Deep Network Approximation Characterized by Number of Neurons*

Zuowei Shen†   Haizhao Yang‡   Shijun Zhang§

July 31, 2020

Abstract

This paper quantitatively characterizes the approximation power of deep feed-forward neural networks (FNNs) in terms of the number of neurons. It is shown by construction that ReLU FNNs with width $C_1 \max\{d\lfloor N^{1/d}\rfloor, N+1\}$ and depth $12L + C_2$ can approximate an arbitrary Hölder continuous function of order α with a constant λ on $[0,1]^d$ with a nearly tight approximation rate $19\sqrt{d}\,\lambda N^{-2\alpha/d}L^{-2\alpha/d}$ measured in L^p-norm for any given $N, L \in \mathbb{N}^+$ and $p \in [1, \infty]$. The constructive approximation is a corollary of a more general result for an arbitrary continuous function f in terms of its modulus of continuity $\omega_f(\cdot)$. In particular, the approximation rate of ReLU FNNs with width $C_1 \max\{d\lfloor N^{1/d}\rfloor, N+1\}$ and depth $12L + C_2$ for a general continuous function f is $19\sqrt{d}\,\omega_f(N^{-2/d}L^{-2/d})$. We also extend our analysis to f on irregular domains or those localized in an ε-neighborhood of a $d_{\mathcal{M}}$-dimensional smooth manifold $\mathcal{M} \subseteq [0,1]^d$ with $d_{\mathcal{M}} \ll d$. Especially, in the case of an essentially low-dimensional domain, we show an approximation rate $2\omega_f\left(\frac{2\varepsilon}{1-\delta}\sqrt{\frac{d}{d_\delta}} + 2\varepsilon\right) + 19\sqrt{d}\,\omega_f\left(\frac{2\sqrt{d}}{(1-\delta)\sqrt{d_\delta}}N^{-2/d_\delta}L^{-2/d_\delta}\right)$ for ReLU FNNs to approximate f in the ε-neighborhood, where $d_\delta = \mathcal{O}\left(d_{\mathcal{M}}\frac{\ln(d/\delta)}{\delta^2}\right)$ for any given $\delta \in (0,1)$ as a relative error for a projection to approximate an isometry when we project $\mathcal{M}$ to a d_δ-dimensional domain.

Key words. Deep ReLU Neural Networks, Hölder Continuity, Modulus of Continuity, Approximation Theory, Low-Dimensional Manifold, Parallel Computing.

1 Introduction

The approximation theory of neural networks has been an active research topic in the past few decades. Previously, as a special kind of ridge function approximation, shallow neural networks with one hidden layer and various activation functions (e.g., wavelets pursuits [10, 46], adaptive splines [19, 55], radial basis functions [8, 18, 25, 53, 65], sigmoid

---

functions [7,13–15,29,37,38,41,45]) were widely discussed and admit good approximation properties, e.g., the universal approximation property [16, 29, 30], lessening the curse of dimensionality [4, 21, 22], and providing attractive approximation rate in nonlinear approximation [10,18,19,25,46,55,65].

The introduction of deep networks with more than one hidden layers has made significant impacts in many fields in computer science and engineering including computer vision [35] and natural language processing [1]. New scientific computing tools based on deep networks have also emerged and facilitated large-scale and high-dimensional problems that were impractical previously [20,24]. The design of deep ReLU FNNs is the key of such a revolution. These breakthroughs have stimulated broad research topics from different points of views to study the power of deep ReLU FNNs, e.g. in terms of combinatorics [51], topology [6], Vapnik-Chervonenkis (VC) dimension [5,27,58], fat-shattering dimension [2,34], information theory [54], classical approximation theory [4,16,30,62,67], optimization [32,33,52] etc.

Particularly in approximation theory, **non-quantitative and asymptotic** approximation rates of ReLU FNNs have been proposed for various types of functions. For example, smooth functions [23,39,43,66], piecewise smooth functions [54], band-limited functions [50], continuous functions [67], solutions to partial differential equations [31]. However, to the best of our knowledge, existing theories [17,23,39,43,48,50,54,63,66,67] can only provide implicit formulas in the sense that the approximation error contains an unknown prefactor, or work only for sufficiently large N and L larger than some unknown numbers. For example, [67] estimated an approximation rate $c(d)L^{-2\alpha/d}$ via a narrow and deep ReLU FNN, where $c(d)$ is an unknown number depending on d, and L is required to be larger than a sufficiently large unknown number $\mathscr{L}$. For another example, given an approximation error ε, [54] proved the existence of a ReLU FNN with a constant but still unknown number of layers approximating a C^β function within the target error. These works can be divided into two cases: 1) FNNs with varying width and only one hidden layer [18,25,40,65] (visualized by the region in ▬ in Figure 1); 2) FNNs with a fixed width of $\mathcal{O}(d)$ and a varying depth larger than an unknown number $\mathscr{L}$ [44,67] (represented by the region in ▬ in Figure 1).

As far as we know, the first **quantitative and non-asymptotic** approximation rate of deep ReLU FNNs was obtained in [62]. Specifically, [62] identified an explicit formulas of the approximation rate

$$
\begin{cases}
2\lambda N^{-2\alpha}, & \text{when } L \geq 2 \text{ and } d = 1, \\
2(2\sqrt{d})^\alpha \lambda N^{-2\alpha/d}, & \text{when } L \geq 3 \text{ and } d \geq 2,
\end{cases}
\tag{1.1}
$$

for ReLU FNNs with an arbitrary width $N \in \mathbb{N}^+$ and a fixed depth $L \in \mathbb{N}^+$ to approximate a Hölder continuous function f of order α with a Hölder constant λ (visualized in the region shown by ▬ in Figure 1). The approximation rate $\mathcal{O}(N^{-2\alpha/d})$ is tight in terms of N and increasing L cannot improve the approximation rate in N. The success of deep FNNs in a broad range of applications has motivated a well-known conjecture that the depth L has an important role in improving the approximation power of deep FNNs. In particular, a very important question in practice would be, given an arbitrary L and N, what is the explicit formula to characterize the approximation error so as to see whether the network is large enough to meet the accuracy requirement. Due to the highly

nonlinear structure of deep FNNs, it is still a challenging open problem to characterize N and L simultaneously in the approximation rate.

To answer the question just above, we establish the first framework that is able to quantify the approximation power of deep ReLU FNNs essentially with arbitrary width N and depth L, achieving a nearly optimal approximation rate, $19\sqrt{d}\,\omega_f(N^{-2/d}L^{-2/d})$, for continuous functions $f \in C([0,1]^d)$. Our result is based on new analysis techniques merely based on the structure of FNNs and a modified bit extraction technique inspired by [5], instead of designing FNNs to approximate traditional approximation basis like polynomials and splines as in the existing literature [26, 39, 43, 48, 49, 54, 56, 57, 60, 63, 66, 67]. The approximation rate obtained here admits an explicit formula to compute the prefactor when $\omega_f(\cdot)$ is known. For example, in the case of Hölder continuous functions of order α with a Hölder constant λ (denoted as the class $\mathrm{Höld}([0,1]^d, \alpha, \lambda)$), $\omega_f(r) \le \lambda r^{\alpha}$ for $r \ge 0$, resulting in the approximation rate $19\sqrt{d}\,\lambda N^{-2\alpha/d}L^{-2\alpha/d}$ as mentioned previously. As a consequence, existing works for the function class $C([0,1]^d)$ are special cases of our result (see Figure 1 for a comparison).
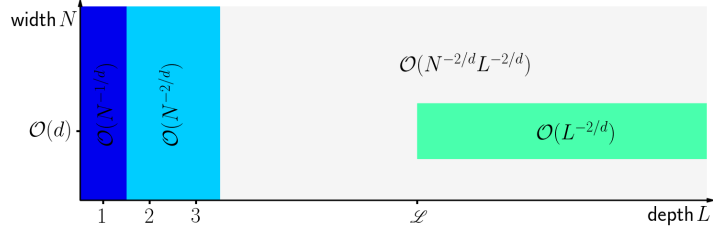

Figure 1: A summary of existing and our new results on the approximation rate of ReLU FNNs for continuous functions. Existing results [18, 25, 40, 44, 62, 65, 67] are applicable in the areas in ■, ■, and ■; our new result is suitable for almost all areas when $L \ge 2$.

Our key contributions can be summarized as follows.

1. Lower bound: We provide a quantitative and non-asymptotic approximation rate $19\sqrt{d}\,\omega_f(N^{-2/d}L^{-2/d})$ in terms of width $\mathcal{O}(N)$ and depth $\mathcal{O}(L)$ for functions in $C([0,1]^d)$ in Theorem 1.1.

2. Upper bound: Through the nearly tight VC-dimension bounds of ReLU FNNs [27], we show that the approximation rate $19\sqrt{d}\,\omega_f(N^{-2\alpha/d}L^{-2\alpha/d})$ in terms of N and L is nearly optimal for $\mathrm{Höld}([0,1]^d, \alpha, \lambda)$ in Theorem 2.3.

3. The approximation rate in terms of the width and depth in this paper is more generic and useful than the one characterized by the number of nonzero parameters denoted as W in the literature. First, the characterization in terms of width and depth implies the one in terms of W, while it is not true the other way around. Second, our theory can provide practical guidance for choosing network sizes in realistic applications while theories in terms of W cannot tell how large a network should be to guarantee a target accuracy, since there are too many networks of different sizes sharing the same number of parameters but with different accuracies.

3

4. Finally, three aspects of neural networks in practice are discussed: 1) neural network approximation in a high-dimensional irregular domain; 2) neural network approximation in the case of a low-dimensional data structure; 3) the optimal ReLU FNN in parallel computation.

Our main result, Theorem 1.1 below, shows that ReLU FNNs with width $\mathcal{O}(N)$ and depth $\mathcal{O}(L)$ can approximate f with an approximation rate $19\sqrt{d}\,\omega_f(N^{-2/d}L^{-2/d})$, where $\omega_f(\cdot)$ is the modulus of continuity of f defined via

$$\omega_f(r) \coloneqq \sup\left\{|f(\boldsymbol{x}) - f(\boldsymbol{y})| : \boldsymbol{x}, \boldsymbol{y} \in [0,1]^d, \ \|\boldsymbol{x} - \boldsymbol{y}\|_2 \le r\right\}, \quad \text{for any } r \ge 0.$$

Theorem 1.1. *Given $f \in C([0,1]^d)$, for any $L \in \mathbb{N}^+$, $N \in \mathbb{N}^+$, and $p \in [1,\infty]$, there exists a function ϕ implemented by a ReLU FNN with width $C_1 \max\left\{d\lfloor N^{1/d}\rfloor, N+1\right\}$ and depth $12L + C_2$ such that*

$$\|f - \phi\|_{L^p([0,1]^d)} \le 19\sqrt{d}\,\omega_f(N^{-2/d}L^{-2/d}),$$

where $C_1 = 12$ and $C_2 = 14$ if $p \in [1,\infty)$; $C_1 = 3^{d+3}$ and $C_2 = 14 + 2d$ if $p = \infty$.

When Theorem 1.1 is applied to $f \in \text{Höld}([0,1]^d, \alpha, \lambda)$, the approximation rate is $19\sqrt{d}\,\lambda N^{-2\alpha/d}L^{-2\alpha/d}$, because $\omega_f(r) \le \lambda r^\alpha$ for any $r \ge 0$. An immediate question following the constructive approximation is how much we can improve the approximation rate. In fact, the approximation rate of $f \in \text{Höld}([0,1]^d, \alpha, \lambda)$ is asymptotically tight based on VC-dimension as we shall see later.

In most real applications of neural networks, though the target function f is defined in a high-dimensional domain, e.g., $[0,1]^d$, where d could be tens of thousands or even millions, only the approximation error of f in a neighborhood of a $d_{\mathcal{M}}$-dimensional manifold $\mathcal{M}$ with $d_{\mathcal{M}} \ll d$ is concerned. Hence, we extend Theorem 1.1 to the case when the domain of f is localized in an ε-neighborhood of a compact $d_{\mathcal{M}}$-dimensional Riemannian submanifold $\mathcal{M} \subseteq [0,1]^d$ having condition number $1/\tau$, volume V, and geodesic covering regularity $\mathcal{R}$. The ε-neighborhood is defined as

$$\mathcal{M}_\varepsilon \coloneqq \left\{\boldsymbol{x} \in [0,1]^d : \inf\{\|\boldsymbol{x} - \boldsymbol{y}\|_2 : \boldsymbol{y} \in \mathcal{M}\} \le \varepsilon\right\}, \quad \text{for } \varepsilon \in (0,1). \tag{1.2}$$

Let $d_\delta = \mathcal{O}\left(\frac{d_{\mathcal{M}}\ln(dV\mathcal{R}\tau^{-1}\delta^{-1})}{\delta^2}\right) = \mathcal{O}\left(d_{\mathcal{M}}\frac{\ln(d/\delta)}{\delta^2}\right)$ be an integer for any $\delta \in (0,1)$ such that $d_{\mathcal{M}} \le d_\delta \le d$. We show an approximation rate

$$2\omega_f\left(\frac{2\varepsilon}{1-\delta}\sqrt{\frac{d}{d_\delta}} + 2\varepsilon\right) + 19\sqrt{d}\,\omega_f\left(\frac{2\sqrt{d}}{(1-\delta)\sqrt{d_\delta}}N^{-2/d_\delta}L^{-2/d_\delta}\right)$$

for ReLU FNNs to pointwisely approximate f on $\mathcal{M}_\varepsilon$. The key ideas of the proof is the application of Theorem 3.1 in [3], which provides a nearly isometric projection $\boldsymbol{A} \in \mathbb{R}^{d_\delta \times d}$ that maps points in $\mathcal{M} \subseteq [0,1]^d$ to a d_δ-dimensional domain with

$$(1-\delta)|\boldsymbol{x}_1 - \boldsymbol{x}_2| \le |\boldsymbol{A}\boldsymbol{x}_1 - \boldsymbol{A}\boldsymbol{x}_2| \le (1+\delta)|\boldsymbol{x}_1 - \boldsymbol{x}_2| \quad \text{for any } \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{M},$$

and the application of Theorem 1.1 in this paper, which constructs the desired ReLU FNN with a size depending on d_δ instead of d to lessen the curse of dimensionality. When δ is closer to 1, d_δ is closer to $d_{\mathcal{M}}$ but the isometric property of the projection is weakened; when δ is closer to 0, the isometric property becomes better but d_δ could be larger than d, in which case we can simply enforce $d_\delta = d$ and choose the identity map as the projection. Hence, $\delta \in (0,1)$ is a parameter to make a balance between isometry and dimension reduction.

Theorem 1.2. *Let f be a continuous function on $[0,1]^d$ and $\mathcal{M} \subseteq [0,1]^d$ be a compact $d_\mathcal{M}$-dimensional Riemannian submanifold. For any $N \in \mathbb{N}^+$, $L \in \mathbb{N}^+$, $\varepsilon \in (0,1)$, and $\delta \in (0,1)$, there exists a function ϕ implemented by a ReLU FNN with width $3^{d_\delta+3} \max\left\{ d_\delta \lfloor N^{1/d_\delta} \rfloor, N+1 \right\}$ and depth $12L + 14 + 2d_\delta$ such that*

$$|f(\boldsymbol{x}) - \phi(\boldsymbol{x})| \le 2\omega_f\left(\tfrac{2\varepsilon}{1-\delta}\sqrt{\tfrac{d}{d_\delta}} + 2\varepsilon\right) + 19\sqrt{d}\,\omega_f\left(\tfrac{2\sqrt{d}}{(1-\delta)\sqrt{d_\delta}}N^{-2/d_\delta}L^{-2/d_\delta}\right), \tag{1.3}$$

for any $\boldsymbol{x} \in \mathcal{M}_\varepsilon$, where $\mathcal{M}_\varepsilon$ is defined in Equation (1.2)

The approximation rate of deep neural networks for functions defined precisely on low-dimensional smooth manifolds has been studied in [61] for C^2 functions and in [9,11] for Lipschitz continuous functions. Considering that it might be more reasonable to assume data located in a small neighborhood of low-dimensional smooth manifold in real applications, we introduce the ε-neighborhood of the manifold $\mathcal{M}$ in Theorem 1.2. In general, existing results are again asymptotic and they cannot be applied to estimate the approximation accuracy of a ReLU FNN with arbitrarily given width N and depth L, since there is no explicit formula without unknown constants to specify the exact error bound. For example, [9] provides an approximation rate $c_1 (NL)^{-c_2/d_\delta}$ with unknown constants (e.g., c_1 and c_2) and requires NL greater than an unknown large number. The demand of an explicit error estimation motivates Theorem 1.2 in this paper. When data are concentrating around $\mathcal{M}$, ε is very small and the dominant term of the approximation error in (1.3) is $19\sqrt{d}\,\omega_f\left(\tfrac{2\sqrt{d}}{(1-\delta)\sqrt{d_\delta}}N^{-2/d_\delta}L^{-2/d_\delta}\right)$ implying that the approximation via deep ReLU FNNs can lessen the curse of dimensionality.

The analysis above provides a general guide for selecting the width and depth of ReLU FNNs to approximate continuous functions, especially when the computation is conducted with parallel computing, which is usually the case in real applications [12,59]. As we shall see later, when the approximation accuracy and the parallel computing efficiency are considered together, very deep FNNs become less attractive than those with $\mathcal{O}(1)$ depth.

The approximation theories in this paper assume that the target function f is fully accessible, making it possible to estimate the approximation error and identify an asymptotically optimal ReLU FNN with a given budget of neurons to minimize the approximation error. In real applications, usually only a limited number of possibly noisy observations of f is available, resulting in a regression problem in statistics. In the latter case, the problem is usually formulated in a stochastic setting with randomly generated noisy observations and the regression error contains mainly two components: bias and variance. The bias is the difference of the expectation of an estimated function and its ground truth f. The approximation theories in this paper play an important role in characterizing the power of neural networks when they are applied to solve regression problems by providing a lower bound of the regression bias.

The rest of this paper is organized as follows. We first prove Theorem 1.1 and show its optimality in Section 2 when assuming Theorem 2.1 is true. Next, Theorem 2.1 is proved in Section 3. In Section 4, three aspects of neural networks in practice will be discussed: 1) neural network approximation in a high-dimensional irregular domain; 2) neural network approximation in the case of a low-dimensional data structure; 3) the optimal ReLU FNN in parallel computation. Finally, Section 5 concludes this paper with a short discussion.

2 Approximation of continuous functions

In this section, we prove Theorem 1.1 and discuss its optimality when assume Theorem 2.1 is true. Notations throughout the proof will be summarized in Section 2.1.

2.1 Notations

Let us summarize all basic notations used in this paper as follows.

- Matrices are denoted by bold uppercase letters. For instance, $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ is a real matrix of size $m \times n$, and $\boldsymbol{A}^T$ denotes the transpose of $\boldsymbol{A}$.

- Vectors are denoted as bold lowercase letters. For example, $\boldsymbol{v} \in \mathbb{R}^n$ is a column vector of size n. Correspondingly, $\boldsymbol{v}(i)$ is the i-th element of $\boldsymbol{v}$. $\boldsymbol{v} = (v_1, \cdots, v_n) = \begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix}$ are vectors consisting of numbers $\{v_i\}_i$ with $\boldsymbol{v}(i) = v_i$.

- Let $\mu(\cdot)$ be the Lebesgue measure.

- Let 1_S be the characteristic function on a set S, i.e., 1_S equals to 1 on S and 0 outside of S.

- The set difference of two sets A and B is denoted by $A \backslash B := \{x : x \in A, \ x \notin B\}$.

- For any $\xi \in \mathbb{R}$, let $\lfloor \xi \rfloor := \max\{i : i \leq \xi, \ i \in \mathbb{Z}\}$ and $\lceil \xi \rceil := \min\{i : i \geq \xi, \ i \in \mathbb{Z}\}$.

- Assume $\boldsymbol{n} \in \mathbb{N}^n$, then $f(\boldsymbol{n}) = \mathcal{O}(g(\boldsymbol{n}))$ means that there exists positive C independent of $\boldsymbol{n}$, f, and g such that $f(\boldsymbol{n}) \leq Cg(\boldsymbol{n})$ when all entries of $\boldsymbol{n}$ go to $+\infty$.

- Let $\sigma : \mathbb{R} \to \mathbb{R}$ denote the rectified linear unit (ReLU), i.e. $\sigma(x) = \max\{0, x\}$. With the abuse of notations, we define $\sigma : \mathbb{R}^d \to \mathbb{R}^d$ as $\sigma(\boldsymbol{x}) = \begin{bmatrix} \max\{0, x_1\} \\ \vdots \\ \max\{0, x_d\} \end{bmatrix}$ for any $\boldsymbol{x} = (x_1, \cdots, x_d) \in \mathbb{R}^d$.

- Given $K \in N^+$ and $\delta \in (0, \frac{1}{K})$, define a trifling region $\Omega([0,1]^d, K, \delta)$ of $[0,1]^d$ as

$$\Omega([0,1]^d, K, \delta) := \bigcup_{i=1}^{d} \left\{ \boldsymbol{x} = (x_1, x_2, \cdots, x_d) : x_i \in \cup_{k=1}^{K-1}(\tfrac{k}{K} - \delta, \tfrac{k}{K}) \right\}. \tag{2.1}$$

  In particular, $\Omega([0,1]^d, K, \delta) = \varnothing$ if $K = 1$. See Figure 2 for two examples of trifling regions.

- Let $\mathrm{H\ddot{o}ld}([0,1]^d, \alpha, \lambda)$ be the set containing all Hölder continuous functions of order $\alpha \in (0, 1]$ with a Hölder constant λ.

- We will use $\mathcal{NN}$ to denote a function implemented by a ReLU FNN for short and use Python-type notations to specify a class of functions implemented by ReLU FNNs with several conditions, e.g., $\mathcal{NN}(\mathrm{c}_1; \ \mathrm{c}_2; \ \cdots; \ \mathrm{c}_m)$ is a set of functions implemented by ReLU FNNs satisfying m conditions given by $\{\mathrm{c}_i\}_{1 \leq i \leq m}$, each of which may
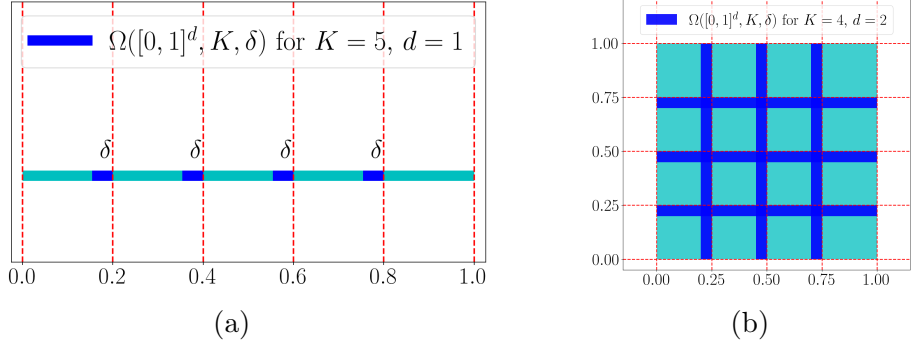
Figure 2: Two examples of trifling regions. (a) $K = 5, d = 1$. (b) $K = 4, d = 2$.

specify the number of inputs (#input), the number of outputs (#output), the total number of nodes in all hidden layers (#neuron), the number of hidden layers (depth), the number of total parameters (#parameter), and the width in each hidden layer (widthvec), the maximum width of all hidden layers (width), etc. For example, if $\phi \in \mathcal{NN}(\#\text{input} = 2; \ \text{widthvec} = [100, 100]; \ \#\text{output} = 1)$, then ϕ is a functions satisfies

- ϕ maps from $\mathbb{R}^2$ to $\mathbb{R}$.
- ϕ can be implemented by a ReLU FNN with two hidden layers and the number of nodes in each hidden layer is 100.

