

Deep Network with Approximation Error Being Reciprocal of Width to Power of Square Root of Depth

Zuowei Shen

matzuows@nus.edu.sg

Department of Mathematics, National University of Singapore

Haizhao Yang

haizhao@purdue.edu

Department of Mathematics, Purdue University

Shijun Zhang

zhangshijun@u.nus.edu

Department of Mathematics, National University of Singapore

Keywords: Exponential Convergence, Curse of Dimensionality, Deep Neural Network, Floor and ReLU Activation Functions, Continuous Function.

Abstract

A new network with super approximation power is introduced. This network is built with Floor ($\lfloor x \rfloor$) or ReLU ($\max\{0, x\}$) activation function in each neuron and hence we call such networks Floor-ReLU networks. For any hyper-parameters $N \in \mathbb{N}^+$ and $L \in \mathbb{N}^+$, it is shown that Floor-ReLU networks with width $\max\{d, 5N + 13\}$ and depth $64dL + 3$ can uniformly approximate a Hölder function f on $[0, 1]^d$ with an approximation error $3\lambda d^{\alpha/2} N^{-\alpha\sqrt{L}}$, where $\alpha \in (0, 1]$ and λ are the Hölder order and constant, respectively. More generally for an arbitrary continuous function f on $[0, 1]^d$ with a modulus of continuity $\omega_f(\cdot)$, the constructive approximation rate is $\omega_f(\sqrt{d} N^{-\sqrt{L}}) + 2\omega_f(\sqrt{d}) N^{-\sqrt{L}}$. As a consequence, this new class of networks overcomes the curse of

dimensionality in approximation power when the variation of $\omega_f(r)$ as $r \rightarrow 0$ is moderate (e.g., $\omega_f(r) \lesssim r^\alpha$ for Hölder continuous functions), since the major term to be considered in our approximation rate is essentially \sqrt{d} times a function of N and L independent of d within the modulus of continuity.

1 Introduction

Recently, there has been a large number of successful real-world applications of deep neural networks in many fields of computer science and engineering, especially for large-scale and high-dimensional learning problems. Understanding the approximation capacity of deep neural networks has become a fundamental research direction for revealing the advantages of deep learning compared to traditional methods. This paper introduces new theories and network architectures achieving root exponential convergence and avoiding the curse of dimensionality simultaneously for (Hölder) continuous functions with an explicit error bound in deep network approximation, which might be two foundational laws supporting the application of deep network approximation in large-scale and high-dimensional problems. The approximation results here are quantitative and apply to networks with essentially arbitrary width and depth. These results suggest considering Floor-ReLU networks as a possible alternative to ReLU networks in deep learning.

Deep ReLU networks with width $\mathcal{O}(N)$ and depth $\mathcal{O}(L)$ can achieve the approximation rate $\mathcal{O}(N^{-L})$ for polynomials on $[0, 1]^d$ (Lu et al., 2020) but it is not true for general functions, e.g., the (nearly) optimal approximation rates of deep ReLU networks for a Lipschitz continuous function and a C^s function f on $[0, 1]^d$ are $\mathcal{O}(\sqrt{d}N^{-2/d}L^{-2/d})$ and $\mathcal{O}(\|f\|_{C^s}N^{-2s/d}L^{-2s/d})$ (Shen et al., 2019b; Lu et al., 2020), respectively. The limitation of ReLU networks motivates us to explore other types of network architectures to answer our curiosity on deep networks: Do deep neural networks with arbitrary width $\mathcal{O}(N)$ and arbitrary depth $\mathcal{O}(L)$ admit an exponential approximation rate $\mathcal{O}(\omega_f(N^{-L^\eta}))$ for some constant $\eta > 0$ for a generic continuous function f on $[0, 1]^d$ with a modulus of continuity $\omega_f(\cdot)$?

To answer this question, we introduce the Floor-ReLU network, which is a fully connected neural network (FNN) built with either Floor ($\lfloor x \rfloor$) or ReLU ($\max\{0, x\}$) activation function¹ in each neuron. Mathematically, if we let $N_0 = d$, $N_{L+1} = 1$, and N_ℓ be the number of neurons in ℓ -th hidden layer of a Floor-ReLU network for $\ell = 1, 2, \dots, L$, then the architecture of this network with input x and output $\phi(x)$ can be

¹Our results can be easily generalized to Ceiling-ReLU networks, namely, feed-forward neural networks with either Ceiling ($\lceil x \rceil$) or ReLU ($\max\{0, x\}$) activation function in each neuron.

described as

$$\mathbf{x} = \tilde{\mathbf{h}}_0 \xrightarrow{\mathbf{W}_0, \mathbf{b}_0} \mathbf{h}_1 \xrightarrow{\sigma \text{ or } \lfloor \cdot \rfloor} \tilde{\mathbf{h}}_1 \quad \dots \quad \xrightarrow{\mathbf{W}_{L-1}, \mathbf{b}_{L-1}} \mathbf{h}_L \xrightarrow{\sigma \text{ or } \lfloor \cdot \rfloor} \tilde{\mathbf{h}}_L \xrightarrow{\mathbf{W}_L, \mathbf{b}_L} \mathbf{h}_{L+1} = \phi(\mathbf{x}),$$

where $\mathbf{W}_\ell \in \mathbb{R}^{N_{\ell+1} \times N_\ell}$, $\mathbf{b}_\ell \in \mathbb{R}^{N_{\ell+1}}$, $\mathbf{h}_{\ell+1} := \mathbf{W}_\ell \cdot \tilde{\mathbf{h}}_\ell + \mathbf{b}_\ell$ for $\ell = 0, 1, \dots, L$, and $\tilde{\mathbf{h}}_{\ell,n}$ is equal to $\sigma(\mathbf{h}_{\ell,n})$ or $\lfloor \mathbf{h}_{\ell,n} \rfloor$ for $\ell = 1, 2, \dots, L$ and $n = 1, 2, \dots, N_\ell$, where $\mathbf{h}_\ell = (\mathbf{h}_{\ell,1}, \dots, \mathbf{h}_{\ell,N_\ell})$ and $\tilde{\mathbf{h}}_\ell = (\tilde{\mathbf{h}}_{\ell,1}, \dots, \tilde{\mathbf{h}}_{\ell,N_\ell})$ for $\ell = 1, 2, \dots, L$. See Figure 1 for an example.

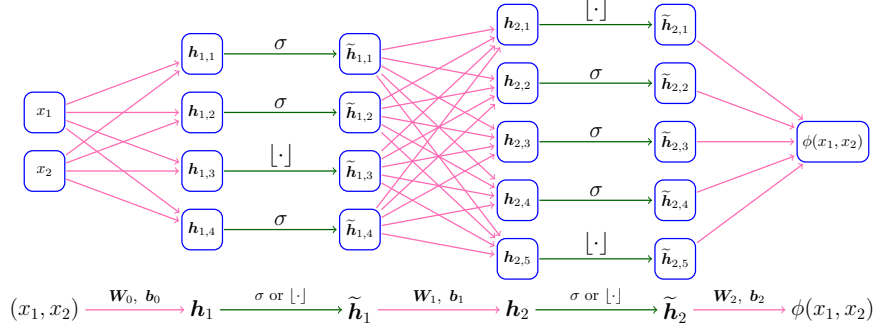


Figure 1: An example of a Floor-ReLU network with width 5 and depth 2.

In Theorem 1.1 below, we show by construction that Floor-ReLU networks with width $\max\{d, 5N + 13\}$ and depth $64dL + 3$ can uniformly approximate a continuous function f on $[0, 1]^d$ with a root exponential approximation rate² $\omega_f(\sqrt{d} N^{-\sqrt{L}}) + 2\omega_f(\sqrt{d})N^{-\sqrt{L}}$, where $\omega_f(\cdot)$ is the modulus of continuity defined as

$$\omega_f(r) := \sup \{ |f(\mathbf{x}) - f(\mathbf{y})| : \|\mathbf{x} - \mathbf{y}\|_2 \leq r, \mathbf{x}, \mathbf{y} \in [0, 1]^d \}, \quad \text{for any } r \geq 0,$$

where $\|\mathbf{x}\|_2 = \sqrt{x_1^2 + x_2^2 + \dots + x_d^2}$ for any $\mathbf{x} = (x_1, x_2, \dots, x_d) \in \mathbb{R}^d$.

Theorem 1.1. *Given any $N, L \in \mathbb{N}^+$ and an arbitrary continuous function f on $[0, 1]^d$, there exists a function ϕ implemented by a Floor-ReLU network with width $\max\{d, 5N + 13\}$ and depth $64dL + 3$ such that*

$$|\phi(\mathbf{x}) - f(\mathbf{x})| \leq \omega_f(\sqrt{d} N^{-\sqrt{L}}) + 2\omega_f(\sqrt{d})N^{-\sqrt{L}}, \quad \text{for any } \mathbf{x} \in [0, 1]^d.$$

With Theorem 1.1, we have an immediate corollary.

Corollary 1.2. *Given an arbitrary continuous function f on $[0, 1]^d$, there exists a function ϕ implemented by a Floor-ReLU network with width \bar{N} and depth \bar{L} such that*

$$|\phi(\mathbf{x}) - f(\mathbf{x})| \leq \omega_f\left(\sqrt{d} \left\lfloor \frac{\bar{N}-13}{5} \right\rfloor^{-\sqrt{\left\lfloor \frac{\bar{L}-3}{64d} \right\rfloor}}\right) + 2\omega_f(\sqrt{d}) \left\lfloor \frac{\bar{N}-13}{5} \right\rfloor^{-\sqrt{\left\lfloor \frac{\bar{L}-3}{64d} \right\rfloor}},$$

for any $\mathbf{x} \in [0, 1]^d$ and $\bar{N}, \bar{L} \in \mathbb{N}^+$ with $\bar{N} \geq \max\{d, 18\}$ and $\bar{L} \geq 64d + 3$.

²All the exponential convergence in this paper is root exponential convergence. Nevertheless, after the introduction, for the convenience of presentation, we will omit the prefix “root”, as in the literature.

87 In Theorem 1.1, the rate in $\omega_f(\sqrt{d}N^{-\sqrt{L}})$ implicitly depends on N and L through
 88 the modulus of continuity of f , while the rate in $2\omega_f(\sqrt{d})N^{-\sqrt{L}}$ is explicit in N and L .
 89 Simplifying the implicit approximation rate to make it explicitly depending on N and
 90 L is challenging in general. However, if f is a Hölder continuous function on $[0, 1]^d$ of
 91 order $\alpha \in (0, 1]$ with a constant λ , i.e., $f(\mathbf{x})$ satisfying

$$92 \quad |f(\mathbf{x}) - f(\mathbf{y})| \leq \lambda \|\mathbf{x} - \mathbf{y}\|_2^\alpha, \quad \text{for any } \mathbf{x}, \mathbf{y} \in [0, 1]^d, \quad (1)$$

93 then $\omega_f(r) \leq \lambda r^\alpha$ for any $r \geq 0$. Therefore, in the case of Hölder continuous functions,
 94 the approximation rate is simplified to $3\lambda d^{\alpha/2} N^{-\alpha\sqrt{L}}$ as shown in the following corol-
 95 lary. In the special case of Lipschitz continuous functions with a Lipschitz constant λ ,
 96 the approximation rate is simplified to $3\lambda\sqrt{d}N^{-\sqrt{L}}$.

