
Neural Network Architecture Beyond Width and Depth

Zuowei Shen

Department of Mathematics
National University of Singapore
matzuows@nus.edu.sg

Haizhao Yang

Department of Mathematics
University of Maryland, College Park
hzyang@umd.edu

Shijun Zhang*

Department of Mathematics
National University of Singapore
zhangshijun@u.nus.edu

Abstract

This paper proposes a new neural network architecture by introducing an additional dimension called height beyond width and depth. Neural network architectures with height, width, and depth as hyper-parameters are called three-dimensional architectures. It is shown that neural networks with three-dimensional architectures are significantly more expressive than the ones with two-dimensional architectures (those with only width and depth as hyper-parameters), e.g., standard fully connected networks. The new network architecture is constructed recursively via a nested structure, and hence we call a network with the new architecture nested network (NestNet). A NestNet of height s is built with each hidden neuron activated by a NestNet of height $\leq s-1$. When $s=1$, a NestNet degenerates to a standard network with a two-dimensional architecture. It is proved by construction that height- s ReLU NestNets with $\mathcal{O}(n)$ parameters can approximate 1-Lipschitz continuous functions on $[0, 1]^d$ with an error $\mathcal{O}(n^{-(s+1)/d})$, while the optimal approximation error of standard ReLU networks with $\mathcal{O}(n)$ parameters is $\mathcal{O}(n^{-2/d})$. Furthermore, such a result is extended to generic continuous functions on $[0, 1]^d$ with the approximation error characterized by the modulus of continuity. Finally, we use numerical experimentation to show the advantages of the super-approximation power of ReLU NestNets.

1 Introduction

In this paper, we design a new neural network architecture by introducing one more dimension, called height, in addition to width and depth in the characterization of dimensions of neural networks. We call neural network architectures with height, width, and depth as hyper-parameters three-dimensional architectures. It is proved by construction that neural networks with three-dimensional architectures improve the approximation power significantly, compared to standard networks with two-dimensional architectures (those with only width and depth as hyper-parameters). The approximation power of standard neural networks has been widely studied in recent years. The optimality of the approximation of standard fully-connected rectified linear unit (ReLU) networks (e.g., see [35, 40, 49, 52]) implies limited room for further improvements. This motivates us to design a new neural network architecture by introducing an additional dimension of height beyond width and depth.

* Corresponding author.

We will focus on the ReLU ($\max\{0, x\}$) activation function and use it to demonstrate our ideas. Our new network architecture is constructed recursively via a nested structure, and hence we call a neural network with the new architecture nested network (**NestNet**). A NestNet of height s is built with each hidden neuron activated by a NestNet of height $\leq s - 1$. In the case of $s = 1$, a NestNet degenerates to a standard network with a two-dimensional architecture. Let us use a simple example to explain the height of a NestNet. We say a network is activated by $\varrho_1, \dots, \varrho_r$ if each hidden neuron of this network is activated by one of $\varrho_1, \dots, \varrho_r$. Here, $\varrho_1, \dots, \varrho_r$ are trainable functions mapping \mathbb{R} to \mathbb{R} . Then, a network of height $s \geq 2$ can be regarded as a $(\varrho_1, \dots, \varrho_r)$ -activated network, where $\varrho_1, \dots, \varrho_r$ are (realized by) networks of height $\leq s - 1$. See an example of a height-2 network in Figure 1. The network therein can be regarded as a (ϱ_1, ϱ_2) -activated network, where ϱ_1 and ϱ_2 are (realized by) networks of height 1 (i.e., standard networks). The number of parameters in the network of Figure 1 is the sum of the numbers of parameters in $\mathcal{L}_0, \mathcal{L}_1, \mathcal{L}_2$ and ϱ_1, ϱ_2 .

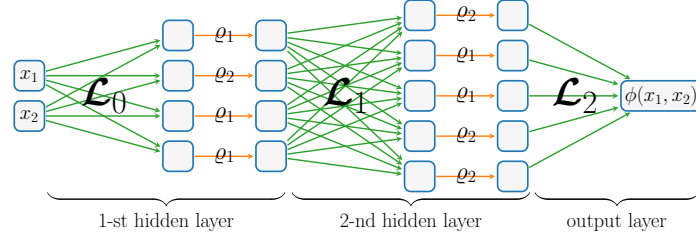


Figure 1: An example of a network of height 2, where ϱ_1 and ϱ_2 are (realized by) networks of height 1 (i.e., standard networks). Here, $\mathcal{L}_0, \mathcal{L}_1$ and \mathcal{L}_2 are affine linear maps.

We remark that a NestNet can be regarded as a sufficiently large standard network by expanding all of its sub-network activation functions. We propose the nested network architecture since it shares the parameters via repetitions of sub-network activation functions. In other words, a NestNet can provide a special parameter-sharing scheme. This is the key reason why the NestNet has much better approximation power than the standard network. If we regard the network in Figure 1 as a NestNet of height 2, then the number of parameters is the sum of the numbers of parameters in $\mathcal{L}_0, \mathcal{L}_1, \mathcal{L}_2$ and ϱ_1, ϱ_2 . However, if we expand the network in Figure 1 to a large standard network, then the number of parameters in ϱ_1 and ϱ_2 will be added many times for computing the total number of parameters.

Next, let us discuss our new network architecture from the perspective of hyper-parameters. We call the network architecture with only width as a hyper-parameter one-dimensional architecture. Its depth and height are both equal to one. Neural networks with this type of architecture are generally called shallow networks. See an example in Figure 2(a). We call the network architecture with only width and depth as hyper-parameters two-dimensional architecture. Its height is equal to one. Neural networks with this type of architecture are generally called deep networks. See an example in Figure 2(b). We call the network architecture with height, width, and depth as hyper-parameters three-dimensional architecture, which is proposed in this paper. Neural networks with this type of architecture are called NestNets. See an example in Figure 2(c). One may refer to Table 1 for the approximation power of networks with these three types of architectures discussed above.

Table 1: Comparison for the approximation error of 1-Lipschitz continuous functions on $[0, 1]^d$ approximated by ReLU NestNets and standard ReLU networks.

	dimension(s)	#parameters	approximation error	remark	reference
one-hidden-layer network	width varies (depth = height = 1)	$\mathcal{O}(n)$	n^{-1} for $d = 1$	linear combination	
deep network	width and depth vary (height = 1)	$\mathcal{O}(n)$	$n^{-2/d}$	composition	[35, 40, 49, 52]
NestNet of height s	width, depth, and height vary	$\mathcal{O}(n)$	$n^{-(s+1)/d}$	nested composition	this paper

Our main contributions are summarized as follows. We first propose a three-dimensional neural network architecture by introducing one more dimension called height beyond width and depth. We show that neural networks with three-dimensional architectures are significantly more expressive than standard networks. In particular, we prove that height- s ReLU NestNets with $\mathcal{O}(n)$ parameters can approximate 1-Lipschitz continuous functions on $[0, 1]^d$ with an error $\mathcal{O}(n^{-(s+1)/d})$, which is much better than the optimal error $\mathcal{O}(n^{-2/d})$ of standard ReLU networks with $\mathcal{O}(n)$ parameters. In

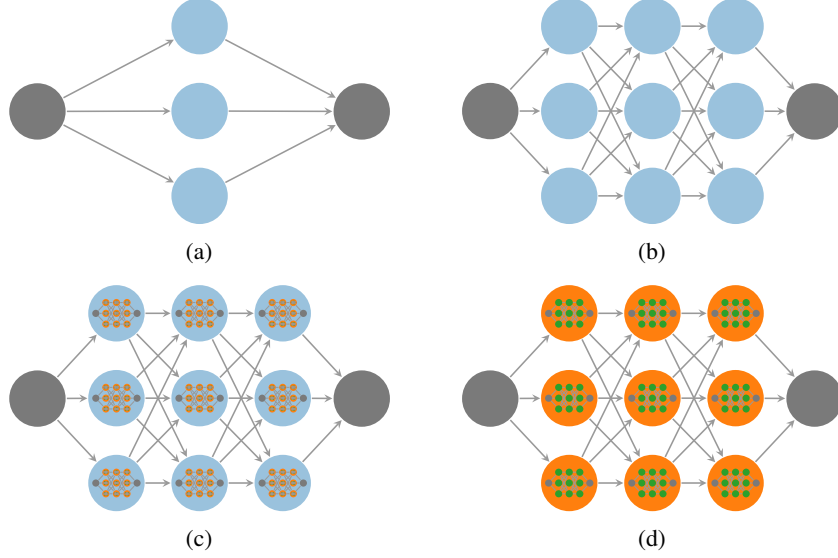


Figure 2: Illustrations of neural networks with one-, two-, and three-dimensional architectures. (a) One-dimensional case (width = 3, depth = height = 1). (b) Two-dimensional case (width = depth = 3, height = 1). (c) Three-dimensional case (width = depth = height = 3). (d) Zoom-in of an activation function of the network in (c). The network in (d) can also be regarded as a network of height 2.

the case of $s + 1 \geq d$, the approximation error is bounded by $\mathcal{O}(n^{-(s+1)/d}) \leq \mathcal{O}(n^{-1})$, which means we overcome the curse of dimensionality. Furthermore, we extend our result to generic continuous functions with the approximation error characterized by the modulus of continuity. See Theorem 2.1 and Corollary 2.2 for more details. Finally, we conduct simple experiments to show the numerical advantages of the super-approximation power of ReLU NestNets.

The rest of this paper is organized as follows. In Section 2, we present the main results, provide the ideas of proving them, and discuss related work. The detailed proofs of the main results are placed in the appendix. Next, we conduct experiments to show the advantages of the super-approximation power of ReLU NestNets in Section 3. Finally, Section 4 concludes this paper with a short discussion.

2 Main results and related work

In this section, we first present our main results and discuss the proof ideas. The detailed proofs of the main results are placed in the appendix. Next, we discuss related work from multiple perspectives.

2.1 Main results

We use $\mathcal{NN}_s\{n\}$ for $n, s \in \mathbb{N}$ to denote the set of functions realized by height- s ReLU NestNets with as most n parameters. We will give the mathematical definition of $\mathcal{NN}_s\{n\}$. We first discuss some notations regarding affine linear maps. We use \mathcal{L} to denote the set of all affine linear maps, i.e.,

$$\mathcal{L} := \left\{ \mathcal{L} : \mathcal{L}(\mathbf{x}) = \mathbf{W}\mathbf{x} + \mathbf{b}, \mathbf{W} \in \mathbb{R}^{d_2 \times d_1}, \mathbf{b} \in \mathbb{R}^{d_2}, d_1, d_2 \in \mathbb{N}^+ \right\}.$$

Let $\#\mathcal{L}$ denote the number of parameters in $\mathcal{L} \in \mathcal{L}$, i.e.,

$$\#\mathcal{L} = (d_1 + 1)d_2 \quad \text{if } \mathcal{L}(\mathbf{x}) = \mathbf{W}\mathbf{x} + \mathbf{b} \quad \text{for } \mathbf{W} \in \mathbb{R}^{d_2 \times d_1} \text{ and } \mathbf{b} \in \mathbb{R}^{d_2}.$$

We use $\vec{g} = (\varrho_1, \dots, \varrho_k)$ to denote an activation function vector, where $\varrho_i : \mathbb{R} \rightarrow \mathbb{R}$ is an activation function for each $i \in \{1, \dots, k\}$. When $\vec{g} = (\varrho_1, \dots, \varrho_k)$ is applied to a vector input $\mathbf{x} = (x_1, \dots, x_k)$,

$$\vec{g}(\mathbf{x}) = \left(\varrho_1(x_1), \dots, \varrho_k(x_k) \right) \quad \text{for any } \mathbf{x} = (x_1, \dots, x_k) \in \mathbb{R}^k.$$

Let $\text{set}(\vec{g})$ denote the function set containing all entries (functions) in \vec{g} . For example, if $\vec{g} = (\varrho_1, \varrho_2, \varrho_3, \varrho_2, \varrho_1)$, then $\text{set}(\vec{g}) = \{\varrho_1, \varrho_2, \varrho_3\}$.

90 To define $\mathcal{NN}_s\{n\}$ for $n, s \in \mathbb{N}$ recursively, we first consider the degenerate case. Define

$$91 \quad \mathcal{NN}_0\{n\} := \{\text{id}_{\mathbb{R}}, \text{ReLU}\} =: \mathcal{NN}_s\{0\} \quad \text{for } n, s \in \mathbb{N},$$

92 where $\text{id}_{\mathbb{R}} : \mathbb{R} \rightarrow \mathbb{R}$ is the identity map. That is, we regard the identity map and ReLU as height-0
93 ReLU NestNets with n parameters or as height- s ReLU NestNets with 0 parameters.

94 Next, let us present the recursive step. For $n, s \in \mathbb{N}^+$, a (vector-valued) function $\phi \in \mathcal{NN}_s\{n\}$ has the
95 following form:

$$96 \quad \phi = \mathcal{L}_m \circ \vec{g}_m \circ \cdots \circ \mathcal{L}_1 \circ \vec{g}_1 \circ \mathcal{L}_0, \quad (1)$$

97 where $\mathcal{L}_0, \dots, \mathcal{L}_m \in \mathcal{L}$ are affine linear maps. Moreover, Equation (1) satisfies the following two
98 conditions:

- 99 • Condition on activation functions:

$$100 \quad \bigcup_{i=1}^m \text{set}(\vec{g}_i) = \{\varrho_1, \dots, \varrho_r\} \quad \text{and} \quad \varrho_j \in \bigcup_{i=0}^{s-1} \mathcal{NN}_i\{n_j\} \quad \text{for } j = 1, \dots, r. \quad (2)$$

101 Here, \vec{g}_i is an activation function vector for each $i \in \{1, \dots, m\}$. All entries in $\vec{g}_1, \dots, \vec{g}_m$
102 form an activation function set $\{\varrho_1, \dots, \varrho_r\}$. For each $j \in \{1, \dots, r\}$, ϱ_j can be realized by a
103 height- i NestNet with $\leq n_j$ parameters for some $i = i_j \leq s-1$. This condition means each
104 hidden neuron is activated by a NestNet of lower height.