- $[n]^L$ is short for $[n, n, \cdots, n] \in \mathbb{N}^L$. For example,

$$\mathcal{NN}(\#\text{input} = d; \ \text{widthvec} = [100, 100]) = \mathcal{NN}(\#\text{input} = d; \ \text{widthvec} = [100]^2).$$

- For a function $\phi \in \mathcal{NN}(\#\text{input} = d; \ \text{widthvec} = [N_1, N_2, \cdots, N_L]; \ \#\text{output} = 1)$, if we set $N_0 = d$ and $N_{L+1} = 1$, then the architecture of the network implementing ϕ can be briefly described as follows:

$$\boldsymbol{x} = \widetilde{\boldsymbol{h}}_0 \xrightarrow{\boldsymbol{W}_0, \ \boldsymbol{b}_0} \boldsymbol{h}_1 \xrightarrow{\sigma} \widetilde{\boldsymbol{h}}_1 \cdots \xrightarrow{\boldsymbol{W}_{L-1}, \ \boldsymbol{b}_{L-1}} \boldsymbol{h}_L \xrightarrow{\sigma} \widetilde{\boldsymbol{h}}_L \xrightarrow{\boldsymbol{W}_L, \ \boldsymbol{b}_L} \boldsymbol{h}_{L+1} = \phi(\boldsymbol{x}),$$

where $\boldsymbol{W}_i \in \mathbb{R}^{N_{i+1} \times N_i}$ and $\boldsymbol{b}_i \in \mathbb{R}^{N_{i+1}}$ are the weight matrix and the bias vector in the i-th linear transform $\mathcal{L}_i$ in ϕ, respectively, i.e.,

$$\boldsymbol{h}_{i+1} := \boldsymbol{W}_i \widetilde{\boldsymbol{h}}_i + \boldsymbol{b}_i =: \mathcal{L}_i(\widetilde{\boldsymbol{h}}_i), \quad \text{for } i = 0, 1, \cdots, L,$$

and

$$\widetilde{\boldsymbol{h}}_i = \sigma(\boldsymbol{h}_i), \quad \text{for } i = 1, \dots, L.$$

In particular, ϕ can be represented in a form of function compositions as follows

$$\phi = \mathcal{L}_L \circ \sigma \circ \mathcal{L}_{L-1} \circ \sigma \circ \cdots \circ \sigma \circ \mathcal{L}_1 \circ \sigma \circ \mathcal{L}_0,$$

which has been illustrated in Figure 3.

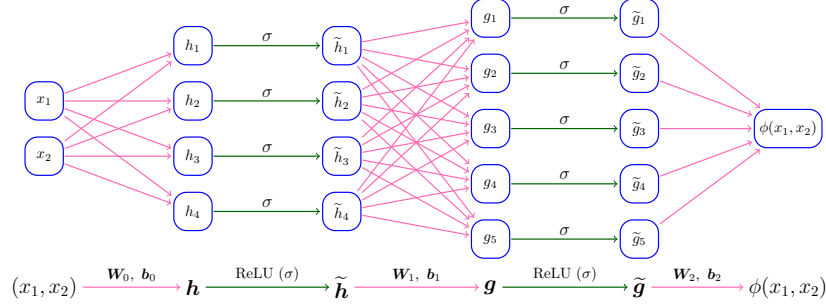- The expression "an FNN with width N and depth L" means

7

Figure 3: An example of ReLU networks with width 5 and depth 2.

- The maximum width of this FNN for all hidden layers is no more than N.
- The number of hidden layers of this FNN is no more than L.

- For $\theta \in [0,1)$, suppose its binary representation is $\theta = \sum_{\ell=1}^{\infty} \theta_\ell 2^{-\ell}$ with $\theta_\ell \in \{0,1\}$, we introduce a special notation $\mathrm{bin}\,0.\theta_1\theta_2\cdots\theta_L$ to denote the L-term binary representation of θ, i.e., $\sum_{\ell=1}^{L} \theta_\ell 2^{-\ell}$.

2.2 Proof of Theorem 1.1

We essentially construct piecewise constant functions to approximate continuous functions in the proof. However, it is impossible to construct a piecewise constant function via ReLU FNNs due to the continuity of ReLU FNNs. Thus, we introduce the trifling region $\Omega([0,1]^d, K, \delta)$, defined in Equation (2.1), and use ReLU FNNs to implement piecewise constant functions outside of the trifling region. To prove Theorem 1.1, we first establish a theorem showing how to construct ReLU FNNs to pointwisely approximate continuous functions except for the trifling region.

Theorem 2.1. *Given $f \in C([0,1]^d)$, for any $L \in \mathbb{N}^+$ and $N \in \mathbb{N}^+$, there exists a function ϕ implemented by a ReLU FNN with width $\max\{4d\lfloor N^{1/d}\rfloor + 3d, 12N + 8\}$ and depth $12L + 14$ such that $\|\phi\|_{L^\infty(\mathbb{R}^d)} \le |f(\mathbf{0})| + \omega_f(\sqrt{d})$ and*

$$|f(\boldsymbol{x}) - \phi(\boldsymbol{x})| \le 18\sqrt{d}\,\omega_f(N^{-2/d}L^{-2/d}), \quad \text{for any } \boldsymbol{x} \in [0,1]^d\backslash\Omega([0,1]^d, K, \delta),$$

where $K = \lfloor N^{1/d}\rfloor^2\lfloor L^{2/d}\rfloor$ and δ is an arbitrary number in $(0, \frac{1}{3K}]$.

With Theorem 2.1 that will be proved in Section 3, we can easily prove Theorem 1.1 for the case $p \in [1, \infty)$. In the early version of this paper, which focuses on continuous functions as target functions, we only considered the case $p \in [1, \infty)$ since it was challenging to control the approximation error in the trifling region. Later in [42] when we considered smooth functions as target functions, we invented a technique that can handle the error in the trifling region as in the lemma below. Therefore, we are now able to control the approximation error for $p = \infty$. The results in this paper are for continuous functions, to which the results in [42] are not applicable; the results in [42] characterize how the smoothness of target functions helps to enhance the approximation capacity of ReLU FNNs, which is not addressed in this paper. It is interesting to point out that the approximation rate $\mathcal{O}(N^{-2/d}L^{-2/d})$ for continuous functions in this paper is even better than the rate $\mathcal{O}((\frac{N}{\ln N})^{-2/d}(\frac{L}{\ln L})^{-2/d})$ for functions in $C^1([0,1]^d)$ in [42].

8

Lemma 2.2 (Theorem 2.1 of [42]). *Given $\varepsilon > 0$, $N, L, K \in \mathbb{N}^+$, and $\delta \in (0, \frac{1}{3K}]$, assume $f \in C([0,1]^d)$ and $\widetilde{\phi}$ can be implemented by a ReLU FNN with width N and depth L. If*

$$|f(\boldsymbol{x}) - \widetilde{\phi}(\boldsymbol{x})| \le \varepsilon, \quad \text{for any } \boldsymbol{x} \in [0,1]^d \backslash \Omega([0,1]^d, K, \delta),$$

then there exists a function ϕ implemented by a new ReLU FNN with width $3^d(N+4)$ and depth $L + 2d$ such that

$$|f(\boldsymbol{x}) - \phi(\boldsymbol{x})| \le \varepsilon + d \cdot \omega_f(\delta), \quad \text{for any } \boldsymbol{x} \in [0,1]^d.$$

Now we are ready to prove Theorem 1.1 by assuming Theorem 2.1 is true, which will be proved later in Section 3.2.

Proof of Theorem 1.1. Let us first consider the case $p \in [1, \infty)$. We may assume f is not a constant function since it is a trivial case. Then $\omega_f(r) > 0$ for any $r > 0$. Set $K = \lfloor N^{1/d} \rfloor^2 \lfloor L^{2/d} \rfloor$ and choose a small $\delta \in (0, \frac{1}{3K}]$ such that

$$Kd\delta\big(|f(\mathbf{0})| + \omega_f(\sqrt{d})\big)^p = \lfloor N^{1/d} \rfloor^2 \lfloor L^{2/d} \rfloor d\delta\big(|f(\mathbf{0})| + \omega_f(\sqrt{d})\big)^p$$
$$\le \big(\omega_f(N^{-2/d} L^{-2/d})\big)^p.$$

By Theorem 2.1, there exists a function ϕ implemented by a ReLU FNN with width

$$\max\big\{4d\lfloor N^{1/d} \rfloor + 3d,\ 12N + 8\big\} \le 12 \max\big\{d\lfloor N^{1/d} \rfloor,\ N + 1\big\}$$

and depth $12L + 14$ such that

$$|f(\boldsymbol{x}) - \phi(\boldsymbol{x})| \le 18\sqrt{d}\,\omega_f(N^{-2/d} L^{-2/d}), \quad \text{for any } \boldsymbol{x} \in [0,1]^d \backslash \Omega([0,1]^d, K, \delta),$$

It follows from $\mu(\Omega([0,1]^d, K, \delta)) \le Kd\delta$ that

$$\|f - \phi\|_{L^p([0,1]^d)}^p = \int_{\Omega([0,1]^d, K, \delta)} |f(\boldsymbol{x}) - \phi(\boldsymbol{x})|^p \mathrm{d}\boldsymbol{x} + \int_{[0,1]^d \backslash \Omega([0,1]^d, K, \delta)} |f(\boldsymbol{x}) - \phi(\boldsymbol{x})|^p \mathrm{d}\boldsymbol{x}$$
$$\le Kd\delta\big(|f(\mathbf{0})| + \omega_f(\sqrt{d})\big)^p + \big(18\sqrt{d}\,\omega_f(N^{-2/d} L^{-2/d})\big)^p$$
$$\le \big(\omega_f(N^{-2/d} L^{-2/d})\big)^p + \big(18\sqrt{d}\,\omega_f(N^{-2/d} L^{-2/d})\big)^p$$
$$\le \big(19\sqrt{d}\,\omega_f(N^{-2/d} L^{-2/d})\big)^p.$$

Hence, $\|f - \phi\|_{L^p([0,1]^d)} \le 19\sqrt{d}\,\omega_f(N^{-2/d} L^{-2/d})$.

Next, let us discuss the case $p = \infty$. Set $K = \lfloor N^{1/d} \rfloor^2 \lfloor L^{2/d} \rfloor$ and choose a small $\delta \in (0, \frac{1}{3K}]$ such that

$$d \cdot \omega_f(\delta) \le \omega_f(N^{-2/d} L^{-2/d}).$$

By Theorem 2.1, there exists a function $\widetilde{\phi}$ implemented by a ReLU FNN with width $\max\big\{4d\lfloor N^{1/d} \rfloor + 3d,\ 12N + 8\big\}$ and depth $12L + 14$ such that

$$|f(\boldsymbol{x}) - \widetilde{\phi}(\boldsymbol{x})| \le 18\sqrt{d}\,\omega_f(N^{-2/d} L^{-2/d}) := \varepsilon, \quad \text{for } \boldsymbol{x} \in [0,1]^d \backslash \Omega([0,1]^d, K, \delta),$$

By Lemma 2.2, there exists a function ϕ implemented by a ReLU FNN with width

$$3^d\Big(\max\big\{4d\lfloor N^{1/d} \rfloor + 3d,\ 12N + 8\big\} + 4\Big) \le 3^{d+3} \max\big\{d\lfloor N^{1/d} \rfloor,\ N + 1\big\}$$

and depth $12L + 14 + 2d$ such that

$$|f(\boldsymbol{x}) - \phi(\boldsymbol{x})| \le \varepsilon + d \cdot \omega_f(\delta) \le 19\sqrt{d}\,\omega_f(N^{-2/d} L^{-2/d}), \quad \text{for any } \boldsymbol{x} \in [0,1]^d.$$

So we finish the proof. $\qquad\square$

2.3 Optimality of Theorem 1.1

This section will show that the approximation rate in Theorem 1.1 is nearly tight and there is no room to improve for the function class $\text{Höld}([0,1]^d, \alpha, \lambda)$. Theorem 2.3 below shows that the approximation rate $\mathcal{O}(\omega_f(N^{-(2/d+\rho)}L^{-(2/d+\rho)}))$ for any $\rho > 0$ is unachievable, implying the approximation rate in Theorem 1.1 is nearly tight for the function class $\text{Höld}([0,1]^d, \alpha, \lambda)$.

Theorem 2.3. *Given any $\rho > 0$ and $C > 0$, there exists $f \in \text{Höld}([0,1]^d, \alpha, \lambda)$ such that, for any $J_0 > 0$, there exist $N, L \in \mathbb{N}$ with $NL \geq J_0$ satisfying*

$$\inf_{\phi \in \mathcal{NN}(\#\text{input}=d;\ \text{width}\leq N;\ \text{depth}\leq L)} \|\phi - f\|_{L^\infty([0,1]^d)} \geq C\lambda N^{-(2\alpha/d+\rho)}L^{-(2\alpha/d+\rho)}.$$

In fact, we can show a stronger result than Theorem 2.3. Under the same conditions as in Theorem 2.3, for any $\mathcal{H} \in [0,1]^d$ with $\mu(\mathcal{H}) \leq 2^{-(d+K^d+1)}K^{-d}$, where $K = \lfloor (NL)^{2/d+\rho/(2\alpha)} \rfloor$, it can be proved that

$$\inf_{\phi \in \mathcal{NN}(\#\text{input}=d;\ \text{width}\leq N;\ \text{depth}\leq L)} \|\phi - f\|_{L^\infty([0,1]^d \setminus \mathcal{H})} \geq C\lambda N^{-(2\alpha/d+\rho)}L^{-(2\alpha/d+\rho)}. \tag{2.2}$$

We will prove (2.2) by contradiction, then Theorem 2.3 holds as a consequence. Assuming Equation (2.2) is false, we have the following claim.

Claim 2.4. *There exist $\rho > 0$ and $C > 0$ such that given any $f \in \text{Höld}([0,1]^d, \alpha, \lambda)$, there exists $J_0 = J_0(\rho, C, f) > 0$ such that, for any $N, L \in \mathbb{N}$ with $NL \geq J_0$, there exist $\phi \in \mathcal{NN}(\#\text{input} = d;\ \text{width} \leq N;\ \text{depth} \leq L)$ and $\mathcal{H} \in [0,1]^d$ with $\mu(\mathcal{H}) \leq 2^{-(d+K^d+1)}K^{-d}$, where $K = \lfloor (NL)^{2/d+\rho/(2\alpha)} \rfloor$, satisfying*

$$\|f - \phi\|_{L^\infty([0,1]^d \setminus \mathcal{H})} \leq C\lambda N^{-(2\alpha/d+\rho)}L^{-(2\alpha/d+\rho)}.$$

Now let us disprove this claim to show Theorem 2.3 and Equation (2.2) are true.

Disproof of Claim 2.4. Without the loss of generality, we assume $\lambda = 1$; in the case of $\lambda \neq 1$, the proof is similar. We will disprove Claim 2.4 using the VC dimension. Recall that the VC dimension of a class of functions is defined as the cardinality of the largest set of points that this class of functions can shatter. Denote the VC dimension of a function set $\mathscr{F}$ by $\text{VCDim}(\mathscr{F})$. By [27] and the fact

$$\mathcal{NN}(\text{width} \leq N;\ \text{depth} \leq L) \subseteq \mathcal{NN}(\#\text{parameter} \leq (LN + d + 2)(N + 1)),$$

there exists $C_1 > 0$ such that

$$\begin{aligned} &\text{VCDim}\big(\mathcal{NN}(\#\text{input} = d;\ \text{width} \leq N;\ \text{depth} \leq L)\big) \\ &\leq C_1(LN + d + 2)(N + 1)L\ln\big((LN + d + 2)(N + 1)\big) \\ &=: b_u(N, L). \end{aligned} \tag{2.3}$$

Then we will use Claim 2.4 to estimate a lower bound of

$$\text{VCDim}\big(\mathcal{NN}(\#\text{input} = d;\ \text{width} \leq N;\ \text{depth} \leq L)\big), \tag{2.4}$$

10

and this lower bound is asymptotically larger than $b_u(N, L)$, which leads to a contradiction.

More precisely, we will construct $\{f_\chi : \chi \in \mathscr{B}\} \subseteq \text{Höld}([0,1]^d, \alpha, 1)$, which can shatter $b_\ell(N, L) := K^d$ points, where $\mathscr{B}$ is a set defined later. Then by Claim 2.4, there exists $\{\phi_\chi : \chi \in \mathscr{B}\}$ such that this set can shatter $b_\ell(N, L)$ points. Finally, $b_\ell(N, L) = K^d = \lfloor (NL)^{2/d+\rho/(2\alpha)} \rfloor^d$ is asymptotically larger than $b_u(N, L) = C_1(LN + d + 2)(N + 1)L \ln\big((LN + d + 2)(N + 1)\big)$, which leads to a contradiction. More details can be found below.

Step 1: Construct $\{f_\chi : \chi \in \mathscr{B}\} \subseteq \text{Höld}([0,1]^d, \alpha, \lambda)$ that scatters $b_\ell(N, L)$ points.

Divide $[0,1]^d$ into K^d non-overlapping sub-cubes $\{Q_{\boldsymbol{\beta}}\}_{\boldsymbol{\beta}}$ as follows:

$$Q_{\boldsymbol{\beta}} := \big\{ \boldsymbol{x} = (x_1, x_2, \cdots, x_d) \in [0,1]^d : x_i \in \big[\tfrac{\beta_i - 1}{K}, \tfrac{\beta_i}{K}\big], \ i = 1, 2, \cdots, d \big\},$$

for any index vector $\boldsymbol{\beta} = (\beta_1, \beta_2, \cdots, \beta_d) \in \{1, 2, \cdots, K\}^d$.

Let $Q(\boldsymbol{x}_0, \eta) \subseteq [0,1]^d$ be a hypercube, whose center and sidelength are $\boldsymbol{x}_0$ and η, respectively. Then we define a function ζ_Q on $[0,1]^d$ corresponding to $Q = Q(\boldsymbol{x}_0, \eta) \subseteq [0,1]^d$ such that:

- $\zeta_Q(\boldsymbol{x}_0) = (\eta/2)^\alpha/2$;

- $\zeta_Q(\boldsymbol{x}) = 0$ for any $\boldsymbol{x} \notin Q \backslash \partial Q$, where ∂Q is the boundary of Q;

- ζ_Q is linear on the line that connects $\boldsymbol{x}_0$ and $\boldsymbol{x}$, for any $\boldsymbol{x} \in \partial Q$.

Define
$$\mathscr{B} := \big\{ \chi : \chi \text{ is a map from } \{1, 2, \cdots, K\}^d \text{ to } \{-1, 1\} \big\}.$$

For each $\chi \in \mathscr{B}$, we define

$$f_\chi(\boldsymbol{x}) := \sum_{\boldsymbol{\beta} \in \{1, 2, \cdots, K\}^d} \chi(\boldsymbol{\beta}) \zeta_{Q_{\boldsymbol{\beta}}}(\boldsymbol{x}),$$

where $\zeta_{Q_{\boldsymbol{\beta}}}(\boldsymbol{x})$ is the associated function introduced just above. It is easy to check that $\{f_\chi : \chi \in \mathscr{B}\} \subseteq \text{Höld}([0,1]^d, \alpha, 1)$ can shatter $b_\ell(N, L) = K^d$ points.

Step 2: Construct $\{\phi_\chi : \chi \in \mathscr{B}\}$ that scatters $b_\ell(N, L)$ points.

