup_{\mathbf{y}\in\Omega_V}\left|G(u)(y)-\sum_{k=1}^{N}f_k(G(u))\widetilde{q}_k(y)\right|+\sup_{u\in U}\sup_{\mathbf{y}\in\Omega_V}\left|\sum_{k=1}^{N}f_k(G(u))\widetilde{q}_k(y)-\sum_{k=1}^{N}\widetilde{a}_k(\mathbf{u})\widetilde{q}_k(y)\right|\\
\leq&\varepsilon_1+(1+1/\gamma_2)\varepsilon_2.
\end{aligned}$$

Set $\varepsilon_2=\frac{\varepsilon}{2(1+1/\gamma_2)},\varepsilon_1=\frac{\varepsilon}{2}$, we have $\delta=C_\delta\varepsilon$ and

$$\sup_{u\in U}\sup_{\mathbf{y}\in\Omega_V}\left|G(u)(\mathbf{y})-\sum_{k=1}^{N}\widetilde{a}_k(\mathbf{u})\widetilde{q}_k(\mathbf{y})\right|\leq\varepsilon,$$

here C_δ is a constant depending on $\gamma_1,\gamma_2,d_1,L_f,L_U$. The resulting network architectures have $N=O(\varepsilon^{-d_2})$,

$$L_1=O\left(\log(\varepsilon^{-1})\right),\ p_1=O(1),\ K_1=O\left(\log(\varepsilon^{-1})\right),\ \kappa_1=O(\varepsilon^{-d_2-1}),\ R_1=1,$$

and

$$\begin{aligned}
&L_2=O\left(c_U^2\log c_U+c_U^2\log(\varepsilon^{-1})\right),\ p_2=O((C_F\sqrt{c_U})^{c_U}\varepsilon^{-c_U}),\\
&K_2=O\left(((C_F\sqrt{c_U})^{c_U}\varepsilon^{-c_U})(c_U^2\log c_U+c_U^2\log(\varepsilon^{-1}))\right),\\
&\kappa_2=O(c_U^{c_U/2+1}\varepsilon^{-c_U-1}),\ R=\beta_V.
\end{aligned}$$

The constant hidden in O depends on $\gamma_1,\gamma_2,\beta_U,\beta_V,d_1,d_2,L_G,L_U,L_V$. ■

5.2 Proof of Theorem 7

Proof [Proof of Theorem 7] We rewrite the error as

$$\begin{aligned}
&\mathbb{E}_{\mathcal{S}}\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}(\widehat{G}(\mathbf{u})(\mathbf{y}_j)-G(u)(\mathbf{y}_j))^2\right]\\
=&\underbrace{2\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}(\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j})-G(u_i)(\mathbf{y}_{i,j}))^2\right]}_{\mathrm{T}_1}\\
&+\mathbb{E}_{\mathcal{S}}\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}(\widehat{G}(\mathbf{u})(\mathbf{y}_j)-G(u)(\mathbf{y}_j))^2\right]\\
&\qquad\underbrace{-2\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}(\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j})-G(u_i)(\mathbf{y}_{i,j}))^2\right]}_{\mathrm{T}_2},
\end{aligned}\tag{33}$$

where u_i and $\mathbf{y}_{i,j}$ are given in the training dataset $\mathcal{S}$.

• Bounding T_1. For T_1, we have

$$
\begin{aligned}
\mathrm{T}_1 =& 2\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}(\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j})-G(u)(\mathbf{y}_{i,j}))^2\right]\\
=& 2\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}(\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j})-v_{i,j}+\xi_{i,j})^2\right]\\
=& 2\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}\left[(\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j})-v_{i,j})^2+2(\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j})-v_{i,j})\xi_{i,j}+\xi_{i,j}^2\right]\right]\\
=& 2\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}(\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j}))-v_{i,j})^2\right]\\
&+4\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}(\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j})-v_{i,j})\xi_{i,j}\right]+2\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}\xi_{i,j}^2\right]\\
=& 2\mathbb{E}_{\mathcal{S}}\inf_{G_{\mathrm{NN}}\in\mathcal{G}_{\mathrm{NN}}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}(G_{\mathrm{NN}}(\mathbf{u}_i)(\mathbf{y}_{i,j})-v_{i,j})^2\right]\\
&+4\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}(\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j})-G(u_i)(\mathbf{y}_{i,j})-\xi_{i,j})\xi_{i,j}\right]+2\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}\xi_{i,j}^2\right]\\
\leq& 2\inf_{G_{\mathrm{NN}}\in\mathcal{G}_{\mathrm{NN}}}\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}(G_{\mathrm{NN}}(\mathbf{u}_i)(\mathbf{y}_{i,j})-v_{i,j})^2\right]\\
&+4\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}(\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j})-G(u_i)(\mathbf{y}_{i,j}))\xi_{i,j}\right]-2\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}\xi_{i,j}^2\right]\\
=& 2\inf_{G_{\mathrm{NN}}\in\mathcal{G}_{\mathrm{NN}}}\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}\left((G_{\mathrm{NN}}(\mathbf{u}_i)(\mathbf{y}_{i,j})-G(u_i)(\mathbf{y}_{i,j})-\xi_{i,j})^2-\xi_{i,j}^2\right)\right]\\
&+4\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}(\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j})-G(u_i)(\mathbf{y}_{i,j}))\xi_{i,j}\right]\\
=& 2\inf_{G_{\mathrm{NN}}\in\mathcal{G}_{\mathrm{NN}}}\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}(G_{\mathrm{NN}}(\mathbf{u})(\mathbf{y}_j)-G(u)(\mathbf{y}_j))^2\right]\\
&+4\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j})\xi_{i,j}\right] \qquad (34)
\end{aligned}
$$

The first term in (34) can be bounded by Corollary 6. More specifically, let C_F be a constant depending on $\gamma_1, \beta_1, d_1, L_G$. For any $\varepsilon > 0$, we choose $n_x = C_1\varepsilon^{-d_1}$ so that $\{\mathcal{B}_\delta(\mathbf{c}_m)\}_{m=1}^{n_x}$ is a cover of Ω_U with $\delta = C_\delta\varepsilon$ for some C_1, C_δ depending on γ_1, d_1, L_f and L_U. According to Corollary 6, we can set network architecture $\mathcal{F}_1 = \mathcal{F}_{\rm NN}(d_2, 1, L_1, p_1, K_1, \kappa_1, R_1)$ and $\mathcal{F}_2 = \mathcal{F}_{\rm NN}(n_x, 1, L_2, p_2, K_2, \kappa_2, R_2)$ with

$$L_1 = O\left(\log(\varepsilon^{-1})\right),\ p_1 = O(1),\ K_1 = O\left(\log(\varepsilon^{-1})\right),\ \kappa_1 = O(\varepsilon^{-d_2-1}),\ R_1 = 1 \tag{35}$$

and

$$\begin{aligned}
&L_2 = O\left(\varepsilon^{-2d_1}\log\varepsilon^{-1}\right),\ p_2 = O\left(C_F^{\varepsilon^{-d_1}}\varepsilon^{-(d_1/2+C_1)\varepsilon^{-d_1}}\right),\\
&K_2 = O\left(C_F^{\varepsilon^{-d_1}}\varepsilon^{-(d_1/2+C_1)\varepsilon^{-d_1}-2d_1}\log\varepsilon^{-1}\right),\ \kappa_2 = O\left((C_1\varepsilon)^{-(C_1d_1/2+d_1)\varepsilon^{-d_1}-d_1-1}\right),\ R = \beta_V,
\end{aligned} \tag{36}$$

The constant in O depends on $\gamma_1, \gamma_2, \beta_U, \beta_V, d_1, d_2, L_G, L_U, L_V$. Set $N = C\varepsilon^{-d_2}$ for some constant C depending on d_2, L_V and γ_2. There are $\{\widetilde{q}_k\}_{k=1}^N$ with $\widetilde{q}_k \in \mathcal{F}_1$ and $\{\widetilde{a}_k\}_{k=1}^N$ with $\widetilde{a}_k \in \mathcal{F}_2$ such that

$$\sup_{u\in U}\sup_{\mathbf{y}\in\Omega_V}\left|G(u)(\mathbf{y}) - \sum_{k=1}^N \widetilde{a}_k(\mathbf{u})\widetilde{q}_k(\mathbf{y})\right| \le \varepsilon.$$

The first term in (34) is bounded by

$$2\inf_{G_{\rm NN}\in\mathcal{G}_{\rm NN}}\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}\left(G_{\rm NN}(\mathbf{u})(\mathbf{y}_j) - G(u)(\mathbf{y}_j)\right)^2\right] \le 2\varepsilon^2. \tag{37}$$

The second term in (34) is bounded by the following lemma (see proof in Section B.6):

Lemma 15 *Denote* $\|G_{\rm NN}\|_n^2 = \frac{1}{n}\sum_{i=1}^n \frac{1}{n_y}\sum_{j=1}^{n_y}|G_{\rm NN}(\mathbf{u}_i)(\mathbf{y}_{i,j})|^2$. *Under the condition of Theorem 7, the second term in (34) is bounded as*

$$\begin{aligned}
&\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^n\frac{1}{n_y}\sum_{j=1}^{n_y}\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j})\xi_{i,j}\right]\\
\le& 2\sigma\left(\sqrt{\mathbb{E}_{\mathcal{S}}\left[\|\widehat{G}-G\|_n^2\right]}+\theta\right)\sqrt{\frac{4\log\mathcal{N}(\theta,\mathcal{G}_{\rm NN},\|\cdot\|_{\infty,\infty})+6}{nn_y}}+\sigma\theta.
\end{aligned} \tag{38}$$

Let $\widehat{G}$ be the network specified in (35) and (36). Substituting (37) and (38) into (34) gives rise to

$$\begin{aligned}
\mathrm{T}_1 =& 2\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^n\frac{1}{n_y}\sum_{j=1}^{n_y}(\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j}) - G(u_i)(\mathbf{y}_{i,j}))^2\right]\\
\le& 2\varepsilon^2 + 8\sigma\left(\sqrt{\mathbb{E}_{\mathcal{S}}\left[\|\widehat{G}-G\|_n^2\right]}+\theta\right)\sqrt{\frac{4\log\mathcal{N}(\theta,\mathcal{G}_{\rm NN},\|\cdot\|_{\infty,\infty})+6}{nn_y}}+4\sigma\theta.
\end{aligned} \tag{39}$$

Denote

$$\eta = \sqrt{\mathbb{E}_{\mathcal{S}}\left[\|\widehat{G} - G\|_n^2\right]},$$
$$a = \varepsilon^2 + 2\sigma\theta + 4\sigma\theta\sqrt{\frac{4\log\mathcal{N}(\theta, \mathcal{G}_{\mathrm{NN}}, \|\cdot\|_{\infty,\infty}) + 6}{nn_y}},$$
$$b = 2\sigma\sqrt{\frac{4\log\mathcal{N}(\theta, \mathcal{G}_{\mathrm{NN}}, \|\cdot\|_{\infty,\infty}) + 6}{nn_y}}.$$

Then (39) can be rewritten as

$$\eta^2 \le a + 2b\eta,$$

from which we deduce that

$$(\eta - b)^2 \le a + b^2 \quad \Rightarrow \quad \eta \le \sqrt{a + b^2} + b \quad \Rightarrow \quad \eta^2 \le 2a + 4b^2.$$

Thus we have

$$\begin{aligned}\mathrm{T}_1 =& 2\eta^2 \\ \le& 4\varepsilon^2 + 8\sigma\theta + 16\sigma\theta\sqrt{\frac{4\log\mathcal{N}(\theta, \mathcal{G}_{\mathrm{NN}}, \|\cdot\|_{\infty,\infty}) + 6}{nn_y}} + 16\sigma^2\frac{4\log\mathcal{N}(\theta, \mathcal{G}_{\mathrm{NN}}, \|\cdot\|_{\infty,\infty}) + 6}{nn_y}.\end{aligned} \tag{40}$$

• Bounding T_2. An upper bound of T_2 is given by the following lemma (see a proof in Section B.7)

Lemma 16 *Under the condition of Theorem 7, we have*

$$\mathrm{T}_2 \le \frac{19\beta_V^2}{n}\log\mathcal{N}\left(\frac{\theta}{4\beta_V}, \mathcal{G}_{\mathrm{NN}}, \|\cdot\|_{\infty,\infty}\right) + 6\theta. \tag{41}$$

• Putting $\mathrm{T}_1, \mathrm{T}_2$ together.

Substituting (40) and (87) into (33) gives rise to

$$\begin{aligned}&\mathbb{E}_{\mathcal{S}}\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}(\widehat{G}(\mathbf{u};\mathbf{y}_j)) - G(\mathbf{u};\mathbf{y}_j))^2\right] \\ \le& 4\varepsilon^2 + 8\sigma\theta + 16\sigma\theta\sqrt{\frac{4\log\mathcal{N}(\theta, \mathcal{G}_{\mathrm{NN}}, \|\cdot\|_{\infty,\infty}) + 6}{nn_y}} + 16\sigma^2\frac{4\log\mathcal{N}(\theta, \mathcal{G}_{\mathrm{NN}}, \|\cdot\|_{\infty,\infty}) + 6}{nn_y} \\ &+ \frac{19\beta_V^2}{n}\log\mathcal{N}\left(\frac{\theta}{4\beta_V}, \mathcal{G}_{\mathrm{NN}}, \|\cdot\|_{\infty,\infty}\right) + 6\theta \\ \le& 4\varepsilon^2 + \left(\frac{64\sigma^2 + 6}{nn_y} + \frac{19\beta_V^2}{n}\right)\log\mathcal{N}\left(\frac{\theta}{4\beta_V}, \mathcal{G}_{\mathrm{NN}}, \|\cdot\|_{\infty,\infty}\right)\end{aligned}$$

$$+ 16\sigma\theta\sqrt{\frac{4\log\mathcal{N}(\frac{\theta}{4\beta_V},\mathcal{G}_{\rm NN},\|\cdot\|_{\infty,\infty})+6}{nn_y}}+(8\sigma+6)\theta. \tag{42}$$

The following lemma (see a proof in Section B.8) gives an upper bound of $\mathcal{N}(\theta,\mathcal{G}_{\rm NN},\|\cdot\|_{\infty,\infty})$:

Lemma 17 *Let $\mathcal{F}_1=\mathcal{F}_{\rm NN}(d_1,1,L_1,p_1,K_1,\kappa_1,R_1)$ and $\mathcal{F}_2=\mathcal{F}_{\rm NN}(d_2,1,L_2,p_2,K_2,\kappa_2,R_2)$. We have*

$$\mathcal{N}(\theta,\mathcal{G}_{\rm NN},\|\cdot\|_{\infty,\infty})\le\left(\frac{2L_1p_1^2\kappa_1H}{\theta}\right)^{NK_1}\left(\frac{2L_2p_2^2\kappa_2H}{\theta}\right)^{NK_2},$$

with $H=N\left(R_1L_1(p_1\gamma_2+2)(\kappa_1p_1)^{L_1-1}+R_2L_2(p_2\beta_U+2)(\kappa_2p_2)^{L_2-1}\right)$.