97 **Corollary 1.3.** *Given any $N, L \in \mathbb{N}^+$ and a Hölder continuous function f on $[0, 1]^d$*
 98 *of order α with a constant λ , there exists a function ϕ implemented by a Floor-ReLU*
 99 *network with width $\max\{d, 5N + 13\}$ and depth $64dL + 3$ such that*

$$100 \quad |\phi(\mathbf{x}) - f(\mathbf{x})| \leq 3\lambda d^{\alpha/2} N^{-\alpha\sqrt{L}}, \quad \text{for any } \mathbf{x} \in [0, 1]^d.$$

101 First, Theorem 1.1 and Corollary 1.3 show that the approximation capacity of deep
 102 networks for continuous functions can be nearly exponentially improved by increasing
 103 the network depth, and the approximation error can be explicitly characterized in terms
 104 of the width $\mathcal{O}(N)$ and depth $\mathcal{O}(L)$. Second, this new class of networks overcomes the
 105 curse of dimensionality in the approximation power when the modulus of continuity is
 106 moderate, since the approximation order is essentially $\omega_f(\sqrt{d}N^{-\sqrt{L}})$. Finally, apply-
 107 ing piecewise constant and integer-valued functions as activation functions and integer
 108 numbers as parameters has been explored in the study of quantized neural networks
 109 (Hubara et al., 2017; Yin et al., 2019; Bengio et al., 2013) with efficient training algo-
 110 rithms for low computational complexity (Wang et al., 2018). The floor function ($\lfloor x \rfloor$) is
 111 a piecewise constant function and can be easily implemented numerically at very little
 112 cost. Hence, the evaluation of the proposed network could be efficiently implemented
 113 in practical computation. Though there might not be an existing optimization algorithm
 114 to identify an approximant with the approximation rate in this paper, Theorem 1.1 can
 115 provide an expected accuracy before a learning task and how much the current opti-
 116 mization algorithms could be improved. Designing an efficient optimization algorithm
 117 for Floor-ReLU networks will be left as future work with several possible directions
 118 discussed later.

119 We would like to remark that an increased smoothness or regularity of the target
 120 function could improve our approximation rate but at the cost of a large prefactor. For
 121 example, to attain better approximation rates for functions in $C^s([0, 1]^d)$, it is common

to use Taylor expansions and derivatives, which are tools that suffer from the curse of dimensionality and will result in a large prefactor like $\mathcal{O}((s+1)^d)$ that is subject to the curse of dimensionality. Furthermore, the prospective approximation rate using smoothness is not attractive. For example, the prospective approximation rate would be $\mathcal{O}(N^{-s\sqrt{L}})$, if we use Floor-ReLU networks with width $\mathcal{O}(N)$ and depth $\mathcal{O}(L)$ to approximate functions in $C^s([0,1]^d)$. However, such a rate $\mathcal{O}(N^{-s\sqrt{L}}) = \mathcal{O}(N^{-\sqrt{s^2L}})$ can be attained by using Floor-ReLU networks with width $\mathcal{O}(N)$ and depth $\mathcal{O}(s^2L)$ to approximate Lipschitz continuous functions. Hence, increasing the network depth can result in the same approximation rate for Lipschitz continuous functions as the rate of smooth functions.

The rest of this paper is organized as follows. In Section 2, we discuss the application scope of our theory and compare related works in the literature. In Section 3, we prove Theorem 1.1 based on Proposition 3.2. Next, this basic proposition is proved in Section 4. Finally, we conclude this paper in Section 5.

2 Discussion

In this section, we will discuss the application scope of our theory in machine learning and its comparison related to existing works.

2.1 Application scope of our theory in machine learning

In supervised learning, an unknown target function $f(\mathbf{x})$ defined on a domain Ω is learned through its finitely many samples $\{(\mathbf{x}_i, f(\mathbf{x}_i))\}_{i=1}^n$. If deep networks are applied in supervised learning, the following optimization problem is solved to identify a deep network $\phi(\mathbf{x}; \boldsymbol{\theta}_S)$, with $\boldsymbol{\theta}_S$ as the set of parameters, to infer $f(\mathbf{x})$ for unseen data samples \mathbf{x} :

$$\boldsymbol{\theta}_S = \arg \min_{\boldsymbol{\theta}} R_S(\boldsymbol{\theta}) := \arg \min_{\boldsymbol{\theta}} \frac{1}{n} \sum_{\{\mathbf{x}_i\}_{i=1}^n} \ell(\phi(\mathbf{x}_i; \boldsymbol{\theta}), f(\mathbf{x}_i)) \quad (2)$$

with a loss function typically taken as $\ell(y, y') = \frac{1}{2}|y - y'|^2$. The inference error is usually measured by $R_D(\boldsymbol{\theta}_S)$, where

$$R_D(\boldsymbol{\theta}) := \mathbb{E}_{\mathbf{x} \sim U(\Omega)} [\ell(\phi(\mathbf{x}; \boldsymbol{\theta}), f(\mathbf{x}))],$$

where the expectation is taken with an unknown data distribution $U(\Omega)$ over Ω .

Note that the best deep network to infer $f(\mathbf{x})$ is $\phi(\mathbf{x}; \boldsymbol{\theta}_D)$ with $\boldsymbol{\theta}_D$ given by

$$\boldsymbol{\theta}_D = \arg \min_{\boldsymbol{\theta}} R_D(\boldsymbol{\theta}).$$

152 The best possible inference error is $R_{\mathcal{D}}(\boldsymbol{\theta}_{\mathcal{D}})$. In real applications, $U(\Omega)$ is unknown
 153 and only finitely many samples from this distribution are available. Hence, the empiri-
 154 cal loss $R_{\mathcal{S}}(\boldsymbol{\theta})$ is minimized hoping to obtain $\phi(\boldsymbol{x}; \boldsymbol{\theta}_{\mathcal{S}})$, instead of minimizing the pop-
 155 ulation loss $R_{\mathcal{D}}(\boldsymbol{\theta})$ to obtain $\phi(\boldsymbol{x}; \boldsymbol{\theta}_{\mathcal{D}})$. In practice, a numerical optimization method to
 156 solve (2) may result in a numerical solution (denoted as $\boldsymbol{\theta}_{\mathcal{N}}$) that may not be a global
 157 minimizer $\boldsymbol{\theta}_{\mathcal{S}}$. Therefore, the actually learned neural network to infer $f(\boldsymbol{x})$ is $\phi(\boldsymbol{x}; \boldsymbol{\theta}_{\mathcal{N}})$
 158 and the corresponding inference error is measured by $R_{\mathcal{D}}(\boldsymbol{\theta}_{\mathcal{N}})$.

159 By the discussion just above, it is crucial to quantify $R_{\mathcal{D}}(\boldsymbol{\theta}_{\mathcal{N}})$ to see how good the
 160 learned neural network $\phi(\boldsymbol{x}; \boldsymbol{\theta}_{\mathcal{N}})$ is, since $R_{\mathcal{D}}(\boldsymbol{\theta}_{\mathcal{N}})$ is the expected inference error over
 161 all possible data samples. Note that

$$\begin{aligned}
 162 \quad R_{\mathcal{D}}(\boldsymbol{\theta}_{\mathcal{N}}) &= [R_{\mathcal{D}}(\boldsymbol{\theta}_{\mathcal{N}}) - R_{\mathcal{S}}(\boldsymbol{\theta}_{\mathcal{N}})] + [R_{\mathcal{S}}(\boldsymbol{\theta}_{\mathcal{N}}) - R_{\mathcal{S}}(\boldsymbol{\theta}_{\mathcal{S}})] + [R_{\mathcal{S}}(\boldsymbol{\theta}_{\mathcal{S}}) - R_{\mathcal{S}}(\boldsymbol{\theta}_{\mathcal{D}})] \\
 163 \quad &\quad + [R_{\mathcal{S}}(\boldsymbol{\theta}_{\mathcal{D}}) - R_{\mathcal{D}}(\boldsymbol{\theta}_{\mathcal{D}})] + R_{\mathcal{D}}(\boldsymbol{\theta}_{\mathcal{D}}) \\
 &\leq R_{\mathcal{D}}(\boldsymbol{\theta}_{\mathcal{D}}) + [R_{\mathcal{S}}(\boldsymbol{\theta}_{\mathcal{N}}) - R_{\mathcal{S}}(\boldsymbol{\theta}_{\mathcal{S}})] \\
 164 \quad &\quad + [R_{\mathcal{D}}(\boldsymbol{\theta}_{\mathcal{N}}) - R_{\mathcal{S}}(\boldsymbol{\theta}_{\mathcal{N}})] + [R_{\mathcal{S}}(\boldsymbol{\theta}_{\mathcal{D}}) - R_{\mathcal{D}}(\boldsymbol{\theta}_{\mathcal{D}})], \tag{3} \\
 165
 \end{aligned}$$

166 where the inequality comes from the fact that $[R_{\mathcal{S}}(\boldsymbol{\theta}_{\mathcal{S}}) - R_{\mathcal{S}}(\boldsymbol{\theta}_{\mathcal{D}})] \leq 0$ since $\boldsymbol{\theta}_{\mathcal{S}}$ is a
 167 global minimizer of $R_{\mathcal{S}}(\boldsymbol{\theta})$. The constructive approximation established in this paper
 168 and in the literature provides an upper bound of $R_{\mathcal{D}}(\boldsymbol{\theta}_{\mathcal{D}})$ in terms of the network size,
 169 e.g., in terms of the network width and depth, or in terms of the number of param-
 170 eters. The second term of (3) is bounded by the optimization error of the numerical
 171 algorithm applied to solve the empirical loss minimization problem in (2). If the nu-
 172 merical algorithm is able to find a global minimizer, the second term is equal to zero.
 173 The theoretical guarantee of the convergence of an optimization algorithm to a global
 174 minimizer $\boldsymbol{\theta}_{\mathcal{S}}$ and the characterization of the convergence belong to the optimization
 175 analysis of neural networks. The third and fourth term of (3) are usually bounded in
 176 terms of the sample size n and a certain norm of $\boldsymbol{\theta}_{\mathcal{N}}$ and $\boldsymbol{\theta}_{\mathcal{D}}$ (e.g., ℓ_1 , ℓ_2 , or the path
 177 norm), respectively. The study of the bounds for the third and fourth terms is referred
 178 to as the generalization error analysis of neural networks.

179 The approximation theory, the optimization theory, and the generalization theory
 180 form the three main theoretical aspects of deep learning with different emphases and
 181 challenges, which have motivated many separate research directions recently. Theorem
 182 1.1 and Corollary 1.3 provide an upper bound of $R_{\mathcal{D}}(\boldsymbol{\theta}_{\mathcal{D}})$. This bound only depends on
 183 the given budget of neurons and layers of Floor-ReLU networks and on the modulus of
 184 continuity of the target function f . Hence, this bound is independent of the empirical
 185 loss minimization in (2) and the optimization algorithm used to compute the numerical
 186 solution of (2). In other words, Theorem 1.1 and Corollary 1.3 quantify the approxima-
 187 tion power of Floor-ReLU networks with a given size. Designing efficient optimization
 188 algorithms and analyzing the generalization bounds for Floor-ReLU networks are two

189 other separate future directions. Although optimization algorithms and generalization
190 analysis are not our focus in this paper, in the next two paragraphs, we discuss several
191 possible research topics in these directions for our Floor-ReLU networks.

192 In this work, we have not analyzed the feasibility of optimization algorithms for the
193 Floor-ReLU network. Typically, stochastic gradient descent (SGD) is applied to solve
194 a network optimization problem. However, the Floor-ReLU network has piecewise
195 constant activation functions making standard SGD infeasible. There are two possible
196 directions to solve the optimization problem for Floor-ReLU networks: 1) gradient-free
197 optimization methods, e.g., Nelder-Mead method (Nelder and Mead, 1965), genetic al-
198 gorithm (Holland, 1992), simulated annealing (Kirkpatrick et al., 1983), particle swarm
199 optimization (Kennedy and Eberhart, 1995), and consensus-based optimization (Pinnau
200 et al., 2017; Carrillo et al., 2019); 2) applying optimization algorithms for quantized
201 networks that also have piecewise constant activation functions (Lin et al., 2019; Boo
202 et al., 2020; Bengio et al., 2013; Wang et al., 2018; Hubara et al., 2017; Yin et al., 2019).
203 It would be interesting future work to explore efficient learning algorithms based on the
204 Floor-ReLU network.

205 Generalization analysis of Floor-ReLU networks is also an interesting future di-
206 rection. Previous works have shown the generalization power of ReLU networks for
207 regression problems (Jacot et al., 2018; Cao and Gu, 2019; Chen et al., 2019b; E et al.,
208 2019; E and Wojtowytsch, 2020) and for solving partial differential equations (Berner
209 et al., 2018; Luo and Yang, 2020). Regularization strategies for ReLU networks to guar-
210 antee good generalization capacity of deep learning have been proposed in (E et al.,
211 2019; E and Wojtowytsch, 2020). It is important to investigate the generalization ca-
212 pacity of our Floor-ReLU networks. Especially, it is of great interest to see whether
213 problem-dependent regularization strategies exist to make the generalization error of
214 our Floor-ReLU networks free of the curse of dimensionality.