By Claim 2.4, there exist $\rho > 0$ and $C_2 > 0$ such that, for any $f_\chi \in \{f_\chi : \chi \in \mathscr{B}\}$ there exists $J_\chi > 0$ such that for all $N, L \in \mathbb{N}$ with $NL \geq J_\chi$, there exist $\phi_\chi \in \mathcal{NN}(\#\text{input} = d; \text{ width} \leq N; \text{ depth} \leq L)$ and $\mathcal{H}_\chi$ with $\mu(\mathcal{H}_\chi) \leq 2^{-(d+K^d+1)}K^{-d}$ such that

$$|f_\chi(\boldsymbol{x}) - \phi_\chi(\boldsymbol{x})| \leq C_2(NL)^{-\alpha(2/d+\rho/\alpha)}, \quad \text{for any } \boldsymbol{x} \in [0,1]^d \backslash \mathcal{H}_\chi.$$

Set $\mathcal{H} = \cup_{\chi \in \mathscr{B}} \mathcal{H}_\chi$ and $J_1 = \max_{\chi \in \mathscr{B}} J_\chi$. Then it holds that

$$\mu(\mathcal{H}) \leq 2^{K^d} 2^{-(d+K^d+1)} K^{-d} = (2K)^{-d}/2. \tag{2.5}$$

It follows that for all $\chi \in \mathscr{B}$ and $N, L \in \mathbb{N}$ with $NL \geq J_1$, we have

$$|f_\chi(\boldsymbol{x}) - \phi_\chi(\boldsymbol{x})| \leq C_2(NL)^{-\alpha(2/d+\rho/\alpha)}, \quad \text{for any } \boldsymbol{x} \in [0,1]^d \backslash \mathcal{H}. \tag{2.6}$$

11

For each index vector $\boldsymbol{\beta} \in \{1, 2, \cdots, K\}^d$ and any $\boldsymbol{x} \in \frac{1}{2}Q_{\boldsymbol{\beta}}$, where $\frac{1}{2}Q_{\boldsymbol{\beta}}$ denotes the cube whose sidelength is half of that of $Q_{\boldsymbol{\beta}}$ sharing the same center of $Q_{\boldsymbol{\beta}}$, since $Q_{\boldsymbol{\beta}}$ has a sidelength $\frac{1}{K} = \lfloor (NL)^{2/d + \rho/(2\alpha)} \rfloor^{-1}$, we have

$$|f_\chi(\boldsymbol{x})| = |\zeta_{Q_{\boldsymbol{\beta}}}(\boldsymbol{x})| \geq |\zeta_{Q_{\boldsymbol{\beta}}}(\boldsymbol{x}_{Q_{\boldsymbol{\beta}}})|/2 = \left(\tfrac{1}{2K}\right)^\alpha / 4 = \tfrac{1}{2^{2+\alpha}}\lfloor (NL)^{2/d+\rho/(2\alpha)} \rfloor^{-\alpha}, \qquad (2.7)$$

where $\boldsymbol{x}_{Q_{\boldsymbol{\beta}}}$ is the center of $Q_{\boldsymbol{\beta}}$. For fixed d, α, and ρ, there exists $J_2 > 0$ large enough such that, for any $N, L \in \mathbb{N}$ with $NL \geq J_2$, we have

$$\tfrac{1}{2^{2+\alpha}}\lfloor (NL)^{2/d+\rho/(2\alpha)} \rfloor^{-\alpha} > C_2(NL)^{-\alpha(2/d+\rho/\alpha)}. \qquad (2.8)$$

By Equation (2.5), for any $\boldsymbol{\beta} \in \{1, 2, \cdots, K\}^d$, we have

$$\mu(\mathcal{H}) \leq (2K)^{-d}/2 < (2K)^{-d} = \mu\left(\tfrac{1}{2}Q_{\boldsymbol{\beta}}\right),$$

which means $\left(\frac{1}{2}Q_{\boldsymbol{\beta}}\right) \cap \left([0,1]^d \backslash \mathcal{H}\right)$ is not empty. Therefore, there exists $\boldsymbol{x}_{\boldsymbol{\beta}} \in \left(\frac{1}{2}Q_{\boldsymbol{\beta}}\right) \cap \left([0,1]^d \backslash \mathcal{H}\right)$ for each $\boldsymbol{\beta} \in \{1, 2, \cdots, K\}^d$ such that

$$|f_\chi(\boldsymbol{x}_{\boldsymbol{\beta}})| \geq \tfrac{1}{2^{2+\alpha}}\lfloor (NL)^{2/d+\rho/(2\alpha)} \rfloor^{-\alpha} > C_2(NL)^{-\alpha(2/d+\rho/\alpha)} \geq |f_\chi(\boldsymbol{x}_{\boldsymbol{\beta}}) - \phi_\chi(\boldsymbol{x}_{\boldsymbol{\beta}})|,$$

where the first, the second, and the last inequalities come from (2.7), (2.8), and (2.6), respectively. In other words, for any $\chi \in \mathscr{B}$ and $\boldsymbol{\beta} \in \{1, 2, \cdots, K\}^d$, $f_\chi(\boldsymbol{x}_{\boldsymbol{\beta}})$ and $\phi_\chi(\boldsymbol{x}_{\boldsymbol{\beta}})$ have the same sign. Then $\{\phi_\chi : \chi \in \mathscr{B}\}$ shatters $\{\boldsymbol{x}_{\boldsymbol{\beta}} : \boldsymbol{\beta} \in \{1, 2, \cdots, K\}^d\}$ since $\{f_\chi : \chi \in \mathscr{B}\}$ shatters $\{\boldsymbol{x}_{\boldsymbol{\beta}} : \boldsymbol{\beta} \in \{1, 2, \cdots, K\}^d\}$ as discussed in Step 1. Hence,

$$\mathrm{VCDim}\left(\{\phi_\chi : \chi \in \mathscr{B}\}\right) \geq K^d = b_\ell(N, L). \qquad (2.9)$$

Step 3: Contradiction.

By Equation (2.3) and (2.9), for any $N, L \in \mathbb{N}$ with $NL \geq J_0 = \max\{J_1, J_2\}$, we have

$$b_\ell(N, L) \leq \mathrm{VCDim}\left(\{\phi_\chi : \chi \in \mathscr{B}\}\right) \leq \mathrm{VCDim}\left(\mathcal{NN}(\#\mathrm{input} = d; \ \mathrm{width} \leq N; \ \mathrm{depth} \leq L)\right) \leq b_u(N, L),$$

implying that

$$\lfloor (NL)^{2/d+\rho/(2\alpha)} \rfloor^d \leq C_1(LN + d + 2)(N + 1)L \ln\left((LN + d + 2)(N + 1)\right),$$

which is a contradiction for sufficiently large $N, L \in \mathbb{N}$. So we finish the proof. $\qquad \square$

By Theorem 2.3, for any $\rho > 0$, the approximation rate cannot be better than $\mathcal{O}\left(N^{-(2\alpha/d+\rho)}L^{-(2/\alpha+\rho)}\right)$, if we use FNNs in $\mathcal{NN}(\#\mathrm{input} = d; \ \mathrm{width} \leq N; \ \mathrm{depth} \leq L)$ to approximate functions in $\mathrm{Höld}([0,1]^d, \alpha, \lambda)$. By a similar argument, we can show that the approximation rate cannot be $\mathcal{O}\left(N^{-2\alpha/d}L^{-(2/\alpha+\rho)}\right)$ nor $\mathcal{O}\left(N^{-(2\alpha/d+\rho)}L^{-2\alpha/d}\right)$. Hence, the approximation rate in Theorem 1.1 is nearly tight.

3 Proof of Theorem 2.1

In this section, we will prove Theorem 2.1. We first present the key ideas in Section 3.1. Based on two propositions in Section 3.1, the detailed proof is presented in Section 3.2. Finally, the proofs of two propositions in Section 3.1 can be found in Section 3.3 and 3.4.
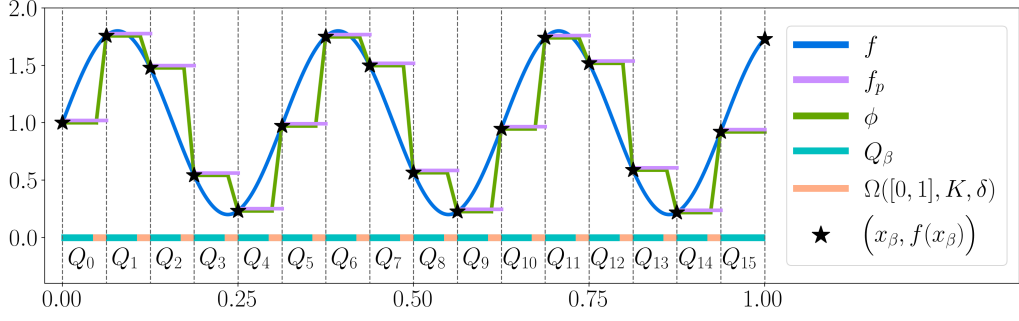
Figure 4: An illustration of f, f_p, ϕ, x_β, Q_β, and the trifling region $\Omega([0,1]^d, K, \delta)$ in the one-dimensional case for $\beta \in \{0, 1, \cdots, K-1\}^d$, where $K = N^2 L^2$ and $d = 1$ with $N = 2$ and $L = 2$. f is the target function; f_p is the piecewise constant function approximating f; ϕ is a function, implemented by a ReLU FNN, approximating f; and x_β is a representative of Q_β. The measure of the trifling region $\Omega([0,1]^d, K, \delta)$ can be arbitrarily small as we shall see in the proof of Theorem 1.1.

3.1 Key idea of proving Theorem 2.1

We will show that an almost piecewise constant function ϕ implemented by a ReLU FNN is enough to achieve the desired approximation rate in Theorem 1.1. Given an arbitrary $f \in C([0,1]^d)$, we introduce a piecewise constant function $f_p \approx f$ serving as an intermediate approximant in our construction in the sense that

$$f \approx f_p \text{ on } [0,1]^d, \quad \text{and} \quad f_p \approx \phi \text{ on } [0,1]^d \backslash \Omega([0,1]^d, K, \delta).$$

The approximation in $f \approx f_p$ is a simple and standard technique in constructive approximation. For example, given arbitrary N and L, uniformly partition $[0,1]^d$ into $\mathcal{O}(N^2 L^2)$ pieces and define f_p using this partition. Then the approximation error of $f_p \approx f$ scales like $\mathcal{O}(N^{-2/d} L^{-2/d})$. We will address the approximation in $f_p \approx \phi$ with the same error scaling and a limited budget of the FNN size, e.g., $\mathcal{O}(NL)$ neurons, based on the fact that f_p can be approximately implemented by a ReLU FNN in $[0,1]^d \backslash \Omega([0,1]^d, K, \delta)$, where $\Omega([0,1]^d, K, \delta)$ is the trifling region near the discontinuous locations of f_p with an arbitrarily small Lebesgue measure (see Figure 4 for an illustration). The introduction of the trifling region is to ease the construction of a deep ReLU FNN to implement the desired ϕ, which is a piecewise linear and continuous function, to approximate the discontinuous function f_p by removing the difficulty near discontinuous points, essentially smoothing f_p by restricting the approximation domain in $[0,1]^d \backslash \Omega([0,1]^d, K, \delta)$.

Now let us discuss the detailed steps of construction. First, divide $[0,1]^d$ into a union of important regions $\{Q_\beta\}_\beta$ and the trifling region $\Omega([0,1]^d, K, \delta)$, where each Q_β is associated with a representative $x_\beta \in Q_\beta$ such that $f(x_\beta) = f_p(x_\beta)$ for each index vector $\beta \in \{0, 1, \ldots, K-1\}^d$, where $K = \mathcal{O}(N^{2/d} L^{2/d})$ is the partition number per dimension (see Figure 6 for examples for $d = 1$ and $d = 2$). Next, we design a vector function $\boldsymbol{\Phi}_1(\boldsymbol{x})$ constructed via $\boldsymbol{\Phi}_1(\boldsymbol{x}) = \big(\phi_1(x_1), \phi_1(x_2), \cdots, \phi_1(x_d)\big)$ to project the whole cube Q_β to a d-dimensional index $\boldsymbol{\beta}$ for each $\boldsymbol{\beta}$, where each one-dimensional function ϕ_1 is a step function implemented by a ReLU FNN. The final step is to solve a point fitting problem. To be precise, we construct a function ϕ_2 implemented by a ReLU FNN to map $\boldsymbol{\beta}$ approximately to $f_p(\boldsymbol{x}_\beta) = f(\boldsymbol{x}_\beta)$. Then $\phi_2 \circ \boldsymbol{\Phi}_1(\boldsymbol{x}) = \phi_2(\boldsymbol{\beta}) \approx f_p(\boldsymbol{x}_\beta) = f(\boldsymbol{x}_\beta)$

13

for any $\boldsymbol{x} \in Q_{\boldsymbol{\beta}}$ and each $\boldsymbol{\beta}$, implying $\phi := \phi_2 \circ \boldsymbol{\Phi}_1 \approx f_p \approx f$ on $[0,1]^d \backslash \Omega([0,1]^d, K, \delta)$. We would like to point out that we only need to care about the values of ϕ_2 at a set of points $\{0, 1, \cdots, K-1\}^d$ during construction, since we design $\phi = \phi_2 \circ \boldsymbol{\Phi}_1$ via a function composition as shown in Figure 5. That means we are unnecessary to care about the values of ϕ_2 outsize of the set of points $\{0, 1, \cdots, K-1\}^d$, which is a key point while designing a ReLU FNN to implement ϕ_2 as we shall see later. This reveals the power of function compositions in a sense.
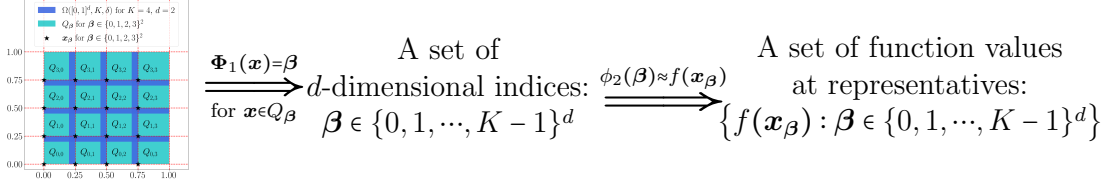

Figure 5: An illustration of the desired function $\phi = \phi_2 \circ \boldsymbol{\Phi}_1$. Note that $\phi \approx f$ on $[0,1]^d \backslash \Omega([0,1]^d, K, \delta)$, since $\phi(\boldsymbol{x}) = \phi_2 \circ \boldsymbol{\Phi}_1(\boldsymbol{x}) = \phi_2(\boldsymbol{\beta}) \approx f(\boldsymbol{x}_{\boldsymbol{\beta}})$ for any $\boldsymbol{x} \in Q_{\boldsymbol{\beta}}$ and each $\boldsymbol{\beta} \in \{0, 1, \cdots, K-1\}^d$.

Finally, we discuss how to implement $\boldsymbol{\Phi}_1$ and ϕ_2 by deep ReLU FNNs with width $\mathcal{O}(N)$ and depth $\mathcal{O}(L)$ using two propositions as we shall prove in Section 3.3 and 3.4 later. We first construct a ReLU FNN with desired width and depth by Proposition 3.1 to implement a one-dimensional step function ϕ_1. Then $\boldsymbol{\Phi}_1$ can be attained via defining

$$\boldsymbol{\Phi}_1(\boldsymbol{x}) = \big(\phi_1(x_1), \phi_1(x_2), \cdots, \phi_1(x_d)\big), \quad \text{for any } \boldsymbol{x} = (x_1, x_2, \cdots, x_d) \in \mathbb{R}^d.$$

Proposition 3.1. *For any $N, L, d \in \mathbb{N}^+$ and $\delta \in (0, \frac{1}{3K}]$ with $K = \lfloor N^{1/d} \rfloor^2 \lfloor L^{2/d} \rfloor$, there exists a one-dimensional function ϕ implemented by a ReLU FNN with width $4\lfloor N^{1/d} \rfloor + 3$ and depth $4L + 5$ such that*

$$\phi(x) = k, \quad \text{if } x \in \big[\tfrac{k}{K}, \tfrac{k+1}{K} - \delta \cdot 1_{\{k \le K-2\}} \big] \text{ for } k = 0, 1, \cdots, K-1.$$

The construction of ϕ_2 is a direct result of Proposition 3.2 below, the proof of which relies on the bit extraction technique in [5].

Proposition 3.2. *Given any $\varepsilon > 0$ and arbitrary $N, L, J \in \mathbb{N}^+$ with $J \le N^2 L^2$, assume $\{y_j \ge 0 : j = 0, 1, \cdots, J-1\}$ is a sample set with $|y_j - y_{j-1}| \le \varepsilon$ for $j = 1, 2, \cdots, J-1$. Then there exists $\phi \in \mathcal{NN}(\#\text{input} = 1; \text{width} \le 12N + 8; \text{depth} \le 4L + 9)$ such that*

(i) $|\phi(j) - y_j| \le \varepsilon$ for $j = 0, 1, \cdots, J-1$;

(ii) $0 \le \phi(x) \le \max\{y_j : j = 0, 1, \cdots, J-1\}$ for any $x \in \mathbb{R}$.

With the above propositions ready, let us prove Theorem 2.1 in Section 3.2. We further assume that $\omega_f(r) > 0$ for any $r > 0$, excluding a simple case when f is a constant function.

3.2 Proof of Theorem 2.1

We essentially construct an almost piecewise constant function implemented by a ReLU FNN with $\mathcal{O}(NL)$ neurons to approximate f. It is clear that $|f(\boldsymbol{x}) - f(\mathbf{0})| \leq \omega_f(\sqrt{d})$ for any $\boldsymbol{x} \in [0,1]^d$. Define $\widetilde{f} = f - f(\mathbf{0}) + \omega_f(\sqrt{d})$, then $0 \leq \widetilde{f}(\boldsymbol{x}) \leq 2\omega_f(\sqrt{d})$ for any $\boldsymbol{x} \in [0,1]^d$. Let $M = N^2 L$, $K = \lfloor N^{1/d} \rfloor^2 \lfloor L^{2/d} \rfloor$, and δ be an arbitrary number in $(0, \frac{1}{3K}]$.

The proof can be divided into four steps as follows:

1. Divide $[0,1]^d$ into a union of sub-cubes $\{Q_{\boldsymbol{\beta}}\}_{\boldsymbol{\beta} \in \{0,1,\cdots,K-1\}^d}$ and the trifling region $\Omega([0,1]^d, K, \delta)$, and denote $\boldsymbol{x}_{\boldsymbol{\beta}}$ as the vertex of $Q_{\boldsymbol{\beta}}$ with minimum $\|\cdot\|_1$ norm

2. Construct a sub-network to implement a vector function $\boldsymbol{\Phi}_1$ projecting the whole cube $Q_{\boldsymbol{\beta}}$ to the d-dimensional index $\boldsymbol{\beta}$ for each $\boldsymbol{\beta}$, i.e., $\boldsymbol{\Phi}_1(\boldsymbol{x}) = \boldsymbol{\beta}$ for all $\boldsymbol{x} \in Q_{\boldsymbol{\beta}}$;

3. Construct a sub-network to implement a function ϕ_2 mapping the index $\boldsymbol{\beta}$ approximately to $\widetilde{f}(\boldsymbol{x}_{\boldsymbol{\beta}})$. This core step can be further divided into three sub-steps:

   3.1. Construct a sub-network to implement ψ_1 bijectively mapping the index set $\{0,1,\cdots,K-1\}^d$ to an auxiliary set $\mathcal{A}_1 \subseteq \left\{ \frac{j}{2K^d} : j = 0,1,\cdots,2K^d \right\}$ defined later, see Figure 7 for a illustration;

   3.2. Determine a continuous piecewise linear function g with a set of breakpoints $\mathcal{A}_1 \cup \mathcal{A}_2 \cup \{1\}$ satisfying: 1) assign the values of g at breakpoints in $\mathcal{A}_1$ based on $\{\widetilde{f}(\boldsymbol{x}_{\boldsymbol{\beta}})\}_{\boldsymbol{\beta}}$, i.e., $g \circ \psi_1(\boldsymbol{\beta}) = \widetilde{f}(\boldsymbol{x}_{\boldsymbol{\beta}})$; 2) assign the values of g at breakpoints in $\mathcal{A}_2 \cup \{1\}$ to reduce the variation of g for applying Proposition 3.2;

   3.3. Apply Proposition 3.2 to construct a sub-network to implement a function ψ_2 approximating g well on $\mathcal{A}_1 \cup \mathcal{A}_2 \cup \{1\}$. Then the desired function ϕ_2 is given by $\phi_2 = \psi_2 \circ \psi_1$ satisfying $\phi_2(\boldsymbol{\beta}) = \psi_2 \circ \psi_1(\boldsymbol{\beta}) \approx g \circ \psi_1(\boldsymbol{\beta}) = \widetilde{f}(\boldsymbol{x}_{\boldsymbol{\beta}})$;

4. Construct the final target network to implement the desired function ϕ such that $\phi(\boldsymbol{x}) = \phi_2 \circ \boldsymbol{\Phi}_1(\boldsymbol{x}) + f(\mathbf{0}) - \omega_f(\sqrt{d}) \approx \widetilde{f}(\boldsymbol{x}_{\boldsymbol{\beta}}) + f(\mathbf{0}) - \omega_f(\sqrt{d}) = f(\boldsymbol{x}_{\boldsymbol{\beta}})$ for $\boldsymbol{x} \in Q_{\boldsymbol{\beta}}$.

The details of these steps can be found below.

Step 1: Divide $[0,1]^d$ into $\{Q_{\boldsymbol{\beta}}\}_{\boldsymbol{\beta} \in \{0,1,\cdots,K-1\}^d}$ and $\Omega([0,1]^d, K, \delta)$.

Define $\boldsymbol{x}_{\boldsymbol{\beta}} \coloneqq \boldsymbol{\beta}/K$ and

$$Q_{\boldsymbol{\beta}} = \left\{ \boldsymbol{x} \coloneqq (x_1, \cdots, x_d) \in [0,1]^d : x_i \in \left[\frac{\beta_i}{K}, \frac{\beta_i+1}{K} - \delta \cdot 1_{\{\beta_i \leq K-2\}} \right], \ i = 1, \cdots, d \right\}$$

for each d-dimensional index $\boldsymbol{\beta} = (\beta_1, \beta_2, \cdots, \beta_d) \in \{0,1,\cdots,K-1\}^d$. Recall that $\Omega([0,1]^d, K, \delta)$ is the trifling region defined in Equation (2.1). Apparently, $\boldsymbol{x}_{\boldsymbol{\beta}}$ is the vertex of $Q_{\boldsymbol{\beta}}$ with minimum $\|\cdot\|_1$ norm and

$$[0,1]^d = \left(\cup_{\boldsymbol{\beta} \in \{0,1,\cdots,K-1\}^d} Q_{\boldsymbol{\beta}} \right) \cup \Omega([0,1]^d, K, \delta),$$

see Figure 6 for illustrations.

Step 2: Construct $\boldsymbol{\Phi}_1$ mapping $\boldsymbol{x} \in Q_{\boldsymbol{\beta}}$ to $\boldsymbol{\beta}$.
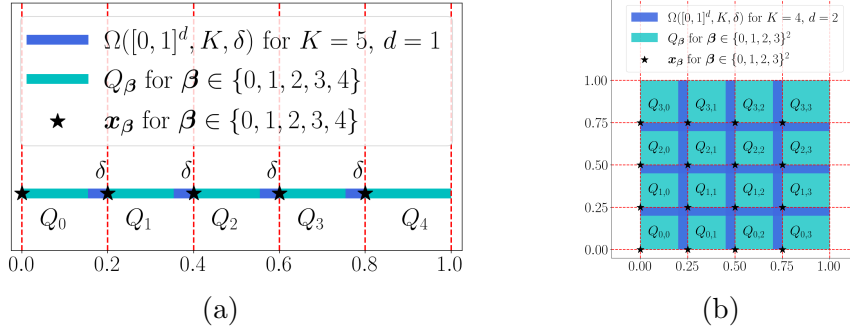
Figure 6: Illustrations of $\Omega([0,1]^d, K, \delta)$, $Q_{\boldsymbol{\beta}}$, and $\boldsymbol{x}_{\boldsymbol{\beta}}$ for $\boldsymbol{\beta} \in \{0, 1, \cdots, K-1\}^d$. (a) $K = 5$ and $d = 1$. (b) $K = 4$ and $d = 2$.

By Proposition 3.1, there exists $\phi_1 \in \mathcal{NN}(\text{width} \le 4\lfloor N^{1/d} \rfloor + 3; \text{ depth} \le 4L + 5)$ such that
$$\phi_1(x) = \tfrac{k}{K}, \quad \text{if } x \in \left[\tfrac{k}{K}, \tfrac{k+1}{K} - \delta \cdot 1_{\{k \le K-2\}}\right] \text{ for } k = 0, 1, \cdots, K-1.$$
It follows that $\phi_1(x_i) = \beta_i$ if $\boldsymbol{x} = (x_1, x_2, \cdots, x_d) \in Q_{\boldsymbol{\beta}}$ for each $\boldsymbol{\beta} = (\beta_1, \beta_2, \cdots, \beta_d)$.