Substituting (35) and (36) into Lemma 17 gives rise to

$$\begin{aligned}
&\log\mathcal{N}(\theta,\mathcal{G}_{\rm NN},\|\cdot\|_{\infty,\infty})\\
=&NK_1(\log 2+\log L_1+2\log p_1+\log\kappa_1+\log H+\log\frac{1}{\theta})\\
&+NK_2(\log 2+\log L_2+2\log p_2+\log\kappa_2+\log H+\log\frac{1}{\theta})\\
\le&C_4N(K_1+K_2)(\log H+\log\frac{1}{\theta})\\
\le&C_4N(K_1+K_2)(\log N+L_2(\log L_2+\log p_2+\log\kappa_2)+\log\frac{1}{\theta})\\
\le&C_4C_F^{\varepsilon^{-d_1}}\varepsilon^{-(d_1/2+C_1)\varepsilon^{-d_1}-2d_1-d_2}\log\frac{1}{\varepsilon}\left(\log\frac{1}{\varepsilon}+\varepsilon^{-2d_1}\log\frac{1}{\varepsilon}\left(\log\frac{1}{\varepsilon}+\varepsilon^{-d_1}\log\frac{1}{\varepsilon}\right)+\log\frac{1}{\theta}\right)\\
\le&C_4C_F^{\varepsilon^{-d_1}}\varepsilon^{-(d_1/2+C_1)\varepsilon^{-d_1}-5d_1-d_2}\log\frac{1}{\varepsilon}(\log^2\frac{1}{\varepsilon}+\log\frac{1}{\theta})
\end{aligned}\tag{43}$$

where C_4 is a constant depending on $\gamma_1,\gamma_2,\beta_U,\beta_V,d_1,d_2,L_G,L_U,L_V$.

Substituting (43) into (42) gives rise to

$$\begin{aligned}
&\mathbb{E}_{\mathcal{S}}\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}(\widehat{G}(\mathbf{u})(\mathbf{y}_j)-G(u)(\mathbf{y}_j))^2\right]\\
\le&4\varepsilon^2+\left(\frac{64\sigma^2+6}{nn_y}+\frac{19\beta_V^2}{n}\right)C_4C_F^{\varepsilon^{-d_1}}\varepsilon^{-(d_1/2+C_1)\varepsilon^{-d_1}-5d_1-d_2}\log\frac{1}{\varepsilon}(\log^2\frac{1}{\varepsilon}+\log\frac{1}{\theta})\\
&+16\sigma\theta\sqrt{\frac{4C_4C_F^{\varepsilon^{-d_1}}\varepsilon^{-(d_1/2+C_1)\varepsilon^{-d_1}-5d_1-d_2}\log\frac{1}{\varepsilon}(\log^2\frac{1}{\varepsilon}+\log\frac{1}{\theta})+6}{nn_y}}+(8\sigma+6)\theta.
\end{aligned}\tag{44}$$

Set $\theta=n^{-1}$. We have

$$\begin{aligned}
&\mathbb{E}_{\mathcal{S}}\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}(\widehat{G}(\mathbf{u})(\mathbf{y}_j)-G(\mathbf{u})(\mathbf{y}_j))^2\right]\\
&\qquad\le C_2\left(\varepsilon^2+\left(\frac{\sigma^2}{n_y}+1\right)\frac{1}{n}C_F^{\varepsilon^{-d_1}}\varepsilon^{-(d_1/2+C_1)\varepsilon^{-d_1}-5d_1-d_2}\right)\log^3\frac{1}{\varepsilon},
\end{aligned}\tag{45}$$

for some C_2 depending on $\gamma_1, \gamma_2, \beta_U, \beta_V, d_1, d_2, L_G, L_U, L_V$

In particular, set $\varepsilon = \left(\frac{d_1}{d_1+2C_1}\frac{\log n}{\log\log n}\right)^{-\frac{1}{d_1}}$. After taking logarithm, ε^2 is of $O(-\log\log n)$, $\frac{1}{n}\varepsilon^{-C_1(d_2+1)\varepsilon^{-d_1}-11d_1/2-d_2}$ is of $O(-\log(nn_y))$. Thus the error inside the parenthesis of (45) is dominated by ε^2. We have

$$\mathbb{E}_{\mathcal{S}}\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}(\widehat{G}(\mathbf{u})(\mathbf{y}_j)-G(\mathbf{u})(\mathbf{y}_j))^2\right] \le C_3\left(\frac{\log n}{\log\log n}\right)^{-\frac{2}{d_1}}. \tag{46}$$

for some constant C_3 depending on $\sigma, \gamma_1, \gamma_2, \beta_U, \beta_V, d_1, d_2, L_G, L_U, L_V$. ■

5.3 Proof of Theorem 8

To prove Theorem 8, we need the following approximation result for functionals defined on U satisfying Assumption 4.

Theorem 18 *Let $d_1, n_x, b_U > 0$ be integers, $\gamma_1, \beta_U, L_U, L_f, R_f > 0$, and U satisfy Assumption 2 and 4 (i), the discretization grids $\{\mathbf{x}_j\}_{j=1}^{n_x}$ satisfy Assumption 4 (ii). There exist constant C depending on $\gamma_1, \beta_U, d_1, L_f, R_f, b_U$ and C_1 depending on γ_1, d_1, β_U such that the following holds: For any $\varepsilon > 0$, set $H = (C\sqrt{b_U})^{b_U}\varepsilon^{-b_U}$ and the network $\mathcal{F}_{\mathrm{NN}}(n_x, 1, L, p, K, \kappa, R)$ with*

$$L = O\left(\log(\varepsilon^{-1})\right),\ p = O(1),\ K = O\left(\log(\varepsilon^{-1})\right),$$
$$\kappa = O(\varepsilon^{-b_U-1}),\ R = 1.$$

There are

$$\sup_{u\in U}\left|f(u) - \sum_{k=1}^{H} a_k\widetilde{q}_k(\mathbf{u})\right| \le \varepsilon, \tag{47}$$

where $\mathbf{u} = [u(\mathbf{x}_1), u(\mathbf{x}_2), ..., u(\mathbf{x}_{n_x})]^\top$, a_k's are coefficients depending on f and satisfying $|a_k| \le C_1$. The constant hidden in O depends on $\gamma_1, \beta_U, d_1, L_f, L_U, b_U, C_A$.

Theorem 18 is proved in Section B.4. Theorem 18 expresses the functional network as a sum of H parallel branches, and the network architecture of each branch is quantified. In the following, we express the functional network as one large network and quantify the network architecture of this large network.

If we set the network architecture $\mathcal{F}_{\mathrm{NN}}(n_x, 1, L, p, K, \kappa, R)$ as

$$L = O\left(\log(\varepsilon^{-1})\right),\ p = O(\varepsilon^{-b_U}),\ K = O\left(\varepsilon^{-b_U}\log(\varepsilon^{-1})\right),\ \kappa = O(\varepsilon^{-b_U-1}),\ R = R_f, \tag{48}$$

for any Lipschitz functional f with Lipschitz constant no more than L_f and $\|f\|_{L^\infty(U)} \le R_f$, there exists $\widetilde{f} \in \mathcal{F}_{\mathrm{NN}}(n_x, 1, L, p, K, \kappa, R)$ such that we have

$$\sup_{u\in U}|f(u) - \widetilde{f}(\mathbf{u})| \le \varepsilon. \tag{49}$$

The constant hidden in O in (48) depends on $\gamma_1, \beta_U, d_1, L_f, L_U, L_f, b_U, C_A$.

Now we are ready to prove Theorem 8.

Proof [Proof of Theorem 8] We follow the proof of Theorem 4 until (28). By Theorem 18, there exists a network architecture $\mathcal{F}_2 = \mathcal{F}_{\rm NN}(n_x, 1, L_2, p_2, K_2, \kappa_2, R_2)$ with

$$L_2 = O\left(\log(\varepsilon_2^{-1})\right),\ p_2 = O(\varepsilon_2^{-b_U}),\ K_2 = O\left(\varepsilon_2^{-b_U}\log(\varepsilon_2^{-1}) + n_x\right),\ \kappa_2 = O(\varepsilon_2^{-b_U-1}),\ R_2 = 1$$

such that for every functional f_k defined in (27), this network architecture gives a network $\widetilde{f}_k$ satisfying

$$\sup_{u\in U} |f_k(G(u)) - \widetilde{f}_k(\mathbf{u})| \le \varepsilon_2.$$

The constant hidden in O depends on $\gamma_1, \beta_U, \beta_V, d_1, L_G, L_U, C_A, b_U$. Since $0 \le \widetilde{q}_k(\mathbf{y}) \le 1$ for any $\mathbf{y} \in \Omega_V$, we deduce

$$\begin{aligned}
&\sup_{\mathbf{y}\in\Omega_V}\left|\sum_{k=1}^N f_k(G(u))\widetilde{q}_k(\mathbf{y}) - \sum_{k=1}^N \widetilde{f}_k(\mathbf{u})\widetilde{q}_k(\mathbf{y})\right| \\
=&\sup_{\mathbf{y}\in\Omega_V}\left|\sum_{k=1}^N \left(f_k(G(u)) - \widetilde{f}_k(\mathbf{u})\right)\widetilde{q}_k(\mathbf{y})\right| \\
\le&\sup_{\mathbf{y}\in\Omega_V}\sum_{k=1}^N \left|f_k(G(u)) - \widetilde{f}_k(\mathbf{u})\right|\widetilde{q}_k(\mathbf{y}) \\
\le&\left\|f_k(G(u)) - \widetilde{f}_k(\mathbf{u})\right\|_{L^\infty(U)}\sup_{\mathbf{y}\in\Omega_V}\sum_{k=1}^N \widetilde{q}_k(\mathbf{y}) \\
\le&\varepsilon_2 \sup_{\mathbf{y}\in\Omega_V}\sum_{k=1}^N \widetilde{q}_k(\mathbf{y}) \\
\le&(1+1/\gamma_2)\varepsilon_2,
\end{aligned} \tag{50}$$

where the last inequality follows from (31). Putting (26) and (50) together, we have

$$\begin{aligned}
&\sup_{u\in U}\sup_{\mathbf{y}\in\Omega_V}\left|G(u)(\mathbf{y}) - \sum_{k=1}^N \widetilde{f}_k(\mathbf{u})\widetilde{q}_k(\mathbf{y})\right| \\
\le&\sup_{u\in U}\sup_{\mathbf{y}\in\Omega_V}\left|G(u)(y) - \sum_{k=1}^N f_k(G(u))\widetilde{q}_k(y)\right| + \sup_{u\in U}\sup_{\mathbf{y}\in\Omega_V}\left|\sum_{k=1}^N f_k(G(u))\widetilde{q}_k(y) - \sum_{k=1}^N \widetilde{a}_k(\mathbf{u})\widetilde{q}_k(y)\right| \\
\le&\varepsilon_1 + (1+1/\gamma_2)\varepsilon_2.
\end{aligned}$$

Set $\varepsilon_2 = \varepsilon/(2(1+1/\gamma_2)), \varepsilon_1 = \frac{\varepsilon}{2}$, we have

$$\sup_{u\in U}\sup_{\mathbf{y}\in\Omega_V}\left|G(u)(\mathbf{y}) - \sum_{k=1}^N \widetilde{a}_k(\mathbf{u})\widetilde{q}_k(\mathbf{y})\right| \le \varepsilon.$$

The resulting network architecture has $N = O(\varepsilon^{-d_2})$ branch and trunk sub-networks. Each branch sub-network has parameters

$$L_1 = O\left(\log(\varepsilon^{-1})\right),\ p_1 = O(1),\ K_1 = O\left(\log(\varepsilon^{-1})\right),\ \kappa_1 = O(\varepsilon^{-1}),\ R_1 = 1,$$

and each trunk sub-network has parameters

$$\begin{aligned}&L_2 = O\left(\log(\varepsilon^{-1})\right),\ p_2 = O(\varepsilon^{-b_U}),\ K_2 = O\left(\varepsilon^{-b_U}(\log(\varepsilon^{-1}) + n_x)\right),\\ &\kappa_2 = O(\varepsilon^{-b_U-1}),\ R = \beta_V.\end{aligned}$$

The constant hidden in O depends on $\gamma_1, \gamma_2, \beta_U, \beta_V, d_1, d_2, L_G, L_U, L_V, C_A, b_U$. ■

5.4 Proof of Theorem 9

Proof [Proof of Theorem 9] We follow the proof of Theorem 7 until (42) and replace Corollary 6 by Theorem 8 and $\mathcal{F}_2$ by $\mathcal{F}_2 = \mathcal{F}_{\text{NN}}(n_x, 1, L_2, p_2, K_2, \kappa_2, R_2)$ with

$$\begin{aligned}&L_2 = O\left(\log(\varepsilon^{-1})\right),\ p_2 = O(\varepsilon^{-b_U}),\ K_2 = O\left(\varepsilon^{-b_U}(\log(\varepsilon^{-1}) + n_x)\right),\\ &\kappa_2 = O(\varepsilon^{-b_U-1}),\ R = \beta_V.\end{aligned} \tag{51}$$