215 2.2 Approximation rates in $\mathcal{O}(N)$ and $\mathcal{O}(L)$ versus $\mathcal{O}(W)$

216 Characterizing deep network approximation in terms of the width $\mathcal{O}(N)$ ³ and depth
217 $\mathcal{O}(L)$ simultaneously is fundamental and indispensable in realistic applications, while
218 quantifying the deep network approximation based on the number of nonzero param-
219 eters W is probably only of interest in theory as far as we know. Theorem 1.1 can
220 provide practical guidance for choosing network sizes in realistic applications while
221 theories in terms of W cannot tell how large a network should be to guarantee a target
222 accuracy. The width and depth are the two most direct and amenable hyper-parameters
223 in choosing a specific network for a learning task, while the number of nonzero parame-

³For simplicity, we omit $\mathcal{O}(\cdot)$ in the following discussion.

ters W is hardly controlled efficiently. Theories in terms of W essentially have a single variable to control the network size in three types of structures: 1) fixing the width N and varying the depth L ; 2) fixing the depth L and changing the width N ; 3) both the width and depth are controlled by the same parameter like the target accuracy ε in a specific way (e.g., N is a polynomial of $\frac{1}{\varepsilon^d}$ and L is a polynomial of $\log(\frac{1}{\varepsilon})$). Considering the non-uniqueness of structures for realizing the same W , it is impractical to develop approximation rates in terms of W covering all these structures. If one network structure has been chosen in a certain application, there might not be a known theory in terms of W to quantify the performance of this structure. Finally, in terms of full error analysis of deep learning including approximation theory, optimization theory, and generalization theory as illustrated in (3), the approximation error characterization in terms of width and depth is more useful than that in terms of the number of parameters, because almost all existing optimization and generalization analysis are based on depth and width instead of the number of parameters (Jacot et al., 2018; Cao and Gu, 2019; Chen et al., 2019b; Arora et al., 2019; Allen-Zhu et al., 2019; E et al., 2019; E and Wojtowysch, 2020; Ji and Telgarsky, 2020), to the best of our knowledge. Approximation results in terms of width and depth are more consistent with optimization and generalization analysis tools to obtain a full error analysis in (3).

Most existing approximation theories for deep neural networks so far focus on the approximation rate in the number of parameters W (Cybenko, 1989; Hornik et al., 1989; Barron, 1993; Liang and Srikant, 2016; Yarotsky, 2017; Poggio et al., 2017; E and Wang, 2018; Petersen and Voigtlaender, 2018; Chui et al., 2018; Yarotsky, 2018; Nakada and Imaizumi, 2019; Gribonval et al., 2019; Gühring et al., 2019; Chen et al., 2019a; Li et al., 2019; Suzuki, 2019; Bao et al., 2019; Opschoor et al., 2019; Yarotsky and Zhevnerchuk, 2019; Bölcskei et al., 2019; Montanelli and Du, 2019; Chen and Wu, 2019; Zhou, 2020; Montanelli and Yang, 2020; Montanelli et al., 2020). From the point of view of theoretical difficulty, controlling two variables N and L in our theory is more challenging than controlling one variable W in the literature. In terms of mathematical logic, the characterization of deep network approximation in terms of N and L can provide an approximation rate in terms of W , while we are not aware of how to derive approximation rates in terms of arbitrary N and L given approximation rates in terms of W , since existing results in terms of W are valid for specific network sizes with width and depth as functions in W without the degree of freedom to take arbitrary values. As we have discussed in the last paragraph, existing theories essentially have a single variable to control the network size in three types of structures. Let us use the first type of structures, which includes the best-known result for a nearly optimal approximation rate, $\mathcal{O}(\omega_f(W^{-2/d}))$, for continuous functions in terms of W using ReLU networks (Yarotsky, 2018) and the best-known result, $\mathcal{O}(\exp(-c_{\alpha,d}\sqrt{W}))$, for Hölder

continuous functions of order α using Sine-ReLU networks (Yarotsky and Zhevnerchuk, 2019), as an example to show how Theorem 1.1 in terms of N and L can be applied to show a better result in terms of W . One can apply Theorem 1.1 in a similar way to obtain other corollaries with other types of structures in terms of W . The main idea is to specify the value of N and L in Theorem 1.1 to show the desired corollary. For example, if we let the width parameter $N = 2$ and the depth parameter $L = W$ in Theorem 1.1, then the width is $\max\{d, 23\}$, the depth is $64dW + 3$, and the total number of parameters is bounded by $\mathcal{O}(\max\{d^2, 23^2\}(64dW + 3)) = \mathcal{O}(W)$. Therefore, we can prove Corollary 2.1 below for the approximation capacity of our Floor-ReLU networks in terms of the total number of parameters as follows.

Corollary 2.1. *Given any $W \in \mathbb{N}^+$ and a continuous function f on $[0, 1]^d$, there exists a function ϕ implemented by a Floor-ReLU network with $\mathcal{O}(W)$ nonzero parameters, a width $\max\{d, 23\}$ and depth $64dW + 3$, such that*

$$|\phi(\mathbf{x}) - f(\mathbf{x})| \leq \omega_f(\sqrt{d}2^{-\sqrt{W}}) + 2\omega_f(\sqrt{d})2^{-\sqrt{W}}, \quad \text{for any } \mathbf{x} \in [0, 1]^d.$$

Corollary 2.1 achieves root exponential convergence without the curse of dimensionality in terms of the number of parameters W with the help of the Floor-ReLU networks. When only ReLU networks are used, the result in (Yarotsky, 2018) suffers from the curse and does not have any kind of exponential convergence. The result in (Yarotsky and Zhevnerchuk, 2019) with Sine-ReLU networks has root exponential convergence but has not excluded the possibility of the curse of dimensionality as we shall discuss later. Furthermore, Corollary 2.1 works for generic continuous functions while (Yarotsky and Zhevnerchuk, 2019) only applies to Hölder continuous functions.

2.3 Further interpretation of our theory

In the interpretation of our theory, there are two more aspects that are important to discuss. The first one is whether it is possible to extend our theory to functions on a more general domain, e.g., $[-M, M]^d$ for some $M > 1$, because $M > 1$ may cause an implicit curse of dimensionality in some existing theory as we shall point out later. The second one is how bad the modulus of continuity would be since it is related to a high-dimensional function f that may lead to an implicit curse of dimensionality in our approximation rate.

First, Theorem 1.1 can be easily generalized to $C([-M, M]^d)$ for any $M > 0$. Let \mathcal{L} be a linear map given by $\mathcal{L}(\mathbf{x}) = 2M(\mathbf{x} - 1/2)$. By Theorem 1.1, for any $f \in C([-M, M]^d)$, there exists ϕ implemented by a Floor-ReLU network with width $\max\{d, 5N + 13\}$ and depth $64dL + 3$ such that

$$|\phi(\mathbf{x}) - f \circ \mathcal{L}(\mathbf{x})| \leq \omega_{f \circ \mathcal{L}}(\sqrt{d}N^{-\sqrt{L}}) + 2\omega_{f \circ \mathcal{L}}(\sqrt{d})N^{-\sqrt{L}}, \quad \text{for any } \mathbf{x} \in [0, 1]^d.$$

It follows from $\mathbf{y} = \mathcal{L}(\mathbf{x}) \in [-M, M]^d$ and $\omega_{f \circ \mathcal{L}}(r) = \omega_f^{[-M, M]^d}(2Mr)$ for any $r \geq 0$ that,⁴ for any $\mathbf{y} \in [-M, M]^d$,

$$|\phi(\frac{\mathbf{y} + M}{2M}) - f(\mathbf{y})| \leq \omega_f^{[-M, M]^d}(2M\sqrt{d}N^{-\sqrt{L}}) + 2\omega_f^{[-M, M]^d}(2M\sqrt{d})N^{-\sqrt{L}}. \quad (4)$$

Hence, the size of the function domain $[-M, M]^d$ only has a mild influence on the approximation rate of our Floor-ReLU networks. Floor-ReLU networks can still avoid the curse of dimensionality and achieve root exponential convergence for continuous functions on $[-M, M]^d$ when $M > 1$. For example, in the case of Hölder continuous functions of order α with a constant λ on $[-M, M]^d$, our approximation rate becomes $3\lambda(2M\sqrt{d}N^{-\sqrt{L}})^\alpha$.

Second, most interesting continuous functions in practice have a good modulus of continuity such that there is no implicit curse of dimensionality hiding in $\omega_f(\cdot)$. For example, we have discussed the case of Hölder continuous functions previously. We would like to remark that the class of Hölder continuous functions implicitly depends on d through its definition in (1), but this dependence is moderate since the ℓ^2 - norm in (1) is the square root of a sum with d terms. Let us now discuss several cases of $\omega_f(\cdot)$ when we cannot achieve exponential convergence or cannot avoid the curse of dimensionality. The first example is $\omega_f(r) = \frac{1}{\ln(1/r)}$ for all small $r > 0$, which leads to an approximation rate

$$3(\sqrt{L} \ln N - \frac{1}{2} \ln d)^{-1}, \quad \text{for large } N, L \in \mathbb{N}^+.$$

Apparently, the above approximation rate still avoids the curse of dimensionality but there is no exponential convergence, which has been canceled out by “ln” in $\omega_f(\cdot)$. The second example is $\omega_f(r) = \frac{1}{\ln^{1/d}(1/r)}$ for all small $r > 0$, which leads to an approximation rate

$$3(\sqrt{L} \ln N - \frac{1}{2} \ln d)^{-1/d}, \quad \text{for large } N, L \in \mathbb{N}^+.$$

The power $\frac{1}{d}$ further weakens the approximation rate and hence the curse of dimensionality occurs. The last example we would like to discuss is $\omega_f(r) = r^{\alpha/d}$ for all small $r > 0$, which results in the approximation rate

$$3d^{\frac{\alpha}{2d}} N^{-\frac{\alpha}{d}\sqrt{L}}, \quad \text{for large } N, L \in \mathbb{N}^+,$$

which achieves the exponential convergence and avoids the curse of dimensionality when we use very deep networks with a fixed width. But if we fix the depth, there is no exponential convergence and the curse occurs. Though we have provided several examples of immoderate $\omega_f(\cdot)$, to the best of our knowledge, we are not aware of practically useful continuous functions with $\omega_f(\cdot)$ that is immoderate.

⁴For an arbitrary set $E \subseteq \mathbb{R}^d$, $\omega_f^E(r)$ is defined via $\omega_f^E(r) := \sup \{|f(\mathbf{x}) - f(\mathbf{y})| : \|\mathbf{x} - \mathbf{y}\|_2 \leq r, \mathbf{x}, \mathbf{y} \in E\}$, for any $r \geq 0$. As defined earlier, $\omega_f(r)$ is short of $\omega_f^{[0,1]^d}(r)$.

2.4 Discussion on the literature

The neural networks constructed here achieve exponential convergence without the curse of dimensionality simultaneously for a function class as general as (Hölder) continuous functions, while—to the best of our knowledge—most existing theories only apply to functions with an intrinsic low complexity. For example, the exponential convergence was studied for polynomials (Yarotsky, 2017; Montanelli et al., 2020; Lu et al., 2020), smooth functions (Montanelli et al., 2020; Liang and Srikant, 2016), analytic functions (E and Wang, 2018), and functions admitting a holomorphic extension to a Bernstein polyellipse (Opschoor et al., 2019). For another example, no curse of dimensionality occurs, or the curse is lessened for Barron spaces (Barron, 1993; E et al., 2019; E and Wojtowytsch, 2020), Korobov spaces (Montanelli and Du, 2019), band-limited functions (Chen and Wu, 2019; Montanelli et al., 2020), compositional functions (Poggio et al., 2017), and smooth functions (Yarotsky and Zhevnerchuk, 2019; Lu et al., 2020; Montanelli and Yang, 2020; Yang and Wang, 2020).

Our theory admits a neat and explicit approximation error bound. For example, our approximation rate in the case of Hölder continuous functions of order α with a constant λ is $3\lambda d^{\alpha/2} N^{-\alpha\sqrt{L}}$, while the prefactor of most existing theories is unknown or grows exponentially in d . Our proof fully explores the advantage of the compositional structure and the nonlinearity of deep networks, while many existing theories were built on traditional approximation tools (e.g., polynomial approximation, multiresolution analysis, and Monte Carlo sampling), making it challenging for existing theories to obtain a neat and explicit error bound with an exponential convergence and without the curse of dimensionality.