- 105 • Condition on the number of parameters:

$$106 \quad \sum_{i=0}^m \#\mathcal{L}_i + \sum_{j=1}^r n_j \leq n. \quad (3)$$

107 This condition means the total number of parameters is no more than n . The total number of
108 parameters is calculated by adding two parts. The first one is the number of parameters in
109 affine linear maps $\mathcal{L}_0, \dots, \mathcal{L}_m$. The other part is the number of parameters in the activation
110 set $\{\varrho_1, \dots, \varrho_r\}$ formed by the entries in activation function vectors $\vec{g}_1, \dots, \vec{g}_m$.

111 Then, with two conditions in Equations (2) and (3), we can define $\mathcal{NN}_s\{n\}$ for $n, s \in \mathbb{N}^+$ as follows:

$$112 \quad \mathcal{NN}_s\{n\} := \left\{ \phi : \phi = \mathcal{L}_m \circ \vec{g}_m \circ \cdots \circ \mathcal{L}_1 \circ \vec{g}_1 \circ \mathcal{L}_0, \quad \mathcal{L}_0, \dots, \mathcal{L}_m \in \mathcal{L}, \quad \bigcup_{i=1}^m \text{set}(\vec{g}_i) = \{\varrho_1, \dots, \varrho_r\}, \right. \\ \left. \varrho_j \in \bigcup_{i=0}^{s-1} \mathcal{NN}_i\{n_j\} \quad \text{for } j = 1, \dots, r, \quad \sum_{i=0}^m \#\mathcal{L}_i + \sum_{j=1}^r n_j \leq n \right\}.$$

113 We remark that, in the definition above, m can be equal to 0. In this case, the function ϕ degenerates
114 to an affine linear map.

115 In the NestNet example in Figure 1, the function ϕ therein is in $\bigcup_{n \in \mathbb{N}} \mathcal{NN}_2\{n\}$ and the activation
116 function vectors \vec{g}_1 and \vec{g}_2 can be represented as

$$117 \quad \vec{g}_1 = (\varrho_1, \varrho_2, \varrho_1, \varrho_1) \quad \text{and} \quad \vec{g}_2 = (\varrho_2, \varrho_1, \varrho_1, \varrho_2, \varrho_2).$$

118 Moreover, the activation function set containing all entries in \vec{g}_1 and \vec{g}_2 is a subset of $\bigcup_{n \in \mathbb{N}} \mathcal{NN}_1\{n\}$,
119 i.e., $\{\varrho_1, \varrho_2\} \subseteq \bigcup_{n \in \mathbb{N}} \mathcal{NN}_1\{n\}$.

120 Let $C([0, 1]^d)$ denote the set of continuous functions on $[0, 1]^d$. By convention, the modulus of
121 continuity of a continuous function $f \in C([0, 1]^d)$ is defined as

$$122 \quad \omega_f(r) := \sup \{ |f(\mathbf{x}) - f(\mathbf{y})| : \|\mathbf{x} - \mathbf{y}\|_2 \leq r, \mathbf{x}, \mathbf{y} \in [0, 1]^d \} \quad \text{for any } r \geq 0.$$

123 Under these settings, we can find a function in $\mathcal{NN}_s\{\mathcal{O}(n)\}$ to approximate $f \in C([0, 1]^d)$ with an
124 approximation error $\mathcal{O}(\omega_f(n^{-(s+1)/d}))$, as shown in the main theorem below.

125 **Theorem 2.1.** *Given a continuous function $f \in C([0, 1]^d)$, for any $n, s \in \mathbb{N}^+$ and $p \in [1, \infty]$, there
126 exists $\phi \in \mathcal{NN}_s\{C_{s,d}(n+1)\}$ such that*

$$127 \quad \|\phi(\mathbf{x}) - f(\mathbf{x})\|_{L^p([0,1]^d)} \leq 7\sqrt{d}\omega_f(n^{-(s+1)/d}),$$

128 where $C_{s,d} = 10^3 d^2 (s+7)^2$ if $p \in [1, \infty)$ and $C_{s,d} = 10^{d+3} d^2 (s+7)^2$ if $p = \infty$.

We remark that the constant $C_{s,d}$ in Theorem 2.1 is valid for all $n \in \mathbb{N}^+$. As we shall see later, $C_{s,d}$ can be greatly reduced if one only cares about large $n \in \mathbb{N}^+$. Generally, it is challenging to simplify the approximation error in Theorem 2.1 to make it explicitly depend on n due to the complexity of $\omega_f(\cdot)$. However, the approximation error can be simplified to an explicit one depending on n in the case of special target function spaces like Hölder continuous function space. To be exact, if f is a Hölder continuous function on $[0, 1]^d$ of order $\alpha \in (0, 1]$ with a Hölder constant $\lambda > 0$, then

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

implying $\omega_f(r) \leq \lambda r^\alpha$ for any $r \geq 0$. This means we can get an exponentially small approximation error $7\lambda\sqrt{d}n^{-(s+1)\alpha/d}$ as shown in Corollary 2.2 below.

Corollary 2.2. *Suppose f is a Hölder continuous function on $[0, 1]^d$ of order $\alpha \in (0, 1]$ with a Hölder constant $\lambda > 0$. For any $n, s \in \mathbb{N}^+$ and $p \in [1, \infty]$, there exists $\phi \in \mathcal{NN}_s\{C_{s,d}(n+1)\}$ such that*

$$\|\phi(\mathbf{x}) - f(\mathbf{x})\|_{L^p([0,1]^d)} \leq 7\lambda\sqrt{d}n^{-(s+1)\alpha/d},$$

where $C_{s,d} = 10^3 d^2 (s+7)^2$ if $p \in [1, \infty)$ and $C_{s,d} = 10^{d+3} d^2 (s+7)^2$ if $p = \infty$.

In Corollary 2.2, if $\alpha = 1$, i.e., f is a Lipschitz continuous function with a Lipschitz constant $\lambda > 0$, then the approximation error can be further simplified to $7\lambda\sqrt{d}n^{-(s+1)/d}$. See Table 1 for the comparison of the approximation error of 1-Lipschitz continuous functions on $[0, 1]^d$ approximated by ReLU NestNets and standard ReLU networks.

2.2 Sketch of proving Theorem 2.1

We will discuss how to prove Theorem 2.1. Given a target function $f \in C([0, 1]^d)$, the key point is to construct an almost piecewise constant function realized by a ReLU NestNet to approximate f well except for a small region. Then we can get the desired result by dealing with the approximation in this small region. We divide the sketch of proving Theorem 2.1 into three main steps.

1. First, we divide $[0, 1]^d$ into a union of cubes $\{Q_\beta\}_{\beta \in \{0,1,\dots,K-1\}^d}$ and a small region Ω with $K = \mathcal{O}(n^{(s+1)/d})$. Each Q_β is associated with a representative $\mathbf{x}_\beta \in Q_\beta$ for each vector index β . See Figure 3 for an illustration for $K = 4$ and $d = 2$.

2. Next, we design a vector-valued function $\Phi_1(\mathbf{x})$ to map the whole cube Q_β to its index β for each β . Here, Φ_1 can be defined/constructed via

$$\Phi_1(\mathbf{x}) = [\phi_1(x_1), \phi_1(x_2), \dots, \phi_1(x_d)]^T,$$

where each one-dimensional function ϕ_1 is a step function outside a small region. We can efficiently construct ReLU NestNets with the desired size to approximate such an almost step function ϕ_1 with sufficiently many “steps” by using the composition architecture of ReLU NestNets. See the appendix for the detailed construction.

3. Finally, we need to construct a function ϕ_2 realized by a ReLU NestNet to map β approximately to $f(\mathbf{x}_\beta)$ for each $\beta \in \{0, 1, \dots, K-1\}^d$. Then we have

$$\phi_2 \circ \Phi_1(\mathbf{x}) = \phi_2(\beta) \approx f(\mathbf{x}_\beta) \approx f(\mathbf{x}) \quad \text{for any } \mathbf{x} \in Q_\beta \text{ and each } \beta,$$

implying

$$\phi := \phi_2 \circ \Phi_1 \approx f \quad \text{on } [0, 1]^d \setminus \Omega.$$

Then, we can get a good approximation on $[0, 1]^d$ by using Lemma 3.4 of our previous paper [24] to deal with the approximation inside Ω . We remark that, in the construction of $\phi_2 : \mathbb{R}^d \rightarrow \mathbb{R}$, we only need to care about the values of ϕ_2 at a set of K^d points $\{0, 1, \dots, K-1\}^d$. As we shall see later, this is the key point to ease the design of a ReLU NestNet with the desired size to realize ϕ_2 .

See Figure 3 for an illustration of the above steps. Observe that in Figure 3, we have

$$\phi(\mathbf{x}) = \phi_2 \circ \Phi_1(\mathbf{x}) = \phi_2(\beta) \stackrel{\mathcal{E}_1}{\approx} f(\mathbf{x}_\beta) \stackrel{\mathcal{E}_2}{\approx} f(\mathbf{x})$$

for any $\mathbf{x} \in Q_\beta$ and each $\beta \in \{0, 1, \dots, K-1\}^d$. That means $\phi - f$ is bounded by $\mathcal{E}_1 + \mathcal{E}_2$ on $[0, 1]^d \setminus \Omega$. For any $\mathbf{x} \in Q_\beta$ and each β , we have

$$\|\mathbf{x}_\beta - \mathbf{x}\|_2 \leq \sqrt{d}/K \implies |f(\mathbf{x}_\beta) - f(\mathbf{x})| \leq \omega_f(\sqrt{d}/K) \implies \mathcal{E}_2 \leq \omega_f(\sqrt{d}/K).$$

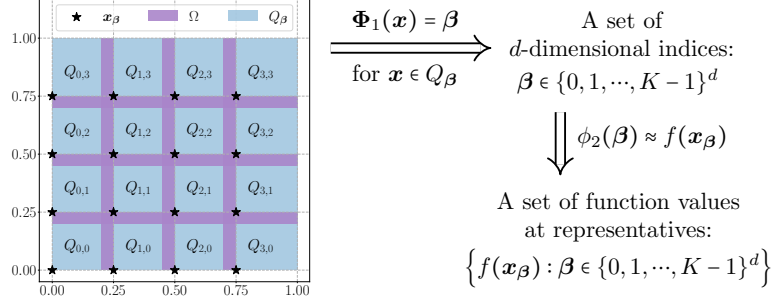


Figure 3: An illustration of the ideas of constructing $\phi = \phi_2 \circ \Phi_1$ to approximate f for $K = 4$ and $d = 2$. Note that $\phi \approx f$ outside Ω since $\phi(x) = \phi_2 \circ \Phi_1(x) = \phi_2(\beta) \approx f(x_{\beta}) \approx f(x)$ for any $x \in Q_{\beta}$ and each $\beta \in \{0, 1, \dots, K-1\}^d$.

The upper bound of \mathcal{E}_1 is determined by the construction of $\phi_2 : \mathbb{R}^d \rightarrow \mathbb{R}$. As stated previously, we only need to care about the values of ϕ_2 at a set of K^d points $\{0, 1, \dots, K-1\}^d \subseteq \mathbb{R}^d$, which gives us much freedom to control \mathcal{E}_1 . As we shall see later, \mathcal{E}_1 can be bounded by $\mathcal{O}(\omega_f(\sqrt{d}/K))$. Therefore, $\phi - f$ is controlled by $\mathcal{O}(\omega_f(\sqrt{d}/K))$ outside Ω , from which we deduce the desired approximation error on $[0, 1]^d \setminus \Omega$ since $K = \mathcal{O}(n^{-(s+1)/d})$. Finally, by using Lemma 3.4 of our previous paper [24] to deal with the approximation inside Ω , we can get the desired approximation error on $[0, 1]^d$.

2.3 Related work

We first compare our results with existing ones from an approximation perspective. Next, we discuss the parameter-sharing schemes of neural networks. Finally, we connect our NestNet architecture to existing trainable activation functions.

Discussion from an approximation perspective

The study of the approximation power of deep neural networks has become an active topic in recent years. This topic has been extensively studied from many perspectives, e.g., in terms of combinatorics [27], topology [7], information theory [29], fat-shattering dimension [1, 21], Vapnik-Chervonenkis (VC) dimension [6, 14, 32], classical approximation theory [3, 4, 8, 9, 10, 11, 12, 13, 18, 22, 24, 25, 28, 34, 35, 38, 39, 42, 48, 49, 52, 53], etc. To the best of our knowledge, the study of neural network approximation has two main stages: shallow (one-hidden-layer) networks and deep networks.

In the early works of neural network approximation, the approximation power of shallow networks is investigated. In particular, the universal approximation theorem [11, 17, 18], without approximation error estimate, showed that a sufficiently large neural network can approximate a target function in a certain function space arbitrarily well. For one-hidden-layer neural networks of width n and sufficiently smooth functions, an asymptotic approximation error $\mathcal{O}(n^{-1/2})$ in the L^2 -norm is proved in [4, 5], leveraging an idea that is similar to Monte Carlo sampling for high-dimensional integrals.

Recently, a large number of works focus on the study of deep neural networks. It is shown in [35, 49, 52] that the optimal approximation error is $\mathcal{O}(n^{-2/d})$ by using ReLU networks with n parameters to approximate 1-Lipschitz continuous functions on $[0, 1]^d$. This optimal approximation error follows a natural question: How can we get a better approximation error? Generally, there are two ideas to get better errors. The first one is to consider smaller function spaces, e.g., smooth functions [24, 50] and band-limited functions [26]. The other one is to introduce new networks, e.g., Floor-ReLU networks [36], Floor-Exponential-Step (FLES) networks [37], and (Sin, ReLU, 2^x)-activated networks [20].

This paper proposes a three-dimensional neural network architecture by introducing one more dimension called height beyond width and depth. As shown in Theorem 2.1 and Corollary 2.2, neural networks with three-dimensional architectures are significantly more expressive than the ones with two-dimensional architectures. We will conduct experiments to explore the numerical properties of NestNets in Section 3.

212 Discussion from a parameter-sharing perspective

213 As discussed previously, our NestNet architecture can be regarded as a sufficiently large standard
214 network architecture with a specific parameter-sharing scheme. Parameter-sharing schemes are
215 used in neural networks to control the overall number of parameters for reducing memory and
216 communication costs. There are two common parameter-sharing schemes for a neural network. The
217 first scheme is to share parameters in the same layer. A typical network example with this scheme is
218 the convolutional neural network (CNN). In CNN architectures, filters in a CNN layer are shared for
219 all channels, which means the parameters in the filters are shared. The second scheme is to share
220 parameters across different layers of networks, e.g., recurrent neural networks.