Substituting (35) and (51) into Lemma 17 gives rise to

$$\begin{aligned}&\log \mathcal{N}(\theta, \mathcal{G}_{\text{NN}}, \|\cdot\|_{\infty,\infty})\\ =&NK_1(\log 2 + \log L_1 + 2\log p_1 + \log \kappa_1 + \log H + \log\frac{1}{\theta})\\ &+ NK_2(\log 2 + \log L_2 + 2\log p_2 + \log \kappa_2 + \log H + \log\frac{1}{\theta})\\ \le& C_4 N(K_1 + K_2)(\log H + \log\frac{1}{\theta})\\ \le& C_4 N(K_1 + K_2)(\log N + L_2(\log L_2 + \log p_2 + \log \kappa_2) + \log\frac{1}{\theta})\\ \le& C_4\varepsilon^{-b_U-d_2}\left(\log\frac{1}{\varepsilon} + n_x\right)\left(\log^2\frac{1}{\varepsilon} + \log\frac{1}{\theta}\right)\end{aligned} \tag{52}$$

where C_4 is a constant depending on $\gamma_1, \gamma_2, \beta_U, \beta_V, d_1, d_2, L_G, L_U, L_V, b_U, C_A$.
Substituting (52) into (42) gives rise to

$$\begin{aligned}&\mathbb{E}_{\mathcal{S}}\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}(\widehat{G}(\mathbf{u})(\mathbf{y}_j) - G(u)(\mathbf{y}_j))^2\right]\\ \le& 4\varepsilon^2 + \left(\frac{64\sigma^2+6}{nn_y} + \frac{19\beta_V^2}{n}\right)C_4\varepsilon^{-b_U-d_2}\left(\log\frac{1}{\varepsilon} + n_x\right)\left(\log^2\frac{1}{\varepsilon} + \log\frac{1}{\theta}\right)\\ &+ 16\sigma\theta\sqrt{\frac{4C_4\varepsilon^{-b_U-d_2}\left(\log\frac{1}{\varepsilon} + n_x\right)\left(\log^2\frac{1}{\varepsilon} + \log\frac{1}{\theta}\right) + 6}{nn_y}} + (8\sigma+6)\theta.\end{aligned} \tag{53}$$

Set $\theta = n^{-1}$ and $\varepsilon = n^{-\frac{1}{2+b_U+d_2}}$, we have

$$\mathbb{E}_{\mathcal{S}}\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}(\widehat{G}(\mathbf{u})(\mathbf{y}_j)-G(\mathbf{u})(\mathbf{y}_j))^2\right] \leq C_2 n^{-\frac{2}{2+b_U+d_2}}\log^3\frac{1}{\varepsilon} \tag{54}$$

for some C_2 depending on $\sigma, \gamma_1, \gamma_2, \beta_U, \beta_V, d_1, d_2, L_G, L_U, L_V, b_U, C_A$. ■

6. Conclusion

In this paper, we have developed mathematical and statistical theories to justify neural scaling laws of DeepONet by analyzing its approximation and generalization error. Our approximation theory can be used to quantify the model scaling law of DeepONet, depicting the scaling between the DeepONet approximation error and the model size. Our generalization theory can be used to quantify the data scaling law of DeepONet, depicting the scaling between the DeepONet generalization error and the training data size. Our general results for learning Lipschitz operators give rise to a slow rate of convergence of the DeepONet error as the model/data size increases. Furthermore, we incorporate low-dimensional structures of the input functions into consideration, and improve the rate of convergence to a power law, which is consistent with the empirical observations in Lu et al. (2021b); de Hoop et al. (2022). Our results provide theoretical foundations of DeepONet, to partially explain the empirical success and neural scaling laws of DeepONet. In the future, we will investigate the optimality of our error bound, and improve it if possible. Another interesting future direction is to incorporate more complicated low-dimensional structures, such as manifold setting, under the framework of DeepONet.

Acknowledgments

Hao Liu was partially supported by HKRGC GRF 12301925 and 12302926, and the Guangdong and Hong Kong Universities "1+1+1" Joint Research Collaboration Scheme 2025A0505000007. Zecheng Zhang was partially supported by the U.S. Department of Energy under grant DE-SC0025440. Wenjing Liao was partially supported by NSF DMS 2145167 and DOE SC0024348. Hayden Schaeffer was partially supported by NSF 2427558 and NSF 2331033.

Appendix

Appendix A. Proof of Examples in Section 3

A.1 Proof of Example 2

Proof [Proof of Example 2]
The exact solution is given as

$$v(\mathbf{x}, T) = u(\mathbf{x} + T\mathbf{c}).$$

For any $u, \bar{u} \in U$ with $u = \sum_{j=1}^{J} a_j w_j, \bar{u} = \sum_{j=1}^{J} \bar{a}_j w_j$, we have

$$\begin{aligned}\|G(u) - G(\bar{u})\|_{L^\infty(\Omega_V)} =& \left\| \sum_{j=1}^{J} |a_j - \bar{a}_j| w_j(\mathbf{x} + T\mathbf{c}) \right\|_{L^\infty(\Omega_V)} \leq \|\mathbf{a} - \bar{\mathbf{a}}\|_{\ell^1} \\ \leq& \sqrt{J}\|\mathbf{a} - \bar{\mathbf{a}}\|_{\ell^2} = \sqrt{J}\|u - \bar{u}\|_{L^2(\Omega_U)}.\end{aligned}$$

■

Appendix B. Proof of Theorems and Lemmata in Section 5

B.1 Proof of Theorem 10

Proof [Proof of Theorem 10] We partition Ω_U into N^{d_1} subcubes for some N to be specified later. We are going to approximate u on each cube by a constant function and then assemble them together to get an approximation of u on Ω_U. Denote the centers of the subcubes by $\{\mathbf{c}_k\}_{k=1}^{N^{d_1}}$ with $\mathbf{c}_k = [c_{k,1}, c_{k,2}, ..., c_{k,d_1}]^\intercal$.

Let $\{\mathbf{c}_k\}_{k=1}^{N^{d_1}}$ be a uniform grid on Ω_U so that each $\mathbf{c}_k \in \left\{-\gamma_1, -\gamma_1 + \frac{2\gamma_1}{N-1}, ..., \gamma_1\right\}^{d_1}$ for each k. Define

$$\psi(a) = \begin{cases} 1, |a| < 1, \\ 0, |a| > 2, \\ 2 - |a|, 1 \leq |a| \leq 2, \end{cases} \tag{55}$$

with $a \in \mathbb{R}$, and

$$\phi_{\mathbf{c}_k}(\mathbf{x}) = \prod_{j=1}^{d_1} \psi\left(\frac{3(N-1)}{2\gamma_1}(x_j - c_{k,j})\right). \tag{56}$$

In this definition, we have $\text{supp}(\phi_{\mathbf{c}_k}) = \left\{\mathbf{x} : \|\mathbf{x} - \mathbf{c}_k\|_\infty \leq \frac{4\gamma_1}{3(N-1)}\right\} \subset \left\{\mathbf{x} : \|\mathbf{x} - \mathbf{c}_k\|_\infty \leq \frac{2\gamma_1}{(N-1)}\right\}$ and

$$\|\phi_{\mathbf{c}_k}\|_{L^\infty(\Omega_U)} = 1, \quad \sum_{k=1}^{N^{d_1}} \phi_{\mathbf{c}_k} = 1.$$

For any $u \in U$, we construct a piecewise constant approximation to u as

$$\bar{u}(\mathbf{x}) = \sum_{k=1}^{N^{d_1}} u(\mathbf{c}_k)\phi_{\mathbf{c}_k}(\mathbf{x}).$$

By utilizing the partition of unity property given by $\sum_{k=1}^{N^{d_1}} \phi_{\mathbf{c}_k} = 1$, it follows that for any $\mathbf{x} \in \Omega_U$,

$$\begin{aligned}
|u(\mathbf{x}) - \bar{u}(\mathbf{x})| &= \left| \sum_{k=1}^{N^{d_1}} \phi_{\mathbf{c}_k}(\mathbf{x})(u(\mathbf{x}) - u(\mathbf{c}_k)) \right| \\
&\leq \sum_{k=1}^{N^{d_1}} \phi_{\mathbf{c}_k}(\mathbf{x})|u(\mathbf{x}) - u(\mathbf{c}_k)| \\
&= \sum_{k:\|\mathbf{c}_k - \mathbf{x}\|_\infty \leq \frac{2\gamma_1}{(N-1)}} \phi_{\mathbf{c}_k}(\mathbf{x})|(u(\mathbf{x}) - u(\mathbf{c}_k))| \\
&\leq \max_{k:\|\mathbf{c}_k - \mathbf{x}\|_\infty \leq \frac{2\gamma_1}{(N-1)}} |(u(\mathbf{x}) - u(\mathbf{c}_k))| \left(\sum_{k:\|\mathbf{c}_k - \mathbf{x}\|_\infty \leq \frac{2\gamma_1}{(N-1)}} \phi_{\mathbf{c}_k}(\mathbf{x}) \right) \\
&\leq \max_{k:\|\mathbf{c}_k - \mathbf{x}\|_\infty \leq \frac{2\gamma_1}{(N-1)}} |(u(\mathbf{x}) - u(\mathbf{c}_k))| \\
&\leq \frac{2\sqrt{d_1}\gamma_1 L_U}{N-1}, \qquad (57)
\end{aligned}$$

where we use the Lipschitz assumption of U in the last inequality. Setting $N = \left\lceil \frac{4\sqrt{d_1}\gamma_1 L_U}{\varepsilon} \right\rceil + 1$ gives rise to

$$|u(\mathbf{x}) - \bar{u}(\mathbf{x})| \leq \frac{\varepsilon}{2}, \quad \forall \mathbf{x} \in \Omega_U. \qquad (58)$$

We then show that $\phi_{\mathbf{c}_k}$ can be approximated by a network with arbitrary accuracy. Note that $\phi_{\mathbf{c}_k}$ is the product of d_1 functions, each of which is piecewise linear and can be realized by 4-layer ReLU networks.

The following lemma shows that a function of the product can be approximated by a network with arbitrary accuracy.

Lemma 19 (Proposition 3 of Yarotsky (2017)) *Given $M > 0$ and $\varepsilon > 0$, there is a ReLU network $\widetilde{\times} : \mathbb{R}^2 \to \mathbb{R}$ in $\mathcal{F}_{\mathrm{NN}}(2, 1, L, p, K, \kappa, R)$ such that for any $|x| \leq M, |y| \leq M$, we have*

$$|\widetilde{\times}(x, y) - xy| < \varepsilon.$$

The network architecture has

$$L = O(\log \varepsilon^{-1}),\ p = 6,\ K = O(\log \varepsilon^{-1}),\ \kappa = O(\varepsilon^{-1}),\ R = M^2. \qquad (59)$$

The constant hidden in O depends on M.

Let $\widetilde{\times}$ be the network defined in Lemma 19 with accuracy δ. We approximate $\phi_{\mathbf{c}_k}$ by $\bar{q}_k$ defined as,

$$\bar{q}_k(\mathbf{x}) = \widetilde{\times}\left(\psi\left(\frac{3(N-1)}{2\gamma_1}(x_1 - c_{k,1})\right), \widetilde{\times}\left(\psi\left(\frac{3(N-1)}{2\gamma_1}(x_2 - c_{k,2})\right), \cdots\right)\right).$$

For each k, $\bar{q}_k \in \mathcal{F}_{\mathrm{NN}}(d_1, 1, L, p, K, \kappa, R)$ with

$$L = O(d_1 \log \delta^{-1}), p = O(1), K = O(d_1 \log \delta^{-1}), \kappa = O(\delta^{-1} + N), R = 1.$$

For any $\mathbf{x} \in \Omega_U$, we have

$$\begin{aligned}
&|\bar{q}_k(\mathbf{x}) - \phi_{\mathbf{c}_k}(\mathbf{x})| \\
\leq &\left|\widetilde{\times}\left(\psi\left(\frac{3(N-1)}{2\gamma_1}(x_1 - c_{k,1})\right), \widetilde{\times}\left(\psi\left(\frac{3(N-1)}{2\gamma_1}(x_2 - c_{k,2})\right), \cdots\right)\right) - \phi_{\mathbf{c}_k}(\mathbf{x})\right| \\
\leq &\left|\widetilde{\times}\left(\psi\left(\frac{3(N-1)}{2\gamma_1}(x_1 - c_{k,1})\right), \widetilde{\times}\left(\psi\left(\frac{3(N-1)}{2\gamma_1}(x_2 - c_{k,2})\right), \cdots\right)\right)\right. \\
&\left. - \psi\left(\frac{3(N-1)}{2\gamma_1}(x_1 - c_{k,1})\right) \widetilde{\times}\left(\psi\left(\frac{3(N-1)}{2\gamma_1}(x_2 - c_{k,2})\right), \cdots\right)\right| \\
&+ \left|\psi\left(\frac{3(N-1)}{2\gamma_1}(x_1 - c_{k,1})\right) \widetilde{\times}\left(\psi\left(\frac{3(N-1)}{2\gamma_1}(x_2 - c_{k,2})\right), \cdots\right) - \phi_{\mathbf{c}_k}(\mathbf{x})\right| \\
\leq &\delta + \mathcal{E}_2,
\end{aligned}$$

where

$$\begin{aligned}
\mathcal{E}_2 &= \left|\psi\left(\frac{3(N-1)}{2\gamma_1}(x_1 - c_{k,1})\right) \widetilde{\times}\left(\psi\left(\frac{3(N-1)}{2\gamma_1}(x_2 - c_{k,2})\right), \cdots\right) - \phi_{\mathbf{c}_k}(\mathbf{x})\right| \\
&= \left|\psi\left(\frac{3(N-1)}{2\gamma_1}(x_1 - c_{k,1})\right)\right| \left|\widetilde{\times}\left(\psi\left(\frac{3(N-1)}{2\gamma_1}(x_2 - c_{k,2})\right), \cdots\right) - \prod_{j=2}^{d_1} \psi\left(\frac{3(N-1)}{2\gamma_1}(x_j - c_{k,j})\right)\right|
\end{aligned}$$