Let us review existing works in more detail below.

Curse of dimensionality. The curse of dimensionality is the phenomenon that approximating a d -dimensional function using a certain parametrization method with a fixed target accuracy generally requires a large number of parameters that is exponential in d and this expense quickly becomes unaffordable when d is large. For example, traditional finite element methods with W parameters can achieve an approximation accuracy $O(W^{-1/d})$ with an explicit indicator of the curse $\frac{1}{d}$ in the power of W . If an approximation rate has a constant independent of W and exponential in d , the curse still occurs implicitly through this prefactor by definition. If the approximation rate has a prefactor C_f depending on f , then the prefactor C_f still depends on d implicitly via f and the curse implicitly occurs if C_f exponentially grows when d increases. Designing a parametrization method that can overcome the curse of dimensionality is an important research topic in approximation theory.

In (Barron, 1993) and its variants or generalization (E et al., 2019; E and Wojtowytsch, 2020; Chen and Wu, 2019; Montanelli et al., 2020), d -dimensional functions

defined on a domain $\Omega \subseteq \mathbb{R}^d$ admitting an integral representation with an integrand as a ridge function on $\tilde{\Omega} \subseteq \mathbb{R}^d$ with a variable coefficient were considered, e.g.,

$$f(\mathbf{x}) = \int_{\tilde{\Omega}} a(\mathbf{w}) K(\mathbf{w} \cdot \mathbf{x}) d\nu(\mathbf{w}), \quad (5)$$

where $\nu(\mathbf{w})$ is a Lebesgue measure in \mathbf{w} . $f(\mathbf{x})$ can be reformulated into the expectation of a high-dimensional random function when \mathbf{w} is treated as a random variable. Then $f(\mathbf{x})$ can be approximated by the average of W samples of the integrand in the same spirit of the law of large numbers with an approximation error essentially bounded by $\frac{C_f \sqrt{\mu(\Omega)}}{\sqrt{W}}$ measured in $L^2(\Omega, \mu)$ (Equation (6) of (Barron, 1993)), where $\mathcal{O}(W)$ is the total number of parameters in the network, C_f is a d -dimensional integral with an integrand related to f , and $\mu(\Omega)$ is the Lebesgue measure of Ω . As pointed out in (Barron, 1993) right after Equation (6), if Ω is not a unit domain in \mathbb{R}^d , $\mu(\Omega)$ would be exponential in d ; at the beginning of Page 932 of (Barron, 1993), it was remarked that C_f can often be exponentially large in d and standard smoothness properties of f alone are not enough to remove the exponential dependence of C_f on d , though there is a large number of examples for which C_f is only moderately large. Therefore, the curse of dimensionality occurs unless C_f and $\mu(\Omega)$ are not exponential in d . It was observed that if the error is measured in the sense of mean squared error in machine learning, which is the square of the $L^2(\Omega, \mu)$ error averaged over $\mu(\Omega)$ resulting in $\frac{C_f^2}{W}$, then the mean squared error has no curse of dimensionality as long as C_f is not exponential in d (Barron, 1993; E et al., 2019; E and Wojtowytsch, 2020).

In (Montanelli and Du, 2019), d -dimensional functions in the Korobov space are approximated by the linear combination of basis functions of a sparse grid, each of which is approximated by a ReLU network. Though the curse of dimensionality has been lessened, target functions have to be sufficiently smooth and the approximation error still contains a factor that is exponential in d , i.e., the curse still occurs. Other works in (Yarotsky, 2017; Yarotsky and Zhevnerchuk, 2019; Lu et al., 2020; Yang and Wang, 2020) study the advantage of smoothness in the network approximation. Polynomials are applied to approximate smooth functions and ReLU networks are constructed to approximate polynomials. The application of smoothness can lessen the curse of dimensionality in the approximation rates in terms of network sizes but also results in a prefactor that is exponentially large in the dimension, which means that the curse still occurs implicitly.

The Kolmogorov-Arnold superposition theorem (KST) (Kolmogorov, 1956; Arnold, 1957; Kolmogorov, 1957) has also inspired a research direction of network approximation (Kůrková, 1992; Maiorov and Pinkus, 1999; Igel'nik and Parikh, 2003; Montanelli and Yang, 2020) for continuous functions. (Kůrková, 1992) provided a quantitative approximation rate of networks with two hidden layers, but the number of neurons scales

exponentially in the dimension and the curse occurs. (Maierov and Pinkus, 1999) relaxes the exact representation in KST to an approximation in a form of two-hidden-layer neural networks with a maximum width $6d + 3$ and a single activation function. This powerful activation function is very complex as described by its authors and its numerical evaluation was not available until a more concrete algorithm was recently proposed in (Guliyev and Ismailov, 2018). Note that there is no available numerical algorithm in (Maierov and Pinkus, 1999; Guliyev and Ismailov, 2018) to compute the whole networks proposed therein. The difficulty is due to the fact that the construction of these networks relies on the outer univariate continuous function of the KST. Though the existence of these outer functions can be shown by construction via a complicated iterative procedure in (Braun and Griebel, 2009), there is no existing numerical algorithm to evaluate them for a given target function yet, even though computation with an arbitrary precision is assumed to be available. Therefore, the networks considered in (Maierov and Pinkus, 1999; Guliyev and Ismailov, 2018) are similar to the original representation in KST in the sense that their existence is proved without an explicit way or numerical algorithm to construct them. (Igel'nik and Parikh, 2003) and (Montanelli and Yang, 2020) apply cubic-splines and piecewise linear functions to approximate the inner and outer functions of KST, resulting in cubic-spline and ReLU networks to approximate continuous functions on $[0, 1]^d$. Due to the pathological outer functions of KST, the approximation bounds still suffer from the curse of dimensionality unless target functions are restricted to a small class of functions with simple outer functions in the KST.

Recently in (Yarotsky and Zhevnerchuk, 2019), Sine-ReLU networks have been applied to approximate Hölder continuous functions of order α on $[0, 1]^d$ with an approximation accuracy $\varepsilon = \exp(-c_{\alpha,d}W^{1/2})$, where W is the number of parameters in the network and $c_{\alpha,d}$ is a positive constant depending on α and d only. Whether or not $c_{\alpha,d}$ exponentially depends on d determines whether or not the curse of dimensionality exists for the Sine-ReLU networks, which is not answered in (Yarotsky and Zhevnerchuk, 2019) and is still an open question.

Finally, we would like to discuss the curse of dimensionality in terms of the continuity of the weight selection as a map $\Sigma : C([0, 1]^d) \rightarrow \mathbb{R}^W$. For a fixed network architecture with a fixed number of parameters W , let $g : \mathbb{R}^W \rightarrow C([0, 1]^d)$ be the map of realizing a DNN from a given set of parameters in \mathbb{R}^W to a function in $C([0, 1]^d)$. Suppose that there is a continuous map Σ from the unit ball of Sobolev space with smoothness s , denoted as $F_{s,d}$, to \mathbb{R}^W such that $\|f - g(\Sigma(f))\|_{L^\infty} \leq \varepsilon$ for all $f \in F_{s,d}$. Then $W \geq c\varepsilon^{-d/s}$ with some constant c depending only on s . This conclusion is given in Theorem 3 of (Yarotsky, 2017), which is a corollary of Theorem 4.2 of (Devore, 1989) in a more general form. Intuitively, this conclusion means that any constructive

443 approximation of ReLU FNNs to approximate $C([0, 1]^d)$ cannot enjoy a continuous
 444 weight selection property if the approximation rate is better than $c\varepsilon^{-d/s}$, i.e., the curse
 445 of dimensionality must occur for constructive approximation for ReLU FNNs with a
 446 continuous weight selection. Theorem 4.2 of (Devore, 1989) can also lead to a new
 447 corollary with a weight selection map $\Sigma : K_{s,d} \rightarrow \mathbb{R}^W$ (e.g., the constructive approxi-
 448 mation of Floor-ReLU networks) and $g : \mathbb{R}^W \rightarrow L^\infty([0, 1]^d)$ (e.g., the realization map
 449 of Floor-ReLU networks), where $K_{s,d}$ is the unit ball of $C^s([0, 1]^d)$ with the Sobolev
 450 norm $W^{s,\infty}([0, 1]^d)$. Then this new corollary implies that the constructive approxima-
 451 tion in this paper cannot enjoy continuous weight selection. However, Theorem 4.2
 452 of (Devore, 1989) is essentially a min-max criterion to evaluate weight selection maps
 453 maintaining continuity: the approximation error obtained by minimizing over all con-
 454 tinuous selection Σ and network realization g and maximizing over all target functions
 455 is bounded below by $\mathcal{O}(W^{-s/d})$. In the worst scenario, a continuous weight selec-
 456 tion cannot enjoy an approximation rate beating the curse of dimensionality. However,
 457 Theorem 4.2 of (Devore, 1989) has not excluded the possibility that most continuous
 458 functions of interest in practice may still enjoy a continuous weight selection without
 459 the curse of dimensionality.

460 **Exponential convergence.** Exponential convergence is referred to as the situation
 461 that the approximation error exponentially decays to zero when the number of param-
 462 eters increases. Designing approximation tools with an exponential convergence is an-
 463 other important topic in approximation theory. In the literature of deep network approx-
 464 imation, when the number of network parameters W is a polynomial of $\mathcal{O}(\log(\frac{1}{\varepsilon}))$, the
 465 terminology “exponential convergence” was also used (E and Wang, 2018; Yarotsky and
 466 Zhevnerchuk, 2019; Opschoor et al., 2019). The exponential convergence in this paper
 467 is root-exponential as in (Yarotsky and Zhevnerchuk, 2019), i.e., $W = \mathcal{O}(\log^2(\frac{1}{\varepsilon}))$. The
 468 exponential convergence in other works is worse than root-exponential.

469 In most cases, the approximation power to achieve exponential approximation rates
 470 in existing works comes from traditional tools for approximating a small class of func-
 471 tions instead of taking advantage of the network structure itself. In (E and Wang, 2018;
 472 Opschoor et al., 2019), highly smooth functions are first approximated by the linear
 473 combination of special polynomials with high degrees (e.g., Chebyshev polynomials,
 474 Legendre polynomials) with an exponential approximation rate, i.e., to achieve an ε -
 475 accuracy, a linear combination of only $\mathcal{O}(p(\log(\frac{1}{\varepsilon})))$ polynomials is required, where p
 476 is a polynomial with a degree that may depend on the dimension d . Then each poly-
 477 nomial is approximated by a ReLU network with $\mathcal{O}(\log(\frac{1}{\varepsilon}))$ parameters. Finally, all
 478 ReLU networks are assembled to form a large network approximating the target func-
 479 tion with an exponential approximation rate. As far as we know, the only existing work
 480 that achieves exponential convergence without taking advantage of special polynomials

481 and smoothness is the Sine-ReLU network in (Yarotsky and Zhevnerchuk, 2019), which
 482 has been mentioned in the paragraph just above. We would like to emphasize that the
 483 result in our paper applies for generic continuous functions including, but not limited
 484 to, the Hölder continuous functions considered in (Yarotsky and Zhevnerchuk, 2019).

485 3 Approximation of continuous functions

486 In this section, we first introduce basic notations in this paper in Section 3.1. Then we
 487 prove Theorem 1.1 based on Proposition 3.2, which will be proved in Section 4.

488 3.1 Notations

489 The main notations of this paper are listed as follows.

490 • Vectors and matrices are denoted in a bold font. Standard vectorization is adopted
 491 in the matrix and vector computation. For example, adding a scalar and a vector
 492 means adding the scalar to each entry of the vector.

493 • Let \mathbb{N}^+ denote the set containing all positive integers, i.e., $\mathbb{N}^+ = \{1, 2, 3, \dots\}$.

494 • Let $\sigma : \mathbb{R} \rightarrow \mathbb{R}$ denote the rectified linear unit (ReLU), i.e. $\sigma(x) = \max\{0, x\}$.
 495 With a slight abuse of notation, we define $\sigma : \mathbb{R}^d \rightarrow \mathbb{R}^d$ as $\sigma(\mathbf{x}) = \begin{bmatrix} \max\{0, x_1\} \\ \vdots \\ \max\{0, x_d\} \end{bmatrix}$

496 for any $\mathbf{x} = (x_1, \dots, x_d) \in \mathbb{R}^d$.