221 In the NestNet architecture, we share parameters via repetitions of sub-network activation functions.
222 Both of parameter-sharing schemes discussed just above are used in the NestNet architecture. The
223 nested architecture of NestNets gives us much freedom to determine how many parameters to share.
224 Beyond parameter-sharing schemes for a neural network, there are also parameter-sharing schemes
225 among different neural networks or models, especially for multi-task Learning. One may refer to
226 [30, 33, 44, 45, 46, 51] for more discussion on parameter sharing in neural networks.

227 Connection to trainable activation functions

228 The key idea of trainable activation functions is to add a small number of trainable parameters to
229 existing activation functions. Let us present several existing trainable activation functions as follows.
230 A ReLU-like function is introduced in [15] by modifying the negative part of ReLU using a trainable
231 parameter α , i.e., the parametric ReLU (PReLU) is defined as $\text{PReLU}(x) := \begin{cases} x & \text{if } x \geq 0 \\ \alpha x & \text{if } x < 0. \end{cases}$ A variant
232 of ELU unit is introduced in [43] by adding two trainable parameters $\beta, \gamma > 0$, i.e., the parametric
233 ELU (PELU) is given by $\text{PELU}(x) := \begin{cases} \beta/\gamma & \text{if } x \geq 0 \\ \beta(\exp(x/\gamma) - 1)x & \text{if } x < 0. \end{cases}$ Authors in [31] propose a type
234 of Flexible ReLU (FReLU), which is defined via $\text{FReLU}(x) := \text{ReLU}(x + \alpha) + \beta$, where α and
235 β are two trainable parameters. One may refer to [2] for a survey of modern trainable activation
236 functions. To the best of our knowledge, most existing trainable activation functions can be regarded
237 as parametric variants of the original activation functions. That is, they are attained via parameterizing
238 the original activation functions with a small number of (typically 1 or 2) trainable parameters.

239 By contrast, activation functions in our NestNets are much more flexible. They can be (realized
240 by) either complicated or simple sub-NestNets. That is, we can freely determine the number of
241 parameters in the activation functions of NestNets. In other words, in NestNets, we can randomly
242 distribute the parameters in the affine linear maps and activation functions. In short, compared to the
243 networks with existing trainable activation functions, our NestNets are more flexible and have much
244 more freedom in the choice of activation functions.

245 3 Experimentation

246 In this section, we will conduct experiments as a proof of concept to explore the numerical properties
247 of ReLU NestNets. It is challenging to tune the hyper-parameters of large NestNets due to their
248 nested architectures. Thus, our experimentation focuses on relatively small NestNets of height
249 2 and we introduce a simple sub-network activation function ϱ , which is realized by a trainable
250 one-hidden-layer ReLU network of width 3. To be exact, ϱ is given by

$$251 \quad \varrho(x) = \mathbf{w}_1^T \cdot (x\mathbf{w}_0 + \mathbf{b}_0) + b_1 \quad \text{for any } x \in \mathbb{R}, \quad (4)$$

252 where $\mathbf{w}_0, \mathbf{w}_1, \mathbf{b}_0 \in \mathbb{R}^3$ and $b_1 \in \mathbb{R}$ are trainable parameters. There are 10 parameters in ϱ . The initial
253 settings for ϱ in our experiments are $\mathbf{w}_0 = (1, 1, 1)$, $\mathbf{w}_1 = (1, 1, -1)$, $\mathbf{b}_0 = (-0.2, -0.1, 0.0)$, and
254 $b_1 = 0$. We believe that NestNets can achieve good results in some real-world applications if proper
255 optimization algorithms are developed for NestNets. In this paper, we only consider two classification
256 problems: a synthetic classification problem based on the Archimedean spiral in Section 3.1 and an
257 image classification problem corresponding to a standard benchmark dataset Fashion-MNIST [47]
258 in Section 3.2. We remark that a classification function can be continuously extended to \mathbb{R}^d if each
259 class of samples are located in a bounded closed subset of \mathbb{R}^d and these subsets are pairwise disjoint.
260 That means we can apply our theory to classification problems.

261 3.1 Archimedean spiral

262 We will design a binary classification experiment by constructing two disjoint sets based on the
 263 Archimedean spiral, which can be described by the equation $r = a + b\theta$ in polar coordinates (r, θ) for
 264 given $a, b \in \mathbb{R}$. Let us first define two curves (Archimedean spirals) as follows:

$$265 \quad \tilde{\mathcal{C}}_i := \left\{ (x, y) : x = r_i \cos \theta, y = r_i \sin \theta, r_i = a_i + b_i \theta, \theta \in [0, s\pi] \right\},$$

266 for $i = 0, 1$, where $a_0 = 0, a_1 = 1, b_0 = b_1 = 1/\pi$, and $s = 30$. To simplify the discussion below, we
 267 normalize $\tilde{\mathcal{C}}_i$ as $\mathcal{C}_i \subseteq [0, 1]^2$, where \mathcal{C}_i is defined by

$$268 \quad \mathcal{C}_i := \left\{ (x, y) : x = \frac{\tilde{x}}{2(s+2)} + \frac{1}{2}, y = \frac{\tilde{y}}{2(s+2)} + \frac{1}{2}, (\tilde{x}, \tilde{y}) \in \tilde{\mathcal{C}}_i \right\},$$

269 for $i = 0, 1$. Then, we can define the two desired sets as follows:

$$270 \quad \mathcal{S}_i := \left\{ (u, v) : \sqrt{(u-x)^2 + (v-y)^2} \leq \varepsilon, (x, y) \in \mathcal{C}_i \right\},$$

271 for $i = 0, 1$, where $\varepsilon = 0.005$ in our experiments. See an illustration for \mathcal{S}_0 and \mathcal{S}_1 in Figure 4.

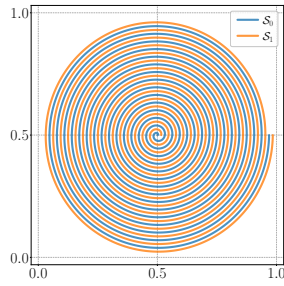


Figure 4: An illustration for \mathcal{S}_0 and \mathcal{S}_1 .

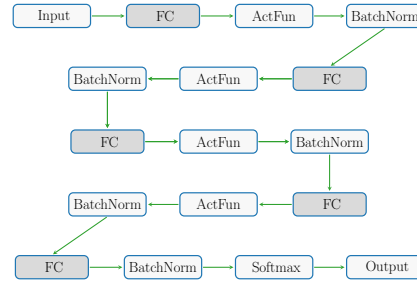


Figure 5: A network architecture illustration.

272 To explore the numerical performance of NestNets, we design NestNets and standard networks to
 273 classify samples in $\mathcal{S}_0 \cup \mathcal{S}_1$. We adopt four-hidden-layer fully connected network architecture of
 274 width 20, 35, or 50. To make the optimization more stable, we add the layers of batch normalization
 275 [19]. See Figure 5 for an illustration of the full network architecture. In Figure 5, FC and ActFun
 276 are short of fully connected layer and activation function, respectively. ActFun is ReLU for standard
 277 networks, while for NestNets, ActFun is the learnable sub-network activation function ϱ given in
 278 Equation (4).

279 Before presenting the experiment results, let us present the hyper-parameters for training the networks
 280 mentioned above. For each $i \in \{0, 1\}$, we randomly choose 3×10^5 training samples and 3×10^4 test
 281 samples in \mathcal{S}_i with label i . Then, we use these 6×10^5 training samples to train the networks and use
 282 these 6×10^4 test samples to compute the test accuracy. We use the cross-entropy loss function to
 283 evaluate the loss between the networks and the target classification function. The number of epochs
 284 and the batch size are set to 500 and 512, respectively. We adopt RAdam [23] as the optimization
 285 method. In epochs $5(i-1) + 1$ to $5i$ for $i = 1, 2, \dots, 100$, the learning rate is $0.2 \times 0.002 \times 0.9^{i-1}$ for
 286 the parameters in ϱ and $0.002 \times 0.9^{i-1}$ for all other parameters. We remark that all training (test)
 287 samples are standardized before training, i.e., we rescale the samples to have a mean of 0 and a
 288 standard deviation of 1.

289 Finally, let us present the experiment results to compare the numerical performances of NestNets
 290 and standard networks. We adopt the average of test accuracies in the last 100 epochs as the target
 291 test accuracy. As we can see from Table 2 and Figure 6, by adding 10 more parameters (stored in ϱ),
 292 NestNets achieve much better test accuracies than standard networks though slightly more training
 293 time is required. In an “unfair” comparison, the test accuracy attained by the NestNet with 1.4×10^3
 294 parameters is still better than that of the standard network with 7.9×10^3 parameters. This numerically
 295 verifies that the NestNet has much better approximation power than the standard network.

296 3.2 Fashion-MNIST

297 We will design convolutional neural network (CNN) architectures activated by ReLU or the sub-
 298 network activation function ϱ given in Equation (4) to classify image samples in Fashion-MNIST [47].

Table 2: Test accuracy comparison.

	width	depth	#parameters	activation function	training time	test accuracy
standard network	20	4	1362	ReLU	≈ 2532 s	0.738290
NestNet	20	4	1362 + 10	sub-network (ϱ)	≈ 4016 s	0.873631
standard network	35	4	3957	ReLU	≈ 2595 s	0.816048
NestNet	35	4	3957 + 10	sub-network (ϱ)	≈ 4104 s	0.995962
standard network	50	4	7902	ReLU	≈ 2642 s	0.866118
NestNet	50	4	7902 + 10	sub-network (ϱ)	≈ 4218 s	0.999984

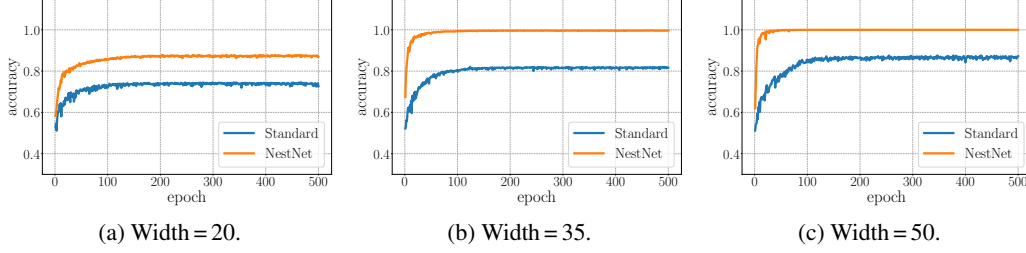


Figure 6: Test accuracy over epochs.

299 This dataset consists of a training set of 6×10^4 samples and a test set of 10^4 samples. Each sample
300 is a 28×28 grayscale image, associated with a label from 10 classes. To compare the numerical
301 performances of NestNets and standard networks, we design a standard CNN architecture and a
302 NestNet architecture that is constructed by replacing a few activation functions of a standard CNN
303 network by the sub-network activation function ϱ . For simplicity, we denote the standard CNN and
304 the NestNet as CNN1 and CNN2. To make the optimization more stable, we add the layers of dropout
305 [16, 41] and batch normalization [19]. See illustrations of CNN1 and CNN2 in Figure 7. We present
306 more details of them in Table 3.

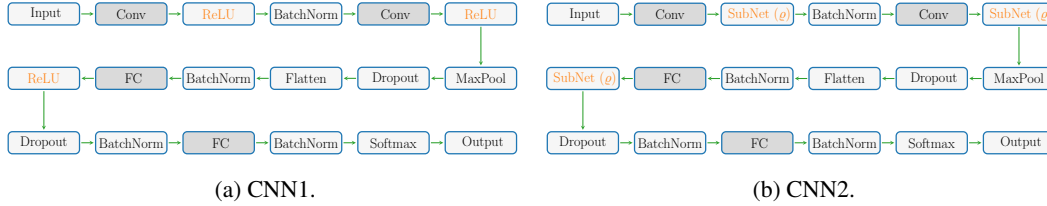


Figure 7: Illustrations of CNN1 and CNN2. Conv and FC represent convolutional and fully connected layers, respectively. CNN2 is indeed a NestNet of height 2.

Table 3: Details of CNN1 and CNN2.

layers	activation function		output size of each layer	dropout	batch normalization
	CNN1	CNN2			
input $\in \mathbb{R}^{28 \times 28}$			28×28		
Conv-1: $1 \times (3 \times 3)$, 12	ReLU	SubNet(ϱ), ReLU ₁	$1 \times (26 \times 26)$ $11 \times (26 \times 26)$		yes
Conv-2: $12 \times (3 \times 3)$, 12	ReLU	SubNet(ϱ), ReLU ₁	$1 \times (24 \times 24)$ $11 \times (24 \times 24)$	1728 (MaxPool & Flatten)	0.25
FC-1: 1728, 48	ReLU	SubNet(ϱ), ReLU ₁	1 47	48	0.5
FC-2: 48, 10				10 (Softmax)	yes
output $\in \mathbb{R}^{10}$					

307 Before presenting the numerical results, let us present the hyper-parameters for training two CNN
308 architectures above. We use the cross-entropy loss function to evaluate the loss between the CNNs
309 and the target classification function. The number of epochs and the batch size are set to 500 and 128,
310 respectively. We adopt RADam [23] as the optimization method and the weight decay of the optimizer
311 is 0.0001. In epochs $5(i-1) + 1$ to $5i$ for $i = 1, 2, \dots, 100$, the learning rate is $0.2 \times 0.002 \times 0.9^{i-1}$

for the parameters in ϱ and $0.002 \times 0.9^{i-1}$ for all other parameters. All training (test) samples in the Fashion-MNIST dataset are standardized in our experiment, i.e., we rescale all training (test) samples to have a mean of 0 and a standard deviation of 1. In the settings above, we repeat the experiment 18 times and discard 3 top-performing and 3 bottom-performing trials by using the average of test accuracy in the last 100 epochs as the performance criterion. For each epoch, we adopt the average of test accuracies in the rest 12 trials as the target test accuracy.

Next, let us present the experiment results to compare the numerical performances of CNN1 and CNN2. The test accuracy comparison of CNN1 and CNN2 is summarized in Table 4.