Repeat this process to estimate $\mathcal{E}_2, \mathcal{E}_3, ..., \mathcal{E}_{d_1+1}$, where $\mathcal{E}_{d_1+1} = \prod_{k=1}^{d_1} \psi\left(\frac{3(N-1)}{2\gamma_1}(x_2 - c_{k,2})\right) - \phi_{\mathbf{c}_k} = 0$. This implies that $\|\phi_{\mathbf{c}_k} - \bar{q}_k\|_{L^\infty(\Omega_U)} \leq d_1\delta$. We define

$$\widetilde{q}_k = \mathrm{ReLU}(\bar{q}_k) = \max\{0, \bar{q}_k\}.$$

Since $\phi_{\mathbf{c}_k} \geq 0$, we have

$$\|\phi_{\mathbf{c}_k} - \widetilde{q}_k\|_{L^\infty(\Omega_U)} \leq \|\phi_{\mathbf{c}_k} - \bar{q}_k\|_{L^\infty(\Omega_U)} \leq d_1\delta$$

and $\widetilde{q}_k \geq 0$. It follows that,

$$\begin{aligned}
\left\|\sum_{k=1}^{N^{d_1}} u(\mathbf{c}_k)\widetilde{q}_k - \bar{u}\right\|_{L^\infty(\Omega_U)} &= \left\|\sum_{k=1}^{N^{d_1}} u(\mathbf{c}_k)\widetilde{q}_k - \sum_{k=1}^{N^{d_1}} u(\mathbf{c}_k)\phi_{\mathbf{c}_k}\right\|_{L^\infty(\Omega_U)} \\
&\leq \sum_{k=1}^{N^{d_1}} |u(\mathbf{c}_k)| \|\widetilde{q}_k - \phi_{\mathbf{c}_k}\|_{L^\infty(\Omega_U)}
\end{aligned}$$

$$\leq d_1 N^{d_1} \beta_U \delta. \tag{60}$$

Setting $\delta = \frac{\varepsilon}{2d_1 N^{d_1} \beta_U}$ and putting (58) and (60) together, we have

$$\left\| u - \sum_{k=1}^{N^{d_1}} u(\mathbf{c}_k)\widetilde{q}_k \right\|_{L^\infty(\Omega_U)} \leq \|u - \bar{u}\|_{L^\infty(\Omega_U)} + \left\| \bar{u} - \sum_{k=1}^{N^{d_1}} u(\mathbf{c}_k)\widetilde{q}_k \right\|_{L^\infty(\Omega_U)} \leq \frac{\varepsilon}{2} + \frac{\varepsilon}{2} = \varepsilon. \tag{61}$$

The network architecture is specified in the theorem. ■

B.2 Proof of Theorem 11

Proof [Proof of Theorem 11]

We let $\{\mathcal{B}_\delta(\mathbf{c}_m)\}_{m=1}^{c_U}$ be a finite cover of Ω_U by c_U Euclidean balls, where c_U can be further estimated in Corollary 12. By Lemma 2, there exists a partition of unity $\{\omega_m(\mathbf{x})\}_{m=1}^{c_U}$ subordinate to the cover $\{\mathcal{B}_\delta(\mathbf{c}_m)\}_{m=1}^{c_U}$. For any $\mathbf{z} = [z_1, ..., z_{c_U}]^\top \in (-\beta_U, \beta_U)^{c_U}$, we can then define a function $z_\omega(\mathbf{x}) : \Omega_U \to \mathbb{R}$ such that

$$z_\omega(\mathbf{x}) = \sum_{m=1}^{c_U} \mathbf{z}_m \omega_m(\mathbf{x}) \quad \forall \mathbf{x} \in \Omega_U. \tag{62}$$

Note that for $u \in U$ and $\mathbf{u} = [u(\mathbf{c}_1), ..., u(\mathbf{c}_{c_U})]^\top$, if we set $\mathbf{z} = \mathbf{u}$, it follows that $u_\omega = z_\omega \in \bar{U}_{\beta_U}$ is an approximation of u with the point-wise error estimation:

$$\begin{aligned} |u(\mathbf{x}) - u_\omega(\mathbf{x})| &\leq \sum_{m=1}^{c_U} |u(\mathbf{x}) - u(\mathbf{c}_m)||\omega_m(\mathbf{x})| \\ &= \sum_{m:\|\mathbf{x}-\mathbf{c}_m\|_2 \leq \delta} |u(\mathbf{x}) - u(\mathbf{c}_m)||\omega_m(\mathbf{x})| \leq L_U \delta \end{aligned}$$

for any $\mathbf{x} \in \Omega_U$. Setting $\delta = \frac{\varepsilon}{2(2\gamma_1)^{d_1/2} L_f L_U}$ and using the Lipschtiz property of f, we have

$$|f(u) - f(u_\omega)| \leq L_f \|u - u_\omega\|_{L^2(\Omega_U)} \leq L_f (2\gamma_1)^{d_1/2} L_U \delta = \frac{\varepsilon}{2}.$$

We next define a function $g : (-\beta_U, \beta_U)^{c_U} \to \mathbb{R}$ such that $g(\mathbf{z}) = f(z_\omega)$, i,e., $g(\mathbf{z}) = f(u_\omega)$. We claim that g is Lipschitz in the following sense: For any $u, \bar{u} \in U$, define u_ω and $\bar{u}_\omega$ as in (62) where $\mathbf{u} = [u(\mathbf{c}_1), ..., u(\mathbf{c}_{c_U})]^\top$ and $\bar{\mathbf{u}} = [\bar{u}(\mathbf{c}_1), ..., \bar{u}(\mathbf{c}_{c_U})]^\top$. Then we have

$$\begin{aligned} |g(\mathbf{u}) - g(\bar{\mathbf{u}})| =& |f(u_\omega) - f(\bar{u}_\omega)| \\ \leq& L_f \|u_\omega - \bar{u}_\omega\|_{L^2(\Omega_U)} \\ =& L_f \sqrt{\int_{\Omega_U} (u_\omega - \bar{u}_\omega)^2 d\mathbf{x}} \\ =& L_f \sqrt{\int_{\Omega_U} \left(\sum_{m=1}^{c_U} \left(u(\mathbf{c}_m) - \bar{u}(\mathbf{c}_m) \right) \omega_m(\mathbf{x}) \right)^2 d\mathbf{x}} \end{aligned}$$

$$
\begin{aligned}
\leq & L_f \sqrt{\int_{\Omega_U} \sum_{m=1}^{c_U} \left(u(\mathbf{c}_m) - \bar{u}(\mathbf{c}_m)\right)^2 \sum_{m=1}^{c_U} \left(\omega_m(\mathbf{x})\right)^2 \, d\mathbf{x}} \\
\leq & L_f \sqrt{\int_{\Omega_U} \sum_{m=1}^{c_U} \left(u(\mathbf{c}_m) - \bar{u}(\mathbf{c}_m)\right)^2 \sum_{m=1}^{c_U} \omega_m(\mathbf{x}) \, d\mathbf{x}} \\
\leq & L_f \sqrt{\int_{\Omega_U} \sum_{m=1}^{c_U} \left(u(\mathbf{c}_m) - \bar{u}(\mathbf{c}_m)\right)^2 \, d\mathbf{x}} \\
= & L_f \left|\Omega_U\right|^{\frac{1}{2}} \|\mathbf{u} - \bar{\mathbf{u}}\|_2 \\
= & L_f (2\gamma_1)^{d_1/2} \|\mathbf{u} - \bar{\mathbf{u}}\|_2
\end{aligned}
$$

where the third equality follows from the property that $\{\omega_m\}_{m=1}^{c_U}$ is a partition of unity. The claim is proved.

By Theorem 10, for $\varepsilon > 0$, if we set $H = (C\sqrt{c_U})^{c_U} \varepsilon^{-c_U}$ for some C depending on d_1, γ_1, β_U and L_f, then there exists a network architecture $\mathcal{F}_{\text{NN}}(c_U, 1, L, p, K, \kappa, R)$ and $\{\widetilde{q}_k\}_{k=1}^H$ with $\widetilde{q}_k \in \mathcal{F}_{\text{NN}}(c_U, 1, L, p, K, \kappa, R)$ for $k = 1, \ldots, H$ such that

$$
\sup_{u \in U} \left| g(\mathbf{u}) - \sum_{k=1}^{H} a_k \widetilde{q}_k(\mathbf{u}) \right| \leq \frac{\varepsilon}{2},
$$

where a_k are constants depending on f with $|a_k| \leq R_f$. Such an architecture has

$$
\begin{aligned}
& L = O\left(c_U^2 \log(c_U) + c_U^2 \log(\varepsilon^{-1})\right), \; p = O(1), \; K = O\left(c_U^2 \log c_U + c_U^2 \log(\varepsilon^{-1})\right), \\
& \kappa = O(c_U^{c_U/2+1} \varepsilon^{-c_U - 1}), \; R = 1.
\end{aligned}
$$

The constant hidden in O depends on d_1, γ_1, β_U and L_f. We have, for any $u \in U$ and $\mathbf{u} = [u(\mathbf{c}_1), ..., u(\mathbf{c}_{c_U})]^\top$

$$
\begin{aligned}
\sup_{u \in U} \left| f(u) - \sum_{k=1}^{H} a_k \widetilde{q}_k(\mathbf{u}) \right| \leq & \sup_{u \in U} |f(u) - g(\mathbf{u})| + \sup_{\boldsymbol{u}} \left| g(\mathbf{u}) - \sum_{k=1}^{H} a_k \widetilde{q}_k(\mathbf{u}) \right| \\
\leq & \frac{\varepsilon}{2} + \frac{\varepsilon}{2} = \varepsilon.
\end{aligned}
$$

■

B.3 Proof of Corollary 12

Proof [Proof of Corollary 12] Ω_U is bounded and closed; hence it is compact. Let $\{\mathcal{B}_\delta(\mathbf{c}_m)\}_{m=1}^{c_U}$ be a finite cover of Ω_U by c_U Euclidean balls, with centers $\{\mathbf{c}_m\}_{m=1}^{c_U}$ and radius δ. By Lemma 5, we have,

$$
c_U \leq C_2 \delta^{-d_1} = C_2 \left(\frac{2(2\gamma_1)^{d_1/2} L_f L_U}{\varepsilon} \right)^{d_1} \tag{63}
$$

for some C_2 depending on γ_1 and d_1. Then Corollary 12 is a direct result of Theorem 11. ■

B.4 Proof of Theorem 18

Proof [Proof of Theorem 18] Under Assumption 3 and Assumption 4(ii), for each $u \in U$, we have $|\alpha_k| \leq C_\alpha$ for $k = 1, \ldots, b_U$ where $C_\alpha = (2\gamma_1)^{d_1/2}\beta_U$. For any $\mathbf{z} = [z_1, ..., z_{b_U}]^\top \in [-C_\alpha, C_\alpha]^{b_U}$, we define the function $z_\omega : \Omega_U \to \mathbb{R}$ such that

$$z_\omega(\mathbf{x}) = \sum_{m=1}^{b_U} \mathbf{z}_m \omega_m(\mathbf{x}), \ \forall \mathbf{x} \in \Omega_U, \tag{64}$$

where $\{\omega_m\}_{m=1}^{b_U}$ are the orthonormal basis in Assumption 4(i).

For $u \in U$ and $\mathbf{u} = [u(\mathbf{x}_1), ..., u(\mathbf{x}_{n_x})]^\top$, if $\mathbf{z} = A\mathbf{u}$ then we have $u = z_\omega$. Let us define the function $g : [-C_\alpha, C_\alpha]^{b_U} \to \mathbb{R}$ such that $g(\mathbf{z}) = f(z_\omega)$ (i.e., $g(A\mathbf{u}) = f(u)$). Then g is Lipschitz in the following sense: For any $u, \bar{u} \in U$, let $\mathbf{u} = [u(\mathbf{x}_1), ..., u(\mathbf{x}_{n_x})]^\top$, $\bar{\mathbf{u}} = [\bar{u}(\mathbf{x}_1), ..., \bar{u}(\mathbf{x}_{n_x})]^\top$, and then we have

$$\begin{aligned}
|g(\mathbf{z}) - g(\bar{\mathbf{z}})| =& |f(z_\omega) - f(\bar{z}_\omega)| \\
\leq& L_f \|z_\omega - \bar{z}_\omega\|_{L^2(\Omega_U)} \\
=& L_f \sqrt{\int_{\Omega_U} (z_\omega - \bar{z}_\omega)^2 d\mathbf{x}} \\
\leq& L_f \sqrt{\int_{\Omega_U} \left(\sum_{m=1}^{b_U} |\mathbf{z}_m - \bar{\mathbf{z}}_m| \omega_m(\mathbf{x}) \right)^2 d\mathbf{x}} \\
\leq& L_f \sqrt{\int_{\Omega_U} \sum_{m=1}^{b_U} |\mathbf{z}_m - \bar{\mathbf{z}}_m|^2 \omega_m^2(\mathbf{x}) d\mathbf{x}} \\
=& L_f \sqrt{\sum_{m=1}^{b_U} |\mathbf{z}_m - \bar{\mathbf{z}}_m|^2 \int_{\Omega_U} \omega_m^2(\mathbf{x}) d\mathbf{x}} \\
=& L_f \|\mathbf{z} - \bar{\mathbf{z}}\|_2.
\end{aligned}$$

By Theorem 10, for $\varepsilon > 0$, set $H = (C\sqrt{b_U})^{b_U} \varepsilon^{-b_U}$ for some C depending on $b_U, d_1, \gamma_1, \beta_U$ and L_f. There exists a network architecture $\mathcal{F}_{\mathrm{NN}}(n_x, 1, L, p, K, \kappa, R)$ and $\{\widetilde{q}_k\}_{k=1}^H$ with $\widetilde{q}_k \in \mathcal{F}_{\mathrm{NN}}(n_x, 1, L, p, K, \kappa, R)$ and $\widetilde{q}_k \geq 0$, for $k = 1, ..., H$ such that

$$\sup_{u \in U} \left| g(A\mathbf{u}) - \sum_{k=1}^{H} a_k \widetilde{q}_k(\mathbf{u}) \right| \leq \varepsilon, \tag{65}$$

where a_k are constants depending on f with $|a_k| \leq R_f$. Such an architecture has

$$L = O(\log(\varepsilon^{-1})), \ p = O(1), \ K = O(\log(\varepsilon^{-1}) + n_x), \ \kappa = O(\varepsilon^{-b_U - 1}), \ R = 1.$$