497 • The floor function (Floor) is defined as $\lfloor x \rfloor := \max\{n : n \leq x, n \in \mathbb{Z}\}$ for any
 498 $x \in \mathbb{R}$.

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

502 • The expression “a network with width N and depth L ” means

503 – The maximum width of this network for all **hidden** layers is no more than
 504 N .

505 – The number of **hidden** layers of this network is no more than L .

3.2 Proof of Theorem 1.1

Theorem 1.1 is an immediate consequence of Theorem 3.1 below.

Theorem 3.1. *Given any $N, L \in \mathbb{N}^+$ and an arbitrary continuous function f on $[0, 1]^d$, there exists a function ϕ implemented by a Floor-ReLU network with width $\max\{d, 2N^2 + 5N\}$ and depth $7dL^2 + 3$ such that*

$$|\phi(\mathbf{x}) - f(\mathbf{x})| \leq \omega_f(\sqrt{d} N^{-L}) + 2\omega_f(\sqrt{d})2^{-NL}, \quad \text{for any } \mathbf{x} \in [0, 1]^d.$$

This theorem will be proved later in this section. Now let us prove Theorem 1.1 based on Theorem 3.1.

Proof of Theorem 1.1. Given any $N, L \in \mathbb{N}^+$, there exist $\tilde{N}, \tilde{L} \in \mathbb{N}^+$ with $\tilde{N} \geq 2$ and $\tilde{L} \geq 3$ such that

$$(\tilde{N} - 1)^2 \leq N < \tilde{N}^2 \quad \text{and} \quad (\tilde{L} - 1)^2 \leq 4L < \tilde{L}^2.$$

By Theorem 3.1, there exists a function ϕ implemented by a Floor-ReLU network with width $\max\{d, 2\tilde{N}^2 + 5\tilde{N}\}$ and depth $7d\tilde{L}^2 + 3$ such that

$$|\phi(\mathbf{x}) - f(\mathbf{x})| \leq \omega_f(\sqrt{d} \tilde{N}^{-\tilde{L}}) + 2\omega_f(\sqrt{d})2^{-\tilde{N}\tilde{L}}, \quad \text{for any } \mathbf{x} \in [0, 1]^d.$$

Note that

$$2^{-\tilde{N}\tilde{L}} \leq \tilde{N}^{-\tilde{L}} = (\tilde{N}^2)^{-\frac{1}{2}\sqrt{\tilde{L}^2}} \leq N^{-\frac{1}{2}\sqrt{4L}} \leq N^{-\sqrt{L}}.$$

Then we have

$$|\phi(\mathbf{x}) - f(\mathbf{x})| \leq \omega_f(\sqrt{d} N^{-\sqrt{L}}) + 2\omega_f(\sqrt{d})N^{-\sqrt{L}}, \quad \text{for any } \mathbf{x} \in [0, 1]^d.$$

For $\tilde{N}, \tilde{L} \in \mathbb{N}^+$ with $\tilde{N} \geq 2$ and $\tilde{L} \geq 3$, we have

$$2\tilde{N}^2 + 5\tilde{N} \leq 5(\tilde{N} - 1)^2 + 13 \leq 5N + 13 \quad \text{and} \quad 7\tilde{L}^2 \leq 16(\tilde{L} - 1)^2 \leq 64L.$$

Therefore, ϕ can be computed by a Floor-ReLU network with width $\max\{d, 2\tilde{N}^2 + 5\tilde{N}\} \leq \max\{d, 5N + 13\}$ and depth $7d\tilde{L}^2 + 3 \leq 64dL + 3$, as desired. So we finish the proof. \square

To prove Theorem 3.1, we first present the proof sketch. Put briefly, we construct piecewise constant functions implemented by Floor-ReLU networks to approximate continuous functions. There are four key steps in our construction.

1. Normalize f as \tilde{f} satisfying $\tilde{f}(\mathbf{x}) \in [0, 1]$ for any $\mathbf{x} \in [0, 1]^d$, divide $[0, 1]^d$ into a set of non-overlapping cubes $\{Q_\beta\}_{\beta \in \{0, 1, \dots, K-1\}^d}$, and denote \mathbf{x}_β as the vertex of Q_β with minimum $\|\cdot\|_1$ norm, where K is an integer determined later. See Figure 2 for the illustrations of Q_β and \mathbf{x}_β .

- 536 2. Construct a Floor-ReLU sub-network to implement a vector-valued function $\Phi_1 : \mathbb{R}^d \rightarrow \mathbb{R}^d$ projecting the whole cube Q_β to the index β for each $\beta \in \{0, 1, \dots, K-1\}^d$, i.e., $\Phi_1(x) = \beta$ for all $x \in Q_\beta$.
537
538
- 539 3. Construct a Floor-ReLU sub-network to implement a function $\phi_2 : \mathbb{R}^d \rightarrow \mathbb{R}$ mapping $\beta \in \{0, 1, \dots, K-1\}^d$ approximately to $\tilde{f}(x_\beta)$ for each β , i.e., $\phi_2(\beta) \approx \tilde{f}(x_\beta)$. Then $\phi_2 \circ \Phi_1(x) = \phi_2(\beta) \approx \tilde{f}(x_\beta)$ for any $x \in Q_\beta$ and each $\beta \in \{0, 1, \dots, K-1\}^d$, implying $\tilde{\phi} := \phi_2 \circ \Phi_1$ approximates \tilde{f} within an error $\mathcal{O}(\omega_f(1/K))$ on $[0, 1]^d$.
540
541
542
543
- 544 4. Re-scale and shift $\tilde{\phi}$ to obtain the desired function ϕ approximating f well and
545 determine the final Floor-ReLU network to implement ϕ .

546 It is not difficult to construct Floor-ReLU networks with the desired width and depth
547 to implement Φ_1 . The most technical part is the construction of a Floor-ReLU network
548 with the desired width and depth computing ϕ_2 , which needs the following proposition
549 based on the “bit extraction” technique introduced in (Bartlett et al., 1998; Harvey et al.,
550 2017).

551 **Proposition 3.2.** *Given any $N, L \in \mathbb{N}^+$ and arbitrary $\theta_m \in \{0, 1\}$ for $m = 1, 2, \dots, N^L$,
552 there exists a function ϕ computed by a Floor-ReLU network with width $2N + 2$ and
553 depth $7L - 2$ such that*

$$554 \quad \phi(m) = \theta_m, \quad \text{for } m = 1, 2, \dots, N^L.$$

555 The proof of this proposition is presented in Section 4. By this proposition and
556 the definition of VC-dimension (e.g., see (Harvey et al., 2017)), it is easy to prove
557 that the VC-dimension of Floor-ReLU networks with a constant width and depth $\mathcal{O}(L)$
558 has a lower bound 2^L . Such a lower bound is much larger than $\mathcal{O}(L^2)$, which is a
559 VC-dimension upper bound of ReLU networks with the same width and depth due to
560 Theorem 8 of (Harvey et al., 2017). This means Floor-ReLU networks are much more
561 powerful than ReLU networks from the perspective of VC-dimension.

562 Based on the proof sketch stated just above, we are ready to give the detailed
563 proof of Theorem 3.1 following similar ideas as in our previous work (Shen et al.,
564 2019a; Shen et al., 2019b; Lu et al., 2020). The main idea of our proof is to re-
565 duce high-dimensional approximation to one-dimensional approximation via a projec-
566 tion. The idea of projection was probably first used in well-established theories, e.g.,
567 KST (Kolmogorov superposition theorem) mentioned in Section 2, where the approxi-
568 mant to high-dimensional functions is constructed by: first, projecting high-dimensional
569 data points to one-dimensional data points; second, construct one-dimensional approx-
570 imants. There has been extensive research based on this idea, e.g., references related

to KST summarized in Section 2, our previous works (Shen et al., 2019a; Shen et al., 2019b; Lu et al., 2020), and (Yarotsky and Zhevnerchuk, 2019). The key to a successful approximant is to construct one-dimensional approximants to deal with a large number of one-dimensional data points; in fact, the number of points is exponential in the dimension d .

Proof of Theorem 3.1. The proof consists of four steps.

Step 1: Set up.

Assume f is not a constant function since it is a trivial case. Then $\omega_f(r) > 0$ for any $r > 0$. Clearly, $|f(\mathbf{x}) - f(\mathbf{0})| \leq \omega_f(\sqrt{d})$ for any $\mathbf{x} \in [0, 1]^d$. Define

$$\tilde{f} := (f - f(\mathbf{0}) + \omega_f(\sqrt{d})) / (2\omega_f(\sqrt{d})). \quad (6)$$

It follows that $\tilde{f}(\mathbf{x}) \in [0, 1]$ for any $\mathbf{x} \in [0, 1]^d$.

Set $K = N^L$, $E_{K-1} = [\frac{K-1}{K}, 1]$, and $E_k = [\frac{k}{K}, \frac{k+1}{K})$ for $k = 0, 1, \dots, K-2$. Define $\mathbf{x}_\beta := \beta/K$ and

$$Q_\beta := \left\{ \mathbf{x} = (x_1, x_2, \dots, x_d) \in \mathbb{R}^d : x_j \in E_{\beta_j} \text{ for } j = 1, 2, \dots, d \right\},$$

for any $\beta = (\beta_1, \beta_2, \dots, \beta_d) \in \{0, 1, \dots, K-1\}^d$. See Figure 2 for the examples of Q_β and \mathbf{x}_β for $\beta \in \{0, 1, \dots, K-1\}^d$ with $K = 4$ and $d = 1, 2$.

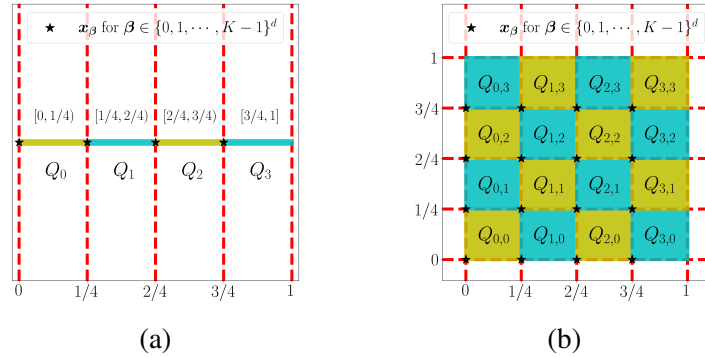


Figure 2: Illustrations of Q_β and \mathbf{x}_β for $\beta \in \{0, 1, \dots, K-1\}^d$. (a) $K = 4, d = 1$. (b) $K = 4, d = 2$.

Step 2: Construct Φ_1 mapping $\mathbf{x} \in Q_\beta$ to β .

Define a step function ϕ_1 as

$$\phi_1(x) := \lfloor -\sigma(-Kx + K - 1) + K - 1 \rfloor, \quad \text{for any } x \in \mathbb{R}.^5$$

See Figure 3 for an example of ϕ_1 when $K = 4$. It follows from the definition of ϕ_1 that

$$\phi_1(x) = k, \quad \text{if } x \in E_k, \text{ for } k = 0, 1, \dots, K-1.$$

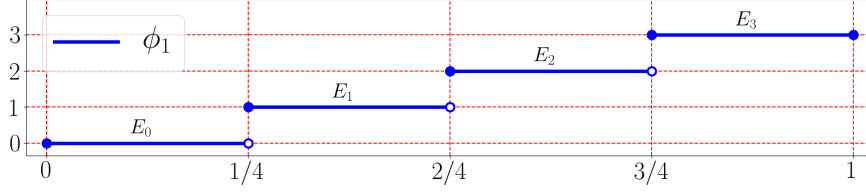


Figure 3: An illustration of ϕ_1 on $[0, 1]$ for the case $K = 4$.

592 Define

593
$$\Phi_1(\mathbf{x}) := (\phi_1(x_1), \phi_1(x_2), \dots, \phi_1(x_d)), \quad \text{for any } \mathbf{x} = (x_1, x_2, \dots, x_d) \in \mathbb{R}^d.$$

594 Clearly, we have, for $\mathbf{x} \in Q_\beta$ and $\beta \in \{0, 1, \dots, K-1\}^d$,

595
$$\Phi_1(\mathbf{x}) = (\phi_1(x_1), \phi_1(x_2), \dots, \phi_1(x_d)) = (\beta_1, \beta_2, \dots, \beta_d) = \beta.$$

596 **Step 3:** Construct ϕ_2 mapping $\beta \in \{0, 1, \dots, K-1\}^d$ approximately to $\tilde{f}(\mathbf{x}_\beta)$.