Table 4: Test accuracy comparison.

	training time	largest accuracy	average of largest 100 accuracies	average accuracy in last 100 epochs
CNN1	≈ 5802 s	0.925290	0.924796	0.924447
CNN2	≈ 7217 s	0.926620	0.926287	0.926032

For each of CNN1 and CNN2, we present the training time, the largest test accuracy, the average of the largest 100 test accuracies, and the average of test accuracies in the last 100 epochs. For an intuitive comparison, we also provide illustrations of the test accuracy over epochs for CNN1 and CNN2 in Figure 8. As we can see from Table 4 and Figure 8, CNN2 performs better than CNN1 though slightly more training time and 10 more parameters are required. This numerically shows that the NestNet is significantly more expressive than the standard network.

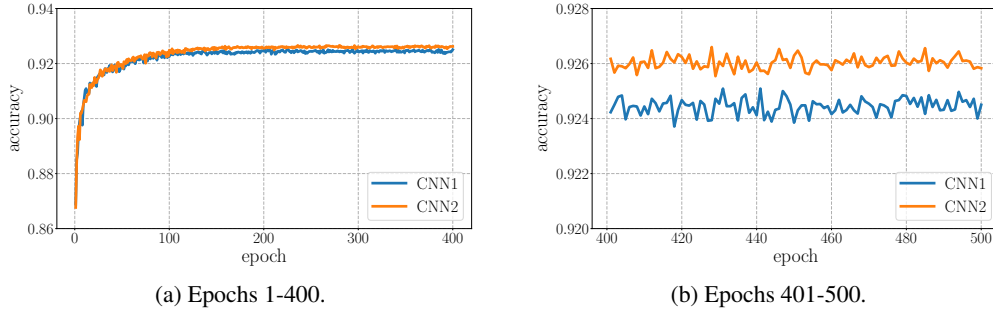


Figure 8: Test accuracy over epochs.

4 Conclusion

This paper proposes a three-dimensional neural network architecture by introducing one more dimension called height beyond width and depth. We show by construction that neural networks with three-dimensional architectures are significantly more expressive than the ones with two-dimensional architectures. We use simple numerical examples to show the advantages of the super-approximation power of ReLU NestNets, which is regarded as a proof of possibility. It would be of great interest to further explore the numerical performance of NestNets to bridge our theoretical results to applications. We believe that NestNets can be further developed and applied to real-world applications.

We remark that our analysis is limited to the ReLU activation function and the (Hölder) continuous function space. It would be interesting to generalize our results to other activation functions (e.g., tanh and sigmoid functions) and other function spaces (e.g., Lebesgue and Sobolev spaces).

Acknowledgments

Z. Shen was supported by Distinguished Professorship of National University of Singapore. H. Yang was partially supported by the US National Science Foundation under award DMS-2244988, DMS-2206333, and the Office of Naval Research Young Investigator Award.

References

- [1] Martin Anthony and Peter L. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, New York, NY, USA, 1st edition, 2009.
- [2] Andrea Apicella, Francesco Donnarumma, Francesco Isgrò, and Roberto Prevete. A survey on modern trainable activation functions. *Neural Networks*, 138:14–32, 2021.
- [3] Chenglong Bao, Qianxiao Li, Zuwei Shen, Cheng Tai, Lei Wu, and Xueshuang Xiang. Approximation analysis of convolutional neural networks. *Semantic Scholar e-Preprint*, page Corpus ID: 204762668, 2019.
- [4] Andrew R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):930–945, May 1993.
- [5] Andrew R. Barron and Jason M. Klusowski. Approximation and estimation for high-dimensional deep learning networks. *arXiv e-prints*, page arXiv:1809.03090, September 2018.
- [6] Peter Bartlett, Vitaly Maiorov, and Ron Meir. Almost linear VC-dimension bounds for piecewise polynomial networks. *Neural Computation*, 10(8):2159–2173, 1998.
- [7] Monica Bianchini and Franco Scarselli. On the complexity of neural network classifiers: A comparison between shallow and deep architectures. *IEEE Transactions on Neural Networks and Learning Systems*, 25(8):1553–1565, Aug 2014.
- [8] Helmut Bölcskei, Philipp Grohs, Gitta Kutyniok, and Philipp Petersen. Optimal approximation with sparsely connected deep neural networks. *SIAM Journal on Mathematics of Data Science*, 1(1):8–45, 2019.
- [9] Minshuo Chen, Haoming Jiang, Wenjing Liao, and Tuo Zhao. Efficient approximation of deep ReLU networks for functions on low dimensional manifolds. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.
- [10] Charles K. Chui, Shao-Bo Lin, and Ding-Xuan Zhou. Construction of neural networks for realization of localized deep learning. *Frontiers in Applied Mathematics and Statistics*, 4:14, 2018.
- [11] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems*, 2:303–314, 1989.
- [12] Rémi Gribonval, Gitta Kutyniok, Morten Nielsen, and Felix Voigtlaender. Approximation spaces of deep neural networks. *Constructive Approximation*, 55:259–367, 2022.
- [13] Ingo Gühring, Gitta Kutyniok, and Philipp Petersen. Error bounds for approximations with deep ReLU neural networks in $W^{s,p}$ norms. *Analysis and Applications*, 18(05):803–859, 2020.
- [14] Nick Harvey, Christopher Liaw, and Abbas Mehrabian. Nearly-tight VC-dimension bounds for piecewise linear neural networks. In Satyen Kale and Ohad Shamir, editors, *Proceedings of the 2017 Conference on Learning Theory*, volume 65 of *Proceedings of Machine Learning Research*, pages 1064–1068, Amsterdam, Netherlands, 07–10 Jul 2017. PMLR.
- [15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, 2015.
- [16] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, abs/1207.0580, 2012.
- [17] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991.
- [18] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.
- [19] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37, ICML’15*, page 448–456. JMLR.org, 2015.
- [20] Yuling Jiao, Yanming Lai, Xiliang Lu, Fengru Wang, Jerry Zhijian Yang, and Yuanyuan Yang. Deep neural networks with ReLU-Sine-Exponential activations break curse of dimensionality on hölder class. *arXiv e-prints*, page arXiv:2103.00542, February 2021.

- [21] Michael J. Kearns and Robert E. Schapire. Efficient distribution-free learning of probabilistic concepts. *J. Comput. Syst. Sci.*, 48(3):464–497, June 1994.
- [22] Qianxiao Li, Ting Lin, and Zuowei Shen. Deep learning via dynamical systems: An approximation perspective. *Journal of the European Mathematical Society*, to appear.
- [23] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. In *International Conference on Learning Representations*, 2020.
- [24] Jianfeng Lu, Zuowei Shen, Haizhao Yang, and Shijun Zhang. Deep network approximation for smooth functions. *SIAM Journal on Mathematical Analysis*, 53(5):5465–5506, 2021.
- [25] Hadrien Montanelli and Haizhao Yang. Error bounds for deep ReLU networks using the Kolmogorov-Arnold superposition theorem. *Neural Networks*, 129:1–6, 2020.
- [26] Hadrien Montanelli, Haizhao Yang, and Qiang Du. Deep ReLU networks overcome the curse of dimensionality for bandlimited functions. *Journal of Computational Mathematics*, 39(6):801–815, 2021.
- [27] Guido F Montufar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. On the number of linear regions of deep neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2924–2932. Curran Associates, Inc., 2014.
- [28] Ryumei Nakada and Masaaki Imaizumi. Adaptive approximation and generalization of deep neural network with intrinsic dimensionality. *Journal of Machine Learning Research*, 21(174):1–38, 2020.
- [29] Philipp Petersen and Felix Voigtlaender. Optimal approximation of piecewise smooth functions using deep ReLU neural networks. *Neural Networks*, 108:296–330, 2018.
- [30] Bryan A. Plummer, Nikoli Dryden, Julius Frost, Torsten Hoeffler, and Kate Saenko. Neural parameter allocation search. In *International Conference on Learning Representations*, 2022.
- [31] Suo Qiu, Xiangmin Xu, and Bolun Cai. Frelu: Flexible rectified linear units for improving convolutional neural networks. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 1223–1228, Los Alamitos, CA, USA, aug 2018. IEEE Computer Society.
- [32] Akito Sakurai. Tight bounds for the VC-dimension of piecewise polynomial networks. In *Advances in Neural Information Processing Systems*, pages 323–329. Neural information processing systems foundation, 1999.
- [33] Pedro Savarese and Michael Maire. Learning implicitly recurrent CNNs through parameter sharing. In *International Conference on Learning Representations*, 2019.
- [34] Zuowei Shen, Haizhao Yang, and Shijun Zhang. Nonlinear approximation via compositions. *Neural Networks*, 119:74–84, 2019.
- [35] Zuowei Shen, Haizhao Yang, and Shijun Zhang. Deep network approximation characterized by number of neurons. *Communications in Computational Physics*, 28(5):1768–1811, 2020.
- [36] Zuowei Shen, Haizhao Yang, and Shijun Zhang. Deep network with approximation error being reciprocal of width to power of square root of depth. *Neural Computation*, 33(4):1005–1036, 03 2021.
- [37] Zuowei Shen, Haizhao Yang, and Shijun Zhang. Neural network approximation: Three hidden layers are enough. *Neural Networks*, 141:160–173, 2021.
- [38] Zuowei Shen, Haizhao Yang, and Shijun Zhang. Deep network approximation: Achieving arbitrary accuracy with fixed number of neurons. *Journal of Machine Learning Research*, 23(276):1–60, 2022.
- [39] Zuowei Shen, Haizhao Yang, and Shijun Zhang. Deep network approximation in terms of intrinsic parameters. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 19909–19934. PMLR, 17–23 Jul 2022.
- [40] Zuowei Shen, Haizhao Yang, and Shijun Zhang. Optimal approximation rate of ReLU networks in terms of width and depth. *Journal de Mathématiques Pures et Appliquées*, 157:101–135, 2022.
- [41] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

- 441 [42] Taiji Suzuki. Adaptivity of deep ReLU network for learning in Besov and mixed smooth Besov spaces:
442 optimal rate and curse of dimensionality. In *International Conference on Learning Representations*, 2019.
- 443 [43] Ludovic Trottier, Philippe Giguère, and Brahim Chaib-draa. Parametric exponential linear unit for deep
444 convolutional neural networks. *2017 16th IEEE International Conference on Machine Learning and*
445 *Applications (ICMLA)*, pages 207–214, 2017.
- 446 [44] Matthew Wallingford, Hao Li, Alessandro Achille, Avinash Ravichandran, Charless Fowlkes, Rahul
447 Bhotika, and Stefano Soatto. Task adaptive parameter sharing for multi-task learning. In *2022 IEEE/CVF*
448 *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7551–7560, 2022.
- 449 [45] Jiaxing Wang, Haoli Bai, Jiaxiang Wu, Xupeng Shi, Junzhou Huang, Irwin King, Michael Lyu, and Jian
450 Cheng. Revisiting parameter sharing for automatic neural channel number search. In H. Larochelle,
451 M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing*
452 *Systems*, volume 33, pages 5991–6002. Curran Associates, Inc., 2020.
- 453 [46] Ze Wang, Xiuyuan Cheng, Guillermo Sapiro, and Qiang Qiu. ACDC: Weight sharing in atom-coefficient
454 decomposed convolution. *arXiv e-prints*, page arXiv:2009.02386, September 2020.
- 455 [47] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: a novel image dataset for benchmarking
456 machine learning algorithms. *arXiv e-prints*, page arXiv:1708.07747, August 2017.
- 457 [48] Dmitry Yarotsky. Error bounds for approximations with deep ReLU networks. *Neural Networks*, 94:103–
458 114, 2017.
- 459 [49] Dmitry Yarotsky. Optimal approximation of continuous functions by very deep ReLU networks. In
460 Sébastien Bubeck, Vianney Perchet, and Philippe Rigollet, editors, *Proceedings of the 31st Conference*
461 *On Learning Theory*, volume 75 of *Proceedings of Machine Learning Research*, pages 639–649. PMLR,
462 06–09 Jul 2018.
- 463 [50] Dmitry Yarotsky and Anton Zhevnerchuk. The phase diagram of approximation rates for deep neural
464 networks. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural*
465 *Information Processing Systems*, volume 33, pages 13005–13015. Curran Associates, Inc., 2020.
- 466 [51] Lijun Zhang, Qizheng Yang, Xiao Liu, and Hui Guan. Rethinking hard-parameter sharing in multi-domain
467 learning. In *2022 IEEE International Conference on Multimedia and Expo (ICME)*, pages 01–06, 2022.
- 468 [52] Shijun Zhang. Deep neural network approximation via function compositions. *PhD Thesis, National*
469 *University of Singapore*, 2020. URL: <https://scholarbank.nus.edu.sg/handle/10635/186064>.
- 470 [53] Ding-Xuan Zhou. Universality of deep convolutional neural networks. *Applied and Computational*
471 *Harmonic Analysis*, 48(2):787–794, 2020.

472	Contents of main article and appendix	
473	1 Introduction	1
474	2 Main results and related work	3
475	2.1 Main results	3
476	2.2 Sketch of proving Theorem 2.1	5
477	2.3 Related work	6
478	3 Experimentation	7
479	3.1 Archimedean spiral	8
480	3.2 Fashion-MNIST	8
481	4 Conclusion	10
482	A Proof of main theorem	15
483	A.1 Notations	15
484	A.2 Detailed proof of Theorem 2.1	16
485	B Proof of auxiliary theorem	18
486	B.1 Key ideas of proving Theorem A.1	18
487	B.2 Detailed proof of Theorem A.1	20
488	C Proof of Proposition B.1	24
489	C.1 Lemmas for proving Proposition B.1	24
490	C.2 Detailed proof of Proposition B.1	27
491	D Proof of Proposition B.2	28
492	D.1 Lemmas for proving Proposition B.2	28
493	D.2 Detailed proof of Proposition B.2	30
494	D.3 Proof of Lemma D.2 for Proposition B.2	32
495	D.3.1 Proof of Lemma D.4 for Lemma D.2	33
496	D.3.2 Proof of Lemma D.5 for Lemma D.2	35

497 A Proof of main theorem

498 In this section, we will prove the main theorem, Theorem 2.1, based on an auxiliary theorem,
 499 Theorem A.1, which will be proved in Section B. Notations throughout this paper are summarized in
 500 Section A.1.