By defining
$$\boldsymbol{\Phi}_1(\boldsymbol{x}) \coloneqq \left(\phi_1(x_1), \phi_1(x_2), \cdots, \phi_1(x_d)\right), \quad \text{for any } \boldsymbol{x} = (x_1, x_2, \cdots, x_d) \in \mathbb{R}^d,$$

we have $\boldsymbol{\Phi}_1(\boldsymbol{x}) = \boldsymbol{\beta}$ if $\boldsymbol{x} \in Q_{\boldsymbol{\beta}}$ for $\boldsymbol{\beta} \in \{0, 1, \cdots, K-1\}^d$.

Step 3: Construct ϕ_2 mapping $\boldsymbol{\beta}$ approximately to $\widetilde{f}(\boldsymbol{x}_{\boldsymbol{\beta}})$.

The construction of the sub-network implementing ϕ_2 is essentially based on Proposition 3.2. To meet the requirements of applying Proposition 3.2, we first define two auxiliary set $\mathcal{A}_1$ and $\mathcal{A}_2$ as
$$\mathcal{A}_1 \coloneqq \left\{ \tfrac{i}{K^{d-1}} + \tfrac{k}{2K^d} : i = 0, 1, \cdots, K^{d-1} \text{ and } k = 0, 1, \cdots, K-1 \right\}$$

and
$$\mathcal{A}_2 \coloneqq \left\{ \tfrac{i}{K^{d-1}} + \tfrac{1}{2K^{d-1}} + \tfrac{k}{2K^d} : i = 0, 1, \cdots, K^{d-1} \text{ and } k = 0, 1, \cdots, K-1 \right\}.$$

Clearly, $\mathcal{A}_1 \cup \mathcal{A}_2 \cup \{1\} = \left\{ \tfrac{j}{2K^d} : j = 0, 1, \cdots, 2K^d \right\}$ and $\mathcal{A}_1 \cap \mathcal{A}_2 = \varnothing$. See Figure 6 for an illustration of $\mathcal{A}_1$ and $\mathcal{A}_2$. Next, we further divide this step into three sub-steps.

Step 3.1: Construct ψ_1 bijectively mapping $\{0, 1, \cdots, K-1\}^d$ to $\mathcal{A}_1$.

Inspired by the binary representation, we define
$$\psi_1(\boldsymbol{x}) \coloneqq \frac{x_d}{2K^d} + \sum_{i=1}^{d-1} \frac{x_i}{K^i}, \quad \text{for any } \boldsymbol{x} = (x_1, x_2, \cdots, x_d) \in \mathbb{R}^d. \tag{3.1}$$

Then ψ_1 is a linear function bijectively mapping the index set $\{0, 1, \cdots, K-1\}^d$ to
$$\left\{ \frac{\beta_d}{2K^d} + \sum_{i=1}^{d-1} \frac{\beta_i}{K^i} : \boldsymbol{\beta} \in \{0, 1, \cdots, K-1\}^d \right\}$$
$$= \left\{ \tfrac{i}{K^{d-1}} + \tfrac{k}{2K^d} : i = 0, 1, \cdots, K^{d-1} \text{ and } k = 0, 1, \cdots, K-1 \right\} = \mathcal{A}_1.$$
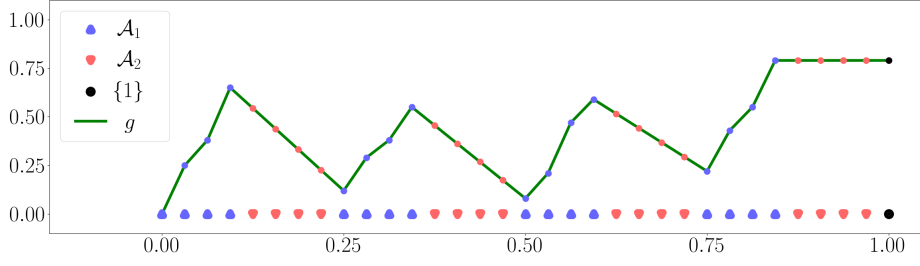
16

Figure 7: An illustration of $\mathcal{A}_1$, $\mathcal{A}_2$, $\{1\}$, and g for $d = 2$ and $K = 4$.

Step 3.2: Construct g to satisfy $g \circ \psi_1(\boldsymbol{\beta}) = \widetilde{f}(\boldsymbol{x}_{\boldsymbol{\beta}})$ and to meet the requirements of applying Proposition 3.2.

Let $g : [0, 1] \to \mathbb{R}$ be a continuous piecewise linear function with a set of breakpoints $\left\{ \frac{j}{2K^d} : j = 0, 1, \cdots, 2K^d \right\} = \mathcal{A}_1 \cup \mathcal{A}_2 \cup \{1\}$ and the values of g at these breakpoints satisfy the following properties:

- The values of g at the breakpoints in $\mathcal{A}_1$ are set as

$$g(\psi_1(\boldsymbol{\beta})) = \widetilde{f}(\boldsymbol{x}_{\boldsymbol{\beta}}), \quad \text{for any } \boldsymbol{\beta} \in \{0, 1, \cdots, K-1\}^d; \tag{3.2}$$

- At the breakpoint 1, let $g(1) = \widetilde{f}(\mathbf{1})$, where $\mathbf{1} = (1, 1, \cdots, 1) \in \mathbb{R}^d$;

- The values of g at the breakpoints in $\mathcal{A}_2 \cup \{1\}$ are assigned to reduce the variation of g, which is a requirement of applying Proposition 3.2. Note that

$$\left\{ \frac{i}{K^{d-1}} - \frac{K+1}{2K^d}, \ \frac{i}{K^{d-1}} \right\} \subseteq \mathcal{A}_1 \cup \{1\}, \quad \text{for } i = 1, 2, \cdots, K^{d-1},$$

  implying the values of g at $\frac{i}{K^{d-1}} - \frac{K+1}{2K^d}$ and $\frac{i}{K^{d-1}}$ have been assigned for $i = 1, 2, \cdots, K^{d-1}$. Thus, the values of g at the breakpoints in $\mathcal{A}_2$ can be successfully assigned by letting g linear on each interval $\left[\frac{i}{K^{d-1}} - \frac{K+1}{2K^d}, \ \frac{i}{K^{d-1}} \right]$ for $i = 1, 2, \cdots, K^{d-1}$, since $\mathcal{A}_2 \subseteq \cup_{i=1}^{K^{d-1}} \left[\frac{i}{K^{d-1}} - \frac{K+1}{2K^d}, \ \frac{i}{K^{d-1}} \right]$.

Apparently, such a function g exists (see Figure 7 for an example) and satisfies

$$\left| g\left(\frac{j}{2K^d} \right) - g\left(\frac{j-1}{2K^d} \right) \right| \leq \max \left\{ \omega_f\left(\frac{1}{K} \right), \omega_f(\sqrt{d})/K \right\} \leq \omega_f\left(\frac{\sqrt{d}}{K} \right), \quad \text{for } j = 1, 2, \cdots, 2K^d,$$

and

$$0 \leq g\left(\frac{j}{2K^d} \right) \leq 2\omega_f(\sqrt{d}), \quad \text{for } j = 0, 1, \cdots, 2K^d.$$

Step 3.3: Construct ψ_2 approximating g well on $\mathcal{A}_1 \cup \mathcal{A}_2 \cup \{1\}$.

Since $2K^d = 2\left(\lfloor N^{1/d} \rfloor^2 \lfloor L^{2/d} \rfloor \right)^d \leq 2\left(N^2 L^2 \right) \leq N^2 \widetilde{L}^2$, where $\widetilde{L} = 2L$, by Proposition 3.2, there exists $\widetilde{\psi}_2 \in \mathcal{NN}(\#\text{input} = 1; \text{ width} \leq 12N + 8; \text{ depth} \leq 4\widetilde{L} + 9) = \mathcal{NN}(\#\text{input} = 1; \text{ width} \leq 12N + 8; \text{ depth} \leq 8L + 9)$ such that

$$\left| \widetilde{\psi}_2(j) - g\left(\frac{j}{2K^d} \right) \right| \leq \omega_f\left(\frac{\sqrt{d}}{K} \right), \quad \text{for } j = 0, 1, \cdots, 2K^d - 1,$$

17

and

$$0 \le \widetilde{\psi}_2(x) \le \max\{g(\tfrac{j}{2K^d}) : j = 0, 1, \cdots, 2K^d - 1\} \le 2\omega_f(\sqrt{d}), \quad \text{for any } x \in \mathbb{R}.$$

By defining $\psi_2(x) \coloneqq \widetilde{\psi}_2(2K^d x)$ for any $x \in \mathbb{R}$, we have $\psi_2 = \psi_2 \circ \psi_1 \in \mathcal{NN}(\#\text{input} = 1; \text{ width} \le 12N + 8; \text{ depth} \le 8L + 9)$,

$$0 \le \psi_2(x) = \widetilde{\psi}_2(2K^d x) \le 2\omega_f(\sqrt{d}), \quad \text{for any } x \in \mathbb{R}, \tag{3.3}$$

and

$$|\psi_2(\tfrac{j}{2K^d}) - g(\tfrac{j}{2K^d})| = |\widetilde{\psi}_2(j) - g(\tfrac{j}{2K^d})| \le \omega_f(\tfrac{\sqrt{d}}{K}), \quad \text{for } j = 0, 1, \cdots, 2K^d - 1. \tag{3.4}$$

Let us end Step 3 by defining the desired function ϕ_2 as $\phi_2 \coloneqq \psi_2 \circ \psi_1$. Note that $\psi_1 : \mathbb{R}^d \to \mathbb{R}$ is a linear function and $\psi_2 \in \mathcal{NN}(\#\text{input} = 1; \text{ width} \le 12N + 8; \text{ depth} \le 8L + 9)$. Thus, $\phi_2 \in \mathcal{NN}(\#\text{input} = d; \text{ width} \le 12N + 8; \text{ depth} \le 8L + 9)$. By Equation (3.2) and (3.4), we have

$$|\phi_2(\boldsymbol{\beta}) - \widetilde{f}(\boldsymbol{x_\beta})| = |\psi_2(\psi_1(\boldsymbol{\beta})) - g(\psi_1(\boldsymbol{\beta}))| \le \omega_f(\tfrac{\sqrt{d}}{K}), \tag{3.5}$$

for any $\boldsymbol{\beta} \in \{0, 1, \cdots, K - 1\}^d$. Equation (3.3) and $\phi_2 = \psi_2 \circ \psi_1$ implies

$$0 \le \phi_2(\boldsymbol{x}) \le 2\omega_f(\sqrt{d}), \quad \text{for any } \boldsymbol{x} \in \mathbb{R}^d. \tag{3.6}$$

Step 4: Construct the final network to implement the desired function ϕ.

Define $\phi \coloneqq \phi_2 \circ \boldsymbol{\Phi}_1 + f(\boldsymbol{0}) - \omega_f(\sqrt{d})$. Since $\phi_1 \in \mathcal{NN}(\text{width} \le 4\lfloor N^{1/d} \rfloor + 3; \text{ depth} \le 4L + 5)$, we have $\boldsymbol{\Phi}_1 \in \mathcal{NN}(\#\text{input} = d; \text{ width} \le 4d\lfloor N^{1/d} \rfloor + 3d; \text{ depth} \le 4L + 5; \#\text{output} = d)$. Note that $\phi_2 \in \mathcal{NN}(\#\text{input} = d; \text{ width} \le 12N + 8; \text{ depth} \le 8L + 9)$. Thus, $\phi = \phi_2 \circ \boldsymbol{\Phi}_1 + f(\boldsymbol{0}) - \omega_f(\sqrt{d})$ is in

$$\mathcal{NN}(\text{width} \le \max\{4d\lfloor N^{1/d} \rfloor + 3d, 12N + 8\}; \text{ depth} \le (4L + 5) + (8L + 9) = 12L + 14).$$

Now let us estimate the approximation error. Note that $f = \widetilde{f} + f(\boldsymbol{0}) - \omega_f(\sqrt{d})$. By Equation (3.5), for any $\boldsymbol{x} \in Q_{\boldsymbol{\beta}}$ and $\boldsymbol{\beta} \in \{0, 1, \cdots, K - 1\}^d$, we have

$$\begin{aligned}
|f(\boldsymbol{x}) - \phi(\boldsymbol{x})| = |\widetilde{f}(\boldsymbol{x}) - \phi_2(\boldsymbol{\Phi}_1(\boldsymbol{x}))| &= |\widetilde{f}(\boldsymbol{x}) - \phi_2(\boldsymbol{\beta})| \\
&\le |\widetilde{f}(\boldsymbol{x}) - \widetilde{f}(\boldsymbol{x_\beta})| + |\widetilde{f}(\boldsymbol{x_\beta}) - \phi_2(\boldsymbol{\beta})| \\
&\le \omega_f(\tfrac{\sqrt{d}}{K}) + \omega_f(\tfrac{\sqrt{d}}{K}) \le 2\omega_f(8\sqrt{d}N^{-2/d}L^{-2/d}),
\end{aligned}$$

where the last inequality comes from the fact $K = \lfloor N^{1/d} \rfloor^2 \lfloor L^{2/d} \rfloor \ge \tfrac{N^{2/d}L^{2/d}}{8}$ for any $N, L \in \mathbb{N}^+$. Recall the fact $\omega_f(nr) \le n\omega_f(r)$ for any $n \in \mathbb{N}^+$ and $r \in [0, \infty)$. Therefore, for any $\boldsymbol{x} \in \cup_{\boldsymbol{\beta} \in \{0, 1, \cdots, K-1\}^d} Q_{\boldsymbol{\beta}} \subseteq [0, 1]^d \backslash \Omega([0, 1]^d, K, \delta)$, we have

$$\begin{aligned}
|f(\boldsymbol{x}) - \phi(\boldsymbol{x})| \le 2\omega_f(8\sqrt{d}N^{-2/d}L^{-2/d}) &\le 2\lceil 8\sqrt{d} \rceil \omega_f(N^{-2/d}L^{-2/d}) \\
&\le 18\sqrt{d}\,\omega_f(N^{-2/d}L^{-2/d}).
\end{aligned}$$

It remains to show the upper bound of ϕ. By Equation (3.6) and $\phi = \phi_2 \circ \boldsymbol{\Phi}_1 + f(\boldsymbol{0}) - \omega_f(\sqrt{d})$, it holds that $\|\phi\|_{L^\infty(\mathbb{R}^d)} \le |f(\boldsymbol{0})| + \omega_f(\sqrt{d})$. Thus, we finish the proof.

3.3 Proof of Proposition 3.1

Lemma 3.3. *For any $N_1, N_2 \in \mathbb{N}^+$, given $N_1(N_2 + 1) + 1$ samples $(x_i, y_i) \in \mathbb{R}^2$ with $x_0 < x_1 < \cdots < x_{N_1(N_2+1)}$ and $y_i \geq 0$ for $i = 0, 1, \cdots, N_1(N_2+1)$, there exists $\phi \in \mathcal{NN}(\#\text{input} = 1; \text{widthvec} = [2N_1, 2N_2 + 1]; \#\text{output} = 1)$ satisfying the following conditions.*

  (i) $\phi(x_i) = y_i$ *for* $i = 0, 1, \cdots, N_1(N_2 + 1)$;

  (ii) ϕ *is linear on each interval* $[x_{i-1}, x_i]$ *for* $i \notin \{(N_2 + 1)j : j = 1, 2, \cdots, N_1\}$.

In fact, Proposition 3.3 is a part of Lemma 2.2 in [62]. For the purpose of being self-contained, we present it as follows.

Lemma (Lemma 2.2 of [62])**.** *For any $m, n \in \mathbb{N}^+$, given any $m(n+1)+1$ samples $(x_i, y_i) \in \mathbb{R}^2$ with $x_0 < x_1 < x_2 < \cdots < x_{m(n+1)}$ and $y_i \geq 0$ for $i = 0, 1, \cdots, m(n + 1)$, there exists $\phi \in \mathcal{NN}(\#\text{input} = 1; \text{widthvec} = [2m, 2n + 1]; \#\text{output} = 1)$ satisfying the following conditions.*

  (i) $\phi(x_i) = y_i$ *for* $i = 0, 1, \cdots, m(n + 1)$;

  (ii) ϕ *is linear on each interval* $[x_{i-1}, x_i]$ *for* $i \notin \{(n + 1)j : j = 1, 2, \cdots, m\}$;

  (iii) $\displaystyle \sup_{x \in [x_0, x_{m(n+1)}]} |\phi(x)| \leq 3 \max_{i \in \{0, 1, \cdots, m(n+1)\}} y_i \prod_{k=1}^{n} \left(1 + \frac{\max\{x_{j(n+1)+n} - x_{j(n+1)+k-1} : j=0,1,\cdots,m-1\}}{\min\{x_{j(n+1)+k} - x_{j(n+1)+k-1} : j=0,1,\cdots,m-1\}}\right).$

Lemma 3.4. *Given any $N, L, d \in \mathbb{N}^+$, it holds that*

$$\mathcal{NN}(\#\text{input} = d; \text{widthvec} = [N, NL]; \#\text{output} = 1)$$
$$\subseteq \mathcal{NN}(\#\text{input} = d; \text{width} \leq 2N + 2; \text{depth} \leq L + 1; \#\text{output} = 1).$$

Proof. The key idea to prove Proposition 3.4 is to re-assemble $\mathcal{O}(L)$ sub-FNNs in the shallower FNN in the left of Figure 8 to form a deeper one with width $\mathcal{O}(N)$ and depth $\mathcal{O}(L)$ on the right of Figure 8.
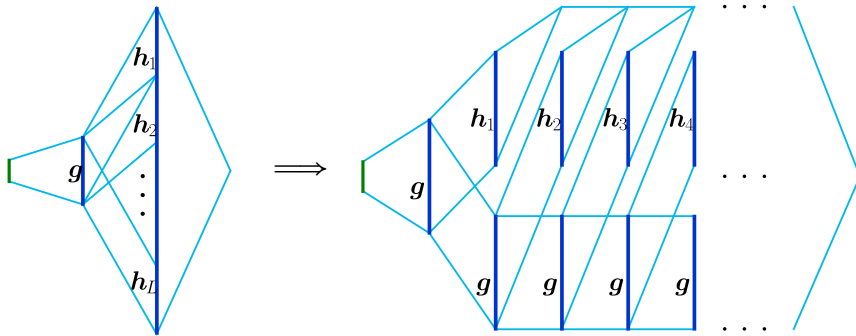

Figure 8: An illustration of the main idea to prove Lemma 3.4.

For any $\phi \in \mathcal{NN}(\#\text{input} = d; \text{widthvec} = [N, NL]; \#\text{output} = 1)$, ϕ can be implemented by a ReLU FNN described as

$$x \xrightarrow[\sigma]{W_0, \, b_0} g \xrightarrow[\sigma]{W_1, \, b_1} h \xrightarrow{W_2, \, b_2} \phi(x),$$

where $\boldsymbol{g}$ and $\boldsymbol{h}$ are the output of the first hidden layer and the second hidden layer, respectively. Note that

$$\boldsymbol{g} = \sigma(\boldsymbol{W}_0 \boldsymbol{x} + \boldsymbol{b}_0), \quad \boldsymbol{h} = \sigma(\boldsymbol{W}_1 \boldsymbol{g} + \boldsymbol{b}_1), \quad \text{and} \quad \phi(\boldsymbol{x}) = \boldsymbol{W}_2 \boldsymbol{h} + \boldsymbol{b}_2. \tag{3.7}$$

We can evenly divide $\boldsymbol{h} \in \mathbb{R}^{NL}$, $\boldsymbol{b}_1 \in \mathbb{R}^{NL}$, $\boldsymbol{W}_1 \in \mathbb{R}^{NL \times N}$, and $\boldsymbol{W}_2 \in \mathbb{R}^{1 \times NL}$ into L parts as follows:

$$\boldsymbol{h} = \begin{bmatrix} \boldsymbol{h}_1 \\ \boldsymbol{h}_2 \\ \vdots \\ \boldsymbol{h}_L \end{bmatrix}, \quad \boldsymbol{b}_1 = \begin{bmatrix} \boldsymbol{b}_{1,1} \\ \boldsymbol{b}_{1,2} \\ \vdots \\ \boldsymbol{b}_{1,L} \end{bmatrix}, \quad \boldsymbol{W}_1 = \begin{bmatrix} \boldsymbol{W}_{1,1} \\ \boldsymbol{W}_{1,2} \\ \vdots \\ \boldsymbol{W}_{1,L} \end{bmatrix},$$

and $\boldsymbol{W}_2 = [\boldsymbol{W}_{2,1}, \boldsymbol{W}_{2,2}, \cdots, \boldsymbol{W}_{2,L}]$, where $\boldsymbol{h}_\ell$, $\boldsymbol{b}_{1,\ell} \in \mathbb{R}^N$, $\boldsymbol{W}_{1,\ell} \in \mathbb{R}^{N \times N}$, and $\boldsymbol{W}_{2,\ell} \in \mathbb{R}^{1 \times N}$ for $\ell = 1, 2, \cdots, L$.

Define

$$s_0 \coloneqq 0, \quad \text{and} \quad s_\ell \coloneqq \sum_{j=1}^{\ell} \boldsymbol{W}_{2,j} \cdot \boldsymbol{h}_j, \quad \text{for } \ell = 1, 2, \cdots, L.$$

Then $\phi(\boldsymbol{x}) = \boldsymbol{W}_2 \cdot \boldsymbol{h} + \boldsymbol{b}_2 = s_L + \boldsymbol{b}_2$ and

$$s_\ell = s_{\ell-1} + \boldsymbol{W}_{2,\ell} \cdot \boldsymbol{h}_\ell, \quad \text{for } \ell = 1, 2, \cdots, L. \tag{3.8}$$

Hence, it is easy to check that ϕ can be also implemented by the deep network shown in Figure 9. It is clear that the network has the architecture of Figure 9 is with width
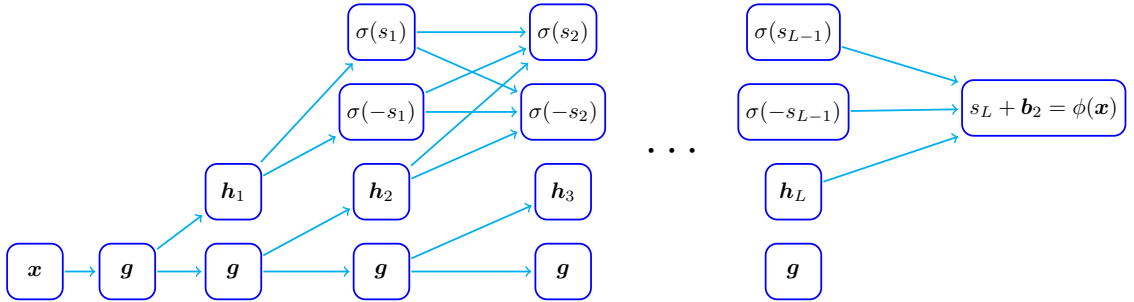

Figure 9: A illustration of the desired network based on Equation (3.7) and (3.8), and the fact $x = \sigma(x) - \sigma(-x)$ for any $x \in \mathbb{R}$. We omit the activation function (σ) if the input is non-negative.

