NONLINEAR APPROXIMATION VIA COMPOSITIONS

Preprint · February 2019

DOI: 10.13140/RG.2.2.30843.98083

Haizhao Yang
National University of Singapore

NONLINEAR APPROXIMATION VIA COMPOSITIONS[*]

ZUOWEI SHEN[†], HAIZHAO YANG[‡], AND SHIJUN ZHANG[§]

Abstract. We study the approximation efficiency of function compositions in nonlinear approximation, especially the case when compositions are implemented using multi-layer feed-forward neural networks (FNNs) with ReLU activation functions. The central question of interest is what are the advantages of function compositions in generating dictionaries and what is the optimal implementation of function compositions via ReLU FNNs especially in modern computing architecture. This question is answered by studying the N-term approximation rate, which is the decrease in error versus the number of computational nodes (neurons) in the approximant, together with parallel efficiency for the first time.

First, for an arbitrary function f on $[0, 1]$, regardless of its smoothness and even the continuity, if f can be approximated via nonlinear approximation using one-hidden-layer ReLU FNNs with an approximation rate $\mathcal{O}(N^{-\eta})$, we quantitatively show that dictionaries with function compositions via deep ReLU FNNs can improve the approximation rate to $\mathcal{O}(N^{-2\eta})$. Second, for Hölder continuous functions of order α with a uniform Lipchitz constant ω on a d-dimensional cube, we show that the N-term approximation via ReLU FNNs with two or three function compositions can achieve an approximation rate $\mathcal{O}(N^{-2\alpha/d})$. The approximation rate can be improved to $\mathcal{O}(L^{-2\alpha/d})$ by composing L times, if N is fixed and sufficiently large; but further compositions cannot achieve the approximation rate $\mathcal{O}(N^{-\alpha L/d})$. Finally, considering the computational efficiency per training iteration in parallel computing, FNNs with $\mathcal{O}(1)$ hidden layers are an optimal choice for approximating Hölder continuous functions if computing resources are enough.

Key words. Deep Neural Networks, ReLU Activation Function, Nonlinear Approximation, Function Composition, Hölder Continuity, Parallel Computing.

AMS subject classifications. 65D15, 65D18, 65D19

1. Introduction.

1.1. Problem Statement.

The goal of approximation theory is to design efficient representations of complicated functions, named as the *target function*, with a small amount of parameters and simple functions, called the *approximants*, which are easy to compute. Obtaining a higher accuracy of the approximation can generally only be achieved by increasing the amount of parameters and computation. The understanding of this trade-off between resolution, parametrization, and computation is a fundamental problem of approximation theory with broad applications in applied and computational mathematics, and computer science. Linear approximation that projects the target function to approximants as linear combinations of basis functions from an N-dimensional linear space has been an useful tool in both analysis and computation (e.g., Fourier series expansion, orthogonal polynomial expansion, finite element analysis, etc). In the case of non-smooth and high-dimensional function approximation, a more favorable technique popularized in recent decades is the nonlinear approximation [13] that does not limit the approximants to come from linear spaces, obtaining sparser representation, cheaper computation, and more robust estimation, and therein emerged the bloom of many breakthroughs in applied mathematics and computer science (e.g., wavelet analysis [9], dictionary learning [50], data compression

[*] Submitted to the editors Feb. 14, 2019.

Funding: H. Yang was supported by the startup of the Department of Mathematics at the National University of Singapore, the grant MOE2018-T2-2-147 from the Ministry of Education in Singapore, and the donation of the Titan Xp GPU from NVIDIA Corporation.

[†] Department of Mathematics, National University of Singapore (matzuows@nus.edu.sg).

[‡] Department of Mathematics, National University of Singapore (haizhao@nus.edu.sg).

[§] Department of Mathematics, National University of Singapore (zhangshijun@u.nus.edu).

and denoising [24, 26], adaptive pursuit [11, 40], compressed sensing [14, 5]).

Typically, nonlinear approximation is a two-stage algorithm that designs a good redundant nonlinear dictionary, $\mathcal{D}$, in its first stage, and identifies the optimal approximant as a linear combination of N elements of $\mathcal{D}$ in the second stage:

$$f(\boldsymbol{x}) \approx g \circ T(\boldsymbol{x}) := \sum_{n=1}^{N} g_n T_n(\boldsymbol{x}), \tag{1.1}$$

where $f(\boldsymbol{x})$ is the target function in a Hilbert space $\mathcal{H}$ associated with a norm denoted as $\|\cdot\|_*$, $\{T_n\} \subseteq \mathcal{D} \subseteq \mathcal{H}$, T is a nonlinear map from $\mathbb{R}^d$ to $\mathbb{R}^N$ with the n-th coordinate being T_n, and g is a linear map from $\mathbb{R}^N$ to $\mathbb{R}$ with the n-th coordinate being $g_n \in \mathbb{R}$. The nonlinear approximation seeks g and T such that

$$\{\{T_n\}, \{g_n\}\} = \underset{\{g_n\}\subseteq\mathbb{R}, \{T_n\}\subseteq\mathcal{D}}{\arg\min} \|f(\boldsymbol{x}) - \sum_{n=1}^{N} g_n T_n(\boldsymbol{x})\|_*,$$

which is also called the N-term approximation. Designing efficient and automatic algorithms for generating dictionary and selecting basis in this kind of nonlinear approximation has been a challenging problem, especially in the multivariate case. One remarkable example of nonlinear approximation is based on one-hidden-layer neural networks. Neural networks give simple and elegant bases of the form $T(\boldsymbol{x}) = \sigma(\boldsymbol{W}\boldsymbol{x}+\boldsymbol{b})$, where $\boldsymbol{W}\boldsymbol{x}+\boldsymbol{b}$ is a linear transform in $\boldsymbol{x}$ with the transformation matrix $\boldsymbol{W}$ (named as the weight matrix) and a shifting vector $\boldsymbol{b}$ (called bias), and σ is a nonlinear function (called the activation function). The N-term approximation of the form

$$f(\boldsymbol{x}) \approx \sum_{n=1}^{N} g_n T_n(\boldsymbol{x}) = \sum_{n=1}^{N} g_n \sigma(\boldsymbol{W}_n \boldsymbol{x} + \boldsymbol{b}_n)$$

includes a wide class of tools, e.g., wavelets pursuits [36, 6], adaptive splines [13, 42], radial basis functions [12, 18, 53], etc. For functions in Besov spaces with smoothness s, [12, 18] constructed an $\mathcal{O}(N^{-s/d})$[1] approximation that is almost optimal [34] and the smoothness cannot be reduced generally [18]. For Hölder continuous functions of order 1 on $[0,1]^d$, [53] essentially constructed an $\mathcal{O}(N^{-\frac{1}{2d}})$ approximation, which is far from the lower bound $\mathcal{O}(N^{-2/d})$ as we shall prove in this paper. Achieving the optimal approximation rate of general continuous functions in constructive approximation, especially in high dimensional spaces, remains an unsolved challenging problem in the literature. We try to tackle this problem via function compositions in the nonlinear approximation and the function composition is implemented with ReLU FNNs.

This paper studies the nonlinear approximation via function compositions, i.e., the approximation of the form

$$f(\boldsymbol{x}) \approx g \circ T^{(L)} \circ T^{(L-1)} \circ \cdots \circ T^{(1)}(\boldsymbol{x}), \tag{1.2}$$

where $T^{(i)}$ is a nonlinear map from $\mathbb{R}^{N_{i-1}}$ to $\mathbb{R}^{N_i}$ for $i = 1, \ldots, L$ with $N_0 = d$ and $N_L = N$. Function compositions enrich the bases of dictionaries in nonlinear approximation and hence could accelerate the convergence rate of the N-term approximation. In terms of numerical computation, there exist efficient algorithms to implement the nonlinear approximation via composition, e.g., multi-layer neural networks, which has

[1] In this paper, we use the big $\mathcal{O}(\cdot)$ notation when we only care about the scaling in terms of the variables inside $(\cdot)$ and the prefactor outside $(\cdot)$ is independent of these variables.

been popularized nowadays with the invention of effective neural network techniques [51, 17, 44], the development of high-performance computing (e.g., hybrid distributed parallel computing with CPUs and GPUs [47, 7]), and the design of new neural architectures [32, 21, 23, 56, 52].

More specifically, given a set of parameters $\boldsymbol{\theta}$ containing all weight matrices and bias vectors, an FNN $\phi(\boldsymbol{x};\boldsymbol{\theta})$ with L hidden layers (the i-th one of which has N_i neurons, i.e., the width of the i-th layer is N_i) to implement the nonlinear approximation via L function compositions in (1.2) can be defined recursively as

$$\boldsymbol{h}_i \coloneqq \boldsymbol{W}_i \tilde{\boldsymbol{h}}_{i-1} + \boldsymbol{b}_i,$$

for $i = 1, \ldots, L+1$, and

$$\tilde{\boldsymbol{h}}_i = \sigma(\boldsymbol{h}_i),$$

for $i = 1, \ldots, L$, where $\boldsymbol{W}_i \in \mathbb{R}^{N_i \times N_{i-1}}$ and $\boldsymbol{b}_i \in \mathbb{R}^{N_i}$ are the weight matrix and the bias vector in the i-th linear transform in ϕ, respectively. To make the notations consistent, we have assumed $N_0 = d$, $N_{L+1} = 1$, $\tilde{\boldsymbol{h}}_0 = \boldsymbol{x}$, and $\phi(\boldsymbol{x};\boldsymbol{\theta}) = \boldsymbol{h}_{L+1}$. To identify the optimal neural network as an approximant to a given target function $f(\boldsymbol{x})$, it is sufficient to solve the following optimization problem

$$\boldsymbol{\theta} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}\big(f(\boldsymbol{x}), \phi(\boldsymbol{x};\boldsymbol{\theta})\big), \tag{1.3}$$

where $\mathcal{L}$ is an appropriate loss function quantifying the difference between $f(\boldsymbol{x})$ and $\phi(\boldsymbol{x};\boldsymbol{\theta})$. With the advance of backpropagation [51] and optimization algorithms [15, 25, 30, 10], good local minima can be identified efficiently for deep neural networks [27, 39, 28].

1.2. Contribution. An immediate question is how to quantify the power of function compositions in nonlinear approximation, if any, and what are the upper and lower bounds of the advantage. Our first result is to show that, for an arbitrary function f on $[0,1]$, if f can be approximated via nonlinear approximation using one-hidden-layer ReLU FNNs with an approximation rate $\mathcal{O}(N^{-\eta})$, we show that dictionaries with two function compositions via deep ReLU FNNs can improve the approximation rate to $\mathcal{O}(N^{-2\eta})$. Second, for Hölder continuous functions of order α with a Lipchitz constant ω on a d-dimensional cube, we show that the N-term approximation via ReLU FNNs with two or three function compositions can achieve the approximation rate $\mathcal{O}(\omega N^{-2\alpha/d})$. Instead of constructing FNNs to approximate traditional approximation tools like polynomials and splines as in existing literature [45, 33, 43, 54, 55, 37, 19, 35, 41, 48, 49] on the approximation theory of deep neural networks, this paper adopts a novel perspective and provides new analysis methods merely based on the structure of FNNs and is able to analyze the approximation capacity of FNNs with $\mathcal{O}(1)$ (i.e., a small constant relative to the width N) layers. More importantly, the approximation rate $\mathcal{O}(\omega N^{-2\alpha/d})$ is **non-asymptotic** (**quantitative**) and valid for all positive integers N with an explicit formula to specify its constant prefactor.

The fact that composing functions once or twice can improve the approximation rate from $\mathcal{O}(N^{-\eta})$ to $\mathcal{O}(N^{-2\eta})$ raises an interesting and widely studied conjecture: whether it is possible to improve the approximation rate to $\mathcal{O}(N^{-L\eta})$ via L times function compositions. Combining existing and our new theories, the answer to this question can be summarized as follows.

(1) If the depth of the composition is $L = \mathcal{O}(1)$, the tight approximation rate is $\mathcal{O}(\omega N^{-2\alpha/d})$, which implies that adding one more layer cannot improve the approximation rate when N is large, although it can improve the approximation accuracy due to the increase of parameters in the approximant.

(2) If FNNs have the same width N, which is a fixed constant number larger than or equal to $2d + 10$, the approximation rate can be improved to a tight rate, $\mathcal{O}(L^{-2\alpha/d})$, as L increases, which is a direct generalization of the result in [55] in terms of nonlinear approximation.

(3) If FNNs have the same width N, which is a constant number less than $d + 1$, the nonlinear approximation in (1.2) loses its approximation capacity according to the theory in [35].

Finally, we analyze the approximation efficiency of neural networks in parallel computing, a very important point of view that was not paid attention to in the literature. In most applications, the efficiency of deep learning computation highly relies on parallel computation and backpropagation. We show that a narrow and very deep neural network is inefficient in practical computation, if its approximation rate is not exponentially better than wide and shallower networks. Hence, neural networks with $\mathcal{O}(1)$ layers is more attractive in modern computational platforms, considering the computational efficiency per training iteration in parallel computing platforms. Our conclusion does not conflict with the current state-of-the-art deep learning research, since most of these successful deep neural networks have depth that is asymptotically $\mathcal{O}(1)$ relative to the width.

1.3. Related work. This paper focuses on the power of compositions in nonlinear approximation and the ReLU FNN is one possible choice to implement the nonlinear approximation. Our analysis on the lower bound of the achievable approximation rate (or equivalently the upper bound of N in the N-term approximation for a given accuracy) is quantitative and proved by construction; while the result for the upper bound of unachievable approximation rate (or equivalently the lower bound of N in the N-term approximation for a given accuracy) is asymptotic. To the best of our knowledge, quantitative analysis in this paper is new; related works in the literature are asymptotic and they can be summarized as follows.

The topic in this paper is related to the study of the expressiveness of neural networks from many other points of views, e.g., in terms of combinatorics [38], topology [4], Vapnik-Chervonenkis (VC) dimension [3, 46, 20] and fat-shattering dimension [29, 1], information theory [41], classical approximation theory [8, 22, 2] etc. Particularly in approximation theory, several recent works have shown the power of depth (or equivalently the power of function compositions) in ReLU FNNs and explained why increasing depth could lead to larger approximation capacity [45, 33, 43, 54, 55, 37, 19, 35, 41, 48, 49].

Approximation theories in [33, 43, 54, 41, 37, 49] aim for target functions with stronger smoothness, e.g. functions in $C^\alpha([0,1]^d)$ with $\alpha \geq 1$, Korobov spaces, or Besev spaces. They send the message that deep FNNs have better approximation capacity than shallow FNNs, especially in the case when the target functions are polynomials. A common idea utilized in these works is to construct FNNs to approximate traditional basis in approximation theory, e.g., polynomials, splines, and sparse grids, which are used to approximate smooth functions. Shallow FNNs therein have minimum depth asymptotically depending on the accuracy ϵ or dimension d.

A more closely related work that proves the optimal rate of approximation of general continuous functions by ReLU FNNs in terms of the number of network weights,

W, and the modulus of continuity of the function was presented in [55]. In particular, if we define the modulus of continuity $\omega_f(r)$ of a function $f:[0,1]^d\to\mathbb{R}$ by

$$\omega_f(r)=\max\{|f(\boldsymbol{x})-f(\boldsymbol{y})|:\boldsymbol{x},\boldsymbol{y}\in[0,1]^d,|\boldsymbol{x}-\boldsymbol{y}|\le r\},$$

then [55] proved that the optimal approximation rate of $f\in C([0,1]^d)$ in the L^∞-norm by a deep ReLU FNN is $\mathcal{O}\big(\omega_f(\mathcal{O}(W^{-2/d}))\big)$, when the FNN has width $\mathcal{O}(d)$, depth $\mathcal{O}(W)$, i.e., the FNN is very deep and narrow.

1.4. Organization. The rest of the paper is organized as follows. Section 2 summarizes the notations throughout this paper. Section 3 presents the main theorems of this paper. In Section 4, numerical tests in parallel computing are presented to support the claims in this paper. Finally, Section 5 concludes this paper with a short discussion.

2. Preliminaries.

For the purpose of convenience, we present notations, definitions, and elementary lemmas used throughout this paper as follows.