501 A.1 Notations

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

- 503 • Let \mathbb{R} , \mathbb{Q} , and \mathbb{Z} denote the set of real numbers, rational numbers, and integers, respectively.
- 504 • Let \mathbb{N} and \mathbb{N}^+ denote the set of natural numbers and positive natural numbers, respectively.
 505 That is, $\mathbb{N}^+ = \{1, 2, 3, \dots\}$ and $\mathbb{N} = \mathbb{N}^+ \cup \{0\}$.
- 506 • For any $x \in \mathbb{R}$, let $\lfloor x \rfloor := \max\{n : n \leq x, n \in \mathbb{Z}\}$ and $\lceil x \rceil := \min\{n : n \geq x, n \in \mathbb{Z}\}$.
- 507 • Let $\mathbb{1}_S$ be the indicator (characteristic) function of a set S , i.e., $\mathbb{1}_S$ is equal to 1 on S and 0
 508 outside S .
- 509 • The set difference of two sets A and B is denoted by $A \setminus B := \{x : x \in A, x \notin B\}$.
- 510 • Matrices are denoted by bold uppercase letters. For instance, $\mathbf{A} \in \mathbb{R}^{m \times n}$ is a real matrix
 511 of size $m \times n$, and \mathbf{A}^T denotes the transpose of \mathbf{A} . Vectors are denoted as bold lowercase
 512 letters. For example, $\mathbf{v} = [v_1, \dots, v_d]^T = \begin{bmatrix} v_1 \\ \vdots \\ v_d \end{bmatrix} \in \mathbb{R}^d$ is a column vector.
- 513 • For any $p \in [1, \infty)$, the p -norm (or ℓ^p -norm) of a vector $\mathbf{x} = [x_1, x_2, \dots, x_d]^T \in \mathbb{R}^d$ is defined
 514 by

$$\|\mathbf{x}\|_p = \|\mathbf{x}\|_{\ell^p} := \left(|x_1|^p + |x_2|^p + \dots + |x_d|^p \right)^{1/p}.$$

516 In the case $p = \infty$,

$$\|\mathbf{x}\|_\infty = \|\mathbf{x}\|_{\ell^\infty} := \max\{|x_i| : i = 1, 2, \dots, d\}.$$

- 518 • By convention, $\sum_{j=n_1}^{n_2} a_j = 0$ if $n_1 > n_2$, no matter what a_j is for each j .
- 519 • Given any $K \in \mathbb{N}^+$ and $\delta \in (0, \frac{1}{K})$, define a trifling region $\Omega([0, 1]^d, K, \delta)$ of $[0, 1]^d$ as

$$520 \quad \Omega([0, 1]^d, K, \delta) := \bigcup_{j=1}^d \left\{ \mathbf{x} = [x_1, x_2, \dots, x_d]^T \in [0, 1]^d : x_j \in \bigcup_{k=1}^{K-1} \left(\frac{k}{K} - \delta, \frac{k}{K} \right) \right\}. \quad (5)$$

521 In particular, $\Omega([0, 1]^d, K, \delta) = \emptyset$ if $K = 1$. See Figure 9 for two examples of trifling
 522 regions.

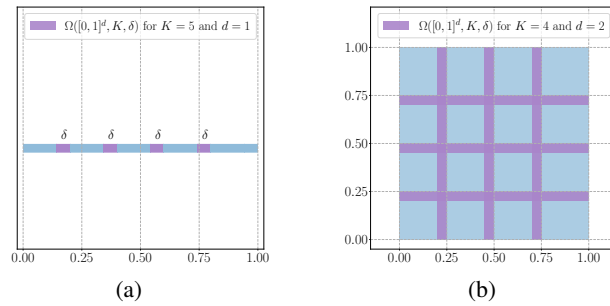


Figure 9: Two examples of trifling regions. (a) $K = 5, d = 1$. (b) $K = 4, d = 2$.

- 523 • For a continuous piecewise linear function $f(x)$, the x values where the slope changes are
 524 typically called **breakpoints**.
- 525 • Let $\sigma : \mathbb{R} \rightarrow \mathbb{R}$ denote the rectified linear unit (ReLU), i.e. $\sigma(x) = \max\{0, x\}$ for any $x \in \mathbb{R}$.
 526 With a slight abuse of notation, we define $\sigma : \mathbb{R}^d \rightarrow \mathbb{R}^d$ as $\sigma(\mathbf{x}) = \begin{bmatrix} \max\{0, x_1\} \\ \vdots \\ \max\{0, x_d\} \end{bmatrix}$ for any
 527 $\mathbf{x} = [x_1, \dots, x_d]^T \in \mathbb{R}^d$.

- Let $\mathcal{NN}_s\{n\}$ for $n, s \in \mathbb{N}^+$ denote the set of functions realized by height- s ReLU NestNets with as most n parameters.
- A function ϕ realized by a ReLU network can be briefly described as follows:

$$x = \tilde{h}_0 \xrightarrow[\mathcal{L}_0]{W_0, b_0} h_1 \xrightarrow{\sigma} \tilde{h}_1 \cdots \xrightarrow[\mathcal{L}_{L-1}]{W_{L-1}, b_{L-1}} h_L \xrightarrow{\sigma} \tilde{h}_L \xrightarrow[\mathcal{L}_L]{W_L, b_L} h_{L+1} = \phi(x),$$

where $W_i \in \mathbb{R}^{N_{i+1} \times N_i}$ and $b_i \in \mathbb{R}^{N_{i+1}}$ are the weight matrix and the bias vector in the i -th affine linear transformation \mathcal{L}_i , respectively, i.e.,

$$h_{i+1} = W_i \cdot \tilde{h}_i + b_i =: \mathcal{L}_i(\tilde{h}_i) \quad \text{for } i = 0, 1, \dots, L,$$

and

$$\tilde{h}_i = \sigma(h_i) \quad \text{for } i = 1, 2, \dots, L.$$

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

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

which has been illustrated in Figure 10.

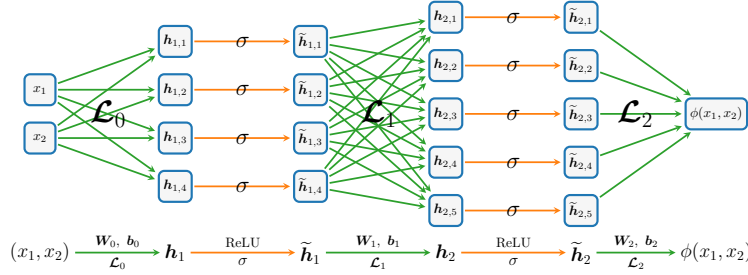


Figure 10: An example of a ReLU network of width 5 and depth 2.

- The expression “a network of width N and depth L ” means
 - The number of neurons in each **hidden** layer of this network (architecture) is no more than N .
 - The number of **hidden** layers of this network (architecture) is no more than L .

A.2 Detailed proof of Theorem 2.1

The key point of proving Theorem 2.1 is to construct a piecewise constant function to approximate the target continuous function. However, ReLU NestNets are unable to approximate piecewise constant functions well the continuity of ReLU NestNets. Thus, we introduce the trifling region $\Omega([0, 1]^d, K, \delta)$, defined in Equation (5), and use ReLU NestNets to implement piecewise constant functions outside the trifling region. To simplify the proof of Theorem 2.1, we introduce an auxiliary theorem, Theorem A.1 below. It can be regarded as a weaker variant of Theorem 2.1, ignoring the approximation in the trifling region.

Theorem A.1. *Given a continuous function $f \in C([0, 1]^d)$, for any $n, s \in \mathbb{N}^+$, there exists $\phi \in \mathcal{NN}_s\{355d^2(s+7)^2(2n+1)\}$ such that $\|\phi\|_{L^\infty(\mathbb{R}^d)} \leq |f(\mathbf{0})| + \omega_f(\sqrt{d})$ and*

$$|\phi(x) - f(x)| \leq 6\sqrt{d}\omega_f(n^{-(s+1)/d}) \quad \text{for any } x \in [0, 1]^d \setminus \Omega([0, 1]^d, K, \delta),$$

where $K = \lfloor n^{(s+1)/d} \rfloor$ and δ is an arbitrary number in $(0, \frac{1}{3K}]$.

The proof of Theorem A.1 can be found in Section B. By assuming Theorem A.1 is true, we can easily prove Theorem 2.1 for the case $p \in [1, \infty)$. To prove Theorem 2.1 for the case $p = \infty$, we need to control the approximation error in the trifling region. To this intent, we introduce a theorem to handle the approximation inside the trifling region.

560 **Theorem A.2** (Lemma 3.11 of [52] or Lemma 3.4 of [24]). Given any $\varepsilon > 0$, $K \in \mathbb{N}^+$, and $\delta \in (0, \frac{1}{3K}]$,
 561 assume $f \in C([0, 1]^d)$ and $g : \mathbb{R}^d \rightarrow \mathbb{R}$ is a general function with

$$562 \quad |g(\mathbf{x}) - f(\mathbf{x})| \leq \varepsilon \quad \text{for any } \mathbf{x} \in [0, 1]^d \setminus \Omega([0, 1]^d, K, \delta).$$

563 Then

$$564 \quad |\phi(\mathbf{x}) - f(\mathbf{x})| \leq \varepsilon + d \cdot \omega_f(\delta) \quad \text{for any } \mathbf{x} \in [0, 1]^d,$$

565 where $\phi := \phi_d$ is defined by induction through $\phi_0 := g$ and

$$566 \quad \phi_{i+1}(\mathbf{x}) := \text{mid}(\phi_i(\mathbf{x} - \delta \mathbf{e}_{i+1}), \phi_i(\mathbf{x}), \phi_i(\mathbf{x} + \delta \mathbf{e}_{i+1})) \quad \text{for } i = 0, 1, \dots, d-1,$$

567 where $\{\mathbf{e}_i\}_{i=1}^d$ is the standard basis in \mathbb{R}^d and $\text{mid}(\cdot, \cdot, \cdot)$ is the function returning the middle value of
 568 three inputs.

569 Now, let us prove Theorem 2.1 by assuming Theorem A.1 is true, the proof of which can be found in
 570 Section B.

571 *Proof of Theorem 2.1.* We may assume f is not a constant function since it is a trivial case. Then
 572 $\omega_f(r) > 0$ for any $r > 0$. Let us first consider the case $p \in [1, \infty)$. Set $K = \lfloor n^{(s+1)/d} \rfloor$ and choose a
 573 sufficiently small $\delta \in (0, \frac{1}{3K}]$ such that

$$574 \quad \begin{aligned} Kd\delta(2|f(\mathbf{0})| + 2\omega_f(\sqrt{d}))^p &= \lfloor n^{(s+1)/d} \rfloor d\delta(2|f(\mathbf{0})| + 2\omega_f(\sqrt{d}))^p \\ &\leq \left(\omega_f(n^{-(s+1)/d})\right)^p. \end{aligned}$$

575 By Theorem A.1, there exists

$$576 \quad \begin{aligned} \phi &\in \mathcal{NN}_s\{355d^2(s+7)^2(2n+1)\} \subseteq \mathcal{NN}_s\{355d^2(s+7)^2 \cdot 2(n+1)\} \\ &\subseteq \mathcal{NN}_s\{10^3d^2(s+7)^2(n+1)\} \end{aligned}$$

577 such that $\|\phi\|_{L^\infty(\mathbb{R}^d)} \leq |f(\mathbf{0})| + \omega_f(\sqrt{d})$ and

$$578 \quad |\phi(\mathbf{x}) - f(\mathbf{x})| \leq 6\sqrt{d}\omega_f(n^{-(s+1)/d}) \quad \text{for any } \mathbf{x} \in [0, 1]^d \setminus \Omega([0, 1]^d, K, \delta).$$

579 Since $\|f\|_{L^\infty([0, 1]^d)} \leq |f(\mathbf{0})| + \omega_f(\sqrt{d})$ and the Lebesgue measure of $\Omega([0, 1]^d, K, \delta)$ is bounded
 580 by $Kd\delta$, we have

$$581 \quad \begin{aligned} \|\phi - f\|_{L^p([0, 1]^d)}^p &= \int_{\Omega([0, 1]^d, K, \delta)} |\phi(\mathbf{x}) - f(\mathbf{x})|^p d\mathbf{x} + \int_{[0, 1]^d \setminus \Omega([0, 1]^d, K, \delta)} |\phi(\mathbf{x}) - f(\mathbf{x})|^p d\mathbf{x} \\ &\leq Kd\delta(2|f(\mathbf{0})| + 2\omega_f(\sqrt{d}))^p + \left(6\sqrt{d}\omega_f(n^{-(s+1)/d})\right)^p \\ &\leq \left(\omega_f(n^{-(s+1)/d})\right)^p + \left(6\sqrt{d}\omega_f(n^{-(s+1)/d})\right)^p \leq \left(7\sqrt{d}\omega_f(n^{-(s+1)/d})\right)^p. \end{aligned}$$

582 Hence, we have $\|f - \phi\|_{L^p([0, 1]^d)} \leq 7\sqrt{d}\omega_f(n^{-(s+1)/d})$.

583 Next, let us discuss the case $p = \infty$. Set $K = \lfloor n^{(s+1)/d} \rfloor$ and choose a sufficiently small $\delta \in (0, \frac{1}{3K}]$
 584 such that

$$585 \quad d \cdot \omega_f(\delta) \leq \omega_f(n^{-(s+1)/d}).$$

586 By Theorem A.1,

$$587 \quad \phi_0 \in \mathcal{NN}_s\{355d^2(s+7)^2(2n+1)\}$$

588 such that

$$589 \quad |\phi_0(\mathbf{x}) - f(\mathbf{x})| \leq 6\sqrt{d}\omega_f(n^{-(s+1)/d}) \quad \text{for any } \mathbf{x} \in [0, 1]^d \setminus \Omega([0, 1]^d, K, \delta).$$

590 By Theorem A.2 with $g = \phi_0$ and $\varepsilon = 6\sqrt{d}\omega_f(n^{-(s+1)/d})$ therein, we have

$$591 \quad |\phi(\mathbf{x}) - f(\mathbf{x})| \leq \varepsilon + d \cdot \omega_f(\delta) \leq 7\sqrt{d}\omega_f(n^{-(s+1)/d}) \quad \text{for any } \mathbf{x} \in [0, 1]^d,$$

592 where $\phi := \phi_d$ is defined by induction through

$$593 \quad \phi_{i+1}(\mathbf{x}) := \text{mid}(\phi_i(\mathbf{x} - \delta \mathbf{e}_{i+1}), \phi_i(\mathbf{x}), \phi_i(\mathbf{x} + \delta \mathbf{e}_{i+1})) \quad \text{for } i = 0, 1, \dots, d-1,$$

where $\{e_i\}_{i=1}^d$ is the standard basis in \mathbb{R}^d and $\text{mid}(\cdot, \cdot, \cdot)$ is the function returning the middle value of three inputs. It remains to estimate the number of parameters in the NestNet realizing $\phi = \phi_d$. By Lemma 3.1 of [37], $\text{mid}(\cdot, \cdot, \cdot)$ can be realized by a ReLU network of width 14 and depth 2, and hence with at most $14 \times (14 + 1) \times (2 + 1) = 630$ parameters.