Note that κ depends on C_A as defined in Assumption 4 as the network weights are scaled up by A. We have for any $u \in U$,

$$\sup_{u\in U}\left|f(u)-\sum_{k=1}^{H}a_k\widetilde{q}_k(\mathbf{u})\right| = \sup_{u\in U}\left|g(A\mathbf{u})-\sum_{k=1}^{H}a_k\widetilde{q}_k(\mathbf{u})\right| = \varepsilon.$$

■

B.5 Proof of Lemma 5

Proof [Proof of Lemma 5] By Conway and Sloane (2013, Chapter 2), we have,

$$c \le \left\lceil \frac{2\gamma}{\delta} \right\rceil^d + 7d\log d \le C\delta^{-d} \tag{66}$$

for some C depending on γ and d. ■

B.6 Proof of Lemma 15

Proof [Proof of Lemma 15] We derive an upper bound of the second term in (34) using the covering number of $\mathcal{G}_{\mathrm{NN}}$. Denote $\|G_{\mathrm{NN}}\|_n^2 = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}|G_{\mathrm{NN}}(\mathbf{u}_i)(\mathbf{y}_{i,j})|^2$. Let $\mathcal{G}^* = \{G_k^*\}_{k=1}^{\mathcal{N}(\theta,\mathcal{G}_{\mathrm{NN}},\|\cdot\|_{\infty,\infty})}$ be a θ cover of $\mathcal{G}_{\mathrm{NN}}$, where $\mathcal{N}(\theta,\mathcal{G}_{\mathrm{NN}},\|\cdot\|_{\infty,\infty})$ is the covering number. Specifically, for any $G_{\mathrm{NN}} \in \mathcal{G}_{\mathrm{NN}}$, there exists $G_{\mathrm{NN}}^* \in \mathcal{G}^*$ satisfying $\|G_{\mathrm{NN}}^* - \widehat{G}\|_{\infty,\infty} \le \theta$. We have

$$\begin{aligned}
&\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j})\xi_{i,j}\right]\\
=&\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}\left(\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j})-G_{\mathrm{NN}}^*(\mathbf{u}_i)(\mathbf{y}_{i,j})+G_{\mathrm{NN}}^*(\mathbf{u}_i)(\mathbf{y}_{i,j})-G(u_i)(\mathbf{y}_{i,j})\right)\xi_{i,j}\right]\\
=&\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}\left(\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j})-G_{\mathrm{NN}}^*(\mathbf{u}_i)(\mathbf{y}_{i,j})\right)\xi_{i,j}\right]\\
&+\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}\left(G_{\mathrm{NN}}^*(\mathbf{u}_i)(\mathbf{y}_{i,j})-G(u_i)(\mathbf{y}_{i,j})\right)\xi_{i,j}\right]\\
\le&\sigma\theta+\mathbb{E}_{\mathcal{S}}\left[\frac{\|G_{\mathrm{NN}}^*-G\|_n}{\sqrt{nn_y}}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n_y}\left(G_{\mathrm{NN}}^*(\mathbf{u}_i)(\mathbf{y}_{i,j})-G(u_i)(\mathbf{y}_{i,j})\right)\xi_{i,j}}{\sqrt{nn_y}\|G_{\mathrm{NN}}^*-G\|_n}\right].
\end{aligned} \tag{67}$$

Note that

$$\|G_{\mathrm{NN}}^*-G\|_n$$

$$
\begin{aligned}
=&\sqrt{\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}|G^*_{\mathrm{NN}}(\mathbf{u}_i)(\mathbf{y}_{i,j})-\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j})+\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j})-G(u_i)(\mathbf{y}_{i,j})|^2}\\
\leq&\sqrt{\frac{2}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}\left(|G^*_{\mathrm{NN}}(\mathbf{u}_i)(\mathbf{y}_{i,j})-\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j})|^2+|\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j})-G(u_i)(\mathbf{y}_{i,j})|^2\right)}\\
\leq&\sqrt{\frac{2}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}\left(\theta^2+|\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j})-G(u_i)(\mathbf{y}_{i,j})|^2\right)}\\
=&\sqrt{2}(\|\widehat{G}-G\|_n+\theta). \qquad (68)
\end{aligned}
$$

Substituting (68) into (67) gives rise to

$$
\begin{aligned}
&\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j})\xi_{i,j}\right]\\
\leq&\sqrt{2}\mathbb{E}_{\mathcal{S}}\left[\frac{\|\widehat{G}-G\|_n+\theta}{\sqrt{nn_y}}\left|\frac{\sum_{i=1}^{n}\sum_{j=1}^{n_y}\left(G^*_{\mathrm{NN}}(\mathbf{u}_i)(\mathbf{y}_{i,j})-G(u_i)(\mathbf{y}_{i,j})\right)\xi_{i,j}}{\sqrt{nn_y}\|G^*_{\mathrm{NN}}-G\|_n}\right|\right]+\sigma\theta. \qquad (69)
\end{aligned}
$$

Recall that $\{G_k^*\}_{k=1}^{\mathcal{N}(\theta,\mathcal{G}_{NN},\|\cdot\|_{\infty,\infty})}$ is a θ cover of $\mathcal{G}_{NN}$, and denote

$$
z_k=\frac{\sum_{i=1}^{n}\sum_{j=1}^{n_y}\left(G_k^*(\mathbf{u}_i)(\mathbf{y}_{i,j})-G(u_i)(\mathbf{y}_{i,j})\right)\xi_{i,j}}{\sqrt{nn_y}\|G^*_{\mathrm{NN}}-G\|_n}.
$$

We have

$$
\begin{aligned}
&\mathbb{E}_{\mathcal{S}}\left[\frac{\|\widehat{G}-G\|_n+\theta}{\sqrt{nn_y}}\left|\frac{\sum_{i=1}^{n}\sum_{j=1}^{n_y}\left(G^*_{\mathrm{NN}}(\mathbf{u}_i)(\mathbf{y}_{i,j})-G(u_i)(\mathbf{y}_{i,j})\right)\xi_{i,j}}{\sqrt{nn_y}\|G^*_{\mathrm{NN}}-G\|_n}\right|\right]\\
\leq&\mathbb{E}_{\mathcal{S}}\left[\frac{\|\widehat{G}-G\|_n+\theta}{\sqrt{nn_y}}\max_k|z_k|\right]\\
\leq&\sqrt{\mathbb{E}_{\mathcal{S}}\left[\left(\|\widehat{G}-G\|_n+\theta\right)^2\right]\mathbb{E}_{\mathcal{S}}\left[\frac{1}{nn_y}\max_k|z_k|^2\right]}\\
\leq&\sqrt{2\mathbb{E}_{\mathcal{S}}\left[\|\widehat{G}-G\|_n^2+\theta^2\right]}\sqrt{\frac{1}{nn_y}\mathbb{E}_{\mathcal{S}}\left[\max_k|z_k|^2\right]}\\
\leq&\sqrt{2}\left(\sqrt{\mathbb{E}_{\mathcal{S}}\left[\|\widehat{G}-G\|_n^2\right]}+\theta\right)\sqrt{\frac{1}{nn_y}\mathbb{E}_{\mathcal{S}}\left[\max_k|z_k|^2\right]}, \qquad (70)
\end{aligned}
$$

where Cauchy-Schwarz inequality is used in the second inequality, and the last inequality uses the relation $\sqrt{a+b^2}\leq\sqrt{a}+b$ for $a,b\geq 0$.

We next derive an upper bound of $\mathbb{E}_{\mathcal{S}}\left[\max_k|z_k|^2\right]$. Since each $\xi_{i,j}$ is a sub-Gaussian variable with variance proxy σ, for given $\left\{u_i,\{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right\}_{i=1}^{n}$, each z_k is a sub-Gaussian

variable with variance proxy σ^2. Let t be a positive number depending on σ and will be made clear later, we deduce,

$$\begin{aligned}
&\mathbb{E}_{\mathcal{S}}\left[\max_k |z_k|^2 \middle| \left\{u_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right\}_{i=1}^{n}\right] \\
=&\frac{1}{t}\log\exp\left(\mathbb{E}_{\mathcal{S}}\left[t\max_k |z_k|^2 \middle| \left\{u_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right\}_{i=1}^{n}\right]\right) \\
\leq&\frac{1}{t}\log\mathbb{E}_{\mathcal{S}}\left[\exp\left(t\max_k |z_k|^2 \middle| \left\{u_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right\}_{i=1}^{n}\right)\right] \\
\leq&\frac{1}{t}\log\mathbb{E}_{\mathcal{S}}\left[\sum_k \exp\left(t|z_k|^2 \middle| \left\{u_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right\}_{i=1}^{n}\right)\right] \\
=&\frac{1}{t}\log\left(\mathcal{N}(\theta, \mathcal{G}_{NN}, \|\cdot\|_{\infty,\infty})\mathbb{E}_{\mathcal{S}}\left[\exp\left(t|z_k|^2 \middle| \left\{u_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right\}_{i=1}^{n}\right)\right]\right) \\
\leq&\frac{1}{t}\log\mathcal{N}(\theta, \mathcal{G}_{NN}, \|\cdot\|_{\infty,\infty}) + \frac{1}{t}\log\mathbb{E}_{\mathcal{S}}\left[\exp\left(t|z_1|^2 \middle| \left\{u_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right\}_{i=1}^{n}\right)\right],
\end{aligned}$$

where we use Jensen's inequality in the first inequality. Due to the i.i.d. assumption of $\left\{u_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right\}_{i=1}^{n}$, we have

$$\mathbb{E}_{\mathcal{S}}\left[\exp\left(t|z_1|^2 \middle| \left\{u_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right\}_{i=1}^{n}\right)\right] = 1 + \sum_{\ell=1}^{\infty}\frac{t^\ell \mathbb{E}_{\mathcal{S}}\left[z_1^{2\ell} \middle| \left\{u_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right\}_{i=1}^{n}\right]}{\ell!}.$$

Since z_1 is sub-Gaussian with variance proxy σ^2, it follows that

$$\begin{aligned}
&1 + \sum_{\ell=1}^{\infty}\frac{t^\ell \mathbb{E}_{\mathcal{S}}\left[z_1^{2\ell} \middle| \left\{u_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right\}_{i=1}^{n}\right]}{\ell!} \\
=&1 + \sum_{\ell=1}^{\infty}\frac{t^\ell}{\ell!}\int_0^{\infty}\mathbb{P}\left(|z_1| \geq \tau^{\frac{1}{2\ell}} \middle| \left\{u_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right\}_{i=1}^{n}\right)d\tau \\
\leq&1 + 2\sum_{\ell=1}^{\infty}\frac{t^\ell}{\ell!}\int_0^{\infty}\exp\left(-\frac{\tau^{1/\ell}}{2\sigma^2}\right)d\tau \\
=&1 + \sum_{\ell=1}^{\infty}\frac{2\ell(2t\sigma^2)^\ell}{\ell!}\Gamma_G(\ell) \\
=&1 + 2\sum_{\ell=1}^{\infty}(2t\sigma^2)^\ell,
\end{aligned}$$

where Γ_G denotes the Gamma function. Setting $t = (4\sigma^2)^{-1}$, we have

$$\begin{aligned}
\mathbb{E}_{\mathcal{S}}\left[\max_k |z_k|^2 \middle| \left\{u_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right\}_{i=1}^{n}\right] \leq&4\sigma^2\log\mathcal{N}(\theta, \mathcal{G}_{\text{NN}}, \|\cdot\|_{\infty,\infty}) + 4\sigma^2\log 3 \\
\leq&4\sigma^2\log\mathcal{N}(\theta, \mathcal{G}_{\text{NN}}, \|\cdot\|_{\infty,\infty}) + 6\sigma^2. \qquad (71)
\end{aligned}$$

Substituting (71) and (70) into (69) gives rise to

$$\begin{aligned}&\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}\widehat{G}(\mathbf{u}_i)(\mathbf{y}_{i,j})\xi_{i,j}\right]\\ \leq&2\sigma\left(\sqrt{\mathbb{E}_{\mathcal{S}}\left[\|\widehat{G}-G\|_n^2\right]}+\theta\right)\sqrt{\frac{4\log\mathcal{N}(\theta,\mathcal{G}_{\mathrm{NN}},\|\cdot\|_{\infty,\infty})+6}{nn_y}}+\sigma\theta.\end{aligned}\tag{72}$$

■

B.7 Proof of Lemma 16

Proof [Proof of Lemma 16]

Denote $\widehat{g}(\mathbf{u})(\mathbf{y})=\left(\widehat{G}(\mathbf{u})(\mathbf{y})-G(u)(\mathbf{y})\right)^2$. Due to the clipping by β_V, $\|\widehat{g}\|_{\infty,\infty}\leq 4\beta_V^2$. Then

$$\begin{aligned}\mathrm{T}_2=&\mathbb{E}_{\mathcal{S}}\left\{\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}\widehat{g}(\mathbf{u})(\mathbf{y}_j)\right]-\frac{2}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}\widehat{g}(\mathbf{u}_i)(\mathbf{y}_{i,j})\right\}\\ =&2\mathbb{E}_{\mathcal{S}}\left\{\frac{1}{2}\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}\widehat{g}(\mathbf{u})(\mathbf{y}_j)\right]-\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}\widehat{g}(\mathbf{u}_i)(\mathbf{y}_{i,j})\right\}\\ =&2\mathbb{E}_{\mathcal{S}}\Bigg\{\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}\widehat{g}(\mathbf{u})(\mathbf{y}_j)\right]-\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}\widehat{g}(\mathbf{u}_i)(\mathbf{y}_{i,j})\\ &-\frac{1}{2}\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}\widehat{g}(\mathbf{u})(\mathbf{y}_j)\right]\Bigg\}.\end{aligned}\tag{73}$$