2.1. Notations.

- $\mathbb{N}^+$ represents the set of positive integers and $\mathbb{N}=\mathbb{N}^+\cup\{0\}$.
- $\mathbb{Q}$ denotes the set of all rational numbers and $\mathbb{R}$ is for the set of all real numbers.
- Matrices are denoted by bold uppercase letters, e.g., $\boldsymbol{A}\in\mathbb{R}^{m\times n}$ is a real matrix of size $m\times n$, and $\boldsymbol{A}^T$ denotes the transpose of $\boldsymbol{A}$. Correspondingly, $\boldsymbol{A}(i,j)$ is the (i,j)-th entry of $\boldsymbol{A}$; $\boldsymbol{A}(:,j)$ is the j-th column of $\boldsymbol{A}$; $\boldsymbol{A}(i,:)$ is the i-th row of $\boldsymbol{A}$.
- Vectors are denoted as bold lowercase letters, e.g., $\boldsymbol{v}\in\mathbb{R}^n$ is a column vector of size n. Correspondingly, $\boldsymbol{v}(i)$ is the i-th element of $\boldsymbol{v}$. $\boldsymbol{v}=[v_1,\cdots,v_n]^T=\begin{bmatrix}v_1\\ \vdots\\ v_n\end{bmatrix}$ are vectors consisting of numbers $\{v_i\}$ with $\boldsymbol{v}(i)=v_i$.
- The Lebesgue measure is denoted as $\mu(\cdot)$.
- The set difference of two sets A and B is denoted by $A\backslash B:=\{x:x\in A,\ x\notin B\}$. A^c denotes $[0,1]^d\backslash A$ for any $A\subseteq[0,1]^d$. $\overline{A}$ is the closure of the set A in the sense of an approximate metric.
- For any $\xi\in\mathbb{R}$, let $\lfloor\xi\rfloor:=\max\{i:i\le\xi,\ i\in\mathbb{N}\}$ and $\lceil\xi\rceil:=\min\{i:i\ge\xi,\ i\in\mathbb{N}\}$.
- 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 $f(\boldsymbol{n})\le Cg(\boldsymbol{n})$ when $\boldsymbol{n}(i)$ goes to $+\infty$ for all i.
- For any $m,n\in\mathbb{N}^+$, index sets $I_0(m,n)$, $I_1(m,n)$, and $I_2(m,n)$ are defined by

  $$I_0(m,n)=\{0,1,\cdots,m(n+1)\},\tag{2.1}$$

  $$I_1(m,n)=\{j(n+1):j=1,2,\cdots,m\},\tag{2.2}$$

  and

  $$I_2(m,n)=I_0(m,n)\backslash I_1(m,n),\tag{2.3}$$

  respectively.
- For a set of numbers A, and a number x, $A-x:=\{y-x:y\in A\}$.
- Define $\mathrm{Lip}(\omega,\alpha,d)$ as the class of functions g defined on $[0,1]^d$ satisfying the uniformly Lipchitz property of order α with a Lipchitz constant $\omega>0$, i.e. $|g(\boldsymbol{x})-g(\boldsymbol{y})|\le\omega\|\boldsymbol{x}-\boldsymbol{y}\|_{l^2}^\alpha$ for any $\boldsymbol{x},\boldsymbol{y}\in[0,1]^d$.

- Let $\mathrm{CPL}(N)$ be the set of continuous piecewise linear functions with $N-1$ pieces mapping $[0,1]$ to $\mathbb{R}$. The end points of each linear piece are called "break points" in this paper.
- Let $\sigma:\mathbb{R}\to\mathbb{R}$ denote the rectified linear unit (ReLU), i.e. $\sigma(x)=\max\{0,x\}$.
- We will use NN as a ReLU neural network for short and use Python-type notations to specify a class of NN's, e.g., $\mathrm{NN}(\mathrm{c}_1;\ \mathrm{c}_2;\ \cdots;\ \mathrm{c}_m)$ is a set of ReLU FNN's satisfying m conditions given by $\{\mathrm{c}_i\}_{1\le i\le m}$, each of which may specify the number of input dimensions (dim), the total number of nodes in all hidden layers (#node), the number of hidden layers (#layer), the number of total weights (#weight), and the width in each hidden layer (width), etc. For example, $\mathrm{NN}(\mathrm{dim}=2;\ \mathrm{width}=[100,100])$ is a set of NN's ϕ satisfying:
  - ϕ maps from $\mathbb{R}^2$ to $\mathbb{R}$.
  - ϕ has 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,

$$\mathrm{NN}(\mathrm{dim}=d;\ \mathrm{width}=[100,100,100])=\mathrm{NN}(\mathrm{dim}=d;\ \mathrm{width}=[100]^3).$$

- For $\phi\in\mathrm{NN}(\mathrm{dim}=d;\ \mathrm{width}=[N_1,N_2,\cdots,N_L])$, if we define $N_0=d$ and $N_{L+1}=1$, then the architecture of ϕ can be briefly described as follows:

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

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

$$\boldsymbol{h}_i := \boldsymbol{W}_i\tilde{\boldsymbol{h}}_{i-1}+\boldsymbol{b}_i$$

  for $i=1,\ldots,L+1$, and

$$\tilde{\boldsymbol{h}}_i=\sigma(\boldsymbol{h}_i)$$

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

2.2. Definitions. The proofs of our theorems mainly fall into two parts: one for the lower bound of the quantitatively achievable approximation rate and the other part is for the lower bound that is not achievable asymptotically. It is easier to prove the lower bound using the L^1-norm[2] than the L^∞-norm; while it is simpler to prove the lower bound in the L^∞-norm than the L^1-norm. Hence, to make our results tight in the same norm, we introduce a new norm, the $L^{1,\infty}$-norm, in between L^1 and L^∞. The main idea is to introduce a small "don't-care" region via the tools below, allow the approximation error to grow within this region, and uniformly control the approximation error in the rest of the "important" region.

Definition 2.1. *Shrinking function.*

$$\lambda(x,y)=\frac{2^{-1-xy}}{xy}$$

for $(x,y)\in(1,\infty)\times\mathbb{N}^+$.

Note that the shrinking function $\lambda:(1,\infty)\times\mathbb{N}^+\to(0,1/4)$ is a strictly decreasing function in x for a fixed y satisfying

$$\sum_{k=N}^{\infty} dk\lambda(k,d)\le 2^{-dN} \text{ for } N\in\mathbb{N}^+. \tag{2.4}$$

[2] It is easy to generalize our results of the L^1-norm to L^p-norm for $p\in[1,\infty)$.

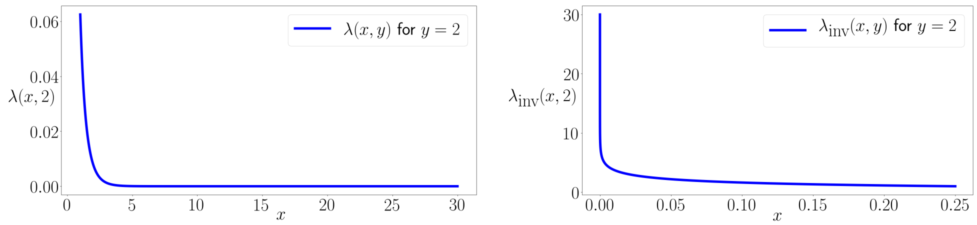
For a fixed $y \in \mathbb{N}^+$, let $\lambda_{\mathrm{inv}}(x, y)$ be the inverse function of $\lambda(x, y)$, i.e.,

$$\lambda_{\mathrm{inv}}(\lambda(x, y), y) = x$$

and

$$\lambda(\lambda_{\mathrm{inv}}(x, y), y) = x.$$

Using the shrinking function $\lambda(x, y)$, we define the shrinking region $\Omega(k, d)$ that gradually shrinks to a set of rational points in $[0, 1]^d$ as k increases in the following definition.



(a) (b)

FIG. 1. *An illustration of $\lambda(x, y)$ in (a) and $\lambda_{\mathrm{inv}}(x, y)$ in (b) for $y = 2$.*

DEFINITION 2.2. *Shrinking region. Let $\mathcal{S}(\ell, d) = \cup_{j=1}^{\ell} [\frac{j}{\ell} - \lambda(\ell, d), \frac{j}{\ell}]$. Then the shrinking region corresponding to integers k and d is defined as*

$$\Omega(k, d) = \cup_{\ell=k}^{\infty} \left(\cup_{i=1}^{d} \{ \boldsymbol{x} = [x_1, \cdots, x_d]^T \in \mathbb{R}^d : x_i \in \mathcal{S}(\ell, d) \} \right). \tag{2.5}$$

Intuitively, the shrinking region $\Omega(k, d)$ can be understood as the "don't-care" region when k is sufficiently large. Figure 2 shows an example of $\mathcal{S}(2, 1)$ (and $\mathcal{S}(3, 1)$), a subset of the shrinking region $\Omega(k, 1)$ when $k \le 2$ (and $k \le 3$).

Now we are ready to introduce the $L^{1,\infty}$-norm using the "don't-care" region.

DEFINITION 2.3. *For any $f \in L^1([0, 1]^d) \cap L^\infty([0, 1]^d)$, we define*[3]

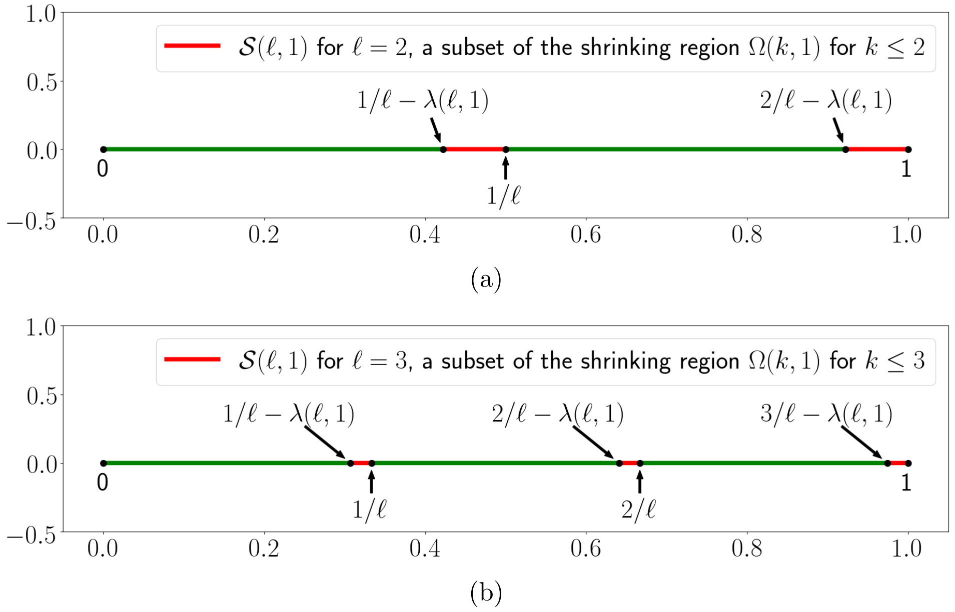
$$\|f\|_{1,\infty} = \sup \left\{ \tfrac{1}{\ln k} \|f\|_{L^\infty(\Omega(k,d)^c)} : k \ge 3,\ k \in \mathbb{N} \right\} + \|f\|_{L^1([0,1]^d)}.$$

The Lebesgue measure of $\Omega(k, d)$ decays quickly and hence $\|f\|_{L^\infty(\Omega(k,d)^c)}$ approaches to $\|f\|_{L^\infty([0,1]^d)}$ as k increases. To obtain a norm weaker than the L^∞-norm, we introduce the term $\ln k$ in Definition 2.3.

To verify that $\|\cdot\|_{1,\infty}$ is a norm in $L^1([0, 1]^d) \cap L^\infty([0, 1]^d)$, it is easy to check the following conditions for any f, f_1, and $f_2 \in L^1([0, 1]^d) \cap L^\infty([0, 1]^d)$:

- First, $\|f\|_{1,\infty} = 0$ implies $0 \le \|f\|_{L^1([0,1]^d)} \le \|f\|_{1,\infty} = 0$. Hence, $f = 0$ iff $\|f\|_{1,\infty} = 0$.
- Second, $\forall\ a \in \mathbb{R}$, we have $\|af\|_{1,\infty} = |a| \cdot \|f\|_{1,\infty}$.

[3] $\frac{1}{\ln k}$ can be replaced with any function in k that goes to 0 when $k \to \infty$, and $\Omega(k, d)$ can be replaced with any set in k and d whose Lebesgue measure goes to 0 when $k \to \infty$ and d is fixed.



(a)

(b)

Fig. 2. *Top: an illustration of $\mathcal{S}(2,1)$ in red, a subset of the shrinking region $\Omega(k,1)$ when $k\le 2$. Bottom: an illustration of $\mathcal{S}(3,1)$ in red, a subset of the shrinking region $\Omega(k,1)$ when $k\le 3$. $\Omega(k,1)$ is the union of all such regions $\mathcal{S}(\ell,1)$ for $\ell\ge k$. We see that $\lambda(x,y)$ is a quickly decreasing function in x and hence the measure of $\Omega(k,1)$ is finite and also quickly decays as k grows.*

- Third, $\forall\ \epsilon>0$, $\exists\ k_0\ge 3$ such that

$$\begin{aligned}
&\|f_1+f_2\|_{1,\infty}\\
&\le \tfrac{1}{\ln k_0}\|f_1+f_2\|_{L^\infty(\Omega(k_0,d)^c)}+\epsilon+\|f_1+f_2\|_{L^1([0,1]^d)}\\
&\le \tfrac{1}{\ln k_0}\|f_1\|_{L^\infty(\Omega(k_0,d)^c)}+\tfrac{1}{\ln k_0}\|f_2\|_{L^\infty(\Omega(k_0,d)^c)}+\|f_1+f_2\|_{L^1([0,1]^d)}+\epsilon\\
&\le \sup\left\{\tfrac{1}{\ln k}\|f_1\|_{L^\infty(\Omega(k,d)^c)}:k\ge 3,\ k\in\mathbb{N}\right\}+\|f_1\|_{L^1([0,1]^d)}\\
&\qquad+\sup\left\{\tfrac{1}{\ln k}\|f_2\|_{L^\infty(\Omega(k,d)^c)}:k\ge 3,\ k\in\mathbb{N}\right\}+\|f_2\|_{L^1([0,1]^d)}+\epsilon\\
&=\|f_1\|_{1,\infty}+\|f_2\|_{1,\infty}+\epsilon.
\end{aligned}$$

Since ϵ is arbitrary, we get $\|f_1+f_2\|_{1,\infty}\le\|f_1\|_{1,\infty}+\|f_2\|_{1,\infty}$.

2.3. Lemmas. Before analyzing deep FNNs, we study the properties of FNNs with only one hidden layer to warm up in Lemma 2.4 below. It indicates that $\mathrm{CPL}(N+1)=\mathrm{NN}(\mathrm{dim}=1;\ \mathrm{width}=[N])$ for any $N\in\mathbb{N}^+$.