By defining a vector-valued function $\Phi_0 : \mathbb{R}^d \rightarrow \mathbb{R}^3$ as

$$\Phi_0(\mathbf{x}) := [\phi_0(\mathbf{x} - \delta \mathbf{e}_1), \phi_0(\mathbf{x}), \phi_0(\mathbf{x} + \delta \mathbf{e}_1)]^T \quad \text{for any } \mathbf{x} \in \mathbb{R}^d,$$

we have $\Phi_0 \in \mathcal{NN}_s\{3^2(355d^2(s+7)^2(2n+1))\}$, implying

$$\begin{aligned} \phi_1 &= \min(\cdot, \cdot, \cdot) \circ \Phi_0 \in \mathcal{NN}_s\left\{630 + 3^2(355d^2(s+7)^2(2n+1))\right\} \\ &\subseteq \mathcal{NN}_s\left\{10(355d^2(s+7)^2(2n+1))\right\}. \end{aligned}$$

Similarly, we have

$$\begin{aligned} \phi &= \phi_d \in \mathcal{NN}_s\left\{10^d(355d^2(s+7)^2(2n+1))\right\} \subseteq \mathcal{NN}_s\left\{10^d(355d^2(s+7)^2 \cdot 2(n+1))\right\} \\ &\subseteq \mathcal{NN}_s\left\{10^{d+3}d^2(s+7)^2(n+1)\right\}. \end{aligned}$$

Thus, we finish the proof of Theorem 2.1.

□

B Proof of auxiliary theorem

We will prove the auxiliary theorem, Theorem A.1, in this section. We first present the key ideas in Section B.1. Next, the detailed proof is presented in Section B.2, based on two propositions in Section B.1, the proofs of which can be found in Sections C and D.

B.1 Key ideas of proving Theorem A.1

Our goal is to construct an almost piecewise constant function realized by a ReLU NestNet to approximate the target function $f \in C([0, 1]^d)$ well. The construction can be divided into three main steps.

1. First, we divide $[0, 1]^d$ into a union of “important” cubes $\{Q_\beta\}_{\beta \in \{0, 1, \dots, K-1\}^d}$ and the trifling region $\Omega([0, 1]^d, K, \delta)$, where $K = \mathcal{O}(n^{(s+1)/d})$. Each Q_β is associated with a representative $\mathbf{x}_\beta \in Q_\beta$ for each vector index β . See Figure 13 for illustrations.
2. Next, we design a vector-valued function $\Phi_1(\mathbf{x})$ to map the whole cube Q_β to its index β for each β . Here, Φ_1 can be defined/constructed via

$$\Phi_1(\mathbf{x}) = [\phi_1(x_1), \phi_1(x_2), \dots, \phi_1(x_d)]^T,$$

where each one-dimensional function ϕ_1 is a step function outside the trifling region and hence can be realized by a ReLU NestNet.

3. The aim of the final step is essentially to solve a point fitting problem. We will construct a function ϕ_2 realized by a ReLU NestNet to map β approximately to $f(\mathbf{x}_\beta)$ for each β . Then we have

$$\phi_2 \circ \Phi_1(\mathbf{x}) = \phi_2(\beta) \approx f(\mathbf{x}_\beta) \approx f(\mathbf{x}) \quad \text{for any } \mathbf{x} \in Q_\beta \text{ and each } \beta,$$

implying

$$\phi := \phi_2 \circ \Phi_1 \approx f \quad \text{on } [0, 1]^d \setminus \Omega([0, 1]^d, K, \delta).$$

We remark that, in the construction of ϕ_2 , we only need to care about the values of ϕ_2 sampled inside the set $\{0, 1, \dots, K-1\}^d$, which is a key point to ease the design of a ReLU NestNet to realize ϕ_2 as we shall see later.

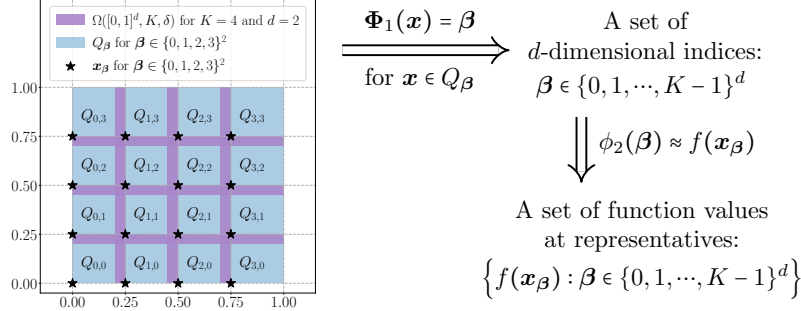


Figure 11: An illustration of the ideas of constructing the desired function $\phi = \phi_2 \circ \Phi_1$. Note that $\phi \approx f$ outside the trifling region since $\phi(x) = \phi_2 \circ \Phi_1(x) = \phi_2(\beta) \approx f(x_\beta) \approx f(x)$ for any $x \in Q_\beta$ and each $\beta \in \{0, 1, \dots, K-1\}^d$.

Observe that in Figure 11, we have

$$\phi(x) = \phi_2 \circ \Phi_1(x) = \phi_2(\beta) \stackrel{\mathcal{E}_1}{\approx} f(x_\beta) \stackrel{\mathcal{E}_2}{\approx} f(x)$$

for any $x \in Q_\beta$ and each $\beta \in \{0, 1, \dots, K-1\}^d$. That means $\phi - f$ is controlled by $\mathcal{E}_1 + \mathcal{E}_2$ on $[0, 1]^d \setminus \Omega([0, 1]^d, K, \delta)$. Since $\|x - x_\beta\|_2 \leq \sqrt{d}/K$ for any $x \in Q_\beta$ and each β , \mathcal{E}_2 is bounded by $\omega_f(\sqrt{d}/K)$. As we shall see later, \mathcal{E}_1 can be bounded by $\mathcal{O}(\omega_f(\sqrt{d}/K))$ by applying Proposition B.2. Therefore, $\phi - f$ is controlled by $\mathcal{O}(\omega_f(\sqrt{d}/K))$ outside the trifling region, from which we deduce the desired approximation error since $K = \mathcal{O}(n^{-(s+1)/d})$.

Finally, we introduce two propositions to simplify the constructions of Φ_1 and ϕ_2 mentioned above. We first show how to construct a ReLU network to implement a one-dimensional step function ϕ_1 in Proposition B.1 below. Then Φ_1 can be defined via

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

Proposition B.1. *Given any $n, r \in \mathbb{N}^+$, $\delta \in (0, 1)$, and $J \in \mathbb{N}^+$ with $J \leq 2^{n^r}$, there exists $\phi \in \mathcal{NN}_r\{36(r+7)n\}$ such that*

$$\phi(x) = \lfloor x \rfloor \quad \text{for any } x \in \bigcup_{j=0}^{J-1} [j, j+1-\delta]$$

and

$$\phi(x) = J \quad \text{for any } x \in [J, J+1].$$

The construction of ϕ_2 is mainly based on Proposition B.2 below, whose proof relies on the bit extraction technique proposed in [6]. As we shall see later, some pre-processing is necessary for meeting the requirements of applying Proposition B.2 to construct ϕ_2 .

Proposition B.2. *Given any $\varepsilon > 0$ and $n, s \in \mathbb{N}^+$, assume $y_j \geq 0$ for $j = 0, 1, \dots, J-1$ are samples with $J \leq n^{s+1}$ and*

$$|y_j - y_{j-1}| \leq \varepsilon \quad \text{for } j = 1, 2, \dots, J-1.$$

Then there exists $\phi \in \mathcal{NN}_s\{350(s+7)^2(n+1)\}$ such that

$$(i) \quad |\phi(j) - y_j| \leq \varepsilon \text{ for } j = 0, 1, \dots, J-1.$$

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

The proofs of these two propositions can be found in Sections C and D. We will give the detailed proof of Theorem A.1 in Section B.2.

657 B.2 Detailed proof of Theorem A.1

658 We essentially construct an almost piecewise constant function realized by a ReLU NestNet with
 659 at most $\mathcal{O}(n)$ parameters to approximate f . We may assume f is not a constant function since
 660 it is a trivial case. Then $\omega_f(r) > 0$ for any $r > 0$. It is clear that $|f(\mathbf{x}) - f(\mathbf{0})| \leq \omega_f(\sqrt{d})$ for
 661 any $\mathbf{x} \in [0, 1]^d$. By defining $\tilde{f} := f - f(\mathbf{0}) + \omega_f(\sqrt{d})$, we have $\omega_{\tilde{f}}(r) = \omega_f(r)$ for any $r \geq 0$ and
 662 $0 \leq \tilde{f}(\mathbf{x}) \leq 2\omega_f(\sqrt{d})$ for any $\mathbf{x} \in [0, 1]^d$.

663 Set $K = \lfloor n^{(s+1)/d} \rfloor$ and let δ be an arbitrary number in $(0, \frac{1}{3K}]$. The proof can be divided into four
 664 main steps as follows:

- 665 1. Divide $[0, 1]^d$ into a union of sub-cubes $\{Q_\beta\}_{\beta \in \{0, 1, \dots, K-1\}^d}$ and the trifling region
 666 $\Omega([0, 1]^d, K, \delta)$, and denote \mathbf{x}_β as the vertex of Q_β with minimum $\|\cdot\|_1$ norm.
- 667 2. Construct a sub-network based on Proposition B.1 to implement a vector function Φ_1
 668 projecting the whole cube Q_β to the d -dimensional index β for each β , i.e., $\Phi_1(\mathbf{x}) = \beta$ for
 669 all $\mathbf{x} \in Q_\beta$.
- 670 3. Construct a sub-network to implement a function ϕ_2 mapping the index β approximately to
 671 $\tilde{f}(\mathbf{x}_\beta)$. This core step can be further divided into three sub-steps:
 - 672 3.1. Construct a sub-network to implement ψ_1 bijectively mapping the index set
 673 $\{0, 1, \dots, K-1\}^d$ to an auxiliary set $\mathcal{A}_1 \subseteq \{\frac{j}{2K^d} : j = 0, 1, \dots, 2K^d\}$ defined later.
 674 See Figure 14 for an illustration.
 - 675 3.2. Determine a continuous piecewise linear function g with a set of breakpoints $\mathcal{A}_1 \cup$
 676 $\mathcal{A}_2 \cup \{1\}$, where $\mathcal{A}_2 \in \{\frac{j}{2K^d} : j = 0, 1, \dots, 2K^d\}$ is a set defined later. Moreover, g
 677 should satisfy two conditions: 1) the values of g at breakpoints in \mathcal{A}_1 is given based on
 678 $\{\tilde{f}(\mathbf{x}_\beta)\}_\beta$, i.e., $g \circ \psi_1(\beta) = \tilde{f}(\mathbf{x}_\beta)$; 2) the values of g at breakpoints in $\mathcal{A}_2 \cup \{1\}$ is
 679 defined to reduce the variation of g , which is necessary for applying Proposition B.2.
 - 680 3.3. Apply Proposition B.2 to construct a sub-network to implement a function ψ_2 approxi-
 681 mating g well on $\mathcal{A}_1 \cup \mathcal{A}_2 \cup \{1\}$. Then the desired function ϕ_2 is given by $\phi_2 = \psi_2 \circ \psi_1$
 682 satisfying $\phi_2(\beta) = \psi_2 \circ \psi_1(\beta) \approx g \circ \psi_1(\beta) = \tilde{f}(\mathbf{x}_\beta)$.
- 683 4. Construct the final network to implement the desired function ϕ via $\phi = \phi_2 \circ \Phi_1 + f(\mathbf{0}) -$
 684 $\omega_f(\sqrt{d})$. Then we have $\phi_2 \circ \Phi_1(\mathbf{x}) = \phi_2(\beta) \approx \tilde{f}(\mathbf{x}_\beta) \approx \tilde{f}(\mathbf{x})$ for any $\mathbf{x} \in Q_\beta$ and
 685 $\beta \in \{0, 1, \dots, K-1\}^d$, implying $\phi(\mathbf{x}) = \phi_2 \circ \Phi_1(\mathbf{x}) + f(\mathbf{0}) - \omega_f(\sqrt{d}) \approx \tilde{f}(\mathbf{x}) + f(\mathbf{0}) -$
 686 $\omega_f(\sqrt{d}) = f(\mathbf{x})$.

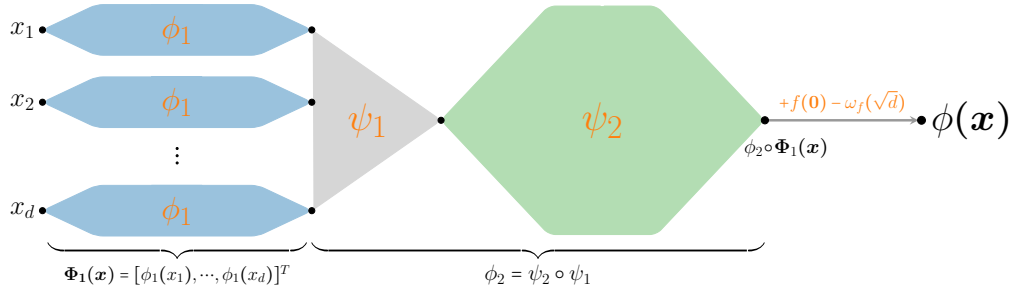


Figure 12: An illustration of the NestNet architecture realizing $\phi = \phi_2 \circ \Phi_1 + f(\mathbf{0}) - \omega_f(\sqrt{d})$. Here, ϕ_1 is implemented via Proposition B.1; $\psi_1 : \mathbb{R}^d \rightarrow \mathbb{R}$ is an affine linear function; ψ_2 is implemented via Proposition B.2.

687 See Figure 12 for an illustration of the NestNet architecture realizing $\phi = \phi_2 \circ \Phi_1 + f(\mathbf{0}) - \omega_f(\sqrt{d})$.
 688 The details of the steps mentioned above can be found below.