597 Using the idea of K -ary representation, we define a linear function ψ_1 via

598
$$\psi_1(\mathbf{x}) := 1 + \sum_{j=1}^d x_j K^{j-1}, \quad \text{for any } \mathbf{x} = (x_1, x_2, \dots, x_d) \in \mathbb{R}^d.$$

599 Then ψ_1 is a bijection from $\{0, 1, \dots, K-1\}^d$ to $\{1, 2, \dots, K^d\}$.

600 Given any $i \in \{1, 2, \dots, K^d\}$, there exists a unique $\beta \in \{0, 1, \dots, K-1\}^d$ such that
601 $i = \psi_1(\beta)$. Then define

602
$$\xi_i := \tilde{f}(\mathbf{x}_\beta) \in [0, 1], \quad \text{for } i = \psi_1(\beta) \text{ and } \beta \in \{0, 1, \dots, K-1\}^d,$$

603 where \tilde{f} is the normalization of f defined in Equation (6). It follows that there exists
604 $\xi_{i,j} \in [0, 1]$ for $j = 1, 2, \dots, NL$ such that

605
$$|\xi_i - \text{bin}_{0, \xi_{i,1} \xi_{i,2} \dots, \xi_{i,NL}}| \leq 2^{-NL}, \quad \text{for } i = 1, 2, \dots, K^d.$$

606 By $K^d = (NL)^d = N^{dL}$ and Proposition 3.2, there exists a function $\psi_{2,j}$ implemented
607 by a Floor-ReLU network with width $2N+2$ and depth $7dL-2$, for each $j = 1, 2, \dots, NL$,
608 such that

609
$$\psi_{2,j}(i) = \xi_{i,j}, \quad \text{for } i = 1, 2, \dots, K^d.$$

610 Define

611
$$\psi_2 := \sum_{j=1}^{NL} 2^{-j} \psi_{2,j} \quad \text{and} \quad \phi_2 := \psi_2 \circ \psi_1.$$

⁵If we just define $\phi_1(x) = \lfloor Kx \rfloor$, then $\phi_1(1) = K \neq K-1$ even though $1 \in E_{K-1}$.

612 Then, for $i = \psi_1(\beta)$ and $\beta \in \{0, 1, \dots, K-1\}^d$, we have

$$\begin{aligned}
613 \quad |\tilde{f}(\mathbf{x}_\beta) - \phi_2(\beta)| &= |\tilde{f}(\mathbf{x}_\beta) - \psi_2(\psi_1(\beta))| = |\xi_i - \psi_2(i)| = \left| \xi_i - \sum_{j=1}^{NL} 2^{-j} \psi_{2,j}(i) \right| \\
&= |\xi_i - \text{bin}_{0.\xi_{i,1}\xi_{i,2}\dots\xi_{i,NL}}| \leq 2^{-NL}.
\end{aligned} \tag{7}$$

614 **Step 4:** Determine the final network to implement the desired function ϕ .

615 Define $\tilde{\phi} := \phi_2 \circ \Phi_1$, i.e., for any $\mathbf{x} = (x_1, x_2, \dots, x_d) \in \mathbb{R}^d$,

$$616 \quad \tilde{\phi}(\mathbf{x}) = \phi_2 \circ \Phi_1(\mathbf{x}) = \phi_2(\phi_1(x_1), \phi_1(x_2), \dots, \phi_1(x_d)).$$

617 Note that $|\mathbf{x} - \mathbf{x}_\beta| \leq \frac{\sqrt{d}}{K}$ for any $\mathbf{x} \in Q_\beta$ and $\beta \in \{0, 1, \dots, K-1\}^d$. Then we have,
618 for any $\mathbf{x} \in Q_\beta$ and $\beta \in \{0, 1, \dots, K-1\}^d$,

$$\begin{aligned}
&|\tilde{f}(\mathbf{x}) - \tilde{\phi}(\mathbf{x})| \leq |\tilde{f}(\mathbf{x}) - \tilde{f}(\mathbf{x}_\beta)| + |\tilde{f}(\mathbf{x}_\beta) - \tilde{\phi}(\mathbf{x})| \\
619 \quad &\leq \omega_{\tilde{f}}\left(\frac{\sqrt{d}}{K}\right) + |\tilde{f}(\mathbf{x}_\beta) - \phi_2(\Phi_1(\mathbf{x}))| \\
&\leq \omega_{\tilde{f}}\left(\frac{\sqrt{d}}{K}\right) + |\tilde{f}(\mathbf{x}_\beta) - \phi_2(\beta)| \leq \omega_{\tilde{f}}\left(\frac{\sqrt{d}}{K}\right) + 2^{-NL},
\end{aligned}$$

620 where the last inequality comes from Equation (7).

621 Note that $\mathbf{x} \in Q_\beta$ and $\beta \in \{0, 1, \dots, K-1\}^d$ are arbitrary. Since $[0, 1]^d = \bigcup_{\beta \in \{0, 1, \dots, K-1\}^d} Q_\beta$,
622 we have

$$623 \quad |\tilde{f}(\mathbf{x}) - \tilde{\phi}(\mathbf{x})| \leq \omega_{\tilde{f}}\left(\frac{\sqrt{d}}{K}\right) + 2^{-NL}, \quad \text{for any } \mathbf{x} \in [0, 1]^d.$$

624 Define

$$625 \quad \phi := 2\omega_f(\sqrt{d})\tilde{\phi} + f(\mathbf{0}) - \omega_f(\sqrt{d}).$$

626 By $K = N^L$ and $\omega_f(r) = 2\omega_f(\sqrt{d}) \cdot \omega_{\tilde{f}}(r)$ for any $r \geq 0$, we have, for any $\mathbf{x} \in [0, 1]^d$,

$$\begin{aligned}
&|f(\mathbf{x}) - \phi(\mathbf{x})| = 2\omega_f(\sqrt{d})|\tilde{f}(\mathbf{x}) - \tilde{\phi}(\mathbf{x})| \leq 2\omega_f(\sqrt{d})\left(\omega_{\tilde{f}}\left(\frac{\sqrt{d}}{K}\right) + 2^{-NL}\right) \\
627 \quad &\leq \omega_f\left(\frac{\sqrt{d}}{K}\right) + 2\omega_f(\sqrt{d})2^{-NL} \\
&\leq \omega_f(\sqrt{d}N^{-L}) + 2\omega_f(\sqrt{d})2^{-NL}.
\end{aligned}$$

628 It remains to determine the width and depth of the Floor-ReLU network implement-
629 ing ϕ . Clearly, ϕ_2 can be implemented by the architecture in Figure 4.

630 As we can see from Figure 4, ϕ_2 can be implemented by a Floor-ReLU network with
631 width $N(2N+2+3) = 2N^2+5N$ and depth $L(7dL-2+1)+2 = L(7dL-1)+2$. With the
632 network architecture implementing ϕ_2 in hand, $\tilde{\phi}$ can be implemented by the network
633 architecture shown in Figure 5. Note that ϕ is defined via re-scaling and shifting $\tilde{\phi}$. As
634 shown in Figure 5, ϕ and $\tilde{\phi}$ can be implemented by a Floor-ReLU network with width
635 $\max\{d, 2N^2+5N\}$ and depth $1+1+L(7dL-1)+2 \leq 7dL^2+3$. So we finish the proof.

636 \square

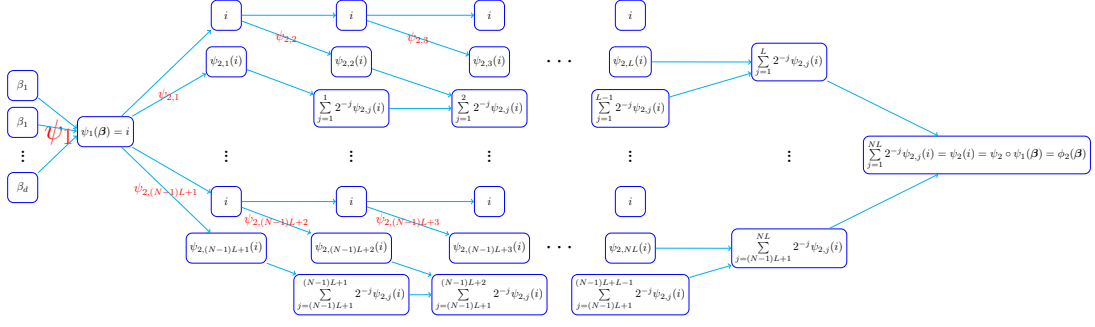


Figure 4: An illustration of the desired network architecture implementing $\phi_2 = \psi_2 \circ \psi_1$ for any input $\beta \in \{0, 1, \dots, K-1\}^d$, where $i = \psi_1(\beta)$.

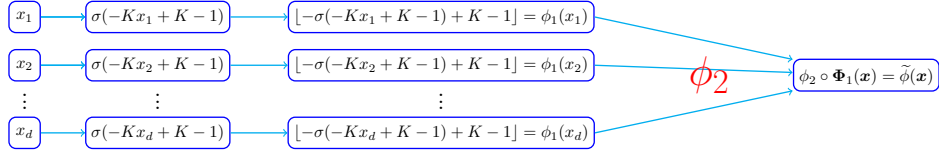


Figure 5: An illustration of the network architecture implementing $\tilde{\phi} = \phi_2 \circ \Phi_1$.

4 Proof of Proposition 3.2

The proof of Proposition 3.2 mainly relies on the “bit extraction” technique. As we shall see later, our key idea is to apply the Floor activation function to make “bit extraction” more powerful to reduce network sizes. In particular, Floor-ReLU networks can extract much more bits than ReLU networks with the same network size.

Let us first establish a basic lemma to extract $1/N$ of the total bits of a binary number; the result is again stored in a binary number.

Lemma 4.1. *Given any $J, N \in \mathbb{N}^+$, there exists a function $\phi : \mathbb{R}^2 \rightarrow \mathbb{R}$ that can be implemented by a Floor-ReLU network with width $2N$ and depth 4 such that, for any $\theta_j \in \{0, 1\}$, $j = 1, \dots, NJ$, we have*

$$\phi(\text{bin}0.\theta_1 \cdots \theta_{NJ}, n) = \text{bin}0.\theta_{(n-1)J+1} \cdots \theta_{nJ}, \quad \text{for } n = 1, 2, \dots, N.$$

Proof. Given any $\theta_j \in \{0, 1\}$ for $j = 1, \dots, NJ$, denote

$$s = \text{bin}0.\theta_1 \cdots \theta_{NJ} \quad \text{and} \quad s_n = \text{bin}0.\theta_{(n-1)J+1} \cdots \theta_{nJ}, \quad \text{for } n = 1, 2, \dots, N.$$

Then our goal is to construct a function $\phi : \mathbb{R}^2 \rightarrow \mathbb{R}$ computed by a Floor-ReLU network with the desired width and depth that satisfies

$$\phi(s, n) = s_n, \quad \text{for } n = 1, 2, \dots, N.$$

Based on the properties of the binary representation, it is easy to check that

$$s_n = \lfloor 2^{nJ} s \rfloor / 2^J - \lfloor 2^{(n-1)J} s \rfloor, \quad \text{for } n = 1, 2, \dots, N. \quad (8)$$

Even with the above formulas to generate s_1, s_2, \dots, s_N , it is still technical to construct a network outputting s_n for a given index $n \in \{1, 2, \dots, N\}$.

Set $\delta = 2^{-J}$ and define g (see Figure 6) as

$$g(x) := \sigma\left(\sigma(x) - \sigma\left(\frac{x+\delta-1}{\delta}\right)\right), \quad \text{where } \sigma(x) = \max\{0, x\}.$$

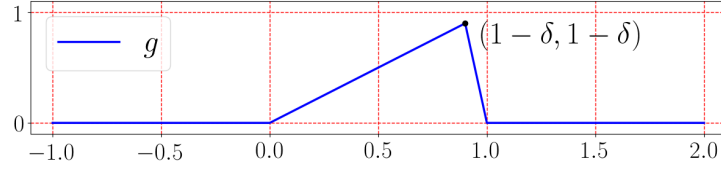


Figure 6: An illustration of $g(x) = \sigma\left(\sigma(x) - \sigma\left(\frac{x+\delta-1}{\delta}\right)\right)$, where $\sigma(x) = \max\{0, x\}$ is the ReLU activation function.