Lemma 2.4. *Suppose $\phi\in\mathrm{NN}(\mathrm{dim}=1;\ \mathrm{width}=[N])$ with an architecture:*

$$x\xrightarrow{\boldsymbol{W}_1,\ \boldsymbol{b}_1}\boldsymbol{h}\xrightarrow{\ \ \sigma\ \ }\tilde{\boldsymbol{h}}\xrightarrow{\boldsymbol{W}_2,\ \boldsymbol{b}_2}\phi(x).$$

Then ϕ is a continuous piecewise linear function. Let $\boldsymbol{W}_1=[1,1,\cdots,1]^T\in\mathbb{R}^{N\times 1}$, then we have:

(1) Given a sequence of strictly increasing numbers $x_0,\ x_1,\cdots,\ x_N$, there exists $\boldsymbol{b}_1\in\mathbb{R}^N$ independent of $\boldsymbol{W}_2$ and $\boldsymbol{b}_2$ such that the break points of ϕ are exactly $x_0,\ \cdots,\ x_N$

on the interval $[x_0, x_N]$[4]*.*

(2) Suppose $\{x_i\}_{i\in\{0,1,\cdots,N\}}$ *and* $\boldsymbol{b}_1$ *are given in (1). Given any sequence* $\{y_i\}_{i\in\{0,1,\cdots,N\}}$*, there exist* $\boldsymbol{W}_2$ *and* $\boldsymbol{b}_2$ *such that* $\phi(x_i) = y_i$ *for* $i = 0, 1, \cdots, N$ *and* $\phi(x)$ *is linear on* $[x_i, x_{i+1}]$ *for* $i = 0, 1, \cdots, N-1$*.*

Proof. Part (1) in this lemma follows by setting $\boldsymbol{b}_1 = [-x_0, -x_1, \cdots, -x_{N-1}]^T$. To prove Part (2), denote

$$\boldsymbol{h} = [h_1, h_2, \cdots, h_N]^T \quad \text{and} \quad \tilde{\boldsymbol{h}} = \sigma(\boldsymbol{h}) = [\sigma(h_1), \sigma(h_2), \cdots, \sigma(h_N)]^T.$$

Note that $\phi(x_i) = y_i$, for $i = 0, 1, \cdots, N$, is equivalent to

$$y_i = \phi(x_i) = \boldsymbol{b}_2 + \sum_{j=1}^{N} \tilde{h}_j(x_i)\boldsymbol{W}_2(1, j) := \boldsymbol{A}(i+1, :)\boldsymbol{u},$$

where

$$\boldsymbol{u} = [\boldsymbol{b}_2, \boldsymbol{W}_2(1,1), \boldsymbol{W}_2(1,2), \cdots, \boldsymbol{W}_2(1,N)]^T,$$

and

$$\boldsymbol{A}(i+1, :) = [1, \tilde{h}_1(x_i), \tilde{h}_2(x_i), \cdots, \tilde{h}_N(x_i)],$$

for $i = 0, 1, \cdots, N$. Let $\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}(1,:) \\ \boldsymbol{A}(2,:) \\ \vdots \\ \boldsymbol{A}(N+1,:) \end{bmatrix}$ and $\boldsymbol{v} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_N \end{bmatrix}$, then

$$\phi(x_i) = y_i, \text{ for } i = 0, 1, \cdots, N \quad \Longleftrightarrow \quad \boldsymbol{A}\boldsymbol{u} = \boldsymbol{v}.$$

What remains to show is that $\boldsymbol{A} \in \mathbb{R}^{(N+1)\times(N+1)}$ is non-singular. Note that $\tilde{h}_i$ is determined by $\boldsymbol{W}_1 = [1, 1, \cdots, 1]^T \in \mathbb{R}^{N\times 1}$ and $\boldsymbol{b}_1 = [-x_0, -x_1, \cdots, -x_{N-1}]^T$. More precisely, $\tilde{h}_i(t) = \sigma(t - x_{i-1})$, for $i = 1, 2, \cdots, N$. Then we have $\tilde{h}_m(x_n) = 0$ if $m > n$, and $\tilde{h}_m(x_n) > 0$ if $m \le n$. Hence,

$$\boldsymbol{A} = \begin{bmatrix} 1 & \tilde{h}_1(x_0) & \cdots & \tilde{h}_N(x_0) \\ 1 & \tilde{h}_1(x_1) & \cdots & \tilde{h}_N(x_1) \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \tilde{h}_1(x_N) & \cdots & \tilde{h}_N(x_N) \end{bmatrix} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 1 & \tilde{h}_1(x_1) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \tilde{h}_1(x_N) & \cdots & \tilde{h}_N(x_N) \end{bmatrix},$$

which means $\det(\boldsymbol{A}) = \prod_{i=1}^{N} \tilde{h}_i(x_i) > 0$. So, $\boldsymbol{A}$ is not singular. □

Next, we study the properties of FNNs with two hidden layers. In fact, we can show that the closure of $\mathrm{NN}(\text{dim} = 1;\ \text{width} = [2m, 2n+1])$ contains $\mathrm{CPL}(mn+1)$ for any $m, n \in \mathbb{N}^+$, where the closure is in the sense of L^p-norm for any $p \in [1, \infty)$. The proof of this property relies on the following lemma and will be given in the next section.

Lemma 2.5. *For any* $m, n \in \mathbb{N}^+$*, given any* $m(n+1)+1$ *samples* $(x_i, y_i) \in \mathbb{R}^2$ *with* $0 = x_0 < x_1 < x_2 < \cdots < x_{m(n+1)} = 1$ *and* $y_i \ge 0$ *for* $i = 0, 1, \cdots, m(n+1)$*, there exists* $\phi \in \mathrm{NN}(\text{dim} = 1;\ \text{width} = [2m, 2n+1])$ *satisfying the following conditions:*

[4] We only consider the interval $[x_0, x_N]$ and hence x_0 and x_N are treated as break points. $\phi(x)$ might not have a real break point in a small open neighborhood of x_0 or x_N.

(1) $\phi(x_i) = y_i$ *for* $i \in I_0(m,n)$*;*
(2) ϕ *is linear on each interval* $[x_{i-1}, x_i]$ *for* $i \in I_2(m,n)\backslash\{0\}$*;*
(3)

$$\sup_{x\in[0,1]} |\phi(x)| \le 3 \max_{i\in I_0(m,n)} y_i \prod_{k=1}^{n} \Big(1 + \tfrac{\max\{x_{j(n+1)+n} - x_{j(n+1)+k-1} : j=0,1,\cdots,m-1\}}{\min\{x_{j(n+1)+k} - x_{j(n+1)+k-1} : j=0,1,\cdots,m-1\}}\Big).$$

Proof. Since $\phi \in \mathrm{NN}(\mathrm{dim} = 1;\ \mathrm{width} = [2m, 2n+1])$, the architecture of ϕ is

$$x \xrightarrow{\boldsymbol{W}_1,\ \boldsymbol{b}_1} \boldsymbol{h} \xrightarrow{\sigma} \tilde{\boldsymbol{h}} \xrightarrow{\boldsymbol{W}_2,\ \boldsymbol{b}_2} \boldsymbol{g} \xrightarrow{\sigma} \tilde{\boldsymbol{g}} \xrightarrow{\boldsymbol{W}_3,\ \boldsymbol{b}_3} \phi(x). \tag{2.6}$$

Note that $\boldsymbol{g}$ maps $x \in \mathbb{R}$ to $\boldsymbol{g}(x) \in \mathbb{R}^{2n+1}$ and hence each entry of $\boldsymbol{g}(x)$ itself is a sub-network with one hidden layer. Denote $\boldsymbol{g} = [g_0, g_1^+, g_1^-, \cdots, g_n^+, g_n^-]^T$, then

$$\{g_0, g_1^+, g_1^-, \cdots, g_n^+, g_n^-\} \subseteq \mathrm{NN}(\mathrm{dim} = 1;\ \mathrm{width} = [2m]).$$

Our proof of Lemma 2.5 is mainly based on the repeated applications of Lemma 2.4 to determine parameters of $\phi(x)$ such that Conditions (1) to (3) hold.

Step 1: Determine $\boldsymbol{W}_1$ and $\boldsymbol{b}_1$.

By Lemma 2.4, there exist $\boldsymbol{W}_1 = [1,1,\cdots,1]^T \in \mathbb{R}^{2m\times 1}$ and $\boldsymbol{b}_1 \in \mathbb{R}^{2m}$ such that all sub-networks in $\{g_0, g_1^+, g_1^-, \cdots, g_n^+, g_n^-\}$ have the same set of break points:

$$\big\{x_i : i \in I_1(m,n) \cup (I_1(m,n) - 1) \cup \{0\}\big\},$$

no matter what $\boldsymbol{W}_2$ and $\boldsymbol{b}_2$ are.

Step 2: Determine $\boldsymbol{W}_2$ and $\boldsymbol{b}_2$.

This is the key step of the proof. Our ultimate goal is to set up

$$\boldsymbol{g} = [g_0, g_1^+, g_1^-, \cdots, g_n^+, g_n^-]^T$$

such that, after a nonlinear activation function, there exists a linear combination in the last step of our network (specified by $\boldsymbol{W}_3$ and $\boldsymbol{b}_3$ as shown in (2.6)) that can generate a desired $\phi(x)$ matching the sample points $\{(x_i, y_i)\}_{0\le i\le m(n+1)}$. In the previous step, we have determined the break points of $\{g_0, g_1^+, g_1^-, \cdots, g_n^+, g_n^-\}$ by setting up W_1 and b_1; in this step, we will identify $\boldsymbol{W}_2 \in \mathbb{R}^{(2n+1)\times 2m}$ and $\boldsymbol{b}_2 \in \mathbb{R}^{2n+1}$ to fully determine $\{g_0, g_1^+, g_1^-, \cdots, g_n^+, g_n^-\}$. This will be conducted in two sub-steps.

Step 2.1: Set up.

Suppose $f_0(x)$ is a continuous piecewise linear function defined on $[0,1]$ fitting the given samples $f_0(x_i) = y_i$ for $i \in I_0(m,n)$, and f_0 is linear between any two adjacent points of $\{x_i : i \in I_0(m,n)\}$.

In this step, we are able to choose $\boldsymbol{W}_2(1,:)$ and $\boldsymbol{b}_2(1)$ such that $g_0(x_i) = f_0(x_i)$ for $i \in I_1(m,n) \cup (I_1(m,n)-1) \cup \{0\}$ by Lemma 2.4, since there are $2m+1$ points in $I_1(m,n) \cup (I_1(m,n)-1) \cup \{0\}$. Define $f_1 \coloneqq f_0 - \tilde{g}_0$, where $\tilde{g}_0 = \sigma(g_0) = g_0$ as shown in Equation (2.6), since g_0 is positive by the construction of Lemma 2.4. Then we have $f_1(x_i) = f_0(x_i) - \tilde{g}_0(x_i) = 0$ for $i \in (I_1(m,n) - n - 1) \cup \{m(n+1)\}$. See Figure 3 (a) for an illustration of f_0, f_1, and g_0.

Step 2.2: Mathematical induction.

For each $k \in \{1, 2, \cdots, n\}$, given f_k, we can determine $\boldsymbol{W}_2(2k,:)$, $\boldsymbol{b}_2(2k)$, $\boldsymbol{W}_2(2k+1,:)$, and $\boldsymbol{b}_2(2k+1)$, to completely specify g_k^+ and g_k^-, which in turn can determine f_{k+1}.

Hence, it is only enough to show how to proceed with an arbitrary k, since the initialization of the induction, i.e., f_1 has been constructed in Step 2.1. See Figure 3 (b)-(d) for the illustration of the first two induction steps.

We recursively rely on the fact of f_k that

- $f_k(x_i) = 0$ for $i \in \cup_{\ell=0}^{k-1}(I_1(m,n) - n - 1 + \ell) \cup \{m(n+1)\}$,
- f_k is linear on each interval $[x_{i-1}, x_i]$ for $i \in I_2(m,n)\backslash\{0\}$,

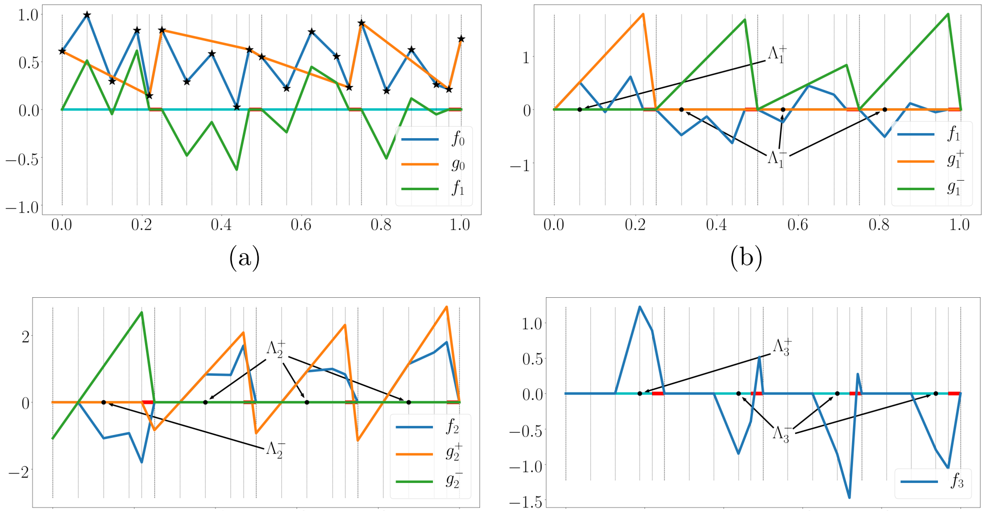
to construct f_{k+1} satisfying similar conditions as follows:

- $f_{k+1}(x_i) = 0$ for $i \in \cup_{\ell=0}^{k}(I_1(m,n) - n - 1 + \ell) \cup \{m(n+1)\}$,
- f_{k+1} is linear on each interval $[x_{i-1}, x_i]$ for $i \in I_2(m,n)\backslash\{0\}$.

The induction process for $\boldsymbol{W}_2(2k,:)$, $\boldsymbol{b}_2(2k)$, $\boldsymbol{W}_2(2k+1,:)$, $\boldsymbol{b}_2(2k+1)$, and f_{k+1} can be divided into four parts.

Step 2.2.1: Define index sets.

Let $\Lambda_k^+ = \{j : f_k(x_{j(n+1)+k}) \geq 0,\ 0 \leq j < m\}$ and $\Lambda_k^- = \{j : f_k(x_{j(n+1)+k}) < 0,\ 0 \leq j < m\}$. The cardinality of $\Lambda_k^+ \cup \Lambda_k^-$ is m. We will use Λ_k^+ and Λ_k^- to generate $2m+1$ samples to determine CPL functions $g_k^+(x)$ and $g_k^-(x)$ in the next step.



(a) (b)

(c) (d)

FIG. 3. *Illustrations of the proof of Lemma 2.5, especially Step 2 of the proof, when $m = n = 4$, with the "don't-care" region in red. (a) Given samples $\{(x_i, y_i) : i = 0, 1, \cdots, m(n+1)\}$ marked with "star" signs, suppose $f_0(x)$ is a CPL function fitting the samples, construct g_0 such that $f_1 = f_0 - \sigma(g_0)$ is closer to 0 than f_0 in the L^∞ sense. (b) Construct g_1^+ and g_1^- such that $f_2 = f_1 - \sigma(g_1^+) + \sigma(g_1^-)$ is closer to 0 than f_1 in the L^∞ sense in a subset of the "important" region. (c) Construct g_2^+ and g_2^- such that $f_3 = f_2 - \sigma(g_2^+) + \sigma(g_2^-)$ is closer to 0 than f_2 in the L^∞ sense in a larger subset of the "important" region. (d) The visualization of f_3, which is 0 in the "important" areas that have been processed and may remain large near the "don't-care" region. f_k will decay quickly outside the "don't-care" region as k increases.*

Step 2.2.2: Determine $\boldsymbol{W}_2(2k,:)$ and $\boldsymbol{b}_2(2k)$.

By Lemma 2.4, we can choose $\boldsymbol{W}_2(2k,:)$ and $\boldsymbol{b}_2(2k)$ to fully determine $g_k^+(x)$ such that each $g_k^+(x_i)$ matches a specific value for $i \in (I_1(m,n) - n - 1) \cup (I_1(m,n) - 1) \cup$

$\{m(n+1)\}$. The values of $\{g_k^+(x_i) : i \in (I_1(m,n)-n-1) \cup (I_1(m,n)-1) \cup \{m(n+1)\}\}$ are specified as:

- If $j \in \Lambda_k^+$, specify the values of $g_k^+(x_{j(n+1)})$ and $g_k^+(x_{j(n+1)+n})$ such that
$$g_k^+(x_{j(n+1)+k-1}) = 0 \quad \text{and} \quad g_k^+(x_{j(n+1)+k}) = f_k(x_{j(n+1)+k}).$$
The existence of the values of $g_k^+(x_{j(n+1)})$ and $g_k^+(x_{j(n+1)+n})$ fulfilling the requirements above comes from the fact that $g_k^+(x)$ is linear on the interval $[x_{j(n+1)}, x_{j(n+1)+n}]$ and $g_k^+(x)$ only depends on the values of $g_k^+(x_{j(n+1)+k-1})$ and $g_k^+(x_{j(n+1)+k})$ on $[x_{j(n+1)}, x_{j(n+1)+n}]$.
Now it is easy to verify that $\tilde{g}_k^+(x) := \sigma(g_k^+(x))$ satisfies
$$\tilde{g}_k^+(x_{j(n+1)+k}) = f_k(x_{j(n+1)+k}) \geq 0 \quad \text{and} \quad \tilde{g}_k^+(x_{j(n+1)+\ell}) = 0$$
for $\ell = 0, 1, \cdots, k-1$, and $\tilde{g}_k^+$ is linear on each interval $[x_{j(n+1)+\ell}, x_{j(n+1)+\ell+1}]$ for $\ell = 0, 1, \cdots, n-1$.
- If $j \in \Lambda_k^-$, specify the values of $g_k^+(x_{j(n+1)})$ and $g_k^+(x_{j(n+1)+n})$ as 0. Then $\tilde{g}_k^+(x) = 0$ on the interval $[x_{j(n+1)}, x_{j(n+1)+n}]$.
- Finally, specify the value of $g_k^+(x)$ at $x = 1$ as 0.