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

690 Define $x_\beta := \beta/K$ and

$$691 \quad Q_\beta := \left\{ \mathbf{x} = [x_1, x_2, \dots, x_d]^T \in [0, 1]^d : x_i \in \left[\frac{\beta_i}{K}, \frac{\beta_i+1}{K} - \delta \cdot \mathbf{1}_{\{\beta_i \leq K-2\}} \right], \quad i = 1, 2, \dots, d \right\}$$

692 for each d -dimensional index $\beta = [\beta_1, \beta_2, \dots, \beta_d]^T \in \{0, 1, \dots, K-1\}^d$. Recall that $\Omega([0, 1]^d, K, \delta)$ is
 693 the trifling region defined in Equation (5). Apparently, $x_\beta = \beta/K$ is the vertex of Q_β with minimum
 694 $\|\cdot\|_1$ norm and

$$695 \quad [0, 1]^d = \left(\cup_{\beta \in \{0, 1, \dots, K-1\}^d} Q_\beta \right) \cup \Omega([0, 1]^d, K, \delta).$$

696 See Figure 13 for illustrations.

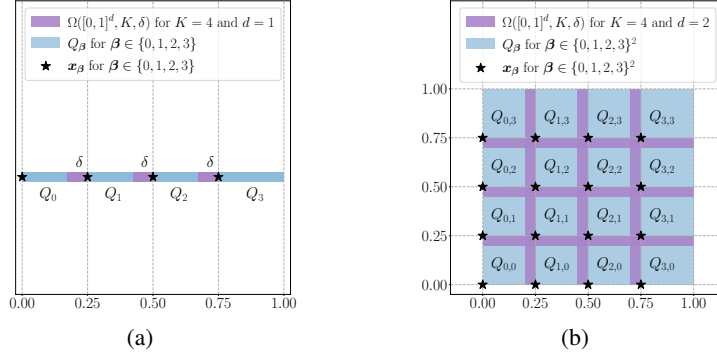


Figure 13: Illustrations of $\Omega([0, 1]^d, K, \delta)$, Q_β , and x_β for $\beta \in \{0, 1, \dots, K-1\}^d$. (a) $K = 4$ and $d = 1$. (b) $K = 4$ and $d = 2$.

697 **Step 2:** Construct Φ_1 mapping $x \in Q_\beta$ to β .

698 Note that

$$699 \quad K-1 = \lfloor n^{(s+1)/d} \rfloor - 1 \leq n^{s+1} \leq (n^s)^2 \leq 4^{(n^s)} = 2^{2(n^s)} \leq 2^{(2n)^s} = 2^{\tilde{n}^s},$$

700 where $\tilde{n} = 2n$. By Proposition B.1 with $r = s$ and $J = K-1 \leq 2^{\tilde{n}^s} = 2^{\tilde{n}^r}$ therein, there exists

$$701 \quad \tilde{\phi}_1 \in \mathcal{NN}_s\{36(s+7)\tilde{n}\} = \mathcal{NN}_s\{36(s+7)(2n)\} = \mathcal{NN}_s\{72(s+7)n\}$$

702 such that

$$703 \quad \tilde{\phi}_1(x) = \lfloor x \rfloor \quad \text{for any } x \in \bigcup_{k=0}^{K-2} [k, k+1-\tilde{\delta}] \text{ with } \tilde{\delta} = K\delta$$

704 and

$$705 \quad \tilde{\phi}_1(x) = K-1 \quad \text{for any } x \in [K-1, K].$$

706 Define $\phi_1(x) := \tilde{\phi}_1(Kx)$ for any $x \in \mathbb{R}$. Then, we have $\phi_1 \in \mathcal{NN}_s\{72(s+7)n\}$ and

$$707 \quad \phi_1(x) = k \quad \text{if } x \in \left[\frac{k}{K}, \frac{k+1}{K} - \delta \cdot \mathbf{1}_{\{k \leq K-2\}} \right] \quad \text{for } k = 0, 1, \dots, K-1.$$

708 It follows that $\phi_1(x_i) = \beta_i$ if $\mathbf{x} = [x_1, x_2, \dots, x_d]^T \in Q_\beta$ for each $\beta = [\beta_1, \beta_2, \dots, \beta_d]^T$.

709 By defining

$$710 \quad \Phi_1(\mathbf{x}) := [\phi_1(x_1), \phi_1(x_2), \dots, \phi_1(x_d)]^T \quad \text{for any } \mathbf{x} = [x_1, x_2, \dots, x_d]^T \in \mathbb{R}^d,$$

711 we have

$$712 \quad \Phi_1(\mathbf{x}) = \beta \quad \text{if } \mathbf{x} \in Q_\beta \quad \text{for each } \beta \in \{0, 1, \dots, K-1\}^d. \quad (6)$$

713 **Step 3:** Construct ϕ_2 mapping β approximately to $\tilde{f}(x_\beta)$.

714 The construction of the sub-network implementing ϕ_2 is essentially based on Proposition B.2. To
 715 meet the requirements of applying Proposition B.2, we first define two auxiliary sets \mathcal{A}_1 and \mathcal{A}_2 as

$$716 \quad \mathcal{A}_1 := \left\{ \frac{i}{K^{d-1}} + \frac{k}{2K^d} : i = 0, 1, \dots, K^{d-1} - 1 \quad \text{and} \quad k = 0, 1, \dots, K-1 \right\}$$

717 and

$$718 \quad \mathcal{A}_2 := \left\{ \frac{i}{K^{d-1}} + \frac{K+k}{2K^d} : i = 0, 1, \dots, K^{d-1}-1 \quad \text{and} \quad k = 0, 1, \dots, K-1 \right\}.$$

719 Clearly,

$$720 \quad \mathcal{A}_1 \cup \mathcal{A}_2 \cup \{1\} = \left\{ \frac{j}{2K^d} : j = 0, 1, \dots, 2K^d \right\} \quad \text{and} \quad \mathcal{A}_1 \cap \mathcal{A}_2 = \emptyset.$$

721 See Figure 13 for an illustration of \mathcal{A}_1 and \mathcal{A}_2 . Next, we further divide this step into three sub-steps.

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

723 Inspired by the binary representation, we define

$$724 \quad \psi_1(\mathbf{x}) := \frac{x_d}{2K^d} + \sum_{i=1}^{d-1} \frac{x_i}{K^i} \quad \text{for any } \mathbf{x} = [x_1, x_2, \dots, x_d]^T \in \mathbb{R}^d. \quad (7)$$

725 Then ψ_1 is a linear function bijectively mapping the index set $\{0, 1, \dots, K-1\}^d$ to

$$726 \quad \begin{aligned} & \left\{ \psi_1(\boldsymbol{\beta}) : \boldsymbol{\beta} \in \{0, 1, \dots, K-1\}^d \right\} = \left\{ \frac{\beta_d}{2K^d} + \sum_{i=1}^{d-1} \frac{\beta_i}{K^i} : \boldsymbol{\beta} \in \{0, 1, \dots, K-1\}^d \right\} \\ & = \left\{ \frac{i}{K^{d-1}} + \frac{k}{2K^d} : i = 0, 1, \dots, K^{d-1}-1 \quad \text{and} \quad k = 0, 1, \dots, K-1 \right\} = \mathcal{A}_1. \end{aligned}$$

727 **Step 3.2:** Construct g to satisfy $g \circ \psi_1(\boldsymbol{\beta}) = \tilde{f}(\mathbf{x}_\beta)$ and to meet the requirements of applying
728 Proposition B.2.

729 Let $g : [0, 1] \rightarrow \mathbb{R}$ be a continuous piecewise linear function with a set of breakpoints

$$730 \quad \left\{ \frac{j}{2K^d} : j = 0, 1, \dots, 2K^d \right\} = \mathcal{A}_1 \cup \mathcal{A}_2 \cup \{1\}.$$

731 Moreover, the values of g at these breakpoints are assigned as follows:

732 • At the breakpoint 1, let $g(1) = \tilde{f}(\mathbf{1})$, where $\mathbf{1} = [1, 1, \dots, 1]^T \in \mathbb{R}^d$.

733 • For the breakpoints in $\mathcal{A}_1 = \left\{ \psi_1(\boldsymbol{\beta}) : \boldsymbol{\beta} \in \{0, 1, \dots, K-1\}^d \right\}$, we set

$$734 \quad g(\psi_1(\boldsymbol{\beta})) = \tilde{f}(\mathbf{x}_\beta) \quad \text{for any } \boldsymbol{\beta} \in \{0, 1, \dots, K-1\}^d. \quad (8)$$

735 • The values of g at the breakpoints in \mathcal{A}_2 are assigned to reduce the variation of g , which is a
736 requirement of applying Proposition B.2. Recall that

$$737 \quad \left\{ \frac{i}{K^{d-1}} - \frac{K+1}{2K^d}, \frac{i}{K^{d-1}} \right\} \subseteq \mathcal{A}_1 \cup \{1\} \quad \text{for } i = 1, 2, \dots, K^{d-1},$$

738 implying the values of g at $\frac{i}{K^{d-1}} - \frac{K+1}{2K^d}$ and $\frac{i}{K^{d-1}}$ have been assigned in the previous
739 cases for. Thus, the values of g at the breakpoints in \mathcal{A}_2 can be successfully assigned
740 by letting g linear on each interval $[\frac{i}{K^{d-1}} - \frac{K+1}{2K^d}, \frac{i}{K^{d-1}}]$ for $i = 1, 2, \dots, K^{d-1}$ since $\mathcal{A}_2 \subseteq$
741 $\bigcup_{i=1}^{K^{d-1}} [\frac{i}{K^{d-1}} - \frac{K+1}{2K^d}, \frac{i}{K^{d-1}}]$. See Figure 14 for an illustration.

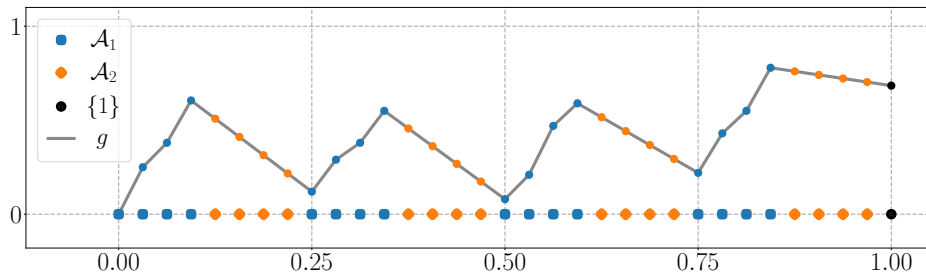


Figure 14: An illustration of \mathcal{A}_1 , \mathcal{A}_2 , $\{1\}$, and g for $K = 4$ and $d = 2$.

742 Apparently, such a function g exists. See Figure 14 for an illustration of g . It is easy to verify that

$$743 \quad \left| g\left(\frac{j}{2K^d}\right) - g\left(\frac{j-1}{2K^d}\right) \right| \leq \max \left\{ \omega_{\tilde{f}}\left(\frac{\sqrt{d}}{K}\right), \frac{\omega_f(\sqrt{d})}{K} \right\} \leq \omega_{\tilde{f}}\left(\frac{\sqrt{d}}{K}\right) = \omega_f\left(\frac{\sqrt{d}}{K}\right)$$

744 for $j = 1, 2, \dots, 2K^d$. Moreover, we have

$$745 \quad 0 \leq g\left(\frac{j}{2K^d}\right) \leq 2\omega_f(\sqrt{d}) \quad \text{for } j = 0, 1, \dots, 2K^d.$$

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

747 Observe that

$$748 \quad 2K^d = 2\left(\lfloor n^{(s+1)/d} \rfloor\right)^d \leq 2n^{s+1} \leq (2n)^{s+1} = \tilde{n}^{s+1}, \quad \text{where } \tilde{n} = 2n.$$

749 By Proposition B.2 with $y_j = g\left(\frac{j}{2K^d}\right)$ and $\varepsilon = \omega_f\left(\frac{\sqrt{d}}{K}\right) > 0$ therein, there exists

$$750 \quad \tilde{\psi}_2 \in \mathcal{NN}_s\left\{350(s+7)^2(\tilde{n}+1)\right\} = \mathcal{NN}_s\left\{350(s+7)^2(2n+1)\right\}$$

751 such that

$$752 \quad |\tilde{\psi}_2(j) - g\left(\frac{j}{2K^d}\right)| \leq \omega_f\left(\frac{\sqrt{d}}{K}\right) \quad \text{for } j = 0, 1, \dots, 2K^d - 1$$

753 and

$$754 \quad 0 \leq \tilde{\psi}_2(x) \leq \max \left\{ g\left(\frac{j}{2K^d}\right) : j = 0, 1, \dots, 2K^d - 1 \right\} \leq 2\omega_f(\sqrt{d}) \quad \text{for any } x \in \mathbb{R}.$$

755 By defining $\psi_2(x) := \tilde{\psi}_2(2K^d x)$ for any $x \in \mathbb{R}$, we have

$$756 \quad 0 \leq \psi_2(x) = \tilde{\psi}_2(2K^d x) \leq 2\omega_f(\sqrt{d}) \quad \text{for any } x \in \mathbb{R} \quad (9)$$

757 and

$$758 \quad \left| \psi_2\left(\frac{j}{2K^d}\right) - g\left(\frac{j}{2K^d}\right) \right| = |\tilde{\psi}_2(j) - g\left(\frac{j}{2K^d}\right)| \leq \omega_f\left(\frac{\sqrt{d}}{K}\right) \quad \text{for } j = 0, 1, \dots, 2K^d - 1. \quad (10)$$

759 Let us end Step 3 by defining the desired function ϕ_2 as $\phi_2 := \psi_2 \circ \psi_1$. Recall that $\psi_1(\beta) = \mathcal{A}_1 \subseteq$
760 $\left\{ \frac{j}{2K^d} : j = 0, 1, \dots, 2K^d - 1 \right\}$. Then, by Equations (8) and (10), we have

$$761 \quad \left| \phi_2(\beta) - \tilde{f}(\mathbf{x}_\beta) \right| = \left| \psi_2(\psi_1(\beta)) - g(\psi_1(\beta)) \right| \leq \omega_f\left(\frac{\sqrt{d}}{K}\right) \quad (11)$$