A lower bound of $\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}\widehat{g}(\mathbf{u})(\mathbf{y}_j)\right]$ is given as

$$\begin{aligned}\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}\widehat{g}(\mathbf{u})(\mathbf{y}_j)\right]=&\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}\frac{4\beta_V^2}{4\beta_V^2}\widehat{g}(\mathbf{u})(\mathbf{y}_j)\right]\\ \geq&\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}\frac{1}{4\beta_V^2}\widehat{g}^2(\mathbf{u})(\mathbf{y}_j)\right].\end{aligned}\tag{74}$$

Substituting (74) into (73) implies

$$\mathrm{T}_2\leq 2\mathbb{E}_{\mathcal{S}}\Bigg\{\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}\widehat{g}(\mathbf{u})(\mathbf{y}_j)\right]-\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}\widehat{g}(\mathbf{u}_i)(\mathbf{y}_{i,j})$$

$$- \frac{1}{8\beta_V^2}\mathbb{E}_{u\sim\rho_u}\mathbb{E}_{\{\mathbf{y}_j\}_{j=1}^{n_y}\sim\rho_y}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}\widehat{g}^2(\mathbf{u})(\mathbf{y}_j)\right]\Bigg\}. \tag{75}$$

Denote $\mathcal{S}' = \left\{u_i', \{\mathbf{y}_{i,j}'\}_{j=1}^{n_y}\right\}_{i=1}^{n}$ as an independent copy of $\mathcal{S}$. Define the set

$$\mathcal{R} = \{g(\mathbf{u})(\mathbf{y}) = (G_{\mathrm{NN}}(\mathbf{u})(\mathbf{y}) - G(u)(\mathbf{y}))^2 \text{ for } G_{\mathrm{NN}} \in \mathcal{G}_{\mathrm{NN}}, u \in U, \mathbf{y} \in \Omega_V\}. \tag{76}$$

We have

$$\begin{aligned}
\mathrm{T}_2 \leq & 2\mathbb{E}_{\mathcal{S}}\Bigg\{\sup_{g\in\mathcal{R}}\Bigg(\mathbb{E}_{\mathcal{S}'}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}g(\mathbf{u}_i')(\mathbf{y}_{i,j}')\right] - \frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}g(\mathbf{u}_i)(\mathbf{y}_{i,j}) \\
& - \frac{1}{8\beta_V^2}\mathbb{E}_{\mathcal{S}'}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}g^2(\mathbf{u}_i')(\mathbf{y}_{i,j}')\right]\Bigg)\Bigg\} \\
= & 2\mathbb{E}_{\mathcal{S}}\Bigg\{\sup_{g\in\mathcal{R}}\Bigg(\mathbb{E}_{\mathcal{S}'}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}\left(g(\mathbf{u}_i')(\mathbf{y}_{i,j}') - g(\mathbf{u}_i)(\mathbf{y}_{i,j})\right)\right] \\
& - \frac{1}{16\beta_V^2}\mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_y}\sum_{j=1}^{n_y}\left(g^2(\mathbf{u}_i')(\mathbf{y}_{i,j}') + g^2(\mathbf{u}_i)(\mathbf{y}_{i,j})\right)\right]\Bigg)\Bigg\} \\
\leq & 2\mathbb{E}_{\mathcal{S},\mathcal{S}'}\Bigg\{\sup_{g\in\mathcal{R}}\Bigg(\frac{1}{n}\sum_{i=1}^{n}\Bigg(\frac{1}{n_y}\sum_{j=1}^{n_y}(g(\mathbf{u}_i')(\mathbf{y}_{i,j}') - g(\mathbf{u}_i)(\mathbf{y}_{i,j})) \\
& - \frac{1}{16\beta_V^2}\mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}\left(g^2(\mathbf{u}_i')(\mathbf{y}_{i,j}') + g^2(\mathbf{u}_i)(\mathbf{y}_{i,j})\right)\right]\Bigg)\Bigg)\Bigg\}.
\end{aligned} \tag{77}$$

By Lemma 20, let $\mathcal{R}^* = \{g_k^*\}_{k=1}^{\mathcal{N}(\theta,\mathcal{R},\|\cdot\|_{\infty,\infty})}$ be a θ-cover of $\mathcal{R}$. Then for any $g \in \mathcal{R}$, there exists $g^* \in \mathcal{R}^*$ satisfying $\|g - g^*\|_{\infty,\infty} \leq \theta$. We will derive an upper bound of (77) using g^*.

For the first term in (77), we have

$$\begin{aligned}
& g(\mathbf{u}_i')(\mathbf{y}_{i,j}') - g(\mathbf{u}_i)(\mathbf{y}_{i,j}) \\
= & \left(g(\mathbf{u}_i')(\mathbf{y}_{i,j}') - g^*(\mathbf{u}_i')(\mathbf{y}_{i,j}')\right) + \left(g^*(\mathbf{u}_i')(\mathbf{y}_{i,j}') - g^*(\mathbf{u}_i)(\mathbf{y}_{i,j})\right) + \left(g^*(\mathbf{u}_i)(\mathbf{y}_{i,j}) - g(\mathbf{u}_i)(\mathbf{y}_{i,j})\right) \\
\leq & (g^*(\mathbf{u}_i')(\mathbf{y}_{i,j}') - g^*(\mathbf{u}_i)(\mathbf{y}_{i,j})) + 2\theta.
\end{aligned} \tag{78}$$

For the second term in (77), we have

$$\begin{aligned}
& g^2(\mathbf{u}_i')(\mathbf{y}_{i,j}') + g^2(\mathbf{u}_i)(\mathbf{y}_{i,j}) \\
= & \left(g^2(\mathbf{u}_i')(\mathbf{y}_{i,j}') - (g^*)^2(\mathbf{u}_i')(\mathbf{y}_{i,j}')\right) + \left(g^2(\mathbf{u}_i)(\mathbf{y}_{i,j}) - (g^*)^2(\mathbf{u}_i)(\mathbf{y}_{i,j})\right) \\
& + \left((g^*)^2(\mathbf{u}_i)(\mathbf{y}_{i,j}) + (g^*)^2(\mathbf{u}_i')(\mathbf{y}_{i,j}')\right) \\
\geq & (g^*)^2(\mathbf{u}_i)(\mathbf{y}_{i,j}) + (g^*)^2(\mathbf{u}_i')(\mathbf{y}_{i,j}') - |g(\mathbf{u}_i')(\mathbf{y}_{i,j}') - g^*(\mathbf{u}_i')(\mathbf{y}_{i,j}')||g(\mathbf{u}_i')(\mathbf{y}_{i,j}') + g^*(\mathbf{u}_i')(\mathbf{y}_{i,j}')| \\
& - |g(\mathbf{u}_i)(\mathbf{y}_{i,j}) - g^*(\mathbf{u}_i)(\mathbf{y}_{i,j})||g(\mathbf{u}_i)(\mathbf{y}_{i,j}) + g^*(\mathbf{u}_i)(\mathbf{y}_{i,j})|
\end{aligned}$$

$$\geq (g^*)^2(\mathbf{u}_i)(\mathbf{y}_{i,j}) + (g^*)^2(\mathbf{u}'_i)(\mathbf{y}'_{i,j}) - 16\beta_V^2\theta. \tag{79}$$

Substituting (78) and (79) into (77) gives rise to

$$\begin{aligned}
\mathrm{T}_2 \leq & 2\mathbb{E}_{\mathcal{S},\mathcal{S}'}\Bigg\{ \sup_{g\in\mathcal{R}} \Bigg(\frac{1}{n}\sum_{i=1}^{n} \Bigg(\frac{1}{n_y}\sum_{j=1}^{n_y} (g(\mathbf{u}'_i)(\mathbf{y}'_{i,j}) - g(\mathbf{u}_i)(\mathbf{y}_{i,j})) \\
& - \frac{1}{16\beta_V^2}\mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[\frac{1}{n_y}\left((g^*)^2(\mathbf{u}_i)(\mathbf{y}_{i,j}) + (g^*)^2(\mathbf{u}'_i)(\mathbf{y}'_{i,j}) \right) \right] \Bigg) \Bigg) \Bigg] + 6\theta \\
= & 2\mathbb{E}_{\mathcal{S},\mathcal{S}'}\Bigg[\max_{k} \Bigg(\frac{1}{n}\sum_{i=1}^{n} \Bigg(\frac{1}{n_y}\sum_{j=1}^{n_y} (g_k^*(\mathbf{u}'_i)(\mathbf{y}'_{i,j}) - g_k^*(\mathbf{u}_i)(\mathbf{y}_{i,j})) \\
& - \frac{1}{16\beta_V^2}\mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}\left((g_k^*)^2(\mathbf{u}_i)(\mathbf{y}_{i,j}) + (g_k^*)^2(\mathbf{u}'_i)(\mathbf{y}'_{i,j}) \right) \right] \Bigg) \Bigg) \Bigg] + 6\theta
\end{aligned} \tag{80}$$

Denote $r_k\left(\mathbf{u}'_i, \{\mathbf{y}'_{i,j}\}_{j=1}^{n_y}, \mathbf{u}_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right) = \frac{1}{n_y}\sum_{j=1}^{n_y}(g_k^*(\mathbf{u}'_i)(\mathbf{y}'_{i,j}) - g_k^*(\mathbf{u}_i)(\mathbf{y}_{i,j}))$. We have

$$\mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[r_k\left(\mathbf{u}'_i, \{\mathbf{y}'_{i,j}\}_{j=1}^{n_y}, \mathbf{u}_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)\right] = 0,$$

and thus

$$\begin{aligned}
& \mathrm{Var}\left(r_k\left(\mathbf{u}'_i, \{\mathbf{y}'_{i,j}\}_{j=1}^{n_y}, \mathbf{u}_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)\right) \\
= & \mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[r_k^2\left(\mathbf{u}'_i, \{\mathbf{y}'_{i,j}\}_{j=1}^{n_y}, \mathbf{u}_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)\right] \\
= & \mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[\frac{1}{n_y^2}\left(\sum_{j=1}^{n_y}\left(g_k^*(\mathbf{u}'_i)(\mathbf{y}'_{i,j}) - g_k^*(\mathbf{u}_i)(\mathbf{y}_{i,j})\right)\right)^2\right] \\
= & \mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[\frac{1}{n_y^2}\sum_{j=1}^{n_y}\sum_{\ell=1}^{n_y}\left(g_k^*(\mathbf{u}'_i)(\mathbf{y}'_{i,j}) - g_k^*(\mathbf{u}_i)(\mathbf{y}_{i,j})\right)\left(g_k^*(\mathbf{u}'_i)(\mathbf{y}'_{i,\ell}) - g_k^*(\mathbf{u}_i)(\mathbf{y}_{i,\ell})\right)\right] \\
= & \mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[\frac{1}{n_y^2}\sum_{j=1}^{n_y}\left(g_k^*(\mathbf{u}'_i)(\mathbf{y}'_{i,j}) - g_k^*(\mathbf{u}_i)(\mathbf{y}_{i,j})\right)^2\right] \\
& + \mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[\frac{1}{n_y^2}\sum_{j=1}^{n_y}\sum_{\ell\neq j}\left(g_k^*(\mathbf{u}'_i)(\mathbf{y}'_{i,j}) - g_k^*(\mathbf{u}_i)(\mathbf{y}_{i,j})\right)\left(g_k^*(\mathbf{u}'_i)(\mathbf{y}'_{i,\ell}) - g_k^*(\mathbf{u}_i)(\mathbf{y}_{i,\ell})\right)\right]
\end{aligned} \tag{81}$$

Note that for $\ell \neq j$, we have

$$\begin{aligned}
& \mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[\left(g_k^*(\mathbf{u}'_i)(\mathbf{y}'_{i,j}) - g_k^*(\mathbf{u}_i)(\mathbf{y}_{i,j})\right)\left(g_k^*(\mathbf{u}'_i)(\mathbf{y}'_{i,\ell}) - g_k^*(\mathbf{u}_i)(\mathbf{y}_{i,\ell})\right)\right] \\
= & \mathbb{E}_{u_i\sim\rho_u, u'_i\sim\rho_u}\left[\mathbb{E}_{y_{i,j}, y_{i,\ell}, y'_{i,j}, y'_{y,\ell}\sim\rho_y}\left[\left(g_k^*(\mathbf{u}'_i)(\mathbf{y}'_{i,j}) - g_k^*(\mathbf{u}_i)(\mathbf{y}_{i,j})\right)\left(g_k^*(\mathbf{u}'_i)(\mathbf{y}'_{i,\ell}) - g_k^*(\mathbf{u}_i)(\mathbf{y}_{i,\ell})\right)\Big| u_i, u'_i\right]\right] \\
= & \mathbb{E}_{u_i\sim\rho_u, u'_i\sim\rho_u}\left[\left(\mathbb{E}_{y_{i,j}, y'_{i,j}\sim\rho_y}\left[\left(g_k^*(\mathbf{u}'_i)(\mathbf{y}'_{i,j}) - g_k^*(\mathbf{u}_i)(\mathbf{y}_{i,j})\right)\right]\right)^2\right]
\end{aligned}$$

$$
\begin{aligned}
&\leq \mathbb{E}_{u_i\sim\rho_u, u_i'\sim\rho_u}\left[\mathbb{E}_{y_{i,j}, y_{i,j}'\sim\rho_y}\left[\left(g_k^*(\mathbf{u}_i')(\mathbf{y}_{i,j}') - g_k^*(\mathbf{u}_i)(\mathbf{y}_{i,j})\right)^2\right]\right] \\
&= \mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[\left(g_k^*(\mathbf{u}_i')(\mathbf{y}_{i,j}') - g_k^*(\mathbf{u}_i)(\mathbf{y}_{i,j})\right)^2\right], \qquad (82)
\end{aligned}
$$

where the inequality uses Jensen's inequality. Substituting (82) into (81) gives rise to

$$
\begin{aligned}
&\operatorname{Var}\left(r_k\left(\mathbf{u}_i', \{\mathbf{y}_{i,j}'\}_{j=1}^{n_y}, \mathbf{u}_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)\right) \\
\leq &\mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[\frac{1}{n_y^2}\sum_{j=1}^{n_y}\left(g_k^*(\mathbf{u}_i')(\mathbf{y}_{i,j}') - g_k^*(\mathbf{u}_i)(\mathbf{y}_{i,j})\right)^2\right] + \mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[\frac{1}{n_y^2}\sum_{j=1}^{n_y}\sum_{\ell\neq j}\left(g_k^*(\mathbf{u}_i')(\mathbf{y}_{i,j}') - g_k^*(\mathbf{u}_i)(\mathbf{y}_{i,j})\right)^2\right] \\
= &\mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[\frac{1}{n_y^2}\sum_{j=1}^{n_y}\sum_{\ell=1}^{n_y}\left(g_k^*(\mathbf{u}_i')(\mathbf{y}_{i,j}') - g_k^*(\mathbf{u}_i)(\mathbf{y}_{i,j})\right)^2\right] \\
= &\mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}\left(g_k^*(\mathbf{u}_i')(\mathbf{y}_{i,j}') - g_k^*(\mathbf{u}_i)(\mathbf{y}_{i,j})\right)^2\right] \\
\leq &2\mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[\frac{1}{n_y}\sum_{j=1}^{n_y}\left((g_k^*)^2(\mathbf{u}_i')(\mathbf{y}_{i,j}') + (g_k^*)^2(\mathbf{u}_i)(\mathbf{y}_{i,j})\right)^2\right]. \qquad (83)
\end{aligned}
$$