Step 2.2.3: Determine $\boldsymbol{W}_2(2k+1,:)$ and $\boldsymbol{b}_2(2k+1)$.

Similarly, we choose $\boldsymbol{W}_2(2k+1,:)$ and $\boldsymbol{b}_2(2k+1)$ such that $g_k^-(x)$ matches specific values as follows:

- If $j \in \Lambda_k^-$, specify the values of $g_k^-(x_{j(n+1)})$ and $g_k^-(x_{j(n+1)+n})$ such that
$$g_k^-(x_{j(n+1)+k-1}) = 0 \quad \text{and} \quad g_k^-(x_{j(n+1)+k}) = -f_k(x_{j(n+1)+k}).$$
Then $\tilde{g}_k^-(x) := \sigma(g_k^-(x))$ satisfies
$$\tilde{g}_k^-(x_{j(n+1)+k}) = -f_k(x_{j(n+1)+k}) > 0 \quad \text{and} \quad \tilde{g}_k^-(x_{j(n+1)+\ell}) = 0$$
for $\ell = 0, 1, \cdots, k-1$, and $\tilde{g}_k^-(x)$ is linear on each interval $[x_{j(n+1)+\ell}, x_{j(n+1)+\ell+1}]$ for $\ell = 0, 1, \cdots, n-1$.
- If $j \in \Lambda_k^+$, specify the values of $g_k^-(x_{j(n+1)})$ and $g_k^-(x_{j(n+)+n})$ as 0. Then $\tilde{g}_k^-(x) = 0$ on the interval $[x_{j(n+1)}, x_{j(n+1)+n}]$.
- Finally, specify the value of $g_k^-(x)$ at $x = 1$ as 0.

Step 2.2.4: Construct f_{k+1} from g_k^+ and g_k^-.

For the sake of clarity, the properties of g_k^+ and g_k^- constructed in Step 2.2.3 are summarized below:

(1) $g_k^+(1) = g_k^-(1) = 0$;
(2) $f_k(x_i) = \tilde{g}_k^+(x_i) = \tilde{g}_k^-(x_i) = 0$ for $i \in \cup_{\ell=0}^{k-1}(I_1(m,n)-n-1+\ell) \cup \{m(n+1)\}$;
(3) If $j \in \Lambda_k^+$, $\tilde{g}_k^+(x_{j(n+1)+k}) = f_k(x_{j(n+1)+k}) \geq 0$ and $\tilde{g}_k^-(x_{j(n+1)+k}) = 0$;
(4) If $j \in \Lambda_k^-$, $\tilde{g}_k^-(x_{j(n+1)+k}) = -f_k(x_{j(n+1)+k}) > 0$ and $\tilde{g}_k^+(x_{j(n+1)+k}) = 0$;
(5) $\tilde{g}_k^+$ and $\tilde{g}_k^-$ are linear on each interval $[x_{j(n+1)+\ell}, x_{j(n+1)+\ell+1}]$ for $\ell = 0, 1, \cdots, n-1$, $j \in \Lambda_k^+ \cup \Lambda_k^- = \{0, 1, \cdots, m-1\}$. In other words, $\tilde{g}_k^+$ and $\tilde{g}_k^-$ are linear on each interval $[x_{i-1}, x_i]$ for $i \in I_2(m,n) \backslash \{0\}$.

See Figure 3 (a)-(c) for the illustration of g_0, g_1^+, g_1^-, g_2^+, and g_2^-, and to verify their properties as listed just above.

Note that $\Lambda_k^+ \cup \Lambda_k^- = \{0, 1, \cdots, m-1\}$, so $f_k(x_i) - \tilde{g}_k^+(x_i) + \tilde{g}_k^-(x_i) = 0$ for $i \in \cup_{\ell=0}^{k}(I_1(m,n)-n-1+\ell) \cup \{m(n+1)\}$. Now we define $f_{k+1} := f_k - \tilde{g}_k^+ + \tilde{g}_k^-$, then

- $f_{k+1}(x_i) = 0$ for $i \in \cup_{\ell=0}^{k}(I_1(m,n)-n-1+\ell) \cup \{m(n+1)\}$;

- f_{k+1} is linear on each interval $[x_{i-1}, x_i]$ for $i \in I_2(m,n)\backslash\{0\}$.

See Figure 3 (b)-(d) for the illustration of f_1, f_2, and f_3, and to verify their properties as listed just above. This finishes the mathematical induction process. As we can imagine based on Figure 3, when k increases, the support of f_k shrinks to the "don't-care" region.

Step 3: Determine $\boldsymbol{W}_3$ and $\boldsymbol{b}_3$.

With the special vector function $\boldsymbol{g} = [g_0, g_1^+, g_1^-, \cdots, g_n^+, g_n^-]^T$ constructed in Step 2, we are able to specify $\boldsymbol{W}_3$ and $\boldsymbol{b}_3$ to generate a desired $\phi(x)$ matching the sample points $\{(x_i, y_i)\}_{0\le i\le m(n+1)}$ and with a well-controlled L^∞-norm.

In fact, we can simply set $\boldsymbol{W}_3 = [1, 1, -1, 1, -1, \cdots, 1, -1] \in \mathbb{R}^{1\times(2n+1)}$ and $\boldsymbol{b}_3 = 0$, which finishes the construction of $\phi(x)$. The rest of the proof is to verify the properties of $\phi(x)$. Note that

$$\phi = \tilde{g}_0 + \sum_{l=1}^{n} \tilde{g}_l^+ - \sum_{l=1}^{n} \tilde{g}_l^-.$$

By the mathematical induction, we have:

- $f_{n+1} = f_0 - \tilde{g}_0 - \sum_{\ell=1}^n \tilde{g}_\ell^+ + \sum_{\ell=1}^n \tilde{g}_\ell^-$;
- $f_{n+1}(x_i) = 0$ for $i \in \cup_{\ell=0}^n (I_1(m,n) - n - 1 + \ell) \cup \{m(n+1)\} = I_0(m,n)$;
- f_{n+1} is linear on each interval $[x_{i-1}, x_i]$ for $i \in I_2(m,n)\backslash\{0\}$.

Hence

$$\phi = \tilde{g}_0 + \sum_{\ell=1}^{n} \tilde{g}_\ell^+ - \sum_{\ell=1}^{n} \tilde{g}_\ell^- = f_0 - f_{n+1}.$$

Then ϕ satisfies Conditions (1) and (2) of this lemma. It remains to check that ϕ satisfies Condition (3).

By the definition of f_1, we have

$$\sup_{x\in[0,1]} |f_1(x)| \le 2\max\{y_i : i \in I_0(m,n)\}. \tag{2.7}$$

By the induction process in Step 2, for $k \in \{1, 2, \cdots, n\}$, it holds that

$$\sup_{x\in[0,1]} |\tilde{g}_k^+(x)| \le \frac{\max\{x_{j(n+1)+n} - x_{j(n+1)+k-1} : j=0,1,\cdots,m-1\}}{\min\{x_{j(n+1)+k} - x_{j(n+1)+k-1} : j=0,1,\cdots,m-1\}} \sup_{x\in[0,1]} |f_k(x)| \tag{2.8}$$

and

$$\sup_{x\in[0,1]} |\tilde{g}_k^-(x)| \le \frac{\max\{x_{j(n+1)+n} - x_{j(n+1)+k-1} : j=0,1,\cdots,m-1\}}{\min\{x_{j(n+1)+k} - x_{j(n+1)+k-1} : j=0,1,\cdots,m-1\}} \sup_{x\in[0,1]} |f_k(x)|. \tag{2.9}$$

Since either $\tilde{g}_k^+(x)$ or $\tilde{g}_k^-(x)$ is equal to 0 on $[0,1]$, we have

$$\sup_{x\in[0,1]} |\tilde{g}_k^+(x) - \tilde{g}_k^-(x)| \le \frac{\max\{x_{j(n+1)+n} - x_{j(n+1)+k-1} : j=0,1,\cdots,m-1\}}{\min\{x_{j(n+1)+k} - x_{j(n+1)+k-1} : j=0,1,\cdots,m-1\}} \sup_{x\in[0,1]} |f_k(x)|. \tag{2.10}$$

Note that $f_{k+1} = f_k - \tilde{g}_k^+ + \tilde{g}_k^-$, which means

$$\begin{aligned} \sup_{x\in[0,1]} |f_{k+1}(x)| &\le \sup_{x\in[0,1]} |\tilde{g}_k^+(x) - \tilde{g}_k^-(x)| + \sup_{x\in[0,1]} |f_k(x)| \\ &\le \left(\frac{\max\{x_{j(n+1)+n} - x_{j(n+1)+k-1} : j=0,1,\cdots,m-1\}}{\min\{x_{j(n+1)+k} - x_{j(n+1)+k-1} : j=0,1,\cdots,m-1\}} + 1 \right) \sup_{x\in[0,1]} |f_k(x)| \end{aligned} \tag{2.11}$$

for $k \in \{1, 2, \cdots, n\}$. Then we have

$$\sup_{x \in [0,1]} |f_{n+1}(x)| \le 2 \max_{i \in I_0(m,n)} y_i \prod_{k=1}^{n} \Big(\tfrac{\max\{x_{j(n+1)+n} - x_{j(n+1)+k-1} : j=0,1,\cdots,m-1\}}{\min\{x_{j(n+1)+k} - x_{j(n+1)+k-1} : j=0,1,\cdots,m-1\}} + 1 \Big).$$

Hence

$$\begin{aligned} \sup_{x \in [0,1]} |\phi(x)| &= \sup_{x \in [0,1]} |f_0(x) - f_{n+1}(x)| \\ &\le \sup_{x \in [0,1]} |f_0(x)| + \sup_{x \in [0,1]} |f_{n+1}(x)| \\ &\le 3 \max_{i \in I_0(m,n)} y_i \prod_{k=1}^{n} \Big(\tfrac{\max\{x_{j(n+1)+n} - x_{j(n+1)+k-1} : j=0,1,\cdots,m-1\}}{\min\{x_{j(n+1)+k} - x_{j(n+1)+k-1} : j=0,1,\cdots,m-1\}} + 1 \Big). \end{aligned}$$

So, we finish the proof. □

Finally, we present the last lemma below to estimate the number of weights of an FNN if the numbers of nodes and layers are given.

LEMMA 2.6. $\mathrm{NN}(\text{dim} = d;\ \#\text{node} \le N;\ \#\text{layer} \le L) \subseteq \mathrm{NN}(\text{dim} = d;\ \#\text{weight} \le (N+d)^2/2 + N;\ \#\text{layer} \le L)$.

Proof. We only discuss the case when $\#\text{layer} = L$ and the case when $\#\text{layer} < L$ is true similarly. For any $\phi \in \mathrm{NN}(\text{dim} = d;\ \#\text{node} \le N;\ \#\text{layer} = L)$, there exist N_1, N_2, $\cdots$, $N_L \in \mathbb{N}^+$ with $\sum_{i=1}^{L} N_i \le N$ s.t. $\phi \in \mathrm{NN}(\text{dim} = d;\ \text{width} = [N_1, N_2, \cdots, N_L];\ \#\text{layer} = L)$. Then the number of weights of ϕ is bounded by

$$\begin{aligned} &(d+1)N_1 + (N_1+1)N_2 + (N_2+1)N_3 + \cdots + (N_{L-1}+1)N_L \\ &\le dN_1 + N_1N_2 + N_2N_3 + \cdots + N_{L-1}N_L + \sum_{i=1}^{L} N_i \\ &\le \Big(\sum_{i=1}^{L} N_i + d \Big)^2 / 2 + N \\ &\le (N+d)^2/2 + N. \end{aligned}$$

□

3. Main Results.

We present our main results in details in this section. In the first part, we quantitatively prove an achievable approximation rate in the N-term nonlinear approximation by construction, i.e., the lower bound of the approximation rate; while in the second part, we show a lower bound of the approximation rate asymptotically, i.e., no approximant exists asymptotically following the approximation rate. Hence, our constructive approximation in the first part is asymptotically tight. In the third part, we discuss the efficiency of the nonlinear approximation considering the approximation rate and parallel computing in FNNs together.

3.1. Quantitative Achievable Approximation Rate.

THEOREM 3.1. *For any $N \in \mathbb{N}^+$ and $f \in \mathrm{Lip}(\omega, \alpha, d)$ with $\alpha \in (0, 1]$, we have:*

(1) If $d = 1$, $\exists\ \phi \in \mathrm{NN}(\text{dim} = 1;\ \text{width} = [2N, 2N+1])$ such that

$$\|\phi - f\|_{1,\infty} \le 5\omega N^{-2\alpha} \quad \text{for any} \quad N \in \mathbb{N}^+;$$

(2) If $d > 1$, $\exists\ \phi \in \mathrm{NN}(\text{dim} = d;\ \text{width} = [2d\lfloor N^{2/d} \rfloor, 2N+2, 2N+3])$ such that

$$\|\phi - f\|_{1,\infty} \le 3(2\sqrt{d})^{\alpha} \omega N^{-2\alpha/d} \quad \text{for any} \quad N \in \mathbb{N}^+.$$

Proof. Without loss of generality, we assume $f(0) = 0$ and $\omega = 1$. The approximation to a more general case can be easily obtained by changing the bias of the output of ϕ that approximate $f(x) - f(0)$, and then rescale the network by ω.

Step 1: The case $d = 1$.

First, we consider the case $d = 1$. Given any $f \in \text{Lip}(\omega, \alpha, d)$ and $N \in \mathbb{N}^+$, we know $|f(x)| \leq 1$ for any $x \in [0,1]$ since $f(0) = 0$ and $\omega = 1$. Set $\bar{f} = f + 1 \geq 0$, then $0 \leq \sup_{x\in[0,1]} |\bar{f}(x)| \leq 2$. Let $X = \{\frac{i}{N^2} : i = 0, 1, \cdots, N^2\} \cup \{\frac{i}{N} - \delta : i = 1, 2, \cdots, N\}$, where δ is a sufficiently small positive number depending on N, and satisfying (3.2), (3.3), and (3.4). Let us order X as $0 = x_0 < x_1 < \cdots < x_{N(N+1)} = 1$. By Lemma 2.5, given the set of samples $\{(x_i, \bar{f}(x_i)) : i \in \{0, 1, \cdots, N(N+1)\}\}$, there exists $\phi \in \text{NN}(\text{dim} = 1;\ \text{width} = [2N, 2N+1])$ such that

- $\phi(x_i) = \bar{f}(x_i)$ for $i \in I_0(N,N)$;
- ϕ is linear on each interval $[x_{i-1}, x_i]$ for $i \in I_2(N,N)\backslash\{0\}$;
- ϕ has an upper bound estimation:

$$\sup\{\phi(x) : x \in [0,1]\} \leq 6(N+1)!.$$

Recall that the definitions of $I_0(N,N)$, $I_1(N,N)$, and $I_2(N,N)$ are given in Equation (2.1), (2.2), and (2.3), respectively. Define

$$H_0 = \cup_{i\in I_1(N,N)}[x_{i-1}, x_i],$$

which is a part of the “don't-care” region, then it is obvious that in the “important” region,

$$|\bar{f}(x) - \phi(x)| \leq 2N^{-2\alpha} \quad \text{for any } x \in [0,1]\backslash H_0, \tag{3.1}$$

by the fact that $\bar{f} \in \text{Lip}(\omega, \alpha, d)$ and points in X are equispaced. Therefore,

$$\begin{aligned}
\|\bar{f} - \phi\|_{L^1([0,1])} &= \int_0^1 |\bar{f}(x) - \phi(x)| dx \\
&= \sum_{i\in I_0(N,N)\backslash\{0\}} \int_{x_{i-1}}^{x_i} |\bar{f}(x) - \phi(x)| dx \\
&= \sum_{i\in I_1(N,N)} \int_{x_{i-1}}^{x_i} |\bar{f}(x) - \phi(x)| dx + \sum_{i\in I_2(N,N)\backslash\{0\}} \int_{x_{i-1}}^{x_i} |\bar{f}(x) - \phi(x)| dx \\
&\leq N\delta(2 + 6(N+1)!) + N^2(2N^{-2\alpha})N^{-2} \\
&\leq 3N^{-2\alpha},
\end{aligned}$$

where the last inequality comes from the fact that δ is small enough satisfying

$$N\delta(2 + 6(N+1)!) \leq N^{-2\alpha}. \tag{3.2}$$

In fact, we also require

$$\tfrac{6(N+1)!+2}{\ln \lambda_{\text{inv}}(\delta,1)} \leq 2N^{-2\alpha} \tag{3.3}$$

and

$$\tfrac{\lambda_{\text{inv}}(\delta,1)}{N} \in \mathbb{N}^+, \tag{3.4}$$

where λ and λ_{inv} were introduced in (2.4). Then by (3.3), we have

$$\begin{aligned} &\sup\left\{\tfrac{1}{\ln k}\|\bar{f}-\phi\|_{L^\infty(\Omega(k,d)^c)} : k \geq \lambda_{\text{inv}}(\delta,1),\ k\in\mathbb{N}\right\} \\ \leq &\sup\left\{\tfrac{1}{\ln k}(2+6(N+1)!) : k \geq \lambda_{\text{inv}}(\delta,1),\ k\in\mathbb{N}\right\} \\ \leq &\tfrac{2+6(N+1)!}{\ln \lambda_{\text{inv}}(\delta,1)} \\ \leq &\, 2N^{-2\alpha}. \end{aligned}$$