$2N + 2$ and depth $L + 1$. So, we finish the proof. $\qquad\square$

With Lemma 3.3 and 3.4 in hand, we are ready to present the detailed proof of Proposition 3.1.

Proof of Proposition 3.1. We divide the proof into two cases: $d = 1$ and $d \geq 2$.

Case 1: $d = 1$.

In this case, $K = \lfloor N^{1/d} \rfloor^2 \lfloor L^{2/d} \rfloor = N^2 L^2$. Denote $M = N^2 L$ and consider the sample set

$$\left\{ (1, M-1), (2, 0) \right\} \cup \left\{ \left(\tfrac{m}{M}, m \right) : m = 0, 1, \cdots, M-1 \right\} \cup \left\{ \left(\tfrac{m+1}{M} - \delta, m \right) : m = 0, 1, \cdots, M-2 \right\}.$$

Its size is $2M + 1 = N \cdot \big((2NL - 1) + 1\big) + 1$. By Lemma 3.3 (set $N_1 = N$ and $N_2 = 2NL - 1$ therein), there exists $\phi_1 \in \mathcal{NN}(\text{widthvec} = [2N, 2(2NL - 1) + 1]) = \mathcal{NN}(\text{widthvec} = [2N, 4NL - 1])$ such that

- $\phi_1\big(\frac{M-1}{M}\big) = \phi_1(1) = M - 1$ and $\phi_1\big(\frac{m}{M}\big) = \phi_1\big(\frac{m+1}{M} - \delta\big) = m$ for $m = 0, 1, \cdots, M - 2$;

- ϕ_1 is linear on $\big[\frac{M-1}{M}, 1\big]$ and each interval $\big[\frac{m}{M}, \frac{m+1}{M} - \delta\big]$ for $m = 0, 1, \cdots, M - 2$.

Then
$$\phi_1(x) = m, \quad \text{if } x \in \big[\tfrac{m}{M}, \tfrac{m+1}{M} - \delta \cdot 1_{\{m \leq M-2\}}\big], \quad \text{for } m = 0, 1, \cdots, M - 1. \qquad (3.9)$$

Now consider the another sample set

$$\big\{(\tfrac{1}{M}, L - 1), (2, 0)\big\} \cup \big\{(\tfrac{\ell}{ML}, \ell) : \ell = 0, 1, \cdots, L - 1\big\} \cup \big\{(\tfrac{\ell+1}{ML} - \delta, \ell) : \ell = 0, 1, \cdots, L - 2\big\}.$$

Its size is $2L + 1 = 1 \cdot \big((2L - 1) + 1\big) + 1$. By Lemma 3.3 (set $N_1 = 1$ and $N_2 = 2L - 1$ therein), there exists $\phi_2 \in \mathcal{NN}(\text{widthvec} = [2, 2(2L - 1) + 1]) = \mathcal{NN}(\text{widthvec} = [2, 4L - 1])$ such that

- $\phi_2\big(\frac{L-1}{ML}\big) = \phi_2\big(\frac{1}{M}\big) = L - 1$ and $\phi_2\big(\frac{\ell}{ML}\big) = \phi_2\big(\frac{\ell+1}{ML} - \delta\big) = \ell$ for $\ell = 0, 1, \cdots, L - 2$;

- ϕ_2 is linear on $\big[\frac{L-1}{ML}, \frac{1}{M}\big]$ and each interval $\big[\frac{\ell}{ML}, \frac{\ell+1}{ML} - \delta\big]$ for $\ell = 0, 1, \cdots, L - 2$.

It follows that, for $m = 0, 1, \cdots, M - 1$ and $\ell = 0, 1, \cdots, L - 1$,

$$\phi_2\big(x - \tfrac{m}{M}\big) = \ell, \quad \text{for } x \in \big[\tfrac{mL+\ell}{ML}, \tfrac{mL+\ell+1}{ML} - \delta \cdot 1_{\{\ell \leq L-2\}}\big]. \qquad (3.10)$$

The fact $K = ML$ implies each $k \in \{0, 1, \cdots, K - 1\}$ can be unique represented by $k = mL + \ell$ for $m = 0, 1, \cdots, M - 1$ and $\ell = 0, 1, \cdots, L - 1$. Then the desired function ϕ can be implemented by a ReLU FNN shown in Figure 10. Clearly,

$$\phi(x) = k, \quad \text{if } x \in \big[\tfrac{k}{K}, \tfrac{k}{K} - \delta \cdot 1_{\{k \leq K-2\}}\big] \text{ for } k \in \{0, 1, \cdots, K - 1\}.$$

By Lemma 3.4, $\phi_1 \in \mathcal{NN}(\text{widthvec} = [2N, 4NL - 1]) \subseteq \mathcal{NN}(\text{width} \leq 4N + 2; \text{ depth} \leq 2L + 1)$ and $\phi_2 \in \mathcal{NN}(\text{widthvec} = [2, 4L - 1]) \subseteq \mathcal{NN}(\text{width} \leq 6; \text{ depth} \leq 2L + 1)$, implying $\phi \in \mathcal{NN}(\text{width} \leq \max\{4N + 2 + 1, 6 + 1\} = 4N + 3; \text{ depth} \leq (2L+1) + 2 + (2L+1) + 1 = 4L + 5)$. So we finish the proof for the case $d = 1$
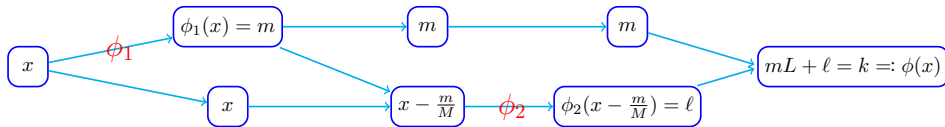

Figure 10: An illustration of the ReLU FNN implementing ϕ based on Equation (3.9) and (3.10) with $x \in \big[\frac{k}{K}, \frac{k}{K} - \delta \cdot 1_{\{k \leq K-2\}}\big] = \big[\frac{mL+\ell}{ML}, \frac{mL+\ell+1}{ML} - \delta \cdot 1_{\{m \leq M-2 \text{ or } \ell \leq L-2\}}\big]$, where $k = mL + \ell$ for $m = 0, 1, \cdots, M - 1$ and $\ell = 0, 1, \cdots, L - 1$. "$\phi_1$" and "$\phi_2$" near "$\longrightarrow$" represent the respective ReLU FNN implementing itself. We omit the activation function ReLU if the input of a neuron is non-negative.

Case 2: $d \geq 2$.

Now we consider the case when $d \geq 2$. Consider the sample set

$$\left\{(1, K-1), (2, 0)\right\} \cup \left\{\left(\tfrac{k}{K}, \tfrac{k}{K}\right) : k = 0, 1, \cdots, K-1\right\} \cup \left\{\left(\tfrac{k+1}{K} - \delta, k\right) : k = 0, 1, \cdots, K-2\right\},$$

whose size is $2K + 1 = \lfloor N^{1/d} \rfloor \big((2\lfloor N^{1/d} \rfloor \lfloor L^{2/d} \rfloor - 1) + 1 \big) + 1$. By Lemma 3.3 (set $N_1 = \lfloor N^{1/d} \rfloor$ and $N_2 = 2\lfloor N^{1/d} \rfloor \lfloor L^{2/d} \rfloor - 1$ therein), there exists ϕ in

$$\mathcal{NN}\big(\text{widthvec} = [2\lfloor N^{1/d} \rfloor, 2(2\lfloor N^{1/d} \rfloor \lfloor L^{2/d} \rfloor - 1) + 1]\big)$$
$$\subseteq \mathcal{NN}\big(\text{widthvec} = [2\lfloor N^{1/d} \rfloor, 4\lfloor N^{1/d} \rfloor \lfloor L^{2/d} \rfloor - 1]\big)$$

such that

- $\phi\left(\tfrac{K-1}{K}\right) = \phi(1) = K - 1$, and $\phi\left(\tfrac{k}{K}\right) = \phi\left(\tfrac{k+1}{K} - \delta\right) = k$ for $k = 0, 1, \cdots, K-2$;

- ϕ is linear on $\left[\tfrac{K-1}{K}, 1\right]$ and each interval $\left[\tfrac{k}{K}, \tfrac{k+1}{K} - \delta\right]$ for $k = 0, 1, \cdots, K-2$.

Then
$$\phi(x) = k, \quad \text{if } x \in \left[\tfrac{k}{K}, \tfrac{k+1}{K} - \delta \cdot 1_{\{k \leq K-2\}}\right] \text{ for } k = 0, 1, \cdots, K-1.$$

By Lemma 3.4,

$$\phi \in \mathcal{NN}\big(\text{widthvec} = [2\lfloor N^{1/d} \rfloor, 4\lfloor N^{1/d} \rfloor \lfloor L^{2/d} \rfloor - 1]\big)$$
$$= \mathcal{NN}\big(\text{width} \leq 4\lfloor N^{1/d} \rfloor + 2; \ \text{depth} \leq 2\lfloor L^{2/d} \rfloor + 1\big)$$
$$\subseteq \mathcal{NN}\big(\text{width} \leq 4\lfloor N^{1/d} \rfloor + 3; \ \text{depth} \leq 4L + 5\big).$$

which means we finish the proof for the case $d \geq 2$. $\qquad\square$

3.4 Proof of Proposition 3.2

The proof of Proposition 3.2 is based on the bit extraction technique in [5, 27]. In fact, we modify this technique to extract the sum of many bits rather than one bit and this modification can be summarized in Lemma 3.5 and 3.6 below.

Lemma 3.5. *For any $L \in \mathbb{N}^+$, there exists a function ϕ in*

$$\mathcal{NN}(\#\text{input} = 2; \ \text{width} \leq 7; \ \text{depth} \leq 2L + 1; \ \#\text{output} = 1)$$

such that, for any $\theta_1, \theta_2, \cdots, \theta_L \in \{0, 1\}$, we have

$$\phi(\text{bin}\,0.\theta_1\theta_2\cdots\theta_L, \ell) = \sum_{j=1}^{\ell} \theta_j, \quad \text{for } \ell = 1, 2, \cdots, L.$$

Proof. Given $\theta_1, \theta_2, \cdots, \theta_L \in \{0, 1\}$, define

$$\xi_j := \text{bin}\,0.\theta_j\theta_{j+1}\cdots\theta_L, \quad \text{for } j = 1, 2, \cdots, L$$

and

$$\mathcal{T}(x) := \begin{cases} 1, & x \geq 0, \\ 0, & x < 0. \end{cases}$$

22

Then we have
$$\theta_j = \mathcal{T}(\xi_j - 1/2), \quad \text{for } j = 1, 2, \cdots, L,$$
and
$$\xi_{j+1} = 2\xi_j - \theta_j, \quad \text{for } j = 1, 2, \cdots, L - 1.$$

I would like to point out that, by above two iteration equations, we can iteratively get $\xi_1, \theta_1, \xi_2, \theta_2, \cdots, \xi_L, \theta_L$ when ξ_1 is given. Based on this iteration idea, the rest proof can be divided into three steps.

Step 1: Simplify the iteration equations.

Note that $\mathcal{T}(x) = \sigma(x/\delta + 1) - \sigma(x/\delta)$ for any $x \notin (-\delta, 0)$. By setting $\delta = 1/2 - \sum_{j=2}^{L} 2^{-j} = 2^{-L}$, we have $\xi_j - 1/2 \notin (-\delta, 0)$ for all j, implying

$$
\begin{aligned}
\theta_j = \mathcal{T}(\xi_j - 1/2) &= \sigma\big((\xi_j - 1/2)/\delta + 1\big) - \sigma\big((\xi_j - 1/2)/\delta\big) \\
&= \sigma\big(\mathcal{L}(\xi_j) + 1\big) - \sigma\big(\mathcal{L}(\xi_j)\big),
\end{aligned}
\tag{3.11}
$$

for $j = 1, 2, \cdots, L$, where $\mathcal{L}$ is the linear map given by $\mathcal{L}(x) = (x - 1/2)/\delta$. It follows that, for $j = 1, 2, \cdots, L - 1$,

$$\xi_{j+1} = 2\xi_j - \theta_j = 2\xi_j - \sigma\big(\mathcal{L}(\xi_j) + 1\big) + \sigma\big(\mathcal{L}(\xi_j)\big). \tag{3.12}$$

Step 2: Design a ReLU FNN to output $\sum_{j=1}^{\ell} \theta_j$.

It is easy to design a ReLU FNN to output $\theta_1, \theta_2, \cdots, \theta_L$ by Equation (3.11) and (3.12) when using $\xi_1 = \text{bin} 0.\theta_1\theta_2\cdots\theta_L$ as the input. However, it is highly non-trivial to construct a ReLU FNN to output $\sum_{j=1}^{\ell} \theta_j$ with another input ℓ, since many operations like multiplication and comparison are not allowed in designing ReLU FNNs.

Now let us establish a formula to represent $\sum_{j=1}^{\ell} \theta_j$ in a form of a ReLU FNN as follows:

The fact that $x_1 x_2 = \sigma(x_1 + x_2 - 1)$ for any $x_1, x_2 \in \{0, 1\}$ implies

$$
\begin{aligned}
\sum_{j=1}^{\ell} \theta_j = \sum_{j=1}^{L} \theta_j \mathcal{T}(\ell - j) &= \sum_{j=1}^{L} \sigma\big(\theta_j + \mathcal{T}(\ell - j) - 1\big) \\
&= \sum_{j=1}^{L} \sigma\big(\theta_j + \sigma(\ell - j + 1) - \sigma(\ell - j) - 1\big),
\end{aligned}
$$

for $\ell = 1, 2, \cdots, L$, where the last equality comes from the fact $\mathcal{T}(n) = \sigma(n + 1) - \sigma(n)$ for any integer n.

To simplify the notations, we define

$$z_{\ell,j} := \sigma\big(\theta_j + \sigma(\ell - j + 1) - \sigma(\ell - j) - 1\big), \tag{3.13}$$

for $\ell = 1, 2, \cdots, L$ and $j = 1, 2, \cdots, L$. Then,

$$\sum_{j=1}^{\ell} \theta_j = \sum_{j=1}^{L} z_{\ell,j}, \quad \text{for } \ell = 1, 2, \cdots, L. \tag{3.14}$$
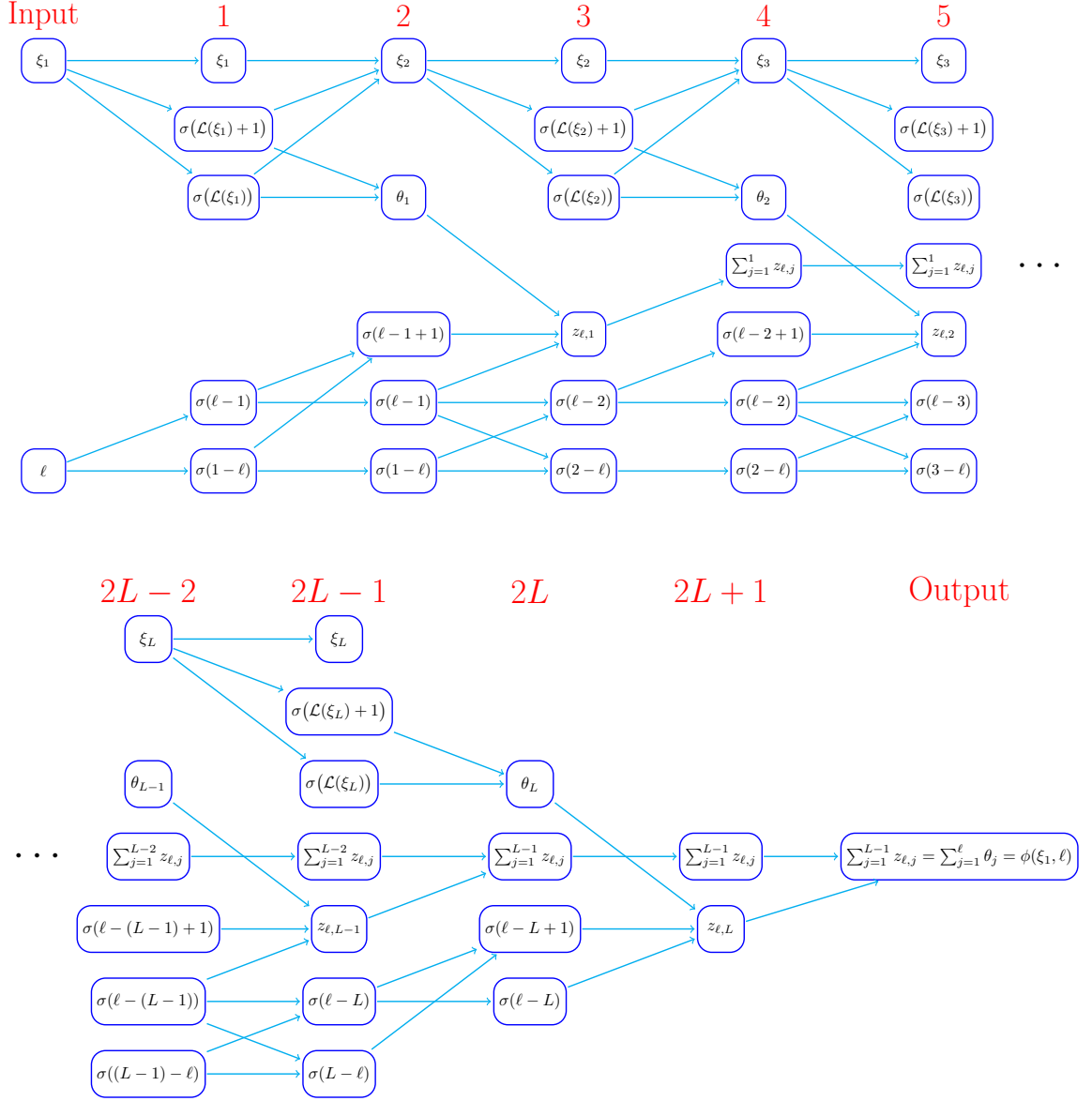
23

Figure 11: A illustration of the target ReLU FNN implementing ϕ to output $\sum_{j=1}^{L} z_{j,\ell} = \sum_{j=1}^{\ell} \theta_j = \phi(\xi_1, \ell)$ given the input $(\xi_1, \ell) = (\text{bin} \, 0.\theta_1\theta_2\cdots\theta_L, \ell)$ for $\ell \in \{1, 2, \cdots, L\}$ and $\theta_1, \theta_2, \cdots, \theta_L \in \{0, 1\}$. The construction is mainly based on Equation (3.11), (3.12), (3.13), and (3.14). The numbers above the architecture indicate the order of hidden layers. It builds a whole iteration step for every two layers. We output both $\sigma(\ell - j)$ and $\sigma(j - \ell)$ in the hidden layers for $j = 1, 2, \cdots, L$ because of the fact $x = \sigma(x) - \sigma(-x)$ for any $x \in \mathbb{R}$. We omit the activation function (σ) if the input of a neuron is non-negative. Note that the parameters are independent of $\theta_1, \theta_2, \cdots, \theta_L$ when using this network architecture to implement the desired function ϕ.

With Equation (3.11), (3.12), (3.13), and (3.14) in hand, it is easy to construct a function ϕ implemented by a ReLU FNN with the desired width and depth outputting $\sum_{j=1}^{\ell} \theta_j = \sum_{j=1}^{L} z_{\ell,j}$ given the input $(\xi_1, \ell) = (\text{bin}\, 0.\theta_1\theta_2\cdots\theta_L, \ell)$ for $\ell \in \{1, 2, \cdots, L\}$ and $\theta_1, \theta_2, \cdots, \theta_L \in \{0, 1\}$. The details of construction is shown in Figure 11. Clearly, the network in Figure 11 is with width 7 and depth $2L + 1$, which implies

$$\phi \in \mathcal{NN}(\#\text{input} = 2; \text{ width} \leq 7; \text{ ; depth} \leq 2L + 1; \#\text{output} = 1).$$

So we finish the proof. $\qquad\square$

Next, we introduce Lemma 3.6 as an advanced version of Lemma 3.5.

Lemma 3.6. *For any $N, L \in \mathbb{N}^+$, any $\theta_{m,\ell} \in \{0, 1\}$ for $m = 0, 1, \cdots, M-1$ and $\ell = 0, 1, \cdots, L-1$, where $M = N^2L$, there exists a function ϕ implemented by a ReLU FNN with width $4N + 3$ and depth $3L + 3$ such that*

$$\phi(m, \ell) = \sum_{j=0}^{\ell} \theta_{m,j}, \quad \text{for } m = 0, 1, \cdots, M - 1 \text{ and } \ell = 0, 1, \cdots, L - 1.$$

Proof. Define
$$y_m := \text{bin}\, 0.\theta_{m,0}\theta_{m,1}\cdots\theta_{m,L-1}, \quad \text{for } m = 0, 1, \cdots, M - 1.$$
Consider the sample set $\{(m, y_m) : m = 0, 1, \cdots, M\}$, whose cardinality is $M+1 = N\big((NL-1) + 1\big) + 1$. By Lemma 3.3 (set $N_1 = N$ and $N_2 = NL - 1$ therein), there exists

$$\phi_1 \in \mathcal{NN}(\#\text{input} = 1; \text{ widthvec} = [2N, 2(NL - 1) + 1])$$
$$= \mathcal{NN}(\#\text{input} = 1; \text{ widthvec} = [2N, 2NL - 1])$$

such that
$$\phi_1(m) = y_m, \quad \text{for } m = 0, 1, \cdots, M - 1.$$

By Lemma 3.5, there exists

$$\phi_2 \in \mathcal{NN}(\#\text{input} = 2; \text{ width} \leq 7; \text{ depth} \leq 2L + 1)$$

such that, for any $\xi_1, \xi_2, \cdots, \xi_L \in \{0, 1\}$, we have

$$\phi_2(\text{bin}\, 0.\xi_1\xi_2\cdots\xi_L, \ell) = \sum_{j=1}^{\ell} \xi_j, \quad \text{for } \ell = 1, 2, \cdots, L.$$

It follows that, for any $\xi_0, \xi_1, \cdots, \xi_{L-1} \in \{0, 1\}$, we have

$$\phi_2(\text{bin}\, 0.\xi_0\xi_1\cdots\xi_{L-1}, \ell + 1) = \sum_{j=0}^{\ell} \xi_j, \quad \text{for } \ell = 0, 1, \cdots, L - 1.$$

Thus, for $m = 0, 1, \cdots, M - 1$ and $\ell = 0, 1, \cdots, L - 1$, we have

$$\phi_2(\phi_1(m), \ell + 1) = \phi_2(y_m, \ell + 1) = \phi_2(0.\theta_{m,0}\theta_{m,1}\cdots\theta_{m,L-1}, \ell + 1) = \sum_{j=0}^{\ell} \theta_{m,j}.$$

Hence, the desired function function ϕ can be implemented by the network shown in Figure 12. By Lemma 3.4, $\phi_1 \in \mathcal{NN}(\text{widthvec} = [2N, 2NL - 1]) \subseteq \mathcal{NN}(\text{width} \leq 4N + 2; \text{depth} \leq L + 1)$, implying the network in Figure 12 is with width $\max\{(4N + 2) + 1, 7\} = 4N + 3$ and depth $(2L + 1) + 1 + (L + 1) = 3L + 3$. So we finish the proof. $\qquad\square$
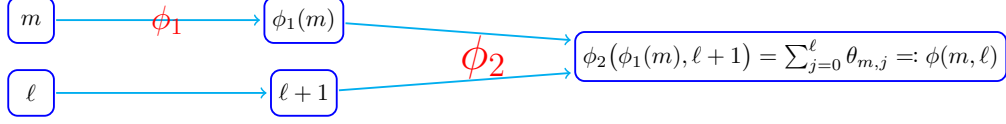
Figure 12: A illustration of the network implementing the desired function ϕ. "ϕ_1" and "ϕ_2" near "$\longrightarrow$" represent the respective ReLU FNN implementing itself. We omit the activation function ReLU if the input of a neuron is non-negative.