Since $s_n \in [0, 1 - \delta]$ for $n = 1, 2, \dots, N$, we have

$$s_n = \sum_{k=1}^N g(s_k + k - n), \quad \text{for } n = 1, 2, \dots, N. \quad (9)$$

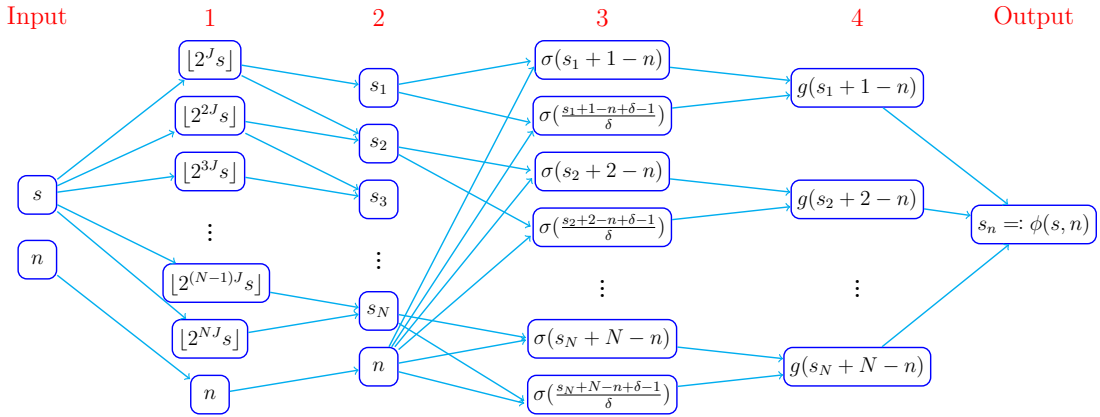


Figure 7: An illustration of the desired network architecture implementing ϕ based on Equation (8) and (9). We omit some ReLU (σ) activation functions when inputs are obviously non-negative. All parameters in this network are essentially determined by Equation (8) and (9), which are valid no matter what $\theta_1, \dots, \theta_{NJ} \in \{0, 1\}$ are. Thus, the desired function ϕ implemented by this network is independent of $\theta_1, \dots, \theta_{NJ} \in \{0, 1\}$.

As shown in Figure 7, the desired function ϕ can be computed by a Floor-ReLU network with width $2N$ and depth 4. Moreover, it holds that

$$\phi(s, n) = s_n, \quad \text{for } n = 1, 2, \dots, N.$$

So we finish the proof. \square

665 The next lemma constructs a Floor-ReLU network that can extract any bit from a
 666 binary representation according to a specific index.

667 **Lemma 4.2.** *Given any $N, L \in \mathbb{N}^+$, there exists a function $\phi : \mathbb{R}^2 \rightarrow \mathbb{R}$ implemented by a*
 668 *Floor-ReLU network with width $2N + 2$ and depth $7L - 3$ such that, for any $\theta_m \in \{0, 1\}$,*
 669 *$m = 1, 2, \dots, N^L$, we have*

$$670 \quad \phi(\text{bin}0.\theta_1\theta_2\cdots\theta_{N^L}, m) = \theta_m, \quad \text{for } m = 1, 2, \dots, N^L.$$

671 *Proof.* The proof is based on repeated applications of Lemma 4.1. Specifically, we
 672 inductively construct a sequence of functions $\phi_1, \phi_2, \dots, \phi_L$ implemented by Floor-ReLU
 673 networks to satisfy the following two conditions for each $\ell \in \{1, 2, \dots, L\}$.

674 (i) $\phi_\ell : \mathbb{R}^2 \rightarrow \mathbb{R}$ can be implemented by a Floor-ReLU network with width $2N + 2$
 675 and depth $7\ell - 3$.

676 (ii) For any $\theta_m \in \{0, 1\}$, $m = 1, 2, \dots, N^\ell$, we have

$$677 \quad \phi_\ell(\text{bin}0.\theta_1\theta_2\cdots\theta_{N^\ell}, m) = \text{bin}0.\theta_m, \quad \text{for } m = 1, 2, \dots, N^\ell.$$

678 Firstly, consider the case $\ell = 1$. By Lemma 4.1 (set $J = 1$ therein), there exists a
 679 function ϕ_1 implemented by a Floor-ReLU network with width $2N \leq 2N + 2$ and depth
 680 $4 = 7 - 3$ such that, for any $\theta_m \in \{0, 1\}$, $m = 1, 2, \dots, N$, we have

$$681 \quad \phi_1(\text{bin}0.\theta_1\theta_2\cdots\theta_N, m) = \text{bin}0.\theta_m, \quad \text{for } m = 1, 2, \dots, N.$$

682 It follows that Condition (i) and (ii) hold for $\ell = 1$.

683 Next, assume Condition (i) and (ii) hold for $\ell = k$. We would like to construct ϕ_{k+1}
 684 to make Condition (i) and (ii) true for $\ell = k + 1$. By Lemma 4.1 (set $J = N^k$ therein),
 685 there exists a function ψ implemented by a Floor-ReLU network with width $2N$ and
 686 depth 4 such that, for any $\theta_m \in \{0, 1\}$, $m = 1, 2, \dots, N^{k+1}$, we have

$$687 \quad \psi(\text{bin}0.\theta_1\cdots\theta_{N^{k+1}}, n) = \text{bin}0.\theta_{(n-1)N^{k+1}+1}\cdots\theta_{(n-1)N^{k+1}+N^k}, \quad \text{for } n = 1, 2, \dots, N. \quad (10)$$

688 By the hypothesis of induction, we have

689 • $\phi_k : \mathbb{R}^2 \rightarrow \mathbb{R}$ can be implemented by a Floor-ReLU network with width $2N + 2$
 690 and depth $7k - 3$.

691 • For any $\theta_j \in \{0, 1\}$, $j = 1, 2, \dots, N^k$, we have

$$692 \quad \phi_k(\text{bin}0.\theta_1\theta_2\cdots\theta_{N^k}, j) = \text{bin}0.\theta_j, \quad \text{for } j = 1, 2, \dots, N^k. \quad (11)$$

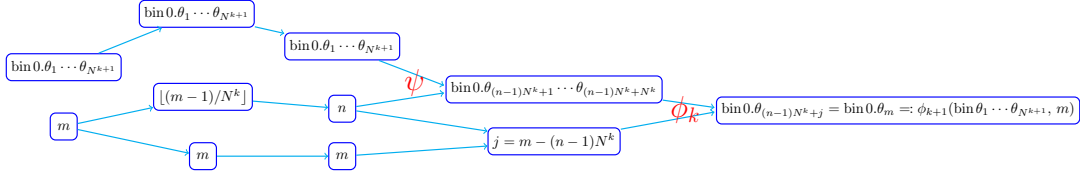


Figure 8: An illustration of the desired network architecture implementing ϕ_{k+1} based on (10), (11), and (12). We omit ReLU (σ) for neurons with non-negative inputs.

Given any $m \in \{1, 2, \dots, N^{k+1}\}$, there exist $n \in \{1, 2, \dots, N\}$ and $j \in \{1, 2, \dots, N^k\}$ such that $m = (n-1)N^k + j$, and such n, j can be obtained by

$$n = \lfloor (m-1)/N^k \rfloor + 1 \quad \text{and} \quad j = m - (n-1)N^k. \quad (12)$$

Then the desired architecture of the Floor-ReLU network implementing ϕ_{k+1} is shown in Figure 8.

Note that ψ can be computed by a Floor-ReLU network of width $2N$ and depth 4. By Figure 8, we have

- $\phi_{k+1} : \mathbb{R}^2 \rightarrow \mathbb{R}$ can be implemented by a Floor-ReLU network with width $2N + 2$ and depth $2 + 4 + 1 + (7k - 3) = 7(k + 1) - 3$, which implies Condition (i) for $\ell = k + 1$.

- For any $\theta_m \in \{0, 1\}$, $m = 1, 2, \dots, N^{k+1}$, we have

$$\phi_{k+1}(\text{bin } 0.\theta_1\theta_2\cdots\theta_{N^{k+1}}, m) = \text{bin } 0.\theta_m, \quad \text{for } m = 1, 2, \dots, N^{k+1}.$$

That is, Condition (ii) holds for $\ell = k + 1$.

So we finish the process of induction.

By the principle of induction, there exists a function $\phi_L : \mathbb{R}^2 \rightarrow \mathbb{R}$ such that

- ϕ_L can be implemented by a Floor-ReLU network with width $2N + 2$ and depth $7L - 3$.

- For any $\theta_m \in \{0, 1\}$, $m = 1, 2, \dots, N^L$, we have

$$\phi_L(\text{bin } 0.\theta_1\theta_2\cdots\theta_{N^L}, m) = \text{bin } 0.\theta_m, \quad \text{for } m = 1, 2, \dots, N^L.$$

Finally, define $\phi := 2\phi_L$. Then ϕ can also be implemented by a Floor-ReLU network with width $2N + 2$ and depth $7L - 3$. Moreover, for any $\theta_m \in \{0, 1\}$, $m = 1, 2, \dots, N^L$, we have

$$\phi(\text{bin } 0.\theta_1\theta_2\cdots\theta_{N^L}, m) = 2 \cdot \phi_L(\text{bin } 0.\theta_1\theta_2\cdots\theta_{N^L}, m) = 2 \cdot \text{bin } 0.\theta_m = \theta_m,$$

for $m = 1, 2, \dots, N^L$. So we finish the proof. \square

717 With Lemma 4.2 in hand, we are ready to prove Proposition 3.2.

718 *Proof of Proposition 3.2.* By Lemma 4.2, there exists a function $\tilde{\phi} : \mathbb{R}^2 \rightarrow \mathbb{R}$ computed
 719 by a Floor-ReLU network with a fixed architecture with width $2N + 2$ and depth $7L - 3$
 720 such that, for any $z_m \in \{0, 1\}$, $m = 1, 2, \dots, N^L$, we have

$$721 \quad \tilde{\phi}(\text{bin}0.z_1 z_2 \dots z_{N^L}, m) = z_m, \quad \text{for } m = 1, 2, \dots, N^L.$$

722 Based on $\theta_m \in \{0, 1\}$ for $m = 1, 2, \dots, N^L$ given in Proposition 3.2, we define the final
 723 function ϕ as

$$724 \quad \phi(x) := \tilde{\phi}(\sigma(x \cdot 0 + \text{bin}0.\theta_1 \theta_2 \dots \theta_{N^L}), \sigma(x)), \quad \text{where } \sigma(x) = \max\{0, x\}.$$

725 Clearly, ϕ can be implemented by a Floor-ReLU network with width $2N + 2$ and depth
 726 $(7L - 3) + 1 = 7L - 2$. Moreover, we have, for any $m \in \{1, 2, \dots, N^L\}$,

$$727 \quad \phi(m) := \tilde{\phi}(\sigma(m \cdot 0 + \text{bin}0.\theta_1 \theta_2 \dots \theta_{N^L}), \sigma(m)) = \tilde{\phi}(\text{bin}0.\theta_1 \theta_2 \dots \theta_{N^L}, m) = \theta_m.$$

728 So we finish the proof. □

729 We finally point out that only the properties of Floor on $[0, \infty)$ are used in our proof.
 730 Thus, the Floor can be replaced by the truncation function that can be easily computed
 731 by truncating the decimal part.

732 5 Conclusion

733 This paper has introduced a theoretical framework to show that deep network approxi-
 734 mation can achieve root exponential convergence and avoid the curse of dimensionality
 735 for approximating functions as general as (Hölder) continuous functions. Given a Lip-
 736 schitz continuous function f on $[0, 1]^d$, it was shown by construction that Floor-ReLU
 737 networks with width $\max\{d, 5N + 13\}$ and depth $64dL + 3$ can achieve a uniform ap-
 738 proximation error bounded by $3\lambda\sqrt{d}N^{-\sqrt{L}}$, where λ is the Lipschitz constant of f .
 739 More generally for an arbitrary continuous function f on $[0, 1]^d$ with a modulus of con-
 740 tinuity $\omega_f(\cdot)$, the approximation error is bounded by $\omega_f(\sqrt{d}N^{-\sqrt{L}}) + 2\omega_f(\sqrt{d})N^{-\sqrt{L}}$.
 741 The results in this paper provide a theoretical lower bound of the power of deep network
 742 approximation. Whether or not this bound is achievable in actual computation relies on
 743 advanced algorithm design as a separate line of research.