By (3.4) and the definition of $\Omega(k,d)$ in (2.5), if $k \leq \lambda_{\text{inv}}(\delta,1)$, we have

$$\begin{aligned} \Omega(k,1) &\supseteq \Omega(\lambda_{\text{inv}}(\delta,1),1) \\ &\supseteq \cup_{j=1}^{\lambda_{\text{inv}}(\delta,1)}\left[\tfrac{j}{\lambda_{\text{inv}}(\delta,1)} - \lambda(\lambda_{\text{inv}}(\delta,1),1),\ \tfrac{j}{\lambda_{\text{inv}}(\delta,1)}\right] \\ &= \cup_{j=1}^{\lambda_{\text{inv}}(\delta,1)}\left[\tfrac{j}{\lambda_{\text{inv}}(\delta,1)} - \delta,\ \tfrac{j}{\lambda_{\text{inv}}(\delta,1)}\right] \\ &\supseteq \cup_{j=1}^{N}\left[\tfrac{j}{N}-\delta,\ \tfrac{j}{N}\right] \\ &= H_0. \end{aligned}$$

By (3.1), it holds that

$$\begin{aligned} &\sup\left\{\tfrac{1}{\ln k}\|\bar{f}-\phi\|_{L^\infty(\Omega(k,1)^c)} : 3 \leq k \leq \lambda_{\text{inv}}(\delta,1),\ k\in\mathbb{N}\right\} \\ \leq &\sup\left\{\tfrac{1}{\ln k}\|\bar{f}-\phi\|_{L^\infty(H_0^c)} : 3 \leq k \leq \lambda_{\text{inv}}(\delta,1),\ k\in\mathbb{N}\right\} \\ \leq &\tfrac{1}{\ln 3}\|\bar{f}-\phi\|_{L^\infty(H_0^c)} \\ \leq &\, 2N^{-2\alpha}. \end{aligned}$$

Then we get

$$\begin{aligned} &\sup\left\{\tfrac{1}{\ln k}\|\bar{f}-\phi\|_{L^\infty(\Omega(k,1)^c)} : k \geq 3,\ k\in\mathbb{N}\right\} \\ = &\max\Bigg\{\sup\left\{\tfrac{1}{\ln k}\|\bar{f}-\phi\|_{L^\infty(\Omega(k,1)^c)} : k \geq \lambda_{\text{inv}}(\delta,1),\ k\in\mathbb{N}\right\}, \\ &\qquad \sup\left\{\tfrac{1}{\ln k}\|\bar{f}-\phi\|_{L^\infty(\Omega(k,1)^c)} : 3 \leq k \leq \lambda_{\text{inv}}(\delta,d),\ k\in\mathbb{N}\right\}\Bigg\} \\ \leq &\, 2N^{-2\alpha}. \end{aligned}$$

Therefore, we have

$$\begin{aligned} \|\bar{f}-\phi\|_{1,\infty} &= \sup_k\left\{\tfrac{1}{\ln k}\|\bar{f}-\phi\|_{L^\infty(\Omega(k,1)^c)} : k \geq 3,\ k\in\mathbb{N}\right\} + \|\bar{f}-\phi\|_{L^1([0,1])} \\ &\leq 2N^{-2\alpha} + 3N^{-2\alpha} \\ &\leq 5N^{-2\alpha}. \end{aligned}$$

Note that $f-(\phi-1) = f+1-\phi = \bar{f}-\phi$. Hence, $\phi-1 \in \mathrm{NN}(\dim = 1;\ \text{width} = [2N, 2N+1])$ and $\|f-(\phi-1)\| \leq 5N^{-2\alpha}$. So, we finish the proof for the case $d=1$.

Step 2: The case $d > 1$.

Next, we consider the case $d > 1$. The main idea is to project the d-dimensional approximation problem into a one-dimensional approximation problem and use the results proved above. For any $N \in \mathbb{N}^+$, let $n = \lfloor N^{2/d} \rfloor$ and δ be a sufficiently small positive number depending on N and d, and satisfying (3.10), (3.11), and (3.12).

We will divide the d-dimensional cube into n^d small non-overlapping sub-cubes (see Figure 4 for an illustration when $d = 3$ and $n = 3$), each of which is associated with a representative point, e.g., a vertex of the sub-cube. Due to the continuity, the target function f can be represented by their values at the representative points. We project these representatives to one-dimensional samples via a ReLU FNN ψ and construct a ReLU FNN $\bar{\phi}$ to fit them. Finally, the ReLU FNN ϕ on the d-dimensional space approximating f can be constructed by $\phi = \bar{\phi} \circ \psi$. The precise construction can be found below.

By Lemma 2.4, there exists $\psi_0 \in \mathrm{NN}(\text{dim} = 1;\ \text{width} = [2n])$ such that

- $\psi_0(1) = n - 1$, and $\psi_0(\frac{i}{n}) = \psi_0(\frac{i+1}{n} - \delta) = i$ for $i = 0, 1, \cdots, n-1$;
- ψ_0 is linear between any two adjacent points of $\{\frac{i}{n} : i = 0, 1, \cdots, n\} \cup \{\frac{i}{n} - \delta : i = 1, 2, \cdots, n\}$.

Define the projection map ψ by

$$\psi(\boldsymbol{x}) = \sum_{i=1}^{d} \tfrac{1}{n^i} \psi_0(x_i), \ \ \boldsymbol{x} = [x_1, x_2, \cdots, x_d]^T \in [0,1]^d. \tag{3.5}$$

Note that ψ is in fact a ReLU FNN in $\mathrm{NN}(\text{dim} = 1;\ \text{width} = [2dn])$.

Given $f \in \mathrm{Lip}(\omega, \alpha, d)$, then $|f(\boldsymbol{x})| \le \sqrt{d}$ for any $\boldsymbol{x} \in [0,1]^d$ since $f(0) = 0$, $\omega = 1$, and $\alpha \in (0,1]$. Define $\bar{f} = f + \sqrt{d}$, then $0 \le \bar{f}(\boldsymbol{x}) \le 2\sqrt{d}$ for any $\boldsymbol{x} \in [0,1]^d$. Hence, we have obtained

$$\left\{ \Big(\sum_{i=1}^{d} \tfrac{\theta_i}{n^i},\ \bar{f}(\tfrac{\boldsymbol{\theta}}{n}) \Big) : \boldsymbol{\theta} = [\theta_1, \theta_2, \cdots, \theta_d]^T \in \{0, 1, \cdots, n-1\}^d \right\} \cup \{(1, 0)\}$$

as a set of $n^d + 1$ samples of a one-dimensional function. By Lemma 2.4, there exists $\bar{\phi} \in \mathrm{NN}(\text{dim} = 1;\ \text{width} = [2\lceil n^{d/2} \rceil, 2\lceil n^{d/2} \rceil + 1])$ such that

$$\bar{\phi}\Big(\sum_{i=1}^{d} \tfrac{\theta_i}{n^i} \Big) = \bar{f}\left(\tfrac{\boldsymbol{\theta}}{n}\right), \quad \text{where } \boldsymbol{\theta} = [\theta_1, \theta_2, \cdots, \theta_d]^T \in \{0, 1, \cdots, n-1\}^d, \tag{3.6}$$

and

$$\sup_{t \in [0,1]} |\bar{\phi}(t)| \le 6\sqrt{d}\left(\lceil n^{d/2} \rceil + 1\right)!. \tag{3.7}$$

Since the range of ψ on $[0,1]^d$ is a subset of $[0,1]$, there exists an FNN

$$\phi \in \mathrm{NN}\left(\text{dim} = d;\ \text{width} = [2nd, 2\lceil n^{d/2} \rceil, 2\lceil n^{d/2} \rceil + 1]\right)$$

defined via

$$\phi(\boldsymbol{x}) = \bar{\phi} \circ \psi(\boldsymbol{x}) \quad \text{ for } \boldsymbol{x} \in [0,1]^d,$$

such that

$$\sup_{\boldsymbol{x} \in [0,1]^d} |\phi(\boldsymbol{x})| \le 6\sqrt{d}\left(\lceil n^{d/2} \rceil + 1\right)!. \tag{3.8}$$

Define

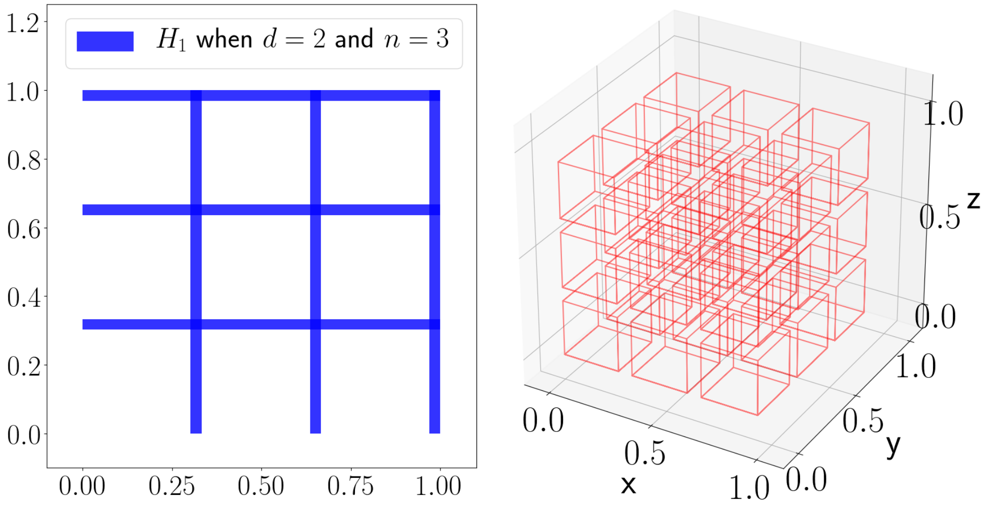
$$H_1 = \cup_{j=1}^{d} \left\{ \boldsymbol{x} = [x_1, x_2, \cdots, x_d]^T \in [0,1]^d : x_j \in \cup_{i=1}^{n} [\tfrac{i}{n} - \delta, \tfrac{i}{n}] \right\},$$

which is a part of the "don't-care" region and it separates the d-dimensional cube into n^d important sub-cubes as illustrated in Figure 4. To index these resulting d-dimensional smaller sub-cubes, define

$$Q_{\boldsymbol{\theta}} = \left\{ \boldsymbol{x} = [x_1, x_2, \cdots, x_d]^T \in [0,1]^d : x_i \in [\tfrac{\theta_i}{n}, \tfrac{\theta_i + 1}{n} - \delta],\ i = 1, 2, \cdots, d \right\}$$

for each d-dimensional index $\boldsymbol{\theta} = [\theta_1, \theta_2, \cdots, \theta_d]^T \in \{0, 1, \cdots, n-1\}^d$. By (3.5), (3.6), and the definition of ψ_0, for any $\boldsymbol{x} = [x_1, x_2, \cdots, x_d]^T \in Q_{\boldsymbol{\theta}}$, we have

$$\phi(\boldsymbol{x}) = \bar{\phi}(\psi(\boldsymbol{x})) = \bar{\phi}\Big(\sum_{i=1}^{d} \tfrac{1}{n^i}\psi_0(x_i)\Big) = \bar{\phi}\Big(\sum_{i=1}^{d} \tfrac{1}{n^i}\theta_i\Big) = \bar{f}\left(\tfrac{\boldsymbol{\theta}}{n}\right).$$



(a) (b)

FIG. 4. *An illustration of H_1 and n^d small non-overlapping sub-cubes that H_1 separates when $n = 3$. (a) In the case when $d = 2$, H_1 in blue separates $[0,1]^2$ into $n^d = 9$ small sub-cubes. (b) In the case when $d = 3$, H_1 (no color) separates $[0,1]^3$ into $n^d = 27$ small sub-cubes in red.*

Then

$$|\bar{f}(\boldsymbol{x}) - \phi(\boldsymbol{x})| = |\bar{f}(\boldsymbol{x}) - \bar{f}\left(\tfrac{\boldsymbol{\theta}}{n}\right)| \le (\sqrt{d}/n)^{\alpha}, \tag{3.9}$$

for any $\boldsymbol{x} \in Q_{\boldsymbol{\theta}}$. Because the Lebesgue measure $\mu(H_1) \le dn\delta$,

$$[0,1]^d = \cup_{\boldsymbol{\theta} \in \{0,1,\cdots,n-1\}^d} Q_{\boldsymbol{\theta}} \cup H_1,$$

(3.8), and (3.9), we have

$$\begin{aligned}
\|\bar{f} - \phi\|_{L^1([0,1]^d)} &= \int_{[0,1]^d} |\bar{f} - \phi| d\boldsymbol{x} \\
&= \int_{H_1} |\bar{f} - \phi| d\boldsymbol{x} + \int_{[0,1]^d \backslash H_1} |\bar{f} - \phi| d\boldsymbol{x} \\
&\le \mu(H_1)\Big(2\sqrt{d} + 6\sqrt{d}\big(\lceil n^{d/2} \rceil + 1\big)!\Big) + \sum_{\boldsymbol{\theta} \in \{0,1,\cdots,n-1\}^d} \int_{Q_{\boldsymbol{\theta}}} |\bar{f} - \phi| d\boldsymbol{x} \\
&\le 2n\delta d\sqrt{d}\Big(1 + 3\big(\lceil n^{d/2} \rceil + 1\big)!\Big) + \sum_{\boldsymbol{\theta} \in \{0,1,\cdots,n-1\}^d} (\sqrt{d}/n)^{\alpha} \mu(Q_{\boldsymbol{\theta}}) \\
&\le (d^{\alpha/2} + 1) n^{-\alpha},
\end{aligned}$$

where the last inequality comes from the fact that δ is small enough such that

$$2n\delta d\sqrt{d}\Big(1 + 3\big(\lceil n^{d/2} \rceil + 1\big)!\Big) \le n^{-\alpha}. \tag{3.10}$$

Together with the requirements

$$\frac{2\sqrt{d}+6\sqrt{d}\big(\lceil n^{d/2}\rceil+1\big)!}{\ln\lambda_{\mathrm{inv}}(\delta,d)} \le n^{-\alpha}, \tag{3.11}$$

and

$$\frac{\lambda_{\mathrm{inv}}(\delta,d)}{n} \in \mathbb{N}^{+}, \tag{3.12}$$

by Equation (3.11), we have

$$\begin{aligned}
&\sup\Big\{\tfrac{1}{\ln k}\|\bar{f}-\phi\|_{L^\infty(\Omega(k,d)^c)} : k \ge \lambda_{\mathrm{inv}}(\delta,d),\ k\in\mathbb{N}\Big\}\\
\le{}& \sup\Big\{\tfrac{1}{\ln k}\Big(2\sqrt{d}+6\sqrt{d}\big(\lceil n^{d/2}\rceil+1\big)!\Big) : k \ge \lambda_{\mathrm{inv}}(\delta,d),\ k\in\mathbb{N}\Big\}\\
\le{}& \tfrac{2\sqrt{d}+6\sqrt{d}(\lceil n^{d/2}\rceil+1)!}{\ln\lambda_{\mathrm{inv}}(\delta,d)}\\
\le{}& n^{-\alpha}.
\end{aligned}$$