Next, we apply Lemma 3.6 to prove Lemma 3.7 below, which is a key intermediate conclusion to prove Proposition 3.2.

Lemma 3.7. *For any $\varepsilon > 0$, $L, N \in \mathbb{N}^+$, denote $M = N^2 L$ and assume $\{y_{m,\ell} \geq 0 : m = 0, 1, \cdots, M-1$ and $\ell = 0, 1, \cdots, L-1\}$ is sample set with*

$$|y_{m,\ell} - y_{m,\ell-1}| \leq \varepsilon, \quad \text{for } m = 0, 1, \cdots, M-1 \text{ and } \ell = 1, 2, \cdots, L-1.$$

Then there exists $\phi \in \mathcal{NN}(\#\text{input} = 2; \text{ width} \leq 12N + 8; \text{ depth} \leq 3L + 6)$ such that

(i) $|\phi(m, \ell) - y_{m,\ell}| \leq \varepsilon$, for $m = 0, 1, \cdots, M-1$ and $\ell = 0, 1, \cdots, L-1$;

(ii) $0 \leq \phi(x_1, x_2) \leq \max\{y_{m,\ell} : m = 0, 1, \cdots, M-1$ and $\ell = 0, 1, \cdots, L-1\}$, for any $x_1, x_2 \in \mathbb{R}$.

Proof. Define

$$a_{m,\ell} := \lfloor y_{m,\ell}/\varepsilon \rfloor, \quad \text{for } m = 0, 1, \cdots, M-1 \text{ and } \ell = 0, 1, \cdots, L-1.$$

We will construct a function implemented by a ReLU FNN to map the index (m, ℓ) to $a_{m,\ell}\varepsilon$ for $m = 0, 1, \cdots, M-1$ and $\ell = 0, 1, \cdots, L-1$.

Define $b_{m,0} := 0$ and $b_{m,\ell} := a_{m,\ell} - a_{m,\ell-1}$ for $m = 0, 1, \cdots, M-1$ and $\ell = 1, \cdots, L-1$. Since $|y_{m,\ell} - y_{m,\ell-1}| \leq \varepsilon$ for all m and ℓ, we have $b_{m,\ell} \in \{-1, 0, 1\}$. Hence, there exist $c_{m,\ell}$ and $d_{m,\ell} \in \{0, 1\}$ such that $b_{m,\ell} = c_{m,\ell} - d_{m,\ell}$, which implies

$$a_{m,\ell} = a_{m,0} + \sum_{j=1}^{\ell}(a_{m,j} - a_{m,j-1}) = a_{m,0} + \sum_{j=1}^{\ell} b_{m,j} = a_{m,0} + \sum_{j=0}^{\ell} b_{m,j}$$
$$= a_{m,0} + \sum_{j=0}^{\ell} c_{m,j} - \sum_{j=0}^{\ell} d_{m,j}.$$

for $m = 0, 1, \cdots, M-1$ and $\ell = 1, \cdots, L-1$.

For the sample set $\{(m, a_{m,0}) : m = 0, 1, \cdots, M-1\} \cup \{(M, 0)\}$, whose size is $M + 1 = N \cdot ((NL-1) + 1) + 1$, by Lemma 3.3 (set $N_1 = N$ and $N_2 = NL-1$ therein), there exists $\psi_1 \in \mathcal{NN}(\text{widthvec} = [2N, 2(NL-1) + 1]) = \mathcal{NN}(\text{widthvec} = [2N, 2NL-1])$ such that

$$\psi_1(m) = a_{m,0}, \quad \text{for } m = 0, 1, \cdots, M-1.$$

By Lemma 3.6, there exist $\psi_2, \psi_3 \in \mathcal{NN}(\text{width} \leq 4N + 3; \text{ depth} \leq 3L + 3)$ such that

$$\psi_2(m, \ell) = \sum_{j=0}^{\ell} c_{m,j} \quad \text{and} \quad \psi_3(m, \ell) = \sum_{j=0}^{\ell} d_{m,j},$$

26

for $m = 0, 1, \cdots, M - 1$ and $\ell = 0, 1, \cdots, L - 1$. Hence, it holds that

$$a_{m,\ell} = a_{m,0} + \sum_{j=0}^{\ell} c_{m,j} - \sum_{j=0}^{\ell} d_{m,j} = \psi_1(m) + \psi_2(m, \ell) - \psi_3(m, \ell), \qquad (3.15)$$

for $m = 0, 1, \cdots, M - 1$ and $\ell = 0, 1, \cdots, L - 1$.

Define

$$y_{\max} := \max\{y_{m,\ell} : m = 0, 1, \cdots, M - 1 \text{ and } \ell = 0, 1, \cdots, L - 1\}.$$

Then the desired function can be implemented by two sub-networks shown in Figure 13.
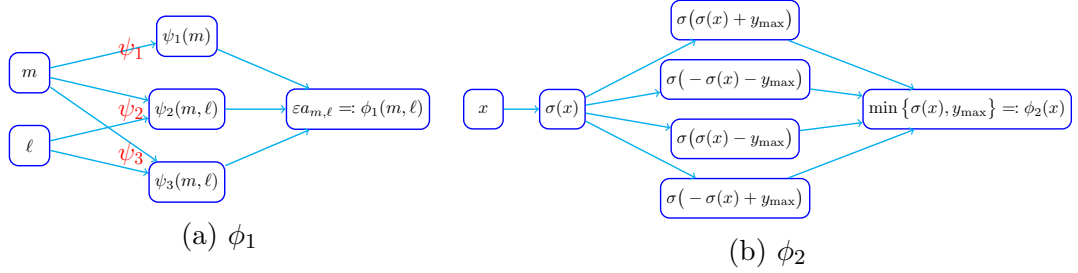

(a) ϕ_1

(b) ϕ_2

Figure 13: Illustrations of two sub-networks implementing the desired function $\phi = \phi_2 \circ \phi_1$ based Equation (3.15) and the fact $\min\{x_1, x_2\} = \frac{x_1 + x_2 - |x_1 - x_2|}{2} = \frac{\sigma(x_1 + x_2) - \sigma(-x_1 - x_2) - \sigma(x_1 - x_2) - \sigma(-x_1 + x_2)}{2}$. $y_{\max}$ is given by $\max\{y_{m,\ell} : m = 0, 1, \cdots, M - 1 \text{ and } \ell = 0, 1, \cdots, L - 1\}$. "$\psi_1$","$\psi_2$", and "$\psi_3$" near "$\longrightarrow$" represent the respective ReLU FNN implementing itself. We omit the activation function ReLU if the input of a neuron is non-negative.

By Lemma 3.4, $\psi_1 \in \mathcal{NN}(\#\text{input} = 1; \text{ widthvec} = [2N, 2NL - 1]) \subseteq \mathcal{NN}(\#\text{input} = 1; \text{ width} \le 4N + 2; \text{depth} \le L + 1)$. Note that $\psi_2, \psi_3 \in \mathcal{NN}(\text{width} \le 4N + 3; \text{ depth} \le 3L + 3)$. Thus, $\phi_1 \in \mathcal{NN}(\text{width} \le (4N + 2) + 2(4N + 3) = 12N + 8; \text{ depth} \le (3L + 3) + 1 = 3L + 4)$ as shown in Figure 13. And it is clear that $\phi_2 \in \mathcal{NN}(\text{width} \le 4; \text{ depth} \le 2)$, implying $\phi = \phi_2 \circ \phi_1 \in \mathcal{NN}(\text{width} \le 12N + 8; \text{ depth} \le (3L + 4) + 2 = 3L + 6)$.

Clearly, $0 \le \phi(x_1, x_2) \le y_{\max}$ for any $x_1, x_2 \in \mathbb{R}$, since $\phi(x_1, x_2) = \phi_2 \circ \phi_1(x_1, x_2) = \max\{\sigma(\phi_1(x_1, x_2)), y_{\max}\}$.

Note that $0 \le \varepsilon a_{m,\ell} = \varepsilon \lfloor y_{m,\ell}/\varepsilon \rfloor \le y_{\max}$. Then we have $\phi(m, \ell) = \phi_2 \circ \phi_1(m, \ell) = \phi_2(\varepsilon a_{m,\ell}) = \max\{\sigma(\varepsilon a_{m,\ell}), y_{\max}\} = \varepsilon a_{m,\ell}$. Therefore,

$$|\phi(m, \ell) - y_{m,\ell}| = |a_{m,\ell}\varepsilon - y_{m,\ell}| = |\lfloor y_{m,\ell}/\varepsilon \rfloor \varepsilon - y_{m,\ell}| \le \varepsilon,$$

for $m = 0, 1, \cdots, M - 1$ and $\ell = 0, 1, \cdots, L - 1$. Hence, we finish the proof. $\square$

Finally, we apply Lemma 3.7 to prove Proposition 3.2.

Proof of Proposition 3.2. Let $M = N^2 L$, then we may assume $J = ML$ since we can set $y_{J-1} = y_J = y_{J+1} = \cdots = y_{ML-1}$ if $J < ML$.

For the sample set

$$\{(mL, m) : m = 0, 1, \cdots, M\} \cup \{(mL + L - 1, m) : m = 0, 1, \cdots, M - 1\},$$

whose size is $2M + 1 = N \cdot \big((2NL - 1) + 1\big) + 1$, by Lemma 3.3 (set $N_1 = N$ and $N_2 = NL - 1$ therein), there exist $\phi_1 \in \mathcal{NN}(\#\text{input} = 1; \text{ widthvec} = [2N, 2(2NL - 1) + 1]) = \mathcal{NN}(\#\text{input} = 1; \text{ widthvec} = [2N, 4NL - 1])$ such that

- $\phi_1(ML) = M$ and $\phi_1(mL) = \phi_1(mL + L - 1) = m$ for $m = 0, 1, \cdots, M - 1$;

- ϕ_1 is linear on each interval $[mL, mL + L - 1]$ for $m = 0, 1, \cdots, M - 1$.

It follows that

$$\phi_1(j) = m, \quad \text{and} \quad j - L\phi_1(j) = \ell, \quad \text{where } j = mL + \ell, \tag{3.16}$$

for $m = 0, 1, \cdots, M - 1$ and $\ell = 0, 1, \cdots, L - 1$.

Note that any number j in $\{0, 1, \ldots, J - 1\}$ can be uniquely indexed as $j = mL + \ell$ for $m = 0, 1, \cdots, M - 1$ and $\ell = 0, 1, \cdots, L - 1$. So we can denote $y_j = y_{mL+\ell}$ as $y_{m,\ell}$. Then by Lemma 3.7, there exists $\phi_2 \in \mathcal{NN}(\text{width} \leq 12N + 8; \text{ depth} \leq 3L + 6)$ such that

$$|\phi_2(m, \ell) - y_{m,\ell}| \leq \varepsilon, \quad \text{for } m = 0, 1, \cdots, M - 1 \text{ and } \ell = 0, 1, \cdots, L - 1, \tag{3.17}$$

and

$$0 \leq \phi_2(x_1, x_2) \leq y_{\max}, \quad \text{for any } x_1, x_2 \in \mathbb{R}, \tag{3.18}$$

where $y_{\max} := \max\{y_{m,\ell} : m = 0, 1, \cdots, M - 1 \text{ and } \ell = 0, 1, \cdots, L - 1\} = \max\{y_j : j = 0, 1, \cdots, ML - 1\}$.
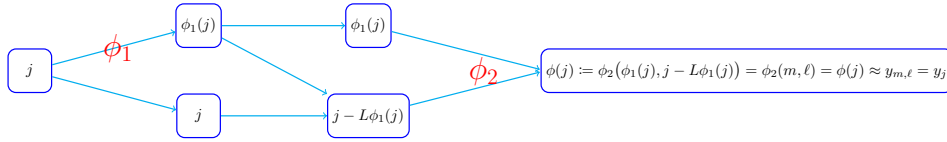

Figure 14: A illustration of the ReLU FNN implementing the desired function ϕ based Equation (3.16). The index $j \in \{0, 1, \cdots, ML - 1\}$ is unique represented by $j = mL + \ell$ for $m = 0, 1, \cdots, M - 1$ and $\ell = 0, 1, \cdots, L - 1$. "$\phi_1$" and "$\phi_2$" near "$\longrightarrow$" represent the respective ReLU FNN implementing itself. We omit the activation function ReLU if the input of a neuron is non-negative.

Note that $\phi_1 \in \mathcal{NN}(\#\text{input} = 1; \text{widthvec} = [2N, 4NL - 1]) \subseteq \mathcal{NN}(\#\text{input} = 1; \text{width} \leq 8N + 2; \text{depth} \leq L + 1)$ by Lemma 3.4 and $\phi_2 \in \mathcal{NN}(\text{width} \leq 12N + 8; \text{ depth} \leq 3L + 6)$. So $\phi \in \mathcal{NN}(\text{width} \leq 12N + 8; \text{ depth} \leq (L + 1) + 2 + (3L + 6) = 4L + 9)$ as shown in Figure 14.

Equation (3.18) implies

$$0 \leq \phi(x) \leq y_{\max}, \quad \text{for any } x \in \mathbb{R},$$

Since ϕ is given by $\phi(x) = \phi_2(\phi_1(x), x - L\phi_1(x))$.

Represent $j \in \{0, 1, \cdots, ML - 1\}$ via $j = mL + \ell$ for $m = 0, 1, \cdots, M - 1$ and $\ell = 0, 1, \cdots, L - 1$, then we have, by Equation (3.17),

$$|\phi(j) - y_j| = |\phi_2(\phi_1(j), j - L\phi_1(j)) - y_j| = |\phi_2(m, \ell) - y_{m,\ell}| \leq \varepsilon.$$

So we finish the proof. $\qquad\square$

We would like to remark that the key idea in the proof of Proposition 3.2 is the bit extraction technique in Lemma 3.5, which allows us to store L bits in a binary number $\text{bin}\, 0.x_1 x_2 \cdots x_L$ and extract each bit x_i. The extraction operator can be efficiently carried out via a deep ReLU neural network demonstrating the power of depth.

4 Neural networks in practice

This section is concerned with neural networks in practice, e.g., approximating functions defined on irregular domains or domains with a low-dimensional structure, and neural network computation in parallel computing.

4.1 Approximation on irregular domain

In this section, we consider approximating continuous functions defined on irregular domains by deep ReLU FNNs. The construction is through extending the target function to a cubic domain, applying Theorem 1.1, and finally restricting the constructed FNN back to the irregular domain.

Given any uniformly continuous and real-valued function f defined on a metric space S with a metric $d_S(\cdot, \cdot)$, we define the (optimal) modulus of continuity of f on a subset $E \subseteq S$ as

$$\omega_f^E(r) := \sup\{|f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)| :, d_S(\boldsymbol{x}_1, \boldsymbol{x}_2) \le r, \, \boldsymbol{x}_1, \boldsymbol{x}_2 \in E\}, \quad \text{for any } r \ge 0.$$

For the purpose of consistency and simplicity, $\omega_f(\cdot)$ is short of $\omega_f^{[0,1]^d}(\cdot)$.

First, let us present two lemmas for (approximately) extending (almost) continuous functions on E to (almost) continuous functions on S. These lemmas are similar to the well-known results for extending Lipschitz or differentiable functions in [47, 64]. We generalize these results to a broader class of functions required in the proof of Theorem 4.3.

Lemma 4.1 (Approximate Extension of Almost-Continuous Functions). *Assume S is a metric space with a metric $d_S(\cdot, \cdot)$ and $\omega : [0, \infty) \to [0, \infty)$ is an increasing function with*

$$\omega(r_1 + r_2) \le \omega(r_1) + \omega(r_2), \quad \text{for any } r_1, r_2 \in [0, \infty). \tag{4.1}$$

Let f be a real-valued function defined on a subset $E \subseteq S$ and satisfy

$$|f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)| \le \omega(d_S(\boldsymbol{x}_1, \boldsymbol{x}_2) + \Delta), \quad \text{for any } \boldsymbol{x}_1, \boldsymbol{x}_2 \in E, \tag{4.2}$$

where Δ is a positive constant independent of f. Then there exists a function g defined on S such that

$$0 \le f(\boldsymbol{x}) - g(\boldsymbol{x}) \le \omega(\Delta), \quad \text{for any } \boldsymbol{x} \in E$$

and

$$|g(\boldsymbol{x}_1) - g(\boldsymbol{x}_2)| \le \omega(d_S(\boldsymbol{x}_1, \boldsymbol{x}_2)), \quad \text{for any } \boldsymbol{x}_1, \boldsymbol{x}_2 \in S.$$

In Lemma 4.1, g is an approximate extension of f defined on E to a new domain S with an approximation error $\omega(\Delta)$. In a special case when $\Delta = 0$ and $\omega(0) = 0$, g is an exact extension of f.

Proof of Lemma 4.1. Define

$$g(\boldsymbol{x}) := \sup_{\boldsymbol{z} \in E} \big(f(\boldsymbol{z}) - \omega(d_S(\boldsymbol{z}, \boldsymbol{x}) + \Delta) \big).$$

By Equation (4.2), we have $f(\boldsymbol{x}_1) - \omega(d_S(\boldsymbol{x}_1, \boldsymbol{x}_2) + \Delta) \leq f(\boldsymbol{x}_2)$ for any $\boldsymbol{x}_1, \boldsymbol{x}_2 \in E$. It holds that $g(\boldsymbol{x}) \leq f(\boldsymbol{x})$ for any $\boldsymbol{x} \in E$. Together with

$$g(\boldsymbol{x}) = \sup_{\boldsymbol{z} \in E} \left(f(\boldsymbol{z}) - \omega(d_S(\boldsymbol{z}, \boldsymbol{x}) + \Delta) \right) \geq f(\boldsymbol{x}) - \omega(d_S(\boldsymbol{x}, \boldsymbol{x}) + \Delta) = f(\boldsymbol{x}) - \omega(\Delta),$$

for any $\boldsymbol{x} \in E$, it follows that $0 \leq f(\boldsymbol{x}) - g(\boldsymbol{x}) \leq \omega(\Delta)$ for any $\boldsymbol{x} \in E$. By Equation (4.1) and the fact

$$\sup_{\boldsymbol{z} \in E} f_1(\boldsymbol{z}) - \sup_{\boldsymbol{z} \in E} f_2(\boldsymbol{z}) \leq \sup_{\boldsymbol{z} \in E} \left(f_1(\boldsymbol{z}) - f_2(\boldsymbol{z}) \right), \quad \text{for any functions } f_1, f_2,$$

we have

$$\begin{aligned}
g(\boldsymbol{x}_1) - g(\boldsymbol{x}_2) &= \sup_{\boldsymbol{z} \in E} \left(f(\boldsymbol{z}) - \omega(d_S(\boldsymbol{z}, \boldsymbol{x}_1)) \right) - \sup_{\boldsymbol{z} \in E} \left(f(\boldsymbol{z}) - \omega(d_S(\boldsymbol{z}, \boldsymbol{x}_2)) \right) \\
&\leq \sup_{\boldsymbol{z} \in E} \left(\omega(d_S(\boldsymbol{z}, \boldsymbol{x}_1)) - \omega(d_S(\boldsymbol{z}, \boldsymbol{x}_2)) \right) \\
&\leq \sup_{\boldsymbol{z} \in E} \omega \left(d_S(\boldsymbol{z}, \boldsymbol{x}_1) - d_S(\boldsymbol{z}, \boldsymbol{x}_2) \right) \\
&\leq \sup_{\boldsymbol{z} \in E} \omega(d_S(\boldsymbol{x}_1, \boldsymbol{x}_2)) = \omega(d_S(\boldsymbol{x}_1, \boldsymbol{x}_2)),
\end{aligned}$$

for any $\boldsymbol{x}_1, \boldsymbol{x}_2 \in S$. Similarly, we have $g(\boldsymbol{x}_2) - g(\boldsymbol{x}_1) \leq \omega(d_S(\boldsymbol{x}_1, \boldsymbol{x}_2))$, which implies

$$|g(\boldsymbol{x}_1) - g(\boldsymbol{x}_2)| \leq \omega(d_S(\boldsymbol{x}_1, \boldsymbol{x}_2)).$$

So we finish the proof. □

Next, we introduce a lemma below for extending continuous functions defined on $E \subseteq S$ to continuous functions defined on S preserving the modulus of continuity.

Lemma 4.2 (Extension of Continuous Functions). *Suppose f is a uniformly continuous function defined on a subset $E \subseteq S$, where S is a metric space with a metric $d_S(\cdot, \cdot)$, then there exists a uniformly continuous function g on S such that $f(x) = g(x)$ for $x \in E$ and $\omega_f^E(r) = \omega_g^S(r)$ for any $r \geq 0$.*

Proof. By the application of Lemma 4.1 with $\omega(r) = \omega_f^E(r)$ for $r \geq 0$ and $\Delta = 0$, we know that there exists $g : S \to \mathbb{R}$ such that

$$0 \leq f(\boldsymbol{x}) - g(\boldsymbol{x}) \leq \omega_f^E(\Delta) = 0, \quad \text{for any } \boldsymbol{x} \in E,$$

and

$$|g(\boldsymbol{x}_1) - g(\boldsymbol{x}_2)| \leq \omega_f^E(d_S(\boldsymbol{x}_1, \boldsymbol{x}_2)), \quad \text{for any } \boldsymbol{x}_1, \boldsymbol{x}_2 \in S.$$

The equation above and the uniform continuity of f imply that g is uniformly continuous. It also follows that

$$f(\boldsymbol{x}) = g(\boldsymbol{x}), \quad \text{for any } \boldsymbol{x} \in E, \quad \text{and} \quad \omega_g^S(r) \leq \omega_f^E(r), \quad \text{for any } r \geq 0,$$

since $\omega_g^S(\cdot)$ is the optimal modulus of continuity of g. Note that $\omega_f^E(\cdot)$ is the optimal modulus of continuity of f and

$$|f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)| = |g(\boldsymbol{x}_1) - g(\boldsymbol{x}_2)| \leq \omega_g^S(d_S(\boldsymbol{x}_1, \boldsymbol{x}_2)), \quad \text{for any } \boldsymbol{x}_1, \boldsymbol{x}_2 \in E.$$

Hence, $\omega_f^E(r) \leq \omega_g^S(r)$ for all $r \geq 0$, which implies $\omega_f^E(r) = \omega_g^S(r)$ since we have proved that $\omega_g^S(r) \leq \omega_f^E(r)$ for all $r \geq 0$. So we finish the proof. □

Now we are ready to introduce and prove the main theorem of this section, which extends Theorem 1.1 to an irregular domain as follows.