744 **Acknowledgments.** Z. Shen is supported by Tan Chin Tuan Centennial Professor-
 745 ship. H. Yang was partially supported by the US National Science Foundation under
 746 award DMS-1945029.

References

- Allen-Zhu, Z., Li, Y., and Liang, Y. (2019). Learning and generalization in overparameterized neural networks, going beyond two layers. *ArXiv*, abs/1811.04918.
- Arnold, V. I. (1957). On functions of three variables. *Dokl. Akad. Nauk SSSR*, pages 679–681.
- Arora, S., Du, S. S., Hu, W., Li, Z., and Wang, R. (2019). Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *ICML*.
- Bao, C., Li, Q., Shen, Z., Tai, C., Wu, L., and Xiang, X. (2019). Approximation analysis of convolutional neural networks. *Semantic Scholar e-Preprint*, page Corpus ID: 204762668.
- Barron, A. R. (1993). Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):930–945.
- Bartlett, P., Maierov, V., and Meir, R. (1998). Almost linear VC-dimension bounds for piecewise polynomial networks. *Neural Computation*, 10:217–3.
- Bengio, Y., Léonard, N., and Courville, A. (2013). Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv e-prints*, page arXiv:1308.3432.
- Berner, J., Grohs, P., and Jentzen, A. (2018). Analysis of the generalization error: Empirical risk minimization over deep artificial neural networks overcomes the curse of dimensionality in the numerical approximation of Black-Scholes partial differential equations. *CoRR*, abs/1809.03062.
- Bölcskei, H., Grohs, P., Kutyniok, G., and Petersen, P. (2019). Optimal approximation with sparsely connected deep neural networks. *SIAM Journal on Mathematics of Data Science*, 1(1):8–45.
- Boo, Y., Shin, S., and Sung, W. (2020). Quantized neural networks: Characterization and holistic optimization. *ArXiv*, abs/2006.00530.
- Braun, J. and Griebel, M. (2009). On a constructive proof of kolmogorov’s superposition theorem. *Constructive Approximation*, 30:653–675.
- Cao, Y. and Gu, Q. (2019). Generalization bounds of stochastic gradient descent for wide and deep neural networks. *CoRR*, abs/1905.13210.

- 778 Carrillo, J. A. T., Jin, S., Li, L., and Zhu, Y. (2019). A consensus-based global optimiza-
779 tion method for high dimensional machine learning problems. *arXiv:1909.09249*.
- 780 Chen, L. and Wu, C. (2019). A note on the expressive power of deep rectified linear
781 unit networks in high-dimensional spaces. *Mathematical Methods in the Applied*
782 *Sciences*, 42(9):3400–3404.
- 783 Chen, M., Jiang, H., Liao, W., and Zhao, T. (2019a). Efficient approximation of
784 deep ReLU networks for functions on low dimensional manifolds. In Wallach, H.,
785 Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors,
786 *Advances in Neural Information Processing Systems 32*, pages 8174–8184. Curran
787 Associates, Inc.
- 788 Chen, Z., Cao, Y., Zou, D., and Gu, Q. (2019b). How much over-parameterization is
789 sufficient to learn deep ReLU networks? *CoRR*, arXiv:1911.12360.
- 790 Chui, C. K., Lin, S.-B., and Zhou, D.-X. (2018). Construction of neural networks
791 for realization of localized deep learning. *Frontiers in Applied Mathematics and*
792 *Statistics*, 4:14.
- 793 Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *MCSS*,
794 2:303–314.
- 795 Devore, R. A. (1989). Optimal nonlinear approximation. *Manuskripta Math*, pages
796 469–478.
- 797 E, W., Ma, C., and Wu, L. (2019). A priori estimates of the population risk for two-layer
798 neural networks. *Communications in Mathematical Sciences*, 17(5):1407 – 1425.
- 799 E, W. and Wang, Q. (2018). Exponential convergence of the deep neural network ap-
800 proximation for analytic functions. *CoRR*, abs/1807.00297.
- 801 E, W. and Wojtowysch, S. (2020). Representation formulas and pointwise properties
802 for barron functions.
- 803 Gribonval, R., Kutyniok, G., Nielsen, M., and Voigtlaender, F. (2019). Approximation
804 spaces of deep neural networks. *arXiv e-prints*, page arXiv:1905.01208.
- 805 Gühring, I., Kutyniok, G., and Petersen, P. (2019). Error bounds for approxi-
806 mations with deep ReLU neural networks in $W^{s,p}$ norms. *arXiv e-prints*, page
807 arXiv:1902.07896.
- 808 Guliyev, N. J. and Ismailov, V. E. (2018). Approximation capability of two hidden layer
809 feedforward neural networks with fixed weights. *Neurocomputing*, 316:262 – 269.

- 810 Harvey, N., Liaw, C., and Mehrabian, A. (2017). Nearly-tight VC-dimension bounds
811 for piecewise linear neural networks. In Kale, S. and Shamir, O., editors, *Proceedings*
812 *of the 2017 Conference on Learning Theory*, volume 65 of *Proceedings of Machine*
813 *Learning Research*, pages 1064–1068, Amsterdam, Netherlands. PMLR.
- 814 Holland, J. H. (1992). Genetic algorithms. *Scientific American*, 267(1):66–73.
- 815 Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks
816 are universal approximators. *Neural Networks*, 2(5):359 – 366.
- 817 Hubara, I., Courbariaux, M., Soudry, D., El-Yaniv, R., and Bengio, Y. (2017). Quan-
818 tized neural networks: Training neural networks with low precision weights and ac-
819 tivations. *J. Mach. Learn. Res.*, 18(1):6869–6898.
- 820 Igel'nik, B. and Parikh, N. (2003). Kolmogorov's spline network. *IEEE Transactions*
821 *on Neural Networks*, 14(4):725–733.
- 822 Jacot, A., Gabriel, F., and Hongler, C. (2018). Neural tangent kernel: Convergence and
823 generalization in neural networks. *CoRR*, abs/1806.07572.
- 824 Ji, Z. and Telgarsky, M. (2020). Polylogarithmic width suffices for gradient de-
825 scent to achieve arbitrarily small test error with shallow ReLU networks. *ArXiv*,
826 abs/1909.12292.
- 827 Kennedy, J. and Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of*
828 *ICNN'95 - International Conference on Neural Networks*, volume 4, pages 1942–
829 1948 vol.4.
- 830 Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Optimization by simulated
831 annealing. *Science*, 220(4598):671–680.
- 832 Kolmogorov, A. N. (1956). On the representation of continuous functions of several
833 variables by superposition of continuous functions of a smaller number of variables.
834 *Dokl. Akad. Nauk SSSR*, pages 179–182.
- 835 Kolmogorov, A. N. (1957). On the representation of continuous functions of several
836 variables by superposition of continuous functions of one variable and addition. *Dokl.*
837 *Akad. Nauk SSSR*, pages 953–956.
- 838 Kůrková, V. (1992). Kolmogorov's theorem and multilayer neural networks. *Neural*
839 *Networks*, 5(3):501 – 506.
- 840 Li, Q., Lin, T., and Shen, Z. (2019). Deep learning via dynamical systems: An approx-
841 imation perspective. *arXiv e-prints*, page arXiv:1912.10382.

- 842 Liang, S. and Srikant, R. (2016). Why deep neural networks? *CoRR*, abs/1610.04161.
- 843 Lin, Y., Lei, M., and Niu, L. (2019). Optimization strategies in quantized neural net-
844 works: A review. In *2019 International Conference on Data Mining Workshops*
845 (*ICDMW*), pages 385–390.
- 846 Lu, J., Shen, Z., Yang, H., and Zhang, S. (2020). Deep network approximation for
847 smooth functions. *arXiv e-prints*, page arXiv:2001.03040.
- 848 Luo, T. and Yang, H. (2020). Two-layer neural networks for partial differential equa-
849 tions: Optimization and generalization theory. *ArXiv*, abs/2006.15733.
- 850 Maierov, V. and Pinkus, A. (1999). Lower bounds for approximation by MLP neural
851 networks. *Neurocomputing*, 25(1):81 – 91.
- 852 Montanelli, H. and Du, Q. (2019). New error bounds for deep ReLU networks using
853 sparse grids. *SIAM Journal on Mathematics of Data Science*, 1(1):78–92.
- 854 Montanelli, H. and Yang, H. (2020). Error bounds for deep ReLU networks using the
855 Kolmogorov-Arnold superposition theorem. *Neural Networks*, 129:1 – 6.
- 856 Montanelli, H., Yang, H., and Du, Q. (2020). Deep ReLU networks overcome the curse
857 of dimensionality for bandlimited functions. *Journal of Computational Mathematics*.
- 858 Nakada, R. and Imaizumi, M. (2019). Adaptive approximation and estimation of deep
859 neural network with intrinsic dimensionality. *arXiv:1907.02177*.
- 860 Nelder, J. and Mead, R. (1965). A simplex method for function minimization. *Comput.*
861 *J.*, 7:308–313.
- 862 Opschoor, J. A. A., Schwab, C., and Zech, J. (2019). Exponential ReLU
863 DNN expression of holomorphic maps in high dimension. Technical Re-
864 port 2019-35, Seminar for Applied Mathematics, ETH Zürich, Switzerland.
865 <https://math.ethz.ch/sam/research/reports.html?id=839>.
- 866 Petersen, P. and Voigtlaender, F. (2018). Optimal approximation of piecewise smooth
867 functions using deep ReLU neural networks. *Neural Networks*, 108:296 – 330.
- 868 Pinnau, R., Totzeck, C., Tse, O., and Martin, S. (2017). A consensus-based model for
869 global optimization and its mean-field limit. *Mathematical Models and Methods in*
870 *Applied Sciences*, 27(01):183–204.
- 871 Poggio, T., Mhaskar, H. N., Rosasco, L., Miranda, B., and Liao, Q. (2017). Why and
872 when can deep—but not shallow—networks avoid the curse of dimensionality: A
873 review. *International Journal of Automation and Computing*, 14:503–519.

- 874 Shen, Z., Yang, H., and Zhang, S. (2019a). Nonlinear approximation via compositions.
875 *Neural Networks*, 119:74 – 84.
- 876 Shen, Z., Yang, H., and Zhang, S. (2019b). Deep network approximation characterized
877 by number of neurons. *arXiv e-prints*, page arXiv:1906.05497.
- 878 Suzuki, T. (2019). Adaptivity of deep ReLU network for learning in Besov and mixed
879 smooth Besov spaces: optimal rate and curse of dimensionality. In *International*
880 *Conference on Learning Representations*.
- 881 Wang, P., Hu, Q., Zhang, Y., Zhang, C., Liu, Y., and Cheng, J. (2018). Two-step quan-
882 tization for low-bit neural networks. In *2018 IEEE/CVF Conference on Computer*
883 *Vision and Pattern Recognition*, pages 4376–4384.
- 884 Yang, Y. and Wang, Y. (2020). Approximation in shift-invariant spaces with deep ReLU
885 neural networks. *arXiv e-prints*, page arXiv:2005.11949.
- 886 Yarotsky, D. (2017). Error bounds for approximations with deep ReLU networks. *Neu-*
887 *ral Networks*, 94:103 – 114.
- 888 Yarotsky, D. (2018). Optimal approximation of continuous functions by very deep
889 ReLU networks. In Bubeck, S., Perchet, V., and Rigollet, P., editors, *Proceedings*
890 *of the 31st Conference On Learning Theory*, volume 75 of *Proceedings of Machine*
891 *Learning Research*, pages 639–649. PMLR.
- 892 Yarotsky, D. and Zhevnerchuk, A. (2019). The phase diagram of approximation rates
893 for deep neural networks. *arXiv e-prints*, page arXiv:1906.09477.
- 894 Yin, P., Lyu, J., Zhang, S., Osher, S., Qi, Y., and Xin, J. (2019). Understand-
895 ing straight-through estimator in training activation quantized neural nets. *ArXiv*,
896 abs/1903.05662.
- 897 Zhou, D.-X. (2020). Universality of deep convolutional neural networks. *Applied and*
898 *Computational Harmonic Analysis*, 48(2):787 – 794.