By the definition of $\Omega(k,d)$, if $k \le \lambda_{\mathrm{inv}}(\delta,d)$, we have

$$\begin{aligned}
&\Omega(k,d)\\
\supseteq{}& \Omega(\lambda_{\mathrm{inv}}(\delta,d),d)\\
\supseteq{}& \cup_{i=1}^{d}\Big\{\boldsymbol{x}\in[0,1]^d : x_i \in \cup_{j=1}^{\lambda_{\mathrm{inv}}(\delta,d)}\big[\tfrac{j}{\lambda_{\mathrm{inv}}(\delta,d)} - \lambda(\lambda_{\mathrm{inv}}(\delta,d),d),\ \tfrac{j}{\lambda_{\mathrm{inv}}(\delta,d)}\big]\Big\}\\
\supseteq{}& \cup_{i=1}^{d}\Big\{\boldsymbol{x}\in[0,1]^d : x_i \in \cup_{j=1}^{n}\big[\tfrac{j}{n}-\delta,\ \tfrac{j}{n}\big]\Big\}\\
={}& H_1.
\end{aligned}$$

By (3.9), it holds that

$$\begin{aligned}
&\sup\Big\{\tfrac{1}{\ln k}\|\bar{f}-\phi\|_{L^\infty(\Omega(k,d)^c)} : 3 \le k \le \lambda_{\mathrm{inv}}(\delta,d),\ k\in\mathbb{N}\Big\}\\
\le{}& \sup\Big\{\tfrac{1}{\ln k}\|\bar{f}-\phi\|_{L^\infty(H_1^c)} : 3 \le k \le \lambda_{\mathrm{inv}}(\delta,d),\ k\in\mathbb{N}\Big\}\\
\le{}& \tfrac{1}{\ln 3}\|\bar{f}-\phi\|_{L^\infty(H_1^c)}\\
\le{}& (\sqrt{d}/n)^{\alpha}.
\end{aligned}$$

Furthermore,

$$\begin{aligned}
&\sup\Big\{\tfrac{1}{\ln k}\|\bar{f}-\phi\|_{L^\infty(\Omega(k,d)^c)} : k \ge 3,\ k\in\mathbb{N}\Big\}\\
={}& \max\bigg\{\sup\Big\{\tfrac{1}{\ln k}\|\bar{f}-\phi\|_{L^\infty(\Omega(k,d)^c)} : 3 \le k \le \lambda_{\mathrm{inv}}(\delta,d),\ k\in\mathbb{N}\Big\},\\
&\qquad \sup\Big\{\tfrac{1}{\ln k}\|\bar{f}-\phi\|_{L^\infty(\Omega(k,d)^c)} : k \ge \lambda_{\mathrm{inv}}(\delta,1),\ k\in\mathbb{N}\Big\}\bigg\}\\
\le{}& \max\{(\sqrt{d}/n)^{\alpha}, n^{-\alpha}\}\\
={}& (\sqrt{d}/n)^{\alpha}.
\end{aligned}$$

Therefore,

$$\begin{aligned}
\|\bar{f}-\phi\|_{1,\infty} &= \sup\Big\{\tfrac{1}{\ln k}\|\bar{f}-\phi\|_{L^\infty(\Omega(k,d)^c)} : k \ge 3,\ k\in\mathbb{N}\Big\} + \|\bar{f}-\phi\|_{L^1([0,1])}\\
&\le (\sqrt{d}/n)^{\alpha} + (d^{\alpha/2}+1)n^{-\alpha}\\
&\le 3d^{\alpha/2}n^{-\alpha}.
\end{aligned}$$

Note that $f-(\phi-\sqrt{d}) = f+\sqrt{d}-\phi = \bar{f}-\phi$. Hence, $\phi-\sqrt{d} \in \mathrm{NN}(\mathrm{dim}=d;\ \mathrm{width}=[2nd, 2\lceil n^{d/2}\rceil, 2\lceil n^{d/2}\rceil+1])$ and $\|f-(\phi-\sqrt{d})\| \le 3d^{\alpha/2}n^{-\alpha}$. Since $n=\lfloor N^{2/d}\rfloor$, we have $\lceil n^{d/2}\rceil \le N+1$. Therefore,

$$\begin{aligned}\phi-\sqrt{d} &\in \mathrm{NN}(\mathrm{dim}=d;\ \mathrm{width}=[2nd, 2\lceil n^{d/2}\rceil, 2\lceil n^{d/2}\rceil+1])\\ &\subseteq \mathrm{NN}(\mathrm{dim}=d;\ \mathrm{width}=[2d\lfloor N^{2/d}\rfloor, 2N+2, 2N+3])\end{aligned}$$

and

$$\|f-(\phi-\sqrt{d})\|_{1,\infty} \le 3d^{\alpha/2}n^{-\alpha} = 3d^{\alpha/2}\lfloor N^{2/d}\rfloor^{-\alpha} \le 3d^{\alpha/2}(N^{2/d}/2)^{-\alpha} = 3(2\sqrt{d})^{\alpha}N^{-2\alpha/d},$$

where the second inequality comes from the fact $\lfloor x\rfloor \ge \frac{x}{2}$ for any $x\in[1,\infty)$. So, we finish the proof for the case $d>1$. □

Theorem 3.1 shows that, for Hölder continuous functions with a Lipchitz constant ω of order α on a d-dimensional cube, the N-term approximation via ReLU FNNs with two or three function compositions can achieve the approximation rate $\mathcal{O}(\omega N^{-2\alpha/d})$. Instead of constructing FNNs to approximate traditional approximation tools like polynomials and splines as in existing literature [45, 33, 43, 54, 55, 37, 19, 35, 41, 48, 49] on the approximation theory of deep neural networks, the analysis techniques in Theorem 3.1 is new and merely based on the structure of FNNs. Hence, a noteworthy advantage of Theorem 3.1 is that, it is able to analyze the approximation capacity of FNNs with $\mathcal{O}(1)$ layers with a non-asymptotic N and an explicit formula for the constant prefactor.

Following the same proof as in Theorem 3.1, we have the following corollary.

Corollary 3.2. *$\forall\ m,n\in\mathbb{N}^+$, the closure of* $\mathrm{NN}(\mathrm{dim}=1;\ \mathrm{width}=[2m,2n+1])$ *contains* $\mathrm{CPL}(mn+1)$ *in the sense of $L^{1,\infty}$-norm.*

An immediate implication of Corollary 3.2 is that, for an arbitrary function f on $[0,1]$, if f can be approximated via nonlinear approximation using one-hidden-layer ReLU FNNs with an approximation rate $\mathcal{O}(N^{-\eta})$, we show that dictionaries with two function compositions via deep ReLU FNNs can improve the approximation rate to $\mathcal{O}(N^{-2\eta})$. We conjecture that this conclusion could be extended to higher dimensions.

3.2. Asymptotic Unachievable Approximation Rate.

In Section 3.1, we have analyzed the approximation capacity of ReLU FNNs in the nonlinear approximation for general continuous functions by construction. In this section, we will show that the construction in Section 3.1 is asymptotically tight via showing the approximation lower bound in Theorem 3.3 below.

Theorem 3.3. *$\forall\ L\in\mathbb{N}^+$, $\rho>0$, and $C>0$, there exists $N_0=N_0(L,\rho,C)>0$ and $f\in\mathrm{Lip}(\omega,\alpha,d)$ with $\alpha\in(0,1]$ such that for any integer $N\ge N_0$*

$$\inf_{\phi\in\mathrm{NN}(\mathrm{dim}=d;\ \#\mathrm{node}\le N;\ \#\mathrm{layer}\le L)} \|\phi(\boldsymbol{x})-f(\boldsymbol{x})\|_{1,\infty} \ge C\omega N^{-(2\alpha/d+\rho)}.$$

Proof. Without loss of generality, we assume $\omega=1$; in the case of $\omega\ne1$, the proof is similar by rescaling $f\in\mathrm{Lip}(\omega,\alpha,d)$ and FNNs with ω.

Let us denote the VC dimension of a function set $\mathcal{F}$ by $\mathrm{VCDim}(\mathcal{F})$. By [20], there exists $C_1>0$ such that

$$\mathrm{VCDim}\big(\mathrm{NN}(\mathrm{dim}=d;\ \#\mathrm{weight}\le W;\ \#\mathrm{layer}\le L)\big) \le C_1 W L \ln W. \tag{3.13}$$

By Lemma 2.6 and Equation (3.13), we have

$$\begin{aligned} &\mathrm{VCDim}\big(\mathrm{NN}(\mathrm{dim}=d;\ \#\mathrm{node}\le N;\ \#\mathrm{layer}\le L)\big)\\ &\le \mathrm{VCDim}\left(\mathrm{NN}(\mathrm{dim}=d;\ \#\mathrm{weight}\le (N+d)^2/2+N;\ \#\mathrm{layer}\le L)\right)\\ &\le C_1\big((N+d)^2/2+N\big)L\ln\left((N+d)^2/2+N\right). \end{aligned} \tag{3.14}$$

Next, we will prove Theorem 3.3 by contradiction. Assuming that Theorem 3.3 is not true, we can show the following claim, which will lead to a contradiction in the end.

Claim 3.4. *There exist $L\in\mathbb{N}^+$, $\rho>0$, and $C_2>0$ such that $\forall\ N_0>0$ and $f\in\mathrm{Lip}(1,\alpha,d)$, $\exists\ N\ge N_0$ and $\phi\in\mathrm{NN}(\mathrm{dim}=d;\ \#\mathrm{node}\le N;\ \#\mathrm{layer}\le L)$ such that*

$$\|f-\phi\|_{1,\infty}\le C_2 N^{-(2\alpha/d+\rho)}.$$

We will estimate a lower bound of

$$\mathrm{VCDim}\big(\mathrm{NN}(\mathrm{dim}=d;\ \#\mathrm{node}\le N;\ \#\mathrm{layer}\le L)\big) \tag{3.15}$$

using Claim 3.4, and this lower bound is asymptotically larger than

$$b_u := C_1\big((N+d)^2/2+N\big)L\ln\big((N+d)^2/2+N\big) \tag{3.16}$$

in (3.14), leading to a contradiction that disproves the assumption that "Theorem 3.3 is not true".

Recall that the VCDim of a class of functions is defined as the cardinality of the largest set of points that this class of functions can shatter. Clearly, the lower bound of (3.15) is larger than or equal to the VCDim of a special subset of FNNs $\mathcal{B}=\{\phi_\beta:\beta\in\mathscr{B}\}\subseteq\mathrm{NN}(\mathrm{dim}=d;\ \#\mathrm{node}\le N;\ \#\mathrm{layer}\le L)$, where $\mathscr{B}$ is a set defined later. Hence, the remaining proof will follow the steps summarized below.

(1) Construct a class of functions $\mathcal{F}=\{f_\beta:\beta\in\mathscr{B}\}\subseteq\mathrm{Lip}(1,\alpha,d)$ such that $\mathcal{F}$ can scatter b_ℓ points, where $\mathscr{B}$ is a set defined below;
(2) Use Claim 3.4 to identify $\mathcal{B}$ such that the function values of $f_\beta\in\mathcal{F}$ and $\phi_\beta\in\mathcal{B}$ have the same sign at b_ℓ points for each $\beta\in\mathscr{B}$. Hence, $\mathcal{B}$ can shatter these b_ℓ points since $\mathcal{F}$ can;
(3) Finally, we reach to a contradiction:

$$b_u<b_\ell\le\mathrm{VCDim}\big(\mathcal{B}\big)\le b_u.$$

More details can be found below.

Step 1: Construct $\mathcal{F}=\{f_\beta:\beta\in\mathscr{B}\}\subset\mathrm{Lip}(1,\alpha,d)$ that scatters b_ℓ points.

Let $Q(\boldsymbol{x}_0,r)\subseteq[0,1]^d$ be a cube, whose center and sidelength are $\boldsymbol{x}_0$ and r, respectively. If Q_1 and Q_2 have the same center and the sidelength of Q_2 is r_0 times that of Q_1, then we denote $Q_2=r_0Q_1$. Besides, for any cube $Q=Q(\boldsymbol{x}_0,r)\in[0,1]^d$ centered at x_0 with a sidelength r, we associate Q with a function $g_Q:[0,1]^d\to\mathbb{R}$, which exists and is uniquely determined by following conditions, such that:

- $g_Q(\boldsymbol{x}_0)=(r/2)^\alpha/2$;
- $g_Q(\boldsymbol{x})=0$ for any $\boldsymbol{x}\notin Q\backslash\partial Q$, where ∂Q is the boundary of Q;
- g_Q is linear on the line that connects $\boldsymbol{x}_0$ and $\boldsymbol{x}$ for any $\boldsymbol{x}\in\partial Q$.

For any N, ρ, d, and α, let $m = \lfloor N^{2/d+\rho/(2\alpha)} \rfloor$. Partition $[0,1]^d$ into m^d non-overlapping sub-cubes $\{Q_J\}_J$ as follows:

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

for any index $\boldsymbol{J} = [j_1, j_2, \cdots, j_d]^T \in \{1, 2, \cdots, m\}^d$. Define

$$\mathscr{B} := \big\{\beta : \beta \text{ is a map that maps}\{1, 2, \cdots, m\}^d \text{ to } \{-1, 1\}\big\}.$$

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

$$f_\beta(\boldsymbol{x}) = \sum_{\boldsymbol{J}\in\{1,2,\cdots,m\}^d} \beta(\boldsymbol{J}) g_{Q_{\boldsymbol{J}}}(\boldsymbol{x}),$$

where $g_{Q_{\boldsymbol{J}}}(\boldsymbol{x})$ is the associated function introduced just above. It is easy to check that $f_\beta \in \text{Lip}(1, \alpha, d)$ and $\mathcal{F} := \{f_\beta : \beta \in \mathscr{B}\}$ can shatter

$$b_\ell := m^d = \lfloor N^{2/d+\rho/(2\alpha)} \rfloor^d \tag{3.17}$$

points, e.g. one point in each $\frac{1}{2}Q_{\boldsymbol{J}}$ for all b_ℓ indices $\boldsymbol{J}$'s.

Step 2: Construct $\mathcal{B}$ that scatters b_ℓ points.

Before constructing $\mathcal{B} \subseteq \text{NN}(\text{dim} = d;\ \#\text{node} \le N;\ \#\text{layer} \le L)$, let us summarize a few inequalities to be applied later.

(1) For each index $\boldsymbol{J} \in \{1, 2, \cdots, m\}^d$ and any $\boldsymbol{x} \in \frac{1}{2}Q_{\boldsymbol{J}}$, since $Q_{\boldsymbol{J}}$ has a sidelength $\frac{1}{m}$, we have

$$|f_\beta(\boldsymbol{x})| = |g_{Q_{\boldsymbol{J}}}(\boldsymbol{x})| \ge |g_{Q_{\boldsymbol{J}}}(\boldsymbol{x}_{Q_{\boldsymbol{J}}})|/2 = \big(\tfrac{1}{2m}\big)^\alpha/4, \tag{3.18}$$

where $\boldsymbol{x}_{Q_{\boldsymbol{J}}}$ is the center of $Q_{\boldsymbol{J}}$.

(2) For fixed d, α, and ρ, there exists N_1 large enough such that for any integer $N \ge N_1$, we have

$$\tfrac{1}{2^{2+\alpha}} \lfloor N^{2/d+\rho/(2\alpha)} \rfloor^{-\alpha} > C_2 N^{-\alpha(2/d+\rho/\alpha)} \ln N. \tag{3.19}$$

By Claim 3.4, for any $N_0 > 0$ and $\beta \in \mathscr{B}$, there exists some integer $N \ge N_0$ and $\phi_\beta \in \text{NN}(\text{dim} = d;\ \#\text{node} \le N;\ \#\text{layer} \le L)$ such that

$$\|f_\beta - \phi_\beta\|_{1,\infty} \le C_2 N^{-\alpha(2/d+\rho/\alpha)},$$

that is,

$$\begin{aligned}\|f_\beta - \phi_\beta\|_{1,\infty} &= \sup\big\{\tfrac{1}{\ln k}\|f_\beta - \phi_\beta\|_{L^\infty(\Omega(k,d)^c)} : k \ge 3,\ k \in \mathbb{N}\big\} + \|f_\beta - \phi_\beta\|_{L^1([0,1]^d)}\\ &\le C_2 N^{-\alpha(2/d+\rho/\alpha)},\end{aligned}$$

which gives

$$\|f_\beta - \phi_\beta\|_{L^\infty(\Omega(N,d)^c)} \le C_2 N^{-\alpha(2/d+\rho/\alpha)} \ln N.$$

Therefore, there exists a set $\widetilde{\Omega}(N,d)$ with $\mu(\widetilde{\Omega}(N,d)) = \mu(\Omega(N,d))$ such that

$$|f_\beta(\boldsymbol{x}) - \phi_\beta(\boldsymbol{x})| \le C_2 N^{-\alpha(2/d+\rho/\alpha)} \ln N \tag{3.20}$$

for any $\boldsymbol{x} \in \widetilde{\Omega}(N,d)^c$.