Theorem 4.3. *Let f be a uniformly continuous function defined on $E \subseteq [-R, R]^d$. For arbitrary $L \in \mathbb{N}^+$ and $N \in \mathbb{N}^+$, there exists a function ϕ implemented by a ReLU FNN with width $3^{d+3} \max\{d\lfloor N^{1/d}\rfloor, N+1\}$ and depth $12L + 14 + 2d$ such that*

$$\|f - \phi\|_{L^\infty(E)} \leq 19\sqrt{d}\,\omega_f^E(2R\sqrt{d}N^{-2/d}L^{-2/d}).$$

Proof. By Lemma 4.2, f can be extended to $\mathbb{R}^d$ such that

$$\omega_f^{\mathbb{R}^d}(r) = \omega_f^E(r), \quad \text{for any } r \geq 0.$$

Define

$$\widetilde{f}(\boldsymbol{x}) \coloneqq f(2R\boldsymbol{x} - R), \quad \text{for any } \boldsymbol{x} \in \mathbb{R}^d.$$

It follows that

$$\omega_{\widetilde{f}}^{\mathbb{R}^d}(r) = \omega_f^{\mathbb{R}^d}(2Rr) = \omega_f^E(2Rr), \quad \text{for any } r \geq 0. \tag{4.3}$$

By Theorem 1.1, there exists a function $\widetilde{\phi}$ implemented by a ReLU FNN with width $3^{d+3} \max\{d\lfloor N^{1/d}\rfloor, N+1\}$ and depth $12L + 14 + 2d$ such that

$$\|\widetilde{f} - \widetilde{\phi}\|_{L^\infty([0,1]^d)} \leq 19\sqrt{d}\,\omega_{\widetilde{f}}^{\mathbb{R}^d}(N^{-2/d}L^{-2/d}).$$

Define

$$\phi(\boldsymbol{x}) \coloneqq \widetilde{\phi}(\tfrac{1}{2R}\boldsymbol{x} + \tfrac{1}{2}), \quad \text{for any } \boldsymbol{x} \in \mathbb{R}^d.$$

Then, by Equation (4.3), for any $\boldsymbol{x} \in E \subseteq [-R, R]^d$, we have

$$|f(\boldsymbol{x}) - \phi(\boldsymbol{x})| = |\widetilde{f}(\tfrac{1}{2R}\boldsymbol{x} + \tfrac{1}{2}) - \widetilde{\phi}(\tfrac{1}{2R}\boldsymbol{x} + \tfrac{1}{2})| \leq \|\widetilde{f} - \widetilde{\phi}\|_{L^\infty([0,1]^d)}$$
$$\leq 19\sqrt{d}\,\omega_{\widetilde{f}}^{\mathbb{R}^d}(N^{-2/d}L^{-2/d}) = 19\sqrt{d}\,\omega_f^E(2RN^{-2/d}L^{-2/d}),$$

which implies

$$\|f - \phi\|_{L^\infty(E)} \leq 19\sqrt{d}\,\omega_f^E(2RN^{-2/d}L^{-2/d}).$$

So we finish the proof. $\qquad\square$

4.2 Approximation in a neighborhood of a low-dimensional manifold

In this section, we study neural network approximation of functions defined in a neighborhood of a low-dimensional manifold and prove Theorem 1.2 in this setting. Let us first introduce Theorem 4.4 which is required to prove Theorem 1.2.

Theorem 4.4 (Theorem 3.1 of [3])**.** *Let $\mathcal{M}$ be a compact $d_{\mathcal{M}}$-dimensional Riemannian submanifold of $\mathbb{R}^d$ having condition number $1/\tau$, volume V, and geodesic covering regularity $\mathcal{R}$. Fix $\delta \in (0, 1)$ and $\gamma \in (0, 1)$. Let $\boldsymbol{A} = \sqrt{\frac{d}{d_\delta}}\Phi$, where $\Phi \in \mathbb{R}^{d_\delta \times d}$ is a random orthoprojector with*

$$d_\delta = \mathcal{O}\left(\frac{d_{\mathcal{M}} \ln(dV\mathcal{R}\tau^{-1}\delta^{-1})\ln(1/\gamma)}{\delta^2}\right).$$

If $d_\delta \leq d$, then with probability at least $1 - \gamma$, the following statement holds: For every $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{M}$,

$$(1 - \delta)|\boldsymbol{x}_1 - \boldsymbol{x}_2| \leq |\boldsymbol{A}\boldsymbol{x}_1 - \boldsymbol{A}\boldsymbol{x}_2| \leq (1 + \delta)|\boldsymbol{x}_1 - \boldsymbol{x}_2|.$$

Theorem 4.4 shows the existence of a linear projector $\boldsymbol{A} \in \mathbb{R}^{d_\delta \times d}$ that maps a low-dimensional manifold in a high-dimensional space to a low-dimensional space nearly preserving distance. With this projection $\boldsymbol{A}$ available, we can prove Theorem 1.2 via constructing a ReLU FNN defined in the low-dimensional space using Theorem 4.3 and hence the curse of dimensionality is lessened. The ideas of the proof are summarized in the following Table 1.

In Table 1 and the detailed proof later, we introduce a new notation $\mathcal{SL}(E)$ for any compact set $E \subseteq \mathbb{R}^d$ as the "smallest" element of E. Specifically, $\mathcal{SL}(E)$ is defined as the unique point in $\cap_{k=1}^d E_k$, where

$$E_k := \{\boldsymbol{x} \in E_{k-1} : x_k = s_k\}, \quad s_k := \inf\{x_k : (x_1, x_2, \cdots, x_d) \in E_{k-1}\}, \quad \text{for } k = 1, 2, \cdots, d,$$

and $E_0 = E$. The compactness of E ensures that $\cap_{k=1}^d E_k$ is in fact one point belonging to E. The introduction of $\mathcal{SL}(\cdot)$ uniquely formulates a low-dimensional function $\widetilde{f}$ representing a high-dimensional function f defined on $\mathcal{M}_\varepsilon$ by

$$\widetilde{f}(\boldsymbol{y}) := f(\boldsymbol{x_y}), \quad \text{where } \boldsymbol{x_y} = \mathcal{SL}(\{\boldsymbol{x} \in \mathcal{M}_\varepsilon : \boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}\}), \quad \text{for any } \boldsymbol{y} \in \boldsymbol{A}(\mathcal{M}_\varepsilon) \subseteq \mathbb{R}^{d_\delta}.$$

As we shall see later, such a definition of $\widetilde{f}$ is reasonable because $\{\boldsymbol{x} \in \mathcal{M}_\varepsilon : \boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}\}$ is contained in a small ball of radius $\mathcal{O}(\varepsilon)$ for any $\boldsymbol{y} \in \boldsymbol{A}(\mathcal{M}_\varepsilon)$. There are many other alternative ways to define $\mathcal{SL}(\cdot)$ as long as the definition ensures that $\mathcal{SL}(E)$ contains only one element. For example, $\mathcal{SL}(E)$ can be defined as any arbitrary point in E. For another example, $\boldsymbol{y} \in \boldsymbol{A}(\mathcal{M})$ cannot guarantee $\boldsymbol{x_y} = \mathcal{SL}(\{\boldsymbol{x} \in \mathcal{M}_\varepsilon : \boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}\}) \in \mathcal{M}$ in the current definition, but in practice we can choose $\mathcal{SL}(\{\boldsymbol{x} \in \mathcal{M} : \boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}\})$ as $\boldsymbol{x_y}$ to ensure that $\boldsymbol{x_y} \in \mathcal{M}$, which might be beneficial for potential applications.

Table 1: Main steps of the proof of Theorem 1.2. Step 1: dimension reduction via the nearly isometric projection operator $\boldsymbol{A}$ provided by Theorem 4.4 to obtain an "equivalent" function $\widetilde{f}$ of f in a low-dimensional domain using $\boldsymbol{x_y} = \mathcal{SL}(\{\boldsymbol{x} \in \mathcal{M}_\varepsilon : \boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}\})$. Step 2: construct a ReLU FNN to implement $\widetilde{\phi} \approx \widetilde{f}$ by Theorem 4.3. Step 3: define a ReLU FNN to implement ϕ in the original high-dimensional domain via the projection $\boldsymbol{A}$. Step 4: verify that the approximation error of $\phi \approx f$ satisfies our requirement.

$f(\boldsymbol{x})$ for $\boldsymbol{x} \in \mathcal{M}_\varepsilon \subseteq [0,1]^d$	$\overset{\text{Step 4}}{\approx}$	$\phi(\boldsymbol{x}) := \widetilde{\phi}(\boldsymbol{A}\boldsymbol{x})$ for $\boldsymbol{x} \in \mathcal{M}_\varepsilon \subseteq [0,1]^d$
Step 1 $\Updownarrow$ $\boldsymbol{x_y} = \mathcal{SL}(\{\boldsymbol{x} \in \mathcal{M}_\varepsilon : \boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}\})$		Step 3 $\Updownarrow$ $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$
$\widetilde{f}(\boldsymbol{y}) := f(\boldsymbol{x_y})$ for $\boldsymbol{y} \in \boldsymbol{A}(\mathcal{M}_\varepsilon) \subseteq \mathbb{R}^{d_\delta}$	$\overset{\text{Step 2}}{\approx}$	$\widetilde{\phi}(\boldsymbol{y})$ for $\boldsymbol{y} \in \boldsymbol{A}(\mathcal{M}_\varepsilon) \subseteq \mathbb{R}^{d_\delta}$

Now we are ready to prove Theorem 1.2.

Proof of Theorem 1.2. By Theorem 4.4, there exists a matrix $\boldsymbol{A} \in \mathbb{R}^{d_\delta \times d}$ such that

$$\boldsymbol{A}\boldsymbol{A}^T = \frac{d}{d_\delta} \boldsymbol{I}_{d_\delta}, \tag{4.4}$$

where $\boldsymbol{I}_{d_\delta}$ is an identity matrix of size $d_\delta \times d_\delta$, and

$$(1-\delta)|\boldsymbol{x}_1 - \boldsymbol{x}_2| \le |\boldsymbol{A}\boldsymbol{x}_1 - \boldsymbol{A}\boldsymbol{x}_2| \le (1+\delta)|\boldsymbol{x}_1 - \boldsymbol{x}_2|, \quad \text{for any } \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{M}. \tag{4.5}$$

Given any $\boldsymbol{y} \in \boldsymbol{A}(\mathcal{M}_\varepsilon)$, then $\{\boldsymbol{x} \in \mathcal{M}_\varepsilon : \boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}\}$ is a nonzero compact set. Let $\boldsymbol{x_y} = \mathcal{SL}(\{\boldsymbol{x} \in \mathcal{M}_\varepsilon : \boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}\})$, then we define $\widetilde{f}$ on $\boldsymbol{A}(\mathcal{M}_\varepsilon)$ as $\widetilde{f}(\boldsymbol{y}) = f(\boldsymbol{x_y})$.

For any $\boldsymbol{y}_1, \boldsymbol{y}_2 \in \boldsymbol{A}(\mathcal{M}_\varepsilon)$, let $\boldsymbol{x}_i = \mathcal{SL}(\{\boldsymbol{x} \in \mathcal{M}_\varepsilon : \boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}_i\})$, then $\boldsymbol{x}_i \in \mathcal{M}_\varepsilon$ for $i = 1, 2$. By the definition of $\mathcal{M}_\varepsilon$, there exist $\widetilde{\boldsymbol{x}}_1, \widetilde{\boldsymbol{x}}_2 \in \mathcal{M}$ such that $|\widetilde{\boldsymbol{x}}_i - \boldsymbol{x}_i| \le \varepsilon$ for $i = 1, 2$. It follows that

$$|\widetilde{f}(\boldsymbol{y}_1) - \widetilde{f}(\boldsymbol{y}_2)| = |f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)| \le \omega_f(|\boldsymbol{x}_1 - \boldsymbol{x}_2|) \le \omega_f(|\widetilde{\boldsymbol{x}}_1 - \widetilde{\boldsymbol{x}}_2| + 2\varepsilon) \le \omega_f(\tfrac{1}{1-\delta}|\boldsymbol{A}\widetilde{\boldsymbol{x}}_1 - \boldsymbol{A}\widetilde{\boldsymbol{x}}_2| + 2\varepsilon),$$

where the last inequality comes from Equation (4.5). By the triangular inequality, we have

$$\begin{aligned}
|\widetilde{f}(\boldsymbol{y}_1) - \widetilde{f}(\boldsymbol{y}_2)| &\le \omega_f(\tfrac{1}{1-\delta}|\boldsymbol{A}\boldsymbol{x}_1 - \boldsymbol{A}\boldsymbol{x}_2| + \tfrac{1}{1-\delta}|\boldsymbol{A}\boldsymbol{x}_1 - \boldsymbol{A}\widetilde{\boldsymbol{x}}_1| + \tfrac{1}{1-\delta}|\boldsymbol{A}\boldsymbol{x}_2 - \boldsymbol{A}\widetilde{\boldsymbol{x}}_2| + 2\varepsilon) \\
&\le \omega_f(\tfrac{1}{1-\delta}|\boldsymbol{A}\boldsymbol{x}_1 - \boldsymbol{A}\boldsymbol{x}_2| + \tfrac{2\varepsilon}{1-\delta}\sqrt{\tfrac{d}{d_\delta}} + 2\varepsilon) \\
&\le \omega_f(\tfrac{1}{1-\delta}|\boldsymbol{y}_1 - \boldsymbol{y}_2| + \tfrac{2\varepsilon}{1-\delta}\sqrt{\tfrac{d}{d_\delta}} + 2\varepsilon).
\end{aligned}$$

Set $\omega(r) = \omega_f(\tfrac{1}{1-\delta}r)$ for any $r \ge 0$ and $\Delta = 2\varepsilon\sqrt{\tfrac{d}{d_\delta}} + 2\varepsilon(1 - \delta)$, then

$$|\widetilde{f}(\boldsymbol{y}_1) - \widetilde{f}(\boldsymbol{y}_2)| \le \omega(|\boldsymbol{y}_1 - \boldsymbol{y}_2| + \Delta), \quad \text{for any } \boldsymbol{y}_1, \boldsymbol{y}_2 \in \boldsymbol{A}(\mathcal{M}_\varepsilon) \subseteq \mathbb{R}^{d_\delta}.$$

By Lemma 4.1, there exists $\widetilde{g}$ defined on $\mathbb{R}^{d_\delta}$ such that

$$|\widetilde{g}(\boldsymbol{y}) - \widetilde{f}(\boldsymbol{y})| \le \omega(\Delta) = \omega_f\big(\tfrac{2\varepsilon}{1-\delta}\sqrt{\tfrac{d}{d_\delta}} + 2\varepsilon\big), \quad \text{for any } \boldsymbol{y} \in \boldsymbol{A}(\mathcal{M}_\varepsilon), \tag{4.6}$$

and

$$|\widetilde{g}(\boldsymbol{y}_1) - \widetilde{g}(\boldsymbol{y}_2)| \le \omega(|\boldsymbol{y}_1 - \boldsymbol{y}_2|) = \omega_f(\tfrac{1}{1-\delta}|\boldsymbol{y}_1 - \boldsymbol{y}_2|), \quad \text{for any } \boldsymbol{y}_1, \boldsymbol{y}_2 \in \mathbb{R}^{d_\delta}.$$

It follows that

$$\omega_{\widetilde{g}}^{\mathbb{R}^{d_\delta}}(r) \le \omega_f(\tfrac{r}{1-\delta}), \quad \text{for any } r \ge 0. \tag{4.7}$$

By Equation (4.4) and the definition of $\mathcal{M}_\varepsilon$ in Equation (1.2), it is easy to check that

$$\boldsymbol{A}(\mathcal{M}_\varepsilon) \subseteq \boldsymbol{A}([0,1]^d) \subseteq [-\sqrt{\tfrac{d}{d_\delta}}, \sqrt{\tfrac{d}{d_\delta}}]^{d_\delta}.$$

By the application of Theorem 4.3 with $E = [-\sqrt{\tfrac{d}{d_\delta}}, \sqrt{\tfrac{d}{d_\delta}}]^{d_\delta}$, there exists a function $\widetilde{\phi}$ implemented by a ReLU FNN with width $3^{d_\delta+3} \max\{d_\delta\lfloor N^{1/d_\delta}\rfloor, N+1\}$ and depth $12L + 14 + 2d_\delta$ such that

$$\|\widetilde{g} - \widetilde{\phi}\|_{L^\infty(E)} \le 19\sqrt{d}\,\omega_{\widetilde{g}}^E\big(2\sqrt{\tfrac{d}{d_\delta}}N^{-2/d_\delta}L^{-2/d_\delta}\big). \tag{4.8}$$

Define $\phi := \widetilde{\phi} \circ \boldsymbol{A}$, i.e., $\phi(\boldsymbol{x}) := \widetilde{\phi}(\boldsymbol{A}\boldsymbol{x})$ for any $\boldsymbol{x} \in \mathbb{R}^d$. Then ϕ is also a ReLU FNN with width $3^{d_\delta+3} \max\{d_\delta\lfloor N^{1/d_\delta}\rfloor, N+1\}$ and depth $12L + 14 + 2d_\delta$.

For any $\boldsymbol{x} \in \mathcal{M}_\varepsilon$, set $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ and $\boldsymbol{x_y} = \mathcal{SL}(\{\boldsymbol{z} \in \mathbb{R}^d : \boldsymbol{A}\boldsymbol{z} = \boldsymbol{y}\})$, there exist $\widetilde{\boldsymbol{x}}, \widetilde{\boldsymbol{x}}_{\boldsymbol{y}} \in \mathcal{M}$ such that $|\widetilde{\boldsymbol{x}} - \boldsymbol{x}| \le \varepsilon$ and $|\widetilde{\boldsymbol{x}}_{\boldsymbol{y}} - \boldsymbol{x_y}| \le \varepsilon$. It follows from Equation (4.5) that

$$\begin{aligned}
|\boldsymbol{x} - \boldsymbol{x_y}| &\le |\widetilde{\boldsymbol{x}} - \widetilde{\boldsymbol{x}}_{\boldsymbol{y}}| + 2\varepsilon \le \tfrac{1}{1-\delta}|\boldsymbol{A}\widetilde{\boldsymbol{x}} - \boldsymbol{A}\widetilde{\boldsymbol{x}}_{\boldsymbol{y}}| + 2\varepsilon \\
&\le \tfrac{1}{1-\delta}\big(|\boldsymbol{A}\widetilde{\boldsymbol{x}} - \boldsymbol{A}\boldsymbol{x}| + |\boldsymbol{A}\boldsymbol{x} - \boldsymbol{A}\boldsymbol{x_y}| + |\boldsymbol{A}\boldsymbol{x_y} - \boldsymbol{A}\widetilde{\boldsymbol{x}}_{\boldsymbol{y}}|\big) + 2\varepsilon \\
&= \tfrac{1}{1-\delta}\big(|\boldsymbol{A}\widetilde{\boldsymbol{x}} - \boldsymbol{A}\boldsymbol{x}| + |\boldsymbol{A}\boldsymbol{x_y} - \boldsymbol{A}\widetilde{\boldsymbol{x}}_{\boldsymbol{y}}|\big) + 2\varepsilon \le \tfrac{2\varepsilon}{1-\delta}\sqrt{\tfrac{d}{d_\delta}} + 2\varepsilon.
\end{aligned} \tag{4.9}$$

In fact, the above equation implies that $\{\boldsymbol{x} \in \mathcal{M}_\varepsilon : \boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}\}$ is contained in a small ball of radius $\mathcal{O}(\varepsilon)$ for $\boldsymbol{y} \in \boldsymbol{A}(\mathcal{M}_\varepsilon)$ as we mentioned previously.

Together with Equation (4.6), (4.7), (4.8), and (4.9), we have, for any $\boldsymbol{x} \in \mathcal{M}_\varepsilon$,

$$
\begin{aligned}
|f(\boldsymbol{x}) - \phi(\boldsymbol{x})| &\le |f(\boldsymbol{x}) - f(\boldsymbol{x_y})| + |f(\boldsymbol{x_y}) - \phi(\boldsymbol{x})| \\
&\le \omega_f\left(\tfrac{2\varepsilon}{1-\delta}\sqrt{\tfrac{d}{d_\delta}} + 2\varepsilon\right) + |\widetilde{f}(\boldsymbol{y}) - \widetilde{\phi}(\boldsymbol{y})| \\
&\le \omega_f\left(\tfrac{2\varepsilon}{1-\delta}\sqrt{\tfrac{d}{d_\delta}} + 2\varepsilon\right) + |\widetilde{f}(\boldsymbol{y}) - \widetilde{g}(\boldsymbol{y})| + |\widetilde{g}(\boldsymbol{y}) - \widetilde{\phi}(\boldsymbol{y})| \\
&\le \omega_f\left(\tfrac{2\varepsilon}{1-\delta}\sqrt{\tfrac{d}{d_\delta}} + 2\varepsilon\right) + \omega_f\left(\tfrac{2\varepsilon}{1-\delta}\sqrt{\tfrac{d}{d_\delta}} + 2\varepsilon\right) + 19\sqrt{d}\,\omega_{\widetilde{g}}^E\left(2\sqrt{\tfrac{d}{d_\delta}}N^{-2/d_\delta}L^{-2/d_\delta}\right) \\
&\le 2\omega_f\left(\tfrac{2\varepsilon}{1-\delta}\sqrt{\tfrac{d}{d_\delta}} + 2\varepsilon\right) + 19\sqrt{d}\,\omega_f\left(\tfrac{2\sqrt{d}}{(1-\delta)\sqrt{d_\delta}}N^{-2/d_\delta}L^{-2/d_\delta}\right).
\end{aligned}
$$

Hence, we have finished the proof of this theorem. $\qquad\square$

It is worth emphasizing that the approximation error

$$
\mathcal{O}\left(\omega_f\big(\mathcal{O}(\varepsilon)\big) + \omega_f\big(\mathcal{O}(N^{-2/d_\delta}L^{-2/d_\delta})\big)\right)
$$

in Theorem 1.2 is equal to $\mathcal{O}\left(\omega_f\big(\mathcal{O}(N^{-2/d_\delta}L^{-2/d_\delta})\big)\right)$ when $\varepsilon = \mathcal{O}(N^{-2/d_\delta}L^{-2/d_\delta})$.