Next we define and estimate $\widetilde{T}_2$,

$$
\begin{aligned}
\widetilde{\mathrm{T}}_2 \leq &2\mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[\max_k\left(\frac{1}{n}\sum_{i=1}^{n}\left(r_k\left(\mathbf{u}_i', \{\mathbf{y}_{i,j}'\}_{j=1}^{n_y}, \mathbf{u}_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)\right.\right.\right. \\
&\left.\left.\left. - \frac{1}{16\beta_V^2}\operatorname{Var}\left[r_k\left(\mathbf{u}_i', \{\mathbf{y}_{i,j}'\}_{j=1}^{n_y}, \mathbf{u}_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)\right]\right)\right)\right].
\end{aligned}
$$

According to (80) and (83),

$$
\mathrm{T}_2 \leq \widetilde{\mathrm{T}}_2 + 6\theta.
$$

We estimate $\widetilde{T}_2$ using the moment generating function of r_k. Note that $\|r_k\|_{\infty,\infty} \leq 4\beta_V^2$. For $0 < t < 3/4\beta_V^2$, we have

$$
\begin{aligned}
&\mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[\exp(tr_k\left(\mathbf{u}_i', \{\mathbf{y}_{i,j}'\}_{j=1}^{n_y}, \mathbf{u}_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right))\right] \\
= &\mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[1 + tr_k\left(\mathbf{u}_i', \{\mathbf{y}_{i,j}'\}_{j=1}^{n_y}, \mathbf{u}_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right) + \sum_{\ell=2}^{\infty}\frac{t^\ell r_k^\ell(\mathbf{u}_i', \{\mathbf{y}_{i,j}'\}_{j=1}^{n_y}, \mathbf{u}_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y})}{\ell!}\right] \\
\leq &\mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[1 + tr_k\left(\mathbf{u}_i', \{\mathbf{y}_{i,j}'\}_{j=1}^{n_y}, \mathbf{u}_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right) + \sum_{\ell=2}^{\infty}\frac{(4\beta_V^2)^{\ell-2}t^\ell r_k^2\left(\mathbf{u}_i', \{\mathbf{y}_{i,j}'\}_{j=1}^{n_y}, \mathbf{u}_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)}{2\times 3^{\ell-2}}\right] \\
= &\mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[1 + tr_k\left(\mathbf{u}_i', \{\mathbf{y}_{i,j}'\}_{j=1}^{n_y}, \mathbf{u}_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right) + \frac{\ell^2 r_k^2\left(\mathbf{u}_i', \{\mathbf{y}_{i,j}'\}_{j=1}^{n_y}, \mathbf{u}_i, \{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)}{2}\sum_{\ell=2}^{\infty}\frac{(4\beta_V^2)^{\ell-2}t^{\ell-2}}{3^{\ell-2}}\right]
\end{aligned}
$$

$$
\begin{aligned}
=&\mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[1+tr_k\left(\mathbf{u}_i',\{\mathbf{y}_{i,j}'\}_{j=1}^{n_y},\mathbf{u}_i,\{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)+\frac{\ell^2 r_k^2\left(\mathbf{u}_i',\{\mathbf{y}_{i,j}'\}_{j=1}^{n_y},\mathbf{u}_i,\{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)}{2}\frac{1}{1-4\beta_V^2 t/3}\right]\\
=&1+t^2\operatorname{Var}[r_k\left(\mathbf{u}_i',\{\mathbf{y}_{i,j}'\}_{j=1}^{n_y},\mathbf{u}_i,\{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)]\frac{1}{2-8\beta_V^2 t/3}\\
\le&\exp\left(\operatorname{Var}[r_k\left(\mathbf{u}_i',\{\mathbf{y}_{i,j}'\}_{j=1}^{n_y},\mathbf{u}_i,\{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)]\frac{3t^2}{6-8\beta_V^2 t}\right),
\end{aligned}
\tag{84}
$$

where the last inequality comes from the relation $1+x\le\exp(x)$ for $x\ge 0$.

For $0<t/n<3/4\beta_V^2$, we have

$$
\begin{aligned}
&\exp\left(\frac{t\widetilde{\mathrm{T}}_2}{2}\right)\\
=&\exp\left(t\mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[\max_k\left(\frac{1}{n}\sum_{i=1}^n\left(r_k\left(\mathbf{u}_i',\{\mathbf{y}_{i,j}'\}_{j=1}^{n_y},\mathbf{u}_i,\{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)\right.\right.\right.\right.\\
&\qquad\left.\left.\left.\left.-\frac{1}{16\beta_V^2}\operatorname{Var}\left[r_k\left(\mathbf{u}_i',\{\mathbf{y}_{i,j}'\}_{j=1}^{n_y},\mathbf{u}_i,\{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)\right]\right)\right)\right]\right)\\
\le&\mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[\exp\left(t\max_k\left(\frac{1}{n}\sum_{i=1}^n\left(r_k\left(\mathbf{u}_i',\{\mathbf{y}_{i,j}'\}_{j=1}^{n_y},\mathbf{u}_i,\{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)\right.\right.\right.\right.\\
&\qquad\left.\left.\left.\left.-\frac{1}{16\beta_V^2}\operatorname{Var}\left[r_k\left(\mathbf{u}_i',\{\mathbf{y}_{i,j}'\}_{j=1}^{n_y},\mathbf{u}_i,\{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)\right]\right)\right)\right)\right]\\
\le&\mathbb{E}_{\mathcal{S},\mathcal{S}'}\left[\sum_k\exp\left(\sum_{i=1}^n\left(\frac{t}{n}r_k\left(\mathbf{u}_i',\{\mathbf{y}_{i,j}'\}_{j=1}^{n_y},\mathbf{u}_i,\{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)\right.\right.\right.\\
&\qquad\left.\left.\left.-\frac{1}{n}\frac{t}{16\beta_V^2}\operatorname{Var}\left[r_k\left(\mathbf{u}_i',\{\mathbf{y}_{i,j}'\}_{j=1}^{n_y},\mathbf{u}_i,\{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)\right]\right)\right)\right]\\
\le&\sum_k\mathbb{E}_{\mathcal{S},\mathcal{S}'}\prod_{i=1}^n\exp\left(\frac{t}{n}r_k\left(\mathbf{u}_i',\{\mathbf{y}_{i,j}'\}_{j=1}^{n_y},\mathbf{u}_i,\{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)\right.\\
&\qquad\left.-\frac{1}{n}\frac{t}{16\beta_V^2}\operatorname{Var}\left[r_k\left(\mathbf{u}_i',\{\mathbf{y}_{i,j}'\}_{j=1}^{n_y},\mathbf{u}_i,\{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)\right]\right)\\
\le&\sum_k\exp\left(\sum_{i=1}^n\left(\operatorname{Var}[r_k\left(\mathbf{u}_i',\{\mathbf{y}_{i,j}'\}_{j=1}^{n_y},\mathbf{u}_i,\{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)]\frac{3(t/n)^2}{6-8\beta_V^2 t/n}\right.\right.\\
&\qquad\left.\left.-\frac{1}{n}\frac{t}{16\beta_V^2}\operatorname{Var}\left[r_k\left(\mathbf{u}_i',\{\mathbf{y}_{i,j}'\}_{j=1}^{n_y},\mathbf{u}_i,\{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)\right]\right)\right)\\
=&\sum_k\exp\left(\sum_{i=1}^n\frac{t}{n}\operatorname{Var}[r_k\left(\mathbf{u}_i',\{\mathbf{y}_{i,j}'\}_{j=1}^{n_y},\mathbf{u}_i,\{\mathbf{y}_{i,j}\}_{j=1}^{n_y}\right)]\left(\frac{3t/n}{6-8\beta_V^2 t/n}-\frac{1}{16\beta_V^2}\right)\right)
\end{aligned}
\tag{85}
$$

where the first inequality follows from Jensen's inequality and the third inequality uses (84) by replacing t by t/n. Setting

$$\frac{3t/n}{6-8\beta_V^2 t/n}-\frac{1}{16\beta_V^2}=0,$$

we have $t=\frac{3n}{28\beta_V^2}$ and $\frac{t}{n}<\frac{3}{4\beta_V^2}$. Substituting the choice of t into (85) gives rise to

$$\frac{t\widetilde{\mathrm{T}}_2}{2}\le\log\sum_k\exp(0).$$

Thus

$$\widetilde{\mathrm{T}}_2\le\frac{2}{t}\log\mathcal{N}(\theta,\mathcal{R},\|\cdot\|_{\infty,\infty})=\frac{56\beta_V^2}{3nn_y}\log\mathcal{N}(\theta,\mathcal{R},\|\cdot\|_{\infty,\infty})$$

and

$$\mathrm{T}_2\le\frac{56\beta_V^2}{3n}\log\mathcal{N}(\theta,\mathcal{R},\|\cdot\|_{\infty,\infty})+6\theta\le\frac{19\beta_V^2}{n}\log\mathcal{N}(\theta,\mathcal{R},\|\cdot\|_{\infty,\infty})+6\theta. \tag{86}$$

The following lemma (see a proof in Section B.9) gives a relation between the covering number of $\mathcal{R}$ and $\mathcal{G}_{\mathrm{NN}}$:

Lemma 20 *Let $\mathcal{G}^*$ be a θ cover with the covering number $\mathcal{N}(\theta,\mathcal{G}_{\mathrm{NN}},\|\cdot\|_{\infty,\infty})$. There exists a finite θ cover $\mathcal{R}^*$ of $\mathcal{R}$, and the covering number is bounded by,*

$$\mathcal{N}(\theta,\mathcal{R},\|\cdot\|_{\infty,\infty})\le\mathcal{N}\left(\frac{\theta}{4\beta_V},\mathcal{G}_{\mathrm{NN}},\|\cdot\|_{\infty,\infty}\right).$$

By Lemma 20, we have

$$\mathrm{T}_2\le\frac{19\beta_V^2}{n}\log\mathcal{N}\left(\frac{\theta}{4\beta_V},\mathcal{G}_{\mathrm{NN}},\|\cdot\|_{\infty,\infty}\right)+6\theta. \tag{87}$$

■

B.8 Proof of Lemma 17

Proof [Proof of Lemma 17] For $h>0$ and each k, let $\widetilde{a}_k'\in\mathcal{F}_2$ be some network so that each nonzero parameter of $\widetilde{a}_k'$ is at most different from the corresponding one in $\widetilde{a}_k$ by h. Similarly, let $\widetilde{q}_k'\in\mathcal{F}_1$ be some network so that each nonzero parameter of $\widetilde{q}_k'$ is at most different from the corresponding one in $\widetilde{q}_k$ by h.