Note that $m = \lfloor N^{2/d+\rho/(2\alpha)} \rfloor$. By the definition of $\Omega(N,d)$ and Equation (2.4), there exists a sufficiently large N_2 such that

$$\mu(\widetilde{\Omega}(N,d)) = \mu(\Omega(N,d)) \le \sum_{k=N}^{\infty} dk\lambda(k,d) \le 2^{-dN} < (2m)^{-d} = \mu(\tfrac{1}{2}Q_{\boldsymbol{J}}), \tag{3.21}$$

for any integer $N \ge N_2$, which means $Q_{\boldsymbol{J}} \cap \widetilde{\Omega}(N,d)^c$ is not empty for any $\boldsymbol{J} \in \{1,2,\cdots,m\}^d$. Hence, there exists $x_{\boldsymbol{J}} \in \frac{1}{2}Q_{\boldsymbol{J}} \cap \widetilde{\Omega}(N,d)^c$ such that

$$\begin{align}
|f_\beta(x_{\boldsymbol{J}})| &\ge \left(\tfrac{1}{2m}\right)^{\alpha}/4 \tag{3.22}\\
&= \tfrac{1}{2^{2+\alpha}} \lfloor N^{2/d+\rho/(2\alpha)} \rfloor^{-\alpha} \notag\\
&> C_2 N^{-\alpha(2/d+\rho/\alpha)} \ln N \tag{3.23}\\
&\ge |f_\beta(x_{\boldsymbol{J}}) - \phi_\beta(x_{\boldsymbol{J}})|, \tag{3.24}
\end{align}$$

where (3.22) comes from (3.18), (3.23) comes from (3.19), and (3.24) comes from (3.20). In other words, for any $\beta \in \mathscr{B}$ and $\boldsymbol{J} \in \{1,2,\cdots,m\}^d$, $f_\beta(x_{\boldsymbol{J}})$ and $\phi_\beta(x_{\boldsymbol{J}})$ have the same sign. Then $\mathcal{B} := \{\phi_\beta : \beta \in \mathscr{B}\}$ shatters $\left\{\boldsymbol{x}_{\boldsymbol{J}} : \boldsymbol{J} \in \{1,2,\cdots,m\}^d\right\}$ since $\mathcal{F} = \{f_\beta : \beta \in \mathscr{B}\}$ shatters $\left\{\boldsymbol{x}_{\boldsymbol{J}} : \boldsymbol{J} \in \{1,2,\cdots,m\}^d\right\}$ as discussed in Step 2. Hence,

$$\text{VCDim}\big(\mathcal{B}\big) \ge m^d = \lfloor N^{2/d+\rho/(2\alpha)} \rfloor^d = b_\ell. \tag{3.25}$$

Step 3: Contradiction.

Let us summarize the conditions about N that make (3.25) true. It is clear that (3.25) is based on (3.18), (3.19), (3.20), and (3.21). By previous step, we have:

- (3.18) is true for any $N \in \mathbb{N}^+$;
- (3.19) is true for $N \ge N_1$, where N_1 is a fixed number;
- For any $N_0 > 0$, there exists $N \ge N_0$ such that (3.20) is true;
- (3.21) is true for $N \ge N_2$, where N_2 is a fixed number.

Combining these conditions, for any $N_0 \ge \max\{N_1, N_2\}$, there always exists $N \ge N_0$ such that (3.25) is true. And we know there exists $N_3 > 0$ such that

$$b_u = C_1\big((N+d)^2/2 + N\big) L \ln\big((N+d)^2/2 + N\big) < \lfloor N^{2/d+\rho/(2\alpha)} \rfloor^d = b_\ell \tag{3.26}$$

for any $N \ge N_3$. Finally, for any $N_0 \ge \max\{N_1, N_2, N_3\}$, there always exists $N \ge N_0$ such that

$$b_u < b_\ell \le \text{VCDim}\big(\mathcal{B}\big) \le \text{VCDim}\big(\text{NN}(\text{dim} = d;\ \#\text{node} \le N;\ \#\text{layer} \le L)\big) \le b_u, \qquad \square$$

which is a contradiction. So, we finish the proof.

Theorem 3.1 shows that the N-term approximation rate via two or three-hidden-layer ReLU FNNs can achieve $\mathcal{O}(N^{-2\alpha/d})$, while Theorem 3.3 shows that the approximation rate cannot be improved to $\mathcal{O}(N^{-(2\alpha/d+\rho)})$ for any $\rho > 0$. Hence, the N-term approximation rate $\mathcal{O}(N^{-2\alpha/d})$ in Theorem 3.1 is tight asymptotically.

It was conjectured in the literature that function compositions can improve the approximation capacity exponentially. By Corollary 3.2, it does be true that composing functions once can improve the approximation rate from $\mathcal{O}(N^{-\eta})$ to $\mathcal{O}(N^{-2\eta})$. However, for general continuous functions, Theorem 3.3 shows that this conjecture is not true, i.e., if the depth of the composition is $L = \mathcal{O}(1)$, the approximation rate cannot be better than $\mathcal{O}(N^{-2\alpha/d})$, not to mention $\mathcal{O}(N^{-L\alpha/d})$, which implies that adding one more layer cannot improve the approximation rate when N is large if $L > 2$.

Following the same proof as in Theorem 3.3, we have the following corollary, which shows that the result in Corollary 3.2 is tight.

COROLLARY 3.5. *$\forall$ $\rho > 0$, $C \in \mathbb{N}^+$, and $L \in \mathbb{N}^+$, there exists $N_0(\rho, C, L) > 0$ such that for any integer $N \geq N_0$,* $\mathrm{CPL}(CN^{2+\rho})$ *is not completely contained in the closure of* $\mathrm{NN}(\dim = 1;\ \#\mathrm{node} \leq N;\ \#\mathrm{layer} \leq L)$ *in the sense of $L^{1,\infty}$-norm.*

3.3. Approximation and Computation Efficiency in Parallel Computing. In this section, we will discuss the efficiency of the N-term approximation via ReLU FNNs in parallel computing. This is of more practical interest than the optimal approximation rate purely based on the number of parameters in the dictionary of the nonlinear approximation, since it is impractical to use FNNs without parallel computing in real applications. Without loss of generality, we assume $\omega = 1$, $N \gg 1$, and $d \gg 1$. We will compare the approximation and computation efficiency of different ReLU FNN structures in the nonlinear approximation in two computing environments: shared memory parallel computing and distributed memory parallel computing.

First, let us summarize standard statistics of the time and memory complexity in one training iteration of ReLU FNNs with $\mathcal{O}(N)$ width and $\mathcal{O}(L)$ depth using m computing cores and $\mathcal{O}(1)$ training data samples per iteration. These statistics can be derived using elementary knowledge in parallel computing [31]. Let $T_s(N, L, m)$ and $T_d(N, L, m)$ denote the time complexity in shared memory and distributed memory parallel computing, respectively. Denote $M_s(N, L, m)$ and $M_d(N, L, m)$ as the memory complexity in shared memory and distributed memory parallel computing, respectively. In shared memory, $M_s(N, L, m)$ is the total memory requirement; while in distributed memory, $M_d(N, L, m)$ is the memory requirement per computing core. Then

$$T_s(N, L, m) = \begin{cases} \mathcal{O}\big(L(N^2/m + \ln \tfrac{m}{N})\big), & m \in [1, N^2], \\ \mathcal{O}(L \ln N), & m \in (N^2, \infty); \end{cases} \tag{3.27}$$

$$T_d(N, L, m) = \begin{cases} \mathcal{O}\big(L(N^2/m + t_s \ln m + \tfrac{t_w N}{\sqrt{m}} \ln m)\big), & m \in [1, N^2], \\ \mathcal{O}(L \ln N), & m \in (N^2, \infty); \end{cases} \tag{3.28}$$

$$M_s(N, L, m) = \mathcal{O}(LN^2) \text{ for all } m \in \mathbb{N}^+; \tag{3.29}$$

and

$$M_d(N, L, m) = \mathcal{O}(LN^2/m + 1) \text{ for all } m \in \mathbb{N}^+, \tag{3.30}$$

where t_s and t_w are the "start-up time" and "per-word transfer time" in the data communication between different computing cores, respectively (see [31] for a detailed introduction).

Finally, we show how to choose ReLU FNN architectures considering the approximation rate and the computational efficiency in parallel computing. In practical applications, a most frequently asked question would be: given a target function

TABLE 1

The comparison of approximation and computation efficiency of different ReLU FNN architectures in shared memory parallel computing with m processors when FNNs nearly have the same approximation accuracy. Note that the analysis in this table is asymptotic in d and N and is optimal up to a log factor; "running time" in this table is the time spent on each training step with $\mathcal{O}(1)$ training data samples.

	NN(width = $[2d\lfloor N^{2/d}\rfloor, 2N, 2N]$)	NN(width = $[N]^L$)	NN(width = $[2d+10]^N$)
accuracy ϵ	$\mathcal{O}(\sqrt{d}N^{-2\alpha/d})$	$\mathcal{O}(N^{-2\alpha/d})$	$\mathcal{O}(C(d)N^{-2\alpha/d})$
number of weights	$\mathcal{O}(N^2)$	$\mathcal{O}(LN^2)$	$\mathcal{O}(d^2N)$
number of nodes	$\mathcal{O}(N)$	$\mathcal{O}(LN)$	$\mathcal{O}(dN)$
running time for $m \in [1, (2d+10)^2]$	$\mathcal{O}(N^2/m)$	$\mathcal{O}(LN^2/m)$	$\mathcal{O}(N(d^2/m + \ln \frac{m}{d}))$
running time for $m \in ((2d+10)^2, N^2]$	$\mathcal{O}(N^2/m + \ln \frac{m}{N})$	$\mathcal{O}(L(N^2/m + \ln \frac{m}{N}))$	$\mathcal{O}(N \ln d)$
running time for $m \in (N^2, \infty)$	$\mathcal{O}(\ln N)$	$\mathcal{O}(L \ln N)$	$\mathcal{O}(N \ln d)$
total memory	$\mathcal{O}(N^2)$	$\mathcal{O}(LN^2)$	$\mathcal{O}(d^2N)$

TABLE 2

The comparison of approximation and computation efficiency of different ReLU FNN architectures in distributed memory parallel computing with m processors when FNNs nearly have the same approximation accuracy. Note that the analysis in this table is asymptotic in d and N and is optimal up to a log factor; "running time" in this table is the time spent on each training step with $\mathcal{O}(1)$ training data samples.

	NN(width = $[2d\lfloor N^{2/d}\rfloor, 2N, 2N]$)	NN(width = $[N]^L$)	NN(width = $[2d+10]^N$)
accuracy ϵ	$\mathcal{O}(\sqrt{d}N^{-2\alpha/d})$	$\mathcal{O}(N^{-2\alpha/d})$	$\mathcal{O}(C(d)N^{-2\alpha/d})$
number of weights	$\mathcal{O}(N^2)$	$\mathcal{O}(LN^2)$	$\mathcal{O}(d^2N)$
number of nodes	$\mathcal{O}(N)$	$\mathcal{O}(LN)$	$\mathcal{O}(dN)$
running time for $m \in [1, (2d+10)^2]$	$\mathcal{O}(N^2/m + t_s \ln m + \frac{t_w N}{\sqrt{m}} \ln m)$	$\mathcal{O}(L(N^2/m + t_s \ln m + \frac{t_w N}{\sqrt{m}} \ln m))$	$\mathcal{O}(N(d^2/m + t_s \ln m + \frac{t_w d}{\sqrt{m}} \ln m))$
running time for $m \in ((2d+10)^2, N^2]$	$\mathcal{O}(N^2/m + t_s \ln m + \frac{t_w N}{\sqrt{m}} \ln m)$	$\mathcal{O}(L(N^2/m + t_s \ln m + \frac{t_w N}{\sqrt{m}} \ln m))$	$\mathcal{O}(N \ln d)$
running time for $m \in (N^2, \infty)$	$\mathcal{O}(\ln N)$	$\mathcal{O}(L \ln N)$	$\mathcal{O}(N \ln d)$
memory per processor	$\mathcal{O}(N^2/m + 1)$	$\mathcal{O}(LN^2/m + 1)$	$\mathcal{O}(d^2N/m + 1)$

$f \in \mathrm{Lip}(\omega, \alpha, d)$, a target approximation accuracy ϵ, and a certain amount of computational resources, e.g., m computer processors, assuming the computer memory is enough, what is a good choice of FNN architecture we should use to reduce the running time of our computers? Certainly, the answer depends on the number of processors m and ideally we hope to increase m by a factor of r to reduce the time (and memory in the distributed environment) complexity by the same factor r, which is the scalability of parallel computing.

We answer the question raised just above using FNN architectures that almost have a uniform width, since the optimal approximation theory of very deep FNNs [55] and this manuscript both utilize a nearly uniform width. Combining the theory in [55] and ours, we summarize several statistics of ReLU FNNs in parallel computing in Table 1 and 2 when FNNs nearly have the same approximation accuracy. In the case of shared memory parallel computing, from Table 1 we see that: if computing resources are enough, shallower FNNs with $\mathcal{O}(1)$ hidden layers require less and even exponentially less running time than very deep FNNs; if computing resources are limited, shallower FNNs might not be applicable or are slower, and hence very deep FNNs are a good choice. In the case of distributed memory parallel computing, the conclusion is almost the same by Table 2, except that the memory limitation is not an issue for shallower FNNs if the number of processors is large enough. In sum, if the approximation rate of very deep FNNs is not exponentially better than shallower FNNs, very deep FNNs are less efficient than shallower FNNs theoretically if computing resources are enough.

4. Numerical Experiments.

In this section, we provide two sets of numerical experiments to compare different ReLU FNN architectures using shared memory

GPU parallel computing. The numerical results for distributed memory parallel computing would be similar if the "start-up time" and "per-word transfer time" in the communication between different computing nodes are small. All numerical tests were conducted using Tensorflow and an NVIDIA P6000 GPU with 3840 CUDA parallel-processing cores and 24 GB memory.

Since it is difficult to generate target functions $f \in \mathrm{Lip}(\omega, \alpha, d)$ with fixed ω and α, and there is no numerical guarantee to identify a best approximant via optimization, we cannot directly verify the nonlinear approximation rate studied in previous sections, but we are able to observe some numerical facts close to the conclusions of approximation rates. Furthermore, we are able to verify the running time estimates in Section 3.3 and show that, to achieve the same theoretical approximation rate, shallow ReLU FNNs are more efficient than very deep FNNs in parallel computing.

In our numerical tests, we generate 50 random smooth functions as our target functions using the algorithm in [16] with a wavelength parameter $\lambda = 0.1$ and an amplitude parameter $\sqrt{(2/\lambda)}$ therein. These target functions are uniformly sampled with 20000 ordered points $\{x_i\}$ in $[0,1]$ to form a data set. The training data set consists of samples with odd indices i's, while the test data set consists of samples with even indices i's. The loss function is defined as the mean square error between the target function and the FNN approximant evaluated on training sample points. The ADAM algorithm [30] with a decreasing learning rate from 0.005 to 0.0005, a batch size 10000, and a maximum number of epochs 20000, is applied to minimize the mean square error. The minimization is randomly initialized by the "normal initialization method"[⑤]. The test error is defined as the mean square error between the target function and the FNN approximant evaluated on test sample points. The training and test data sets are essentially the same in our numerical test since we aim at studying the approximation power of FNNs instead of the generalization capacity of FNNs. Note that due to the highly non-convexity of the optimization problem, there might be chances such that the minimizers we found are bad local minimizers. Hence, we compute the average test error of the best 40 tests among the total 50 tests of each architecture.