The application of Theorem 4.4 and the proof of Theorem 1.2 in fact inspire an efficient two-step algorithm for high-dimensional learning problems: in the first step, high-dimensional data are projected to a low-dimensional space via a random projection; in the second step, a deep learning algorithm is applied to learn from the low-dimensional data. By Theorem 4.4 and 1.2, the deep learning algorithm in the low-dimensional space can still provide good results with a high probability.

4.3 Optimal ReLU FNN structure in parallel computing

In this section, we show how to select the best ReLU FNN to approximate functions in Höld($[0,1]^d, \alpha, \lambda$) on a d-dimensional cube, if the approximation error ε and the number of parallel computing cores (processors) p are given. We choose the best ReLU FNN by minimizing the time complexity in each training iteration. The analysis in this section is valid up to a constant prefactor.

Assume $\phi_{\boldsymbol{\theta}} \in \mathcal{NN}(\#\text{input} = d; \text{widthvec} = [N]^L)$, $N, L \in \mathbb{N}^+$, where $\boldsymbol{\theta}$ is the vector including all parameters of $\phi_{\boldsymbol{\theta}}$. By the basic knowledge of parallel computing (see [36] for more details), we have the following Table 2.

Table 2: Time complexity of one training iteration for an FNN of width N and depth L.

Number of cores p	Time Complexity	
	Evaluating $\phi_{\boldsymbol{\theta}}(\boldsymbol{x})$	Evaluating $\frac{\partial \phi_{\boldsymbol{\theta}}(\boldsymbol{x})}{\partial \boldsymbol{\theta}}$
$p \in [1, N]$	$\mathcal{O}(N^2 L/p)$	$\mathcal{O}(N^2 L/p)$
$p \in (N, N^2]$	$\mathcal{O}\left(L(N^2/p + \ln \frac{p}{N})\right)$	$\mathcal{O}\left(L(N^2/p + \ln \frac{p}{N})\right)$
$p \in (N^2, \infty)$	$\mathcal{O}(L \ln N)$	$\mathcal{O}(L \ln N)$

For the sake of simplicity, we assume that the training batch size is $\mathcal{O}(1)$. Denote the time complexity of each training iteration as $T(n, L)$, then

$$T(N, L) = \begin{cases} \mathcal{O}(N^2 L/p), & p \in [1, N], \\ \mathcal{O}\big(L(N^2/p + \ln \frac{p}{N})\big), & p \in (N, N^2], \\ \mathcal{O}(L \ln N), & p \in (N^2, \infty). \end{cases}$$

Theorem 1.1 and 2.3 imply that the approximation error ε is essentially $\mathcal{O}\big((NL)^{-2\alpha/d}\big)$. Hence, we can get the optimal size of ReLU FNNs via the optimization problem below:

$$(N_{\text{opt}}, L_{\text{opt}}) = \underset{N,\ L}{\arg\min}\ T(N, L)$$

$$\text{subject to} \quad \begin{cases} \varepsilon = \mathcal{O}\big((NL)^{-2\alpha/d}\big), \\ N, L, p \in N^+. \end{cases} \tag{4.10}$$

To simplify the discussion, we have the following assumptions:

- Dropping the notation $\mathcal{O}(\cdot)$ sometimes while assuming asymptotic analysis with the abuse of notations.

- N, L, and p are allowed to be real numbers.

- We denote $\varepsilon = (NL)^{-2\alpha/d}$ since the approximation rate $\mathcal{O}\big((NL)^{-2\alpha/d}\big)$ is both attainable and nearly optimal.

With $\varepsilon = (NL)^{-2\alpha/d}$, we have

$$\overline{T}(N, L) := \begin{cases} N^2 L/p & p \in [1, N], \\ L(N^2/p + \ln \frac{p}{N}), & p \in (N, N^2], \\ L(1 + \ln N), & p \in [N^2, \infty), \end{cases}$$

$$= \begin{cases} N\varepsilon^{-d/(2\alpha)}/p, & N \in [p, \infty), \\ N\varepsilon^{-d/(2\alpha)}/p + \frac{1}{N}\varepsilon^{-d/(2\alpha)} \ln \frac{p}{N}, & N \in [\sqrt{p}, p), \\ \frac{1+\ln N}{N}\varepsilon^{-d/(2\alpha)}, & N \in [1, \sqrt{p}). \end{cases} \tag{4.11}$$

Then we get $T(N, L) = \mathcal{O}(\overline{T}(N, L))$. Therefore, the optimization problem in Equation (4.10) can be simplified to

$$(N_{\text{opt}}, L_{\text{opt}}) = \underset{N,\ L}{\arg\min}\ \overline{T}(N, L)$$

$$\text{subject to} \quad \begin{cases} \varepsilon = (NL)^{-2\alpha/d}, \\ N, L, p \in [1, \infty). \end{cases} \tag{4.12}$$

By Equation (4.11), $\overline{T}(N, L)$ is independent of L on the condition that $\varepsilon = (NL)^{-2\alpha/d}$. Therefore, we may denote $\overline{T}(N, L)$ by $\overline{T}(N)$. Now we consider two cases: the case $p = \mathcal{O}(1)$ and the case $p \gg \mathcal{O}(1)$.

Case 1: The case $p = \mathcal{O}(1)$.

It is clear that $\overline{T}(N)$ is increasing in N when $N \in [p, \infty)$ by Equation (4.11). Together with $p = \mathcal{O}(1)$, then $\mathcal{O}(\sqrt{p}) = \mathcal{O}(p) = \mathcal{O}(1)$. Therefore, $N_{\text{opt}} = \mathcal{O}(1)$ and

$L_{\text{opt}} = \mathcal{O}(\varepsilon^{-d/(2\alpha)})$. Note that we regard d as a constant ($\mathcal{O}(1)$) in above analysis, N_{opt} should be $\mathcal{O}(d)$ in fact.

Case 2: The case $p \gg \mathcal{O}(1)$.

Since $\varepsilon = (NL)^{-2\alpha/d}$, we have $N \le \varepsilon^{-d/(2\alpha)}$. We only need to consider the monotonicity of $\overline{T}(N)$ on $[1, \varepsilon^{-d/(2\alpha)}]$. Together with Equation (4.11), this case can be divided into two sub-cases: the sub-case $\sqrt{p} \le \varepsilon^{-d/(2\alpha)}$ and the sub-case $\sqrt{p} > \varepsilon^{-d/(2\alpha)}$.

Case 2.1: The sub-case $\sqrt{p} > \varepsilon^{-d/(2\alpha)}$.

$\sqrt{p} > \varepsilon^{-d/(2\alpha)}$ implies $[1, \varepsilon^{-d/(2\alpha)}] \subseteq [1, \sqrt{p}]$. Hence, $\overline{T}(N)$ is decreasing in N on $[1, \varepsilon^{-d/(2\alpha)}]$. It follows that $N_{\text{opt}} = \mathcal{O}(\varepsilon^{-d/(2\alpha)})$ and that $L_{\text{opt}} = \mathcal{O}(1)$.

Case 2.2: The sub-case $\sqrt{p} \le \varepsilon^{-d/(2\alpha)}$.

For this sub-case, N_{opt} and N_{opt} are hard to estimate. However, we can give a rough range of N_{opt}. Since $\overline{T}(N)$ is decreasing in N on $[1, \sqrt{p}]$ and increasing in N on $[p, \infty)$, the minimum of $\overline{T}(N)$ is achieved on $[\sqrt{p}, p]$. Hence, $N_{\text{opt}} \in [\mathcal{O}(\sqrt{p}), \mathcal{O}(p)] \cap [\mathcal{O}(\sqrt{p}), \mathcal{O}(\varepsilon^{-d/(2\alpha)})]$ and $L_{\text{opt}} = \mathcal{O}(\varepsilon^{-d/(2\alpha)}/N_{\text{opt}})$.

5 Conclusion and future work

This paper aims at a quantitative and optimal approximation rate of ReLU FNNs in terms of both width and depth simultaneously to approximate continuous functions. It was shown that ReLU FNNs with width $\mathcal{O}(N)$ and depth $\mathcal{O}(L)$ can approximate an arbitrary continuous function on a d-dimensional cube with an approximation rate $19\sqrt{d}\,\omega_f(N^{-2/d}L^{-2/d})$. In particular, when f is a Hölder continuous function of order α with a Hölder constant λ, the approximation rate is $19\sqrt{d}\,\lambda N^{-2\alpha/d}L^{-2\alpha/d}$ and it is nearly asymptotically tight. We also extended our analysis to the case when the domain of f is irregular and showed the same approximation rate. In practical applications, it is usually believed that real data are sampled from an ε-neighborhood of a $d_{\mathcal{M}}$-dimensional smooth manifold $\mathcal{M} \subseteq [0,1]^d$ with $d_{\mathcal{M}} \ll d$. In the case of an essentially low-dimensional domain, we show an approximation rate $2\omega_f\left(\frac{4\varepsilon}{1-\delta}\sqrt{\frac{d}{d_\delta}}\right) + 19\sqrt{d}\,\omega_f\left(\frac{2\sqrt{d}}{(1-\delta)\sqrt{d_\delta}}N^{-2/d_\delta}L^{-2/d_\delta}\right)$ for ReLU FNNs to approximate f in the ε-neighborhood, $d_\delta = \mathcal{O}\left(d_{\mathcal{M}}\frac{\ln(d/\delta)}{\delta^2}\right)$ for any given $\delta \in (0,1)$.

Besides, we studied how to select the best ReLU FNN to approximate continuous function in parallel computing. In particular, ReLU FNNs with depth $\mathcal{O}(1)$ are the best choices if the number of parallel computing cores p is sufficiently large. ReLU FNNs with width $\mathcal{O}(d)$ are best choices if $p = \mathcal{O}(1)$. The width of best ReLU FNNs is between $\mathcal{O}(\sqrt{p})$ and $\mathcal{O}(p)$ if p is moderate.

We would like to remark that our analysis was based on the fully connected feedforward neural networks and the ReLU activation function. It would be very interesting to generalize our conclusions to neural networks with other types of architectures (e.g., convolutional neural networks) and activation functions (e.g., tanh and sigmoid functions). Besides, if identity maps are allowed in the construction of neural networks as in the residual networks [28], the size of FNNs in our construction can be further optimized. Finally, the proposed analysis could be generalized to other function spaces with explicit

formulas to characterize the approximation error. These will be left as future work.

Acknowledgments

References

[1] O. Abdel-Hamid, A. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, *Convolutional neural networks for speech recognition*, IEEE/ACM Transactions on Audio, Speech, and Language Processing, 22 (2014), pp. 1533–1545.

[2] M. Anthony and P. L. Bartlett, *Neural Network Learning: Theoretical Foundations*, Cambridge University Press, New York, NY, USA, 1st ed., 2009.

[3] R. G. Baraniuk and M. B. Wakin, *Random projections of smooth manifolds*, Foundations of Computational Mathematics, 9 (2009), pp. 51–77.

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

[5] P. Bartlett, V. Maiorov, and R. Meir, *Almost linear VC dimension bounds for piecewise polynomial networks*, Neural Computation, 10 (1998), pp. 217–3.

[6] M. Bianchini and F. Scarselli, *On the complexity of neural network classifiers: A comparison between shallow and deep architectures*, IEEE Transactions on Neural Networks and Learning Systems, 25 (2014), pp. 1553–1565.

[7] E. K. Blum and L. K. Li, *Approximation theory and feedforward networks*, Neural Networks, 4 (1991), pp. 511 – 515.

[8] D. S. Broomhead and D. Lowe, *Multivariable Functional Interpolation and Adaptive Networks*, Complex Systems 2, (1988), pp. 321–355.

[9] J. Cai, D. Li, J. Sun, and K. Wang, *Enhanced expressive power and fast training of neural networks by random projections*, CoRR, abs/1811.09054 (2018).

[10] S. Chen and D. Donoho, *Basis pursuit*, in Proceedings of 1994 28th Asilomar Conference on Signals, Systems and Computers, vol. 1, Oct 1994, pp. 41–44 vol.1.

[11] C. K. Chui, S.-B. Lin, and D.-X. Zhou, *Construction of neural networks for realization of localized deep learning*, Frontiers in Applied Mathematics and Statistics, 4 (2018), p. 14.

[12] D. C. Cireşan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber, *Flexible, high performance convolutional neural networks for image classification*, in Proceedings of the Twenty-Second International Joint Conference on

Artificial Intelligence - Volume Volume Two, IJCAI'11, AAAI Press, 2011, pp. 1237–1242.

[13] D. Costarelli and A. R. Sambucini, *Saturation classes for max-product neural network operators activated by sigmoidal functions*, Results in Mathematics, 72 (2017), pp. 1555 – 1569.

[14] D. Costarelli and G. Vinti, *Convergence for a family of neural network operators in orlicz spaces*, Mathematische Nachrichten, 290 (2017), pp. 226–235.

[15] ——, *Approximation results in orlicz spaces for sequences of kantorovich max-product neural network operators*, Results in Mathematics, 73 (2018), pp. 1 – 15.

[16] G. Cybenko, *Approximation by superpositions of a sigmoidal function*, MCSS, 2 (1989), pp. 303–314.

[17] I. Daubechies, R. DeVore, S. Foucart, B. Hanin, and G. Petrova, *Nonlinear approximation and (deep) relu networks*, vol. abs/1905.02199, 2019.

[18] R. DEVORE and A. RON, *Approximation using scattered shifts of a multivariate function*, Transactions of the American Mathematical Society, 362 (2010), pp. 6205–6229.

[19] R. A. DeVore, *Nonlinear approximation*, Acta Numerica, 7 (1998), p. 51150.

[20] W. E, J. Han, and A. Jentzen, *Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations*, Communications in Mathematics and Statistics, 5 (2017), pp. 349–380.

[21] W. E, C. Ma, and Q. Wang, *A priori estimates of the population risk for residual networks*, ArXiv, abs/1903.02154 (2019).

[22] W. E, C. Ma, and L. Wu, *A priori estimates of the population risk for two-layer neural networks*, Communications in Mathematical Sciences, 17 (2019), pp. 1407 – 1425.

[23] W. E and Q. Wang, *Exponential convergence of the deep neural network approximation for analytic functions*, CoRR, abs/1807.00297 (2018).

[24] J. Han, A. Jentzen, and W. E, *Solving high-dimensional partial differential equations using deep learning*, Proceedings of the National Academy of Sciences, 115 (2018), pp. 8505–8510.

[25] T. Hangelbroek and A. Ron, *Nonlinear approximation using gaussian kernels*, Journal of Functional Analysis, 259 (2010), pp. 203 – 219.

[26] B. Hanin and M. Sellke, *Approximating continuous functions by ReLU nets of minimal width*, (2017).

[27] N. Harvey, C. Liaw, and A. Mehrabian, *Nearly-tight VC-dimension bounds for piecewise linear neural networks*, in Proceedings of the 2017 Conference on Learning Theory, S. Kale and O. Shamir, eds., vol. 65 of Proceedings of Machine Learning Research, Amsterdam, Netherlands, 07–10 Jul 2017, PMLR, pp. 1064–1068.

[28] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016, pp. 770–778.

[29] K. Hornik, *Approximation capabilities of multilayer feedforward networks*, Neural Networks, 4 (1991), pp. 251 – 257.

[30] K. Hornik, M. Stinchcombe, and H. White, *Multilayer feedforward networks are universal approximators*, Neural Networks, 2 (1989), pp. 359 – 366.

[31] M. Hutzenthaler, A. Jentzen, T. Kruse, and T. A. Nguyen, *A proof that rectified deep neural networks overcome the curse of dimensionality in the numerical approximation of semilinear heat equations*, SN Partial Differential Equations and Applications, (2020).

[32] K. Kawaguchi, *Deep learning without poor local minima*, in Advances in Neural Information Processing Systems 29, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, eds., Curran Associates, Inc., 2016, pp. 586–594.

[33] K. Kawaguchi and Y. Bengio, *Depth with nonlinearity creates no bad local minima in resnets*, (2018).

[34] M. J. Kearns and R. E. Schapire, *Efficient distribution-free learning of probabilistic concepts*, J. Comput. Syst. Sci., 48 (1994), pp. 464–497.

[35] A. Krizhevsky, I. Sutskever, and G. E. Hinton, *Imagenet classification with deep convolutional neural networks*, in Advances in Neural Information Processing Systems 25, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, eds., Curran Associates, Inc., 2012, pp. 1097–1105.

[36] V. Kumar, *Introduction to Parallel Computing*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2nd ed., 2002.

[37] V. Kůrková, *Kolmogorov's theorem and multilayer neural networks*, Neural Networks, 5 (1992), pp. 501 – 506.

[38] G. Lewicki and G. Marino, *Approximation of functions of finite variation by superpositions of a sigmoidal function*, Applied Mathematics Letters, 17 (2004), pp. 1147 – 1152.

[39] S. Liang and R. Srikant, *Why deep neural networks?*, CoRR, abs/1610.04161 (2016).

[40] S. Lin, X. Liu, Y. Rong, and Z. Xu, *Almost optimal estimates for approximation and learning by radial basis function networks*, Machine Learning, 95 (2014), pp. 147–164.

[41] B. Llanas and F. Sainz, *Constructive approximate interpolation by neural networks*, Journal of Computational and Applied Mathematics, 188 (2006), pp. 283 – 308.

[42] J. Lu, Z. Shen, H. Yang, and S. Zhang, *Deep Network Approximation for Smooth Functions*, arXiv e-prints, (2020), p. arXiv:2001.03040.

[43] Z. Lu, H. Pu, F. Wang, Z. Hu, and L. Wang, *The expressive power of neural networks: A view from the width*, in Advances in Neural Information Processing Systems 30, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds., Curran Associates, Inc., 2017, pp. 6231–6239.

[44] Z. Lu, H. Pu, F. Wang, Z. Hu, and L. Wang, *The expressive power of neural networks: A view from the width*, CoRR, abs/1709.02540 (2017).

[45] V. Maiorov and A. Pinkus, *Lower bounds for approximation by mlp neural networks*, Neurocomputing, 25 (1999), pp. 81 – 91.

[46] S. G. Mallat and Z. Zhang, *Matching pursuits with time-frequency dictionaries*, IEEE Transactions on Signal Processing, 41 (1993), pp. 3397–3415.

[47] E. J. McShane, *Extension of range of functions*, Bull. Amer. Math. Soc., 40 (1934), pp. 837–842.

[48] H. Montanelli and Q. Du, *New error bounds for deep relu networks using sparse grids*, SIAM Journal on Mathematics of Data Science, 1 (2019), pp. 78–92.

[49] H. Montanelli and H. Yang, *Error bounds for deep relu networks using the kolmogorovarnold superposition theorem*, Neural Networks, 129 (2020), pp. 1 – 6.

[50] H. Montanelli, H. Yang, and Q. Du, *Deep relu networks overcome the curse of dimensionality for bandlimited functions*, Journal of Computational Mathematics, (to appear).

[51] G. F. Montufar, R. Pascanu, K. Cho, and Y. Bengio, *On the number of linear regions of deep neural networks*, in Advances in Neural Information Processing Systems 27, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, eds., Curran Associates, Inc., 2014, pp. 2924–2932.

[52] Q. N. Nguyen and M. Hein, *The loss surface of deep and wide neural networks*, CoRR, abs/1704.08045 (2017).

[53] J. Park and I. W. Sandberg, *Universal approximation using radial-basis-function networks*, Neural Computation, 3 (1991), pp. 246–257.

[54] P. Petersen and F. Voigtlaender, *Optimal approximation of piecewise smooth functions using deep ReLU neural networks*, Neural Networks, 108 (2018), pp. 296 – 330.

[55] P. Petrushev, *Multivariate n-term rational and piecewise polynomial approximation*, Journal of Approximation Theory, 121 (2003), pp. 158 – 197.

[56] D. ROLNICK AND M. TEGMARK, *The power of deeper networks for expressing natural functions*, CoRR, abs/1705.05502 (2017).

[57] I. SAFRAN AND O. SHAMIR, *Depth-width tradeoffs in approximating natural functions with neural networks*, in Proceedings of the 34th International Conference on Machine Learning, D. Precup and Y. W. Teh, eds., vol. 70 of Proceedings of Machine Learning Research, International Convention Centre, Sydney, Australia, 06–11 Aug 2017, PMLR, pp. 2979–2987.

[58] A. SAKURAI, *Tight bounds for the VC-dimension of piecewise polynomial networks*, in Advances in Neural Information Processing Systems, Neural information processing systems foundation, 1999, pp. 323–329.

[59] D. SCHERER, A. MÜLLER, AND S. BEHNKE, *Evaluation of pooling operations in convolutional architectures for object recognition*, in Artificial Neural Networks – ICANN 2010, K. Diamantaras, W. Duch, and L. S. Iliadis, eds., Berlin, Heidelberg, 2010, Springer Berlin Heidelberg, pp. 92–101.

[60] J. SCHMIDT-HIEBER, *Nonparametric regression using deep neural networks with ReLU activation function*, (2017).

[61] U. SHAHAM, A. CLONINGER, AND R. R. COIFMAN, *Provable approximation properties for deep neural networks*, Applied and Computational Harmonic Analysis, 44 (2018), pp. 537 – 557.

[62] Z. SHEN, H. YANG, AND S. ZHANG, *Nonlinear approximation via compositions*, Neural Networks, 119 (2019), pp. 74 – 84.

[63] T. SUZUKI, *Adaptivity of deep reLU network for learning in besov and mixed smooth besov spaces: optimal rate and curse of dimensionality*, in International Conference on Learning Representations, 2019.

[64] H. WHITNEY, *Analytic extensions of differentiable functions defined in closed sets*, Transactions of the American Mathematical Society, 36 (1934), pp. 63–89.

[65] T. F. XIE AND F. L. CAO, *The rate of approximation of gaussian radial basis neural networks in continuous function space*, Acta Mathematica Sinica, English Series, 29 (2013), pp. 295–302.

[66] D. YAROTSKY, *Error bounds for approximations with deep ReLU networks*, Neural Networks, 94 (2017), pp. 103 – 114.

[67] D. YAROTSKY, *Optimal approximation of continuous functions by very deep ReLU networks*, in Proceedings of the 31st Conference On Learning Theory, S. Bubeck, V. Perchet, and P. Rigollet, eds., vol. 75 of Proceedings of Machine Learning Research, PMLR, 06–09 Jul 2018, pp. 639–649.