According to Chen et al. (2022, Proof of Lemma 5.3), we have

$$\|\widetilde{a}_k'-\widetilde{a}_k\|_\infty\le hL_2(p_2\beta_U+2)(\kappa_2 p_2)^{L_2-1},\quad \|\widetilde{q}_k'-\widetilde{q}_k\|_\infty\le hL_1(p_1\gamma_2+2)(\kappa_1 p_1)^{L_1-1}.$$

We deduce

$$
\begin{aligned}
&\left\| \mathrm{CL}_c\left(\sum_{k=1}^{N} \widetilde{a}'_k(\mathbf{u})\widetilde{q}'_k(\mathbf{y})\right) - \mathrm{CL}_c\left(\sum_{k=1}^{N} \widetilde{a}_k(\mathbf{u})\widetilde{q}_k(\mathbf{y})\right)\right\|_{\infty,\infty} \\
\leq &\left\| \sum_{k=1}^{N} \widetilde{a}'_k(\mathbf{u})\widetilde{q}'_k(\mathbf{y}) - \sum_{k=1}^{N} \widetilde{a}_k(\mathbf{u})\widetilde{q}_k(\mathbf{y})\right\|_{\infty,\infty} \\
\leq &\sum_{k=1}^{N} \|\widetilde{a}'_k(\mathbf{u})\widetilde{q}'_k(\mathbf{y}) - \widetilde{a}_k(\mathbf{u})\widetilde{q}_k(\mathbf{y})\|_{\infty,\infty} \\
\leq &\sum_{k=1}^{N} \left(\|\widetilde{a}'_k(\mathbf{u})\widetilde{q}'_k(\mathbf{y}) - \widetilde{a}'_k(\mathbf{u})\widetilde{q}_k(\mathbf{y})\|_{\infty,\infty} + \|\widetilde{a}'_k(\mathbf{u})\widetilde{q}_k(\mathbf{y}) - \widetilde{a}_k(\mathbf{u})\widetilde{q}_k(\mathbf{y})\|_{\infty,\infty}\right) \\
\leq &\sum_{k=1}^{N} \left(\|\widetilde{a}'_k\|_\infty \|\widetilde{q}'_k(\mathbf{y}) - \widetilde{q}_k(\mathbf{y})\|_\infty + \|\widetilde{q}_k\|_\infty \|\widetilde{a}'_k(\mathbf{u}) - \widetilde{a}_k(\mathbf{u})\|_\infty\right) \\
\leq &\sum_{k=1}^{N} \left(R_1 h L_1(p_1\gamma_2+2)(\kappa_1 p_1)^{L_1-1} + R_2 h L_2(p_2\beta_U+2)(\kappa_2 p_2)^{L_2-1}\right) \\
= &hN\left(R_1 L_1(p_1\gamma_2+2)(\kappa_1 p_1)^{L_1-1} + R_2 L_2(p_2\beta_U+2)(\kappa_2 p_2)^{L_2-1}\right).
\end{aligned}
$$

Set h so that $hN\left(R_1 L_1(p_1\gamma_2+2)(\kappa_1 p_1)^{L_1-1} + R_2 L_2(p_2\beta_U+2)(\kappa_2 p_2)^{L_2-1}\right) = \theta$ gives

$$
h = \frac{\theta}{H} \text{ with } H = N\left(R_1 L_1(p_1\gamma_2+2)(\kappa_1 p_1)^{L_1-1} + R_2 L_2(p_2\beta_U+2)(\kappa_2 p_2)^{L_2-1}\right).
$$

We uniformly discretize the parameters of $\mathcal{F}_1$ and $\mathcal{F}_2$ by $2\kappa_1/h$ and $2\kappa_2/h$. The collection of all networks corresponding to those grid parameters forms a θ-cover of $\mathcal{G}$. The covering number is bounded by

$$
\begin{aligned}
\mathcal{N}(\theta, \mathcal{G}_{NN}, \|\cdot\|_{\infty,\infty}) \leq &\binom{L_1 p_1^2}{K_1}\left(\frac{2\kappa_1}{h}\right)^{NK_1} \cdot \binom{L_2 p_2^2}{K_2}\left(\frac{2\kappa_2}{h}\right)^{NK_2} \\
\leq &(L_1 p_1^2)^{NK_1}\left(\frac{2\kappa_1}{h}\right)^{NK_1} \cdot (L_2 p_2^2)^{K_2}\left(\frac{2\kappa_2}{h}\right)^{K_2} \\
\leq &\left(\frac{2L_1 p_1^2 \kappa_1 H}{\theta}\right)^{NK_1}\left(\frac{2L_2 p_2^2 \kappa_2 H}{\theta}\right)^{NK_2}.
\end{aligned}
$$

■

B.9 Proof of Lemma 20

Proof [Proof of Lemma 20] For any $g, \bar{g} \in \mathcal{R}$, we have

$$
g(\mathbf{u})(\mathbf{y}) = (G_{\mathrm{NN}}(\mathbf{u})(\mathbf{y}) - G(u)(\mathbf{y}))^2, \quad \bar{g}(\mathbf{u})(\mathbf{y}) = (\bar{G}_{\mathrm{NN}}(\mathbf{u})(\mathbf{y}) - G(u)(\mathbf{y}))^2 \tag{88}
$$

for some $G_{\text{NN}}, \bar{G}_{\text{NN}} \in \mathcal{G}$. We have

$$\begin{aligned}
\|g - \bar{g}\|_{\infty,\infty} =& \sup_{u\in U}\sup_{\mathbf{y}\in\Omega_V} |(G_{\text{NN}}(\mathbf{u})(\mathbf{y}) - G(u)(\mathbf{y}))^2 - (\bar{G}_{\text{NN}}(\mathbf{u})(\mathbf{y}) - G(u)(\mathbf{y}))^2| \\
=& \sup_{u\in U}\sup_{\mathbf{y}\in\Omega_V} |\left(G_{\text{NN}}(\mathbf{u})(\mathbf{y}) - \bar{G}_{\text{NN}}(\mathbf{u})(\mathbf{y})\right)\left(G_{\text{NN}}(\mathbf{u})(\mathbf{y}) + \bar{G}_{\text{NN}}(\mathbf{u})(\mathbf{y}) - 2G(u)(\mathbf{y})\right)| \\
\leq& \sup_{u\in U}\sup_{\mathbf{y}\in\Omega_V} |G_{\text{NN}}(\mathbf{u})(\mathbf{y}) - \bar{G}_{\text{NN}}(\mathbf{u})(\mathbf{y})||G_{\text{NN}}(\mathbf{u})(\mathbf{y}) + \bar{G}_{\text{NN}}(\mathbf{u})(\mathbf{y}) - 2G(u)(\mathbf{y})| \\
\leq& 4\beta_V \|G_{\text{NN}} - \bar{G}_{\text{NN}}\|_{\infty,\infty}.
\end{aligned}$$

We thus have

$$\mathcal{N}(\theta, \mathcal{R}, \|\cdot\|_{\infty,\infty}) \leq \mathcal{N}\left(\frac{\theta}{4\beta_V}, \mathcal{G}_{NN}, \|\cdot\|_{\infty,\infty}\right).$$

■

References

Gang Bao, Dong Wang, and Boyi Zou. Wanco: Weak adversarial networks for constrained optimization problems. *arXiv preprint arXiv:2407.03647*, 2024.

Kaushik Bhattacharya, Bamdad Hosseini, Nikola B Kovachki, and Andrew M Stuart. Model reduction and neural networks for parametric pdes. *The SMAI journal of computational mathematics*, 7:121–157, 2021.

Cristian Bodnar, Wessel P Bruinsma, Ana Lucic, Megan Stanley, Johannes Brandstetter, Patrick Garvan, Maik Riechert, Jonathan Weyn, Haiyu Dong, Anna Vaughan, et al. Aurora: A foundation model of the atmosphere. *arXiv preprint arXiv:2405.13063*, 2024.

Minshuo Chen, Haoming Jiang, Wenjing Liao, and Tuo Zhao. Efficient approximation of deep relu networks for functions on low dimensional manifolds. *Advances in neural information processing systems*, 32, 2019.

Minshuo Chen, Haoming Jiang, Wenjing Liao, and Tuo Zhao. Nonparametric regression on low-dimensional manifolds using deep relu networks: Function approximation and statistical recovery. *Information and Inference: A Journal of the IMA*, 11(4):1203–1253, 2022.

Tianping Chen and Hong Chen. Approximations of continuous functionals by neural networks with application to dynamic systems. *IEEE Transactions on Neural networks*, 4 (6):910–918, 1993.

Tianping Chen and Hong Chen. Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems. *IEEE transactions on neural networks*, 6(4):911–917, 1995.

John Horton Conway and Neil James Alexander Sloane. *Sphere packings, lattices and groups*, volume 290. Springer Science & Business Media, 2013.

Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath. Generative adversarial networks: An overview. *IEEE signal processing magazine*, 35(1):53–65, 2018.

Biraj Dahal, Alexander Havrilla, Minshuo Chen, Tuo Zhao, and Wenjing Liao. On deep generative models for approximation and estimation of distributions on manifolds. *Advances in Neural Information Processing Systems*, 35:10615–10628, 2022.

Maarten V de Hoop, Daniel Zhengyu Huang, Elizabeth Qian, and Andrew M Stuart. The cost-accuracy trade-off in operator learning with neural networks. *arXiv preprint arXiv:2203.13181*, 2022.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. Ieee, 2013.

Jiequn Han, Arnulf Jentzen, and Weinan E. Solving high-dimensional partial differential equations using deep learning. *Proceedings of the National Academy of Sciences*, 115(34): 8505–8510, 2018.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. Deep learning scaling is predictable, empirically. *arXiv preprint arXiv:1712.00409*, 2017.

Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine*, 29(6):82–97, 2012.

Masaaki Imaizumi and Kenji Fukumizu. Deep neural networks learn non-smooth functions effectively. In *The 22nd international conference on artificial intelligence and statistics*, pages 869–878. PMLR, 2019.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Yuehaw Khoo, Jianfeng Lu, and Lexing Ying. Solving parametric pde problems with artificial neural networks. *European Journal of Applied Mathematics*, 32(3):421–435, 2021.

Katiana Kontolati, Somdatta Goswami, George Em Karniadakis, and Michael D Shields. Learning in latent spaces improves the predictive accuracy of deep neural operators. *arXiv preprint arXiv:2304.07599*, 2023.

Samuel Lanthaler. Operator learning with pca-net: upper and lower complexity bounds. *Journal of Machine Learning Research*, 24(318):1–67, 2023.

Samuel Lanthaler and Andrew M Stuart. The parametric complexity of operator learning. *arXiv preprint arXiv:2306.15924*, 2023.

Samuel Lanthaler, Siddhartha Mishra, and George E Karniadakis. Error estimates for deeponets: A deep learning framework in infinite dimensions. *Transactions of Mathematics and Its Applications*, 6(1):tnac001, 2022.

Sanghyun Lee and Yeonjong Shin. On the training and generalization of deep operator networks. *SIAM Journal on Scientific Computing*, 46(4):C273–C296, 2024.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Graph kernel network for partial differential equations. *arXiv preprint arXiv:2003.03485*, 2020.

Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, Anima Anandkumar, et al. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021.

Guang Lin, Christian Moya, and Zecheng Zhang. B-deeponet: An enhanced bayesian deeponet for solving noisy parametric pdes using accelerated replica exchange sgld. *Journal of Computational Physics*, 473:111713, 2023.

Hao Liu, Minshuo Chen, Tuo Zhao, and Wenjing Liao. Besov function approximation and binary classification on low-dimensional manifolds using convolutional residual networks. In *International Conference on Machine Learning*, pages 6770–6780. PMLR, 2021.

Hao Liu, Jiahui Cheng, and Wenjing Liao. Deep neural networks are adaptive to function regularity and data distribution in approximation and estimation. *arXiv preprint arXiv:2406.05320*, 2024a.

Hao Liu, Alex Havrilla, Rongjie Lai, and Wenjing Liao. Deep nonparametric estimation of intrinsic data structures by chart autoencoders: Generalization error and robustness. *Applied and Computational Harmonic Analysis*, 68:101602, 2024b.

Hao Liu, Haizhao Yang, Minshuo Chen, Tuo Zhao, and Wenjing Liao. Deep nonparametric estimation of operators between infinite dimensional spaces. *Journal of Machine Learning Research*, 25(24):1–67, 2024c.

Hao Liu, Biraj Dahal, Rongjie Lai, and Wenjing Liao. Generalization error guaranteed auto-encoder-based nonlinear model reduction for operator learning. *Applied and Computational Harmonic Analysis*, 74:101717, 2025.

Yuxuan Liu, Zecheng Zhang, and Hayden Schaeffer. Prose: Predicting operators and symbolic expressions using multimodal transformers. *arXiv preprint arXiv:2309.16816*, 2023.

Yuxuan Liu, Jingmin Sun, Xinjie He, Griffin Pinney, Zecheng Zhang, and Hayden Schaeffer. Prose-fd: A multimodal pde foundation model for learning multiple operators for forecasting fluid dynamics. *arXiv preprint arXiv:2409.09811*, 2024d.

Jianfeng Lu, Zuowei Shen, Haizhao Yang, and Shijun Zhang. Deep network approximation for smooth functions. *SIAM Journal on Mathematical Analysis*, 53(5):5465–5506, 2021a.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature machine intelligence*, 3(3):218–229, 2021b.

Hrushikesh Narhar Mhaskar and Nahmwoo Hahm. Neural networks for functional approximation and system identification. *Neural Computation*, 9(1):143–159, 1997.

Ryumei Nakada and Masaaki Imaizumi. Adaptive approximation and generalization of deep neural network with intrinsic dimensionality. *Journal of Machine Learning Research*, 21 (174):1–38, 2020.

Kenta Oono and Taiji Suzuki. Approximation and non-parametric estimation of resnet-type convolutional neural networks. In *International conference on machine learning*, pages 4922–4931. PMLR, 2019.

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

Phillip Pope, Chen Zhu, Ahmed Abdelkader, Micah Goldblum, and Tom Goldstein. The intrinsic dimension of images and its impact on learning. *arXiv preprint arXiv:2104.08894*, 2021.

Thomas F Reid and Stephen C King. Pendulum motion and differential equations. *Primus*, 19(2):205–217, 2009.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.

Anselm Johannes Schmidt-Hieber. Nonparametric regression using deep neural networks with relu activation function. *Annals of statistics*, 48(4):1875–1897, 2020.

Linhao Song, Ying Liu, Jun Fan, and Ding-Xuan Zhou. Approximation of smooth functionals using deep relu networks. *Neural Networks*, 166:424–436, 2023.

Shashank Subramanian, Peter Harrington, Kurt Keutzer, Wahid Bhimji, Dmitriy Morozov, Michael W Mahoney, and Amir Gholami. Towards foundation models for scientific machine learning: Characterizing scaling and transfer behavior. *Advances in Neural Information Processing Systems*, 36, 2024.

Jingmin Sun, Zecheng Zhang, and Hayden Schaeffer. Lemon: Learning to learn multi-operator networks. *arXiv preprint arXiv:2408.16168*, 2024.

Joshua B Tenenbaum, Vin de Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

Loring W Tu. Manifolds. In *An Introduction to Manifolds*, pages 47–83. Springer, 2011.

Yunfei Yang, Han Feng, and Ding-Xuan Zhou. On the rates of convergence for learning with convolutional neural networks. *arXiv preprint arXiv:2403.16459*, 2024.

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

Zecheng Zhang, Wing Tat Leung, and Hayden Schaeffer. A discretization-invariant extension and analysis of some deep operator networks. *arXiv preprint arXiv:2307.09738*, 2023a.

Zecheng Zhang, Leung Wing Tat, and Hayden Schaeffer. Belnet: basis enhanced learning, a mesh-free neural operator. *Proceedings of the Royal Society A*, 479(2276):20230043, 2023b.