To observe numerical phenomena in terms of N-term nonlinear approximation, in the first set of numerical experiments, we use two types of FNNs to obtain approximants to each target function: the first type has $L = \mathcal{O}(1)$ layers with different sizes of width N; the second type has a fixed width $N = 12$ with different numbers of layers L. Numerical results are summarized in Table 3. To observe numerical phenomena in terms of the number of parameters in FNNs, in the second set of numerical experiments, we use FNNs with the same number of parameters but different sizes of width N and different numbers of layers L. Numerical results are summarized in Table 4.

By the last columns of Table 3, we verified that as long as the number of computing cores m is larger than or equal to N^2, the running time per iteration of FNNs with $\mathcal{O}(1)$ layers is $\mathcal{O}(\ln N)$, while the running time per iteration of FNNs with $\mathcal{O}(N)$ layers and $\mathcal{O}(1)$ width is $\mathcal{O}(N)$. By the last columns of Table 4, we see that when the number of parameters is the same, very deep FNNs requires much more running time per iteration than shallower FNNs and the difference becomes more significant when the number of parameters increases. Hence, very deep FNNs are much less efficient than shallower FNNs in parallel computing in terms of running time.

Besides, by Table 3 and 4, the test error of very deep FNNs cannot be improved if the depth is increased and the error even becomes larger when depth is larger.

[⑤] See https://medium.com/prateekvishnu/xavier-and-he-normal-he-et-al-initialization-8e3d7a087528.

TABLE 3

Comparison between NN(dim = 1; width = $[N]^L$) *and* NN(dim = 1; width = $[12]^N$) *for* $N = 32, 64, 128$ *and* $L = 2, 4, 8$. *"Time" in this table is the total running time spent on* 20000 *training steps with training batch size* 10000*, and the unit is second(s).*

N	layer	width	test error	improvement ratio	#parameter	time
32	2	32	8.06×10^{-2}	–	1153	3.09×10^{1}
32	4	32	3.98×10^{-4}	–	3265	3.82×10^{1}
32	8	32	1.50×10^{-5}	–	7489	5.60×10^{1}
32	32	12	1.29×10^{-3}	–	4873	1.27×10^{2}
64	2	64	2.51×10^{-2}	3.21	4353	3.45×10^{1}
64	4	64	4.27×10^{-5}	9.32	12673	5.00×10^{1}
64	8	64	2.01×10^{-6}	7.46	29313	7.91×10^{1}
64	64	12	1.16×10^{-1}	0.01	9865	2.37×10^{2}
128	2	128	2.04×10^{-3}	12.3	16897	5.03×10^{1}
128	4	128	1.05×10^{-5}	4.07	49921	8.21×10^{1}
128	8	128	1.47×10^{-6}	1.37	115969	1.41×10^{2}
128	128	12	3.17×10^{-1}	0.37	19849	4.47×10^{2}

However, when the number of layers is fixed, increasing width can reduce the test error. More quantitatively, we define the *improvement ratio* of an FNN with width N and depth L in Table 3 as the ratio of the test error of an FNN in NN(dim = 1; width = $[N/2]^L$) (or NN(dim = 1; width = $[N]^{L/2}$)) over the test error of the current FNN in NN(dim = 1; width = $[N]^L$). Similarly, the improvement ratio of an FNN with a number of parameters W in Table 4 is defined as the ratio of the test error of an FNN with the same type of architecture and a number of parameters $W/2$ over the test error of the current FNN. According to the improvement ratio in Table 3 and 4, when $L = \mathcal{O}(1)$, the numerical approximation rate in terms of N is in a range between 2 to 4. We would like to emphasize that due to the highly non-convexity of the deep learning optimization and the difficulty to generate target functions of the same class with a fixed order α and constant ω, we cannot accurately verify the approximation rate. But the statistics of the improvement ratio can roughly reflect the approximation rate and the numerical results stand in line with our theoretical analysis.

5. Conclusions. We study the approximation and computation efficiency of function compositions in nonlinear approximation, especially the case when the composition is implemented using multi-layer feed-forward neural networks (FNNs) with ReLU activation functions in parallel computing. New analysis techniques have been proposed to quantified the advantages of function compositions in accelerating the approximation rate in nonlinear approximation, achieving the optimal approximation rate of ReLU FNNs with $\mathcal{O}(1)$ hidden layers for approximating Hölder continuous functions. Moreover, for an arbitrary function f on $[0, 1]$, regardless of its smoothness and even the continuity, if f can be approximated via nonlinear approximation using one-hidden-layer ReLU FNNs with an approximation rate $\mathcal{O}(N^{-\eta})$, we show that nonlinear approximation with function compositions via deep ReLU FNNs can improve the approximation rate to $\mathcal{O}(N^{-2\eta})$. Finally, considering the computational efficiency per training iteration in parallel computing platforms, FNNs with $\mathcal{O}(1)$

TABLE 4

Comparison between shallow FNNs and deep FNNs when the total number of parameters (#parameter) is fixed. "Time" in this table is the total running time spent on 20000 *training steps with training batch size* 10000*, and the unit is second(s).*

#parameter	layer	width	test error	improvement ratio	time
5038	2	69	1.13×10^{-2}	–	3.84×10^{1}
5041	4	40	1.65×10^{-4}	–	3.80×10^{1}
4993	8	26	1.69×10^{-5}	–	5.07×10^{1}
5029	33	12	4.77×10^{-3}	–	1.28×10^{2}
9997	2	98	4.69×10^{-3}	2.41	4.40×10^{1}
10090	4	57	7.69×10^{-5}	2.14	4.67×10^{1}
9954	8	37	7.43×10^{-6}	2.27	5.92×10^{1}
10021	65	12	2.80×10^{-1}	0.02	2.31×10^{2}
19878	2	139	1.43×10^{-3}	3.28	5.18×10^{1}
20170	4	81	2.30×10^{-5}	3.34	6.26×10^{1}
20194	8	53	2.97×10^{-6}	2.50	7.08×10^{1}
20005	129	12	3.17×10^{-1}	0.88	4.30×10^{2}

hidden layers are a better choice for approximating Hölder continuous functions if computing resources are enough. Our discussion provides a new point of view for the debate of "deep vs. shallow" in the literature of deep learning research. We would like to conclude our paper with a simple message that depth is good but a very deep ReLU FNN could be less efficient that a relative shallower FNN with $\mathcal{O}(1)$ hidden layers when we consider the approximation rate and parallel computing at the same time.

Acknowledgments. H. Yang thanks the support of the start-up grant by the Department of Mathematics at the National University of Singapore, the support of the Ministry of Education in Singapore under the grant MOE2018-T2-2-147, and the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research. We would like to acknowledge the editor and reviewers for their help in improving this manuscript.

REFERENCES

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

[2] A. R. Barron, *Universal approximation bounds for superpositions of a sigmoidal function*, IEEE Transactions on Information Theory, 39 (1993), pp. 930–945, https://doi.org/10.1109/18.256500.

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

[4] 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, https://doi.org/10.1109/TNNLS.2013.2293637.

[5] E. J. Candes and M. B. Wakin, *An introduction to compressive sampling*, IEEE Signal Processing Magazine, 25 (2008), pp. 21–30, https://doi.org/10.1109/MSP.2007.914731.

[6] 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, https://doi.org/10.1109/ACSSC.1994.471413.

[7] 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, https://doi.org/10.5591/978-1-57735-516-8/IJCAI11-210, http://dx.doi.org/10.5591/978-1-57735-516-8/IJCAI11-210.

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

[9] I. Daubechies, *Ten Lectures on Wavelets*, Society for Industrial and Applied Mathematics, 1992, https://doi.org/10.1137/1.9781611970104, https://epubs.siam.org/doi/abs/10.1137/1.9781611970104, https://arxiv.org/abs/https://epubs.siam.org/doi/pdf/10.1137/1.9781611970104.

[10] Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio, *Identifying and attacking the saddle point problem in high-dimensional non-convex optimization*, in Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2, NIPS'14, Cambridge, MA, USA, 2014, MIT Press, pp. 2933–2941, http://dl.acm.org/citation.cfm?id=2969033.2969154.

[11] G. Davis, *Adaptive nonlinear approximations*, 1994.

[12] R. DEVORE and A. RON, *Approximation using scattered shifts of a multivariate function*, Transactions of the American Mathematical Society, 362 (2010), pp. 6205–6229, http://www.jstor.org/stable/40997201.

[13] R. A. DeVore, *Nonlinear approximation*, Acta Numerica, 7 (1998), p. 51–150, https://doi.org/10.1017/S0962492900002816.

[14] D. L. Donoho, *Compressed sensing*, IEEE Transactions on Information Theory, 52 (2006), pp. 1289–1306, https://doi.org/10.1109/TIT.2006.871582.

[15] J. Duchi, E. Hazan, and Y. Singer, *Adaptive subgradient methods for online learning and stochastic optimization*, J. Mach. Learn. Res., 12 (2011), pp. 2121–2159, http://dl.acm.org/citation.cfm?id=1953048.2021068.

[16] S.-I. Filip, A. Javeed, and L. N. Trefethen, *Smooth random functions, random ODEs, and Gaussian processes.* To appear in SIAM Review., Dec. 2018, https://hal.inria.fr/hal-01944992.

[17] K. Fukushima, *Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position*, Biological Cybernetics, 36 (1980), pp. 193–202, https://doi.org/10.1007/BF00344251, https://doi.org/10.1007/BF00344251.

[18] T. Hangelbroek and A. Ron, *Nonlinear approximation using gaussian kernels*, Journal of Functional Analysis, 259 (2010), pp. 203 – 219, https://doi.org/https://doi.org/10.1016/j.jfa.2010.02.001, http://www.sciencedirect.com/science/article/pii/S0022123610000467.

[19] B. Hanin and M. Sellke, *Approximating continuous functions by ReLU nets of minimal width*, (2017), https://arxiv.org/abs/1710.11278.

[20] 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, http://proceedings.mlr.press/v65/harvey17a.html.

[21] 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, https://doi.org/10.1109/CVPR.2016.90.

[22] K. Hornik, M. Stinchcombe, and H. White, *Multilayer feedforward networks are universal approximators*, Neural Networks, 2 (1989), pp. 359 – 366, https://doi.org/https://doi.org/10.1016/0893-6080(89)90020-8, http://www.sciencedirect.com/science/article/pii/0893608089900208.

[23] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, *Densely connected convolutional networks*, 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), (2017), pp. 2261–2269.

[24] J. Jiang, *Design of neural networks for lossless data compression*, Optical Engineering, 35 (1996), pp. 35 – 35 – 7, https://doi.org/10.1117/1.600614, https://doi.org/10.1117/1.600614.

[25] R. Johnson and T. Zhang, *Accelerating stochastic gradient descent using predictive variance reduction*, in Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 1, NIPS'13, USA, 2013, Curran Associates Inc., pp. 315–323,

http://dl.acm.org/citation.cfm?id=2999611.2999647.

[26] J. Joutsensalo, *Nonlinear data compression and representation by combining self-organizing map and subspace rule*, in Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94), vol. 2, June 1994, pp. 637–640 vol.2, https://doi.org/10.1109/ICNN.1994.374249.

[27] 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, http://papers.nips.cc/paper/6112-deep-learning-without-poor-local-minima.pdf.

[28] K. Kawaguchi and Y. Bengio, *Depth with nonlinearity creates no bad local minima in resnets*, (2018), https://arxiv.org/abs/1810.09038, https://arxiv.org/abs/1810.09038.

[29] M. J. Kearns and R. E. Schapire, *Efficient distribution-free learning of probabilistic concepts*, J. Comput. Syst. Sci., 48 (1994), pp. 464–497, https://doi.org/10.1016/S0022-0000(05)80062-5, http://dx.doi.org/10.1016/S0022-0000(05)80062-5.

[30] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, CoRR, abs/1412.6980 (2014), http://arxiv.org/abs/1412.6980, https://arxiv.org/abs/1412.6980.

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

[32] Y. LeCun, P. Haffner, L. Bottou, and Y. Bengio, *Object recognition with gradient-based learning*, in Shape, Contour and Grouping in Computer Vision, London, UK, UK, 1999, Springer-Verlag, pp. 319–, http://dl.acm.org/citation.cfm?id=646469.691875.

[33] S. Liang and R. Srikant, *Why deep neural networks?*, CoRR, abs/1610.04161 (2016), http://arxiv.org/abs/1610.04161, https://arxiv.org/abs/1610.04161.

[34] 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, https://doi.org/10.1007/s10994-013-5406-z, https://doi.org/10.1007/s10994-013-5406-z.

[35] Z. Lu, H. Pu, F. Wang, Z. Hu, and L. Wang, *The expressive power of neural networks: A view from the width*, vol. abs/1709.02540, 2017, http://arxiv.org/abs/1709.02540, https://arxiv.org/abs/1709.02540.

[36] S. G. Mallat and Z. Zhang, *Matching pursuits with time-frequency dictionaries*, IEEE Transactions on Signal Processing, 41 (1993), pp. 3397–3415, https://doi.org/10.1109/78.258082.

[37] H. Montanelli and Q. Du, *New error bounds for deep networks using sparse grids*, (2017), https://arxiv.org/abs/1712.08688.

[38] 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, http://papers.nips.cc/paper/5422-on-the-number-of-linear-regions-of-deep-neural-networks.pdf.

[39] Q. N. Nguyen and M. Hein, *The loss surface of deep and wide neural networks*, CoRR, abs/1704.08045 (2017), http://arxiv.org/abs/1704.08045, https://arxiv.org/abs/1704.08045.

[40] H. Ohlsson, A. Y. Yang, R. Dong, and S. S. Sastry, *Nonlinear basis pursuit*, in 2013 Asilomar Conference on Signals, Systems and Computers, Nov 2013, pp. 115–119, https://doi.org/10.1109/ACSSC.2013.6810285.

[41] P. Petersen and F. Voigtlaender, *Optimal approximation of piecewise smooth functions using deep ReLU neural networks*, Neural Networks, 108 (2018), pp. 296 – 330, https://doi.org/https://doi.org/10.1016/j.neunet.2018.08.019, http://www.sciencedirect.com/science/article/pii/S0893608018302454.

[42] P. Petrushev, *Multivariate n-term rational and piecewise polynomial approximation*, Journal of Approximation Theory, 121 (2003), pp. 158 – 197, https://doi.org/https://doi.org/10.1016/S0021-9045(02)00060-6, http://www.sciencedirect.com/science/article/pii/S0021904502000606.

[43] D. Rolnick and M. Tegmark, *The power of deeper networks for expressing natural functions*, CoRR, abs/1705.05502 (2017), http://arxiv.org/abs/1705.05502, https://arxiv.org/abs/1705.05502.

[44] D. Rumelhart, J. McClelland, P. R. Group, and S. D. P. R. G. University of California, *Psychological and Biological Models*, no. v. 2 in A Bradford Book, MIT Press, 1986, https://books.google.com.sg/books?id=davmLgzusB8C.

[45] 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, http://proceedings.mlr.press/v70/safran17a.html.

[46] 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.

[47] 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.

[48] J. Schmidt-Hieber, *Nonparametric regression using deep neural networks with ReLU activation function*, (2017), https://arxiv.org/abs/1708.06633.

[49] 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, https://openreview.net/forum?id=H1ebTsActm.

[50] S. Tariyal, A. Majumdar, R. Singh, and M. Vatsa, *Greedy deep dictionary learning*, CoRR, abs/1602.00203 (2016), http://arxiv.org/abs/1602.00203, https://arxiv.org/abs/1602.00203.

[51] P. Werbos, *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*, Harvard University, 1975, https://books.google.com.sg/books?id=z81XmgEACAAJ.

[52] S. Xie, R. B. Girshick, P. Dollár, Z. Tu, and K. He, *Aggregated residual transformations for deep neural networks*, CoRR, abs/1611.05431 (2016), http://arxiv.org/abs/1611.05431, https://arxiv.org/abs/1611.05431.

[53] 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, https://doi.org/10.1007/s10114-012-1369-4, https://doi.org/10.1007/s10114-012-1369-4.

[54] D. Yarotsky, *Error bounds for approximations with deep ReLU networks*, Neural Networks, 94 (2017), pp. 103 – 114, https://doi.org/https://doi.org/10.1016/j.neunet.2017.07.002, http://www.sciencedirect.com/science/article/pii/S0893608017301545.

[55] 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, http://proceedings.mlr.press/v75/yarotsky18a.html.

[56] S. Zagoruyko and N. Komodakis, *Wide residual networks*, CoRR, abs/1605.07146 (2016).