762 for any $\beta \in \{0, 1, \dots, K-1\}^d$. Moreover, by Equation (9) and $\phi_2 = \psi_2 \circ \psi_1$, we have

$$763 \quad 0 \leq \phi_2(\mathbf{x}) = \psi_2(\psi(\mathbf{x})) \leq 2\omega_f(\sqrt{d}) \quad \text{for any } \mathbf{x} \in \mathbb{R}^d. \quad (12)$$

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

765 Define $\phi := \phi_2 \circ \Phi_1 + f(\mathbf{0}) - \omega_f(\sqrt{d})$. By Equation (12), we have

$$766 \quad 0 \leq \phi_2 \circ \Phi_1(\mathbf{x}) \leq 2\omega_f(\sqrt{d})$$

767 for any $\mathbf{x} \in \mathbb{R}^d$, implying

$$768 \quad f(\mathbf{0}) - \omega_f(\sqrt{d}) \leq \phi(\mathbf{x}) = \phi_2 \circ \Phi_1(\mathbf{x}) + f(\mathbf{0}) - \omega_f(\sqrt{d}) \leq f(\mathbf{0}) + \omega_f(\sqrt{d}).$$

769 It follows that $\|\phi\|_{L^\infty(\mathbb{R}^d)} \leq |f(\mathbf{0})| + \omega_f(\sqrt{d})$.

770 Next, let us estimate the approximation error. Recall that $f = \tilde{f} + f(\mathbf{0}) - \omega_f(\sqrt{d})$ and $\phi = \phi_2 \circ \Phi_1 +$
771 $f(\mathbf{0}) - \omega_f(\sqrt{d})$. By Equations (6) and (11), for any $\mathbf{x} \in Q_\beta$ and $\beta \in \{0, 1, \dots, K-1\}^d$, we have

$$\begin{aligned} |f(\mathbf{x}) - \phi(\mathbf{x})| &= |\tilde{f}(\mathbf{x}) - \phi_2 \circ \Phi_1(\mathbf{x})| = |\tilde{f}(\mathbf{x}) - \phi_2(\beta)| \\ &\leq |\tilde{f}(\mathbf{x}) - \tilde{f}(\mathbf{x}_\beta)| + |\tilde{f}(\mathbf{x}_\beta) - \phi_2(\beta)| \\ &\leq \omega_f\left(\frac{\sqrt{d}}{K}\right) + \omega_f\left(\frac{\sqrt{d}}{K}\right) \leq 2\omega_f(2\sqrt{d}n^{-(s+1)/d}), \end{aligned}$$

773 where the last inequality comes from the fact

$$774 \quad K = \lfloor n^{(s+1)/d} \rfloor \geq n^{(s+1)/d}/2 \quad \text{for } n \in \mathbb{N}^+.$$

775 Recall the fact $\omega_f(j \cdot r) \leq j \cdot \omega_f(r)$ for any $j \in \mathbb{N}^+$ and $r \in [0, \infty)$. Therefore, for any $\mathbf{x} \in$
 776 $\bigcup_{\beta \in \{0,1,\dots,K-1\}^d} Q_\beta = [0,1]^d \setminus \Omega([0,1]^d, K, \delta)$, we have

$$777 \quad |\phi(\mathbf{x}) - f(\mathbf{x})| \leq 2\omega_f\left(2\sqrt{d}n^{-(s+1)/d}\right) \leq 2\left\lceil 2\sqrt{d} \right\rceil \omega_f\left(n^{-(s+1)/d}\right) \\ \leq 6\sqrt{d}\omega_f\left(n^{-(s+1)/d}\right).$$

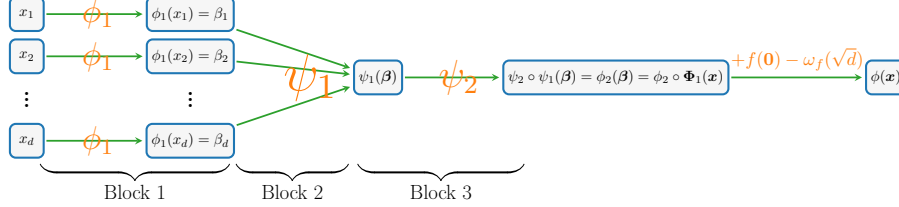


Figure 15: An illustration of the final NestNet realizing $\phi = \phi_2 \circ \Phi_1 + f(\mathbf{0}) - \omega_f(\sqrt{d})$ for $\mathbf{x} = [x_1, x_2, \dots, x_d]^T \in Q_\beta$ for each $\beta \in \{0, 1, \dots, K-1\}^d$.

778 It remains to estimate the number of parameters in the NestNet realizing ϕ , which is shown in
 779 Figure 15. Recall that $\phi_1 \in \mathcal{NN}_s\{72(s+7)n\}$, ψ_1 is an affine linear map, and $\psi_2 \in \mathcal{NN}_s\{350(s+7)^2(2n+1)\}$. Therefore, $\phi = \phi_2 \circ \Phi_1 + f(\mathbf{0}) - \omega_f(\sqrt{d})$ can be realized by a height- s NestNet with
 780 at most
 781

$$782 \quad \underbrace{d^2(72(s+7)n)}_{\text{Block 1}} + \underbrace{(d+1)}_{\text{Block 2}} + \underbrace{350(s+7)^2(2n+1)}_{\text{Block 3}} + 1 \leq 355d^2(s+7)^2(2n+1)$$

783 parameters, which means we finish the proof of Theorem A.1.

784 C Proof of Proposition B.1

785 The key point of proving Proposition B.1 is the composition architecture of neural networks. To
 786 simplify the proof, we first establish several lemmas for proving Proposition B.1 in Section C.1. Next,
 787 we present the detailed proof of Proposition B.1 in Section C.2 based on the lemmas established in
 788 Section C.1.

789 C.1 Lemmas for proving Proposition B.1

790 **Lemma C.1.** Given any $n, r \in \mathbb{N}^+$ and $\delta \in (0, \frac{1}{C(r,n)})$ with $C(r,n) = \prod_{i=1}^r 2^{n^i}$, there exists
 791 $\phi \in \mathcal{NN}_r\{(12r+68)n\}$ such that

$$792 \quad \phi(x) = \lfloor x \rfloor \quad \text{for any } x \in \bigcup_{\ell=0}^{2^{n^r}-1} [\ell, \ell+1 - C(r,n) \cdot \delta].$$

793 We will prove Lemma C.1 by induction. To simplify the proof, we introduce two lemmas for the base
 794 case and the induction step.

795 First, we introduce the following lemma for the base case of proving Lemma C.1.

796 **Lemma C.2.** Given any $n \in \mathbb{N}^+$ and $\delta \in (0, 1)$, there exists a function ϕ realized by a ReLU network
 797 of width 4 and depth $4n-1$ such that

$$798 \quad \phi(x) = \lfloor x \rfloor \quad \text{for any } x \in \bigcup_{\ell=0}^{2^n-1} [\ell, \ell+1 - \delta].$$

799 *Proof.* Set $\tilde{\delta} = 2^{-n}\delta$ and define

$$800 \quad \phi_0(x) := \frac{\sigma(x-1+\tilde{\delta}) - \sigma(x-1)}{\tilde{\delta}} \quad \text{for } x \in \mathbb{R}.$$

801 Clearly, ϕ_0 can be realized by a one-hidden-layer ReLU network of width 2. Moreover, we have

$$802 \quad \phi_0(x) = \frac{\sigma(x-1+\tilde{\delta}) - \sigma(x-1)}{\tilde{\delta}} = \frac{0-0}{\tilde{\delta}} = 0 \quad \text{if } x \in [0, 1-\tilde{\delta}]$$

803 and

$$804 \quad \phi_0(x) = \frac{\sigma(x-1+\tilde{\delta}) - \sigma(x-1)}{\tilde{\delta}} = \frac{(x-1+\tilde{\delta}) - (x-1)}{\tilde{\delta}} = 1 \quad \text{if } x \in [1, 2-\tilde{\delta}].$$

805 By fixing

$$806 \quad x \in \bigcup_{\ell=0}^{2^n-1} [\ell, \ell+1-\delta] = \bigcup_{\ell=0}^{2^n-1} [\ell, \ell+1-2^n\tilde{\delta}],$$

807 we have $\lfloor x \rfloor \in \{0, 1, \dots, 2^n-1\}$, implying that $\lfloor x \rfloor$ can be represented as

$$808 \quad \lfloor x \rfloor = \sum_{i=0}^{n-1} z_i 2^i \quad \text{for } z_0, z_1, \dots, z_{n-1} \in \{0, 1\}.$$

809 Then, for $j = 0, 1, \dots, n-1$, we have $\sum_{i=0}^j z_i 2^i + 1 \leq z_j 2^j + \sum_{i=0}^{j-1} 2^i + 1 \leq z_j 2^j + 2^j$, implying

$$810 \quad \frac{x - \sum_{i=j+1}^{n-1} z_i 2^i}{2^j} \in \left[\frac{\lfloor x \rfloor - \sum_{i=j+1}^{n-1} z_i 2^i}{2^j}, \frac{\lfloor x \rfloor + 1 - 2^n \tilde{\delta} - \sum_{i=j+1}^{n-1} z_i 2^i}{2^j} \right] = \left[\frac{\sum_{i=0}^j z_i 2^i}{2^j}, \frac{\sum_{i=0}^j z_i 2^i + 1 - 2^n \tilde{\delta}}{2^j} \right] \\ \subseteq \left[\frac{z_j 2^j}{2^j}, \frac{z_j 2^j + 2^j - 2^n \tilde{\delta}}{2^j} \right] \subseteq [z_j, z_j + 1 - \tilde{\delta}].$$

811 It follows that

$$812 \quad \phi_0\left(\frac{x - \sum_{i=j+1}^{n-1} z_i 2^i}{2^j}\right) = z_j \quad \text{for } j = 0, 1, \dots, n-1.$$

813 Therefore, the desired function ϕ can be realized by the network in Figure 16.

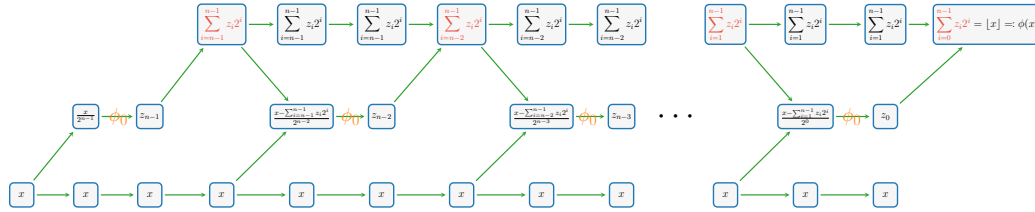


Figure 16: An illustration of the NestNet realizing ϕ . Here, ϕ_0 represent an one-hidden-layer ReLU network of width 2.

814 Clearly,

$$815 \quad \phi(x) = \lfloor x \rfloor \quad \text{for any } x \in \bigcup_{\ell=0}^{2^n-1} [\ell, \ell+1-\delta].$$

816 Moreover, ϕ can be realized by a ReLU network of width $1 + 2 + 1 = 4$ and depth $(1 + 1 + 1) + (1 + 1 + 1)(n-1) = 4n-1$. Hence, we finish the proof of Lemma C.2. \square

818 Next, we introduce the following lemma for the induction step of proving Lemma C.1.

819 **Lemma C.3.** Given any $n, s, \hat{n} \in \mathbb{N}^+$ and $\delta \in (0, \frac{1}{2^{n+s+1}})$, if $g \in \mathcal{NN}_s\{\hat{n}\}$ satisfying

$$820 \quad g(x) = \lfloor x \rfloor \quad \text{for any } x \in \bigcup_{\ell=0}^{2^{n+s}-1} [\ell, \ell+1-\delta].$$

821 Then there exists $\phi \in \mathcal{NN}_{s+1}\{\hat{n} + 12n - 7\}$ such that

$$822 \quad \phi(x) = \lfloor x \rfloor \quad \text{for any } x \in \bigcup_{\ell=0}^{2^{n+s+1}-1} [\ell, \ell+1-2^{n+s+1}\delta].$$

823 *Proof.* By setting $m = 2^{n^s}$, we have $m^n = (2^{n^s})^n = 2^{(n^s)n} = 2^{n^{s+1}}$ and

$$824 \quad g(x) = \lfloor x \rfloor \quad \text{for any } x \in \bigcup_{\ell=0}^{m-1} [\ell, \ell + 1 - \delta]. \quad (13)$$

825 By fixing

$$826 \quad x \in \bigcup_{\ell=0}^{2^{n^{s+1}}-1} [\ell, \ell + 1 - 2^{n^{s+1}}\delta] = \bigcup_{\ell=0}^{m^n-1} [\ell, \ell + 1 - m^n\delta],$$

827 we have $\lfloor x \rfloor \in \{0, 1, \dots, m^n - 1\}$, implying that $\lfloor x \rfloor$ can be represented as

$$828 \quad \lfloor x \rfloor = \sum_{i=0}^{n-1} z_i m^i \quad \text{for } z_0, z_1, \dots, z_{n-1} \in \{0, 1, \dots, m-1\}.$$

829 Then, for $j = 0, 1, \dots, n-1$, we have

$$830 \quad \sum_{i=0}^j z_i m^i + 1 \leq z_j m^j + \sum_{i=0}^{j-1} (m-1)m^i + 1 = z_j m^j + m^j,$$

831 implying

$$\begin{aligned} 832 \quad \frac{x - \sum_{i=j+1}^{n-1} z_i m^i}{m^j} &\in \left[\frac{\lfloor x \rfloor - \sum_{i=j+1}^{n-1} z_i m^i}{m^j}, \frac{\lfloor x \rfloor + 1 - m^n \delta - \sum_{i=j+1}^{n-1} z_i m^i}{m^j} \right] \\ &= \left[\frac{\sum_{i=0}^j z_i m^i}{m^j}, \frac{\sum_{i=0}^j z_i m^i + 1 - m^n \delta}{m^j} \right] \\ &\subseteq \left[\frac{z_j m^j}{m^j}, \frac{z_j m^j + m^j - m^n \delta}{m^j} \right] \subseteq [z_j, z_j + 1 - \delta]. \end{aligned}$$

833 It follows that

$$834 \quad g\left(\frac{x - \sum_{i=j+1}^{n-1} z_i m^i}{m^j}\right) = z_j \quad \text{for } j = 0, 1, \dots, n-1.$$

835 Therefore, the desired function ϕ can be realized by the network in Figure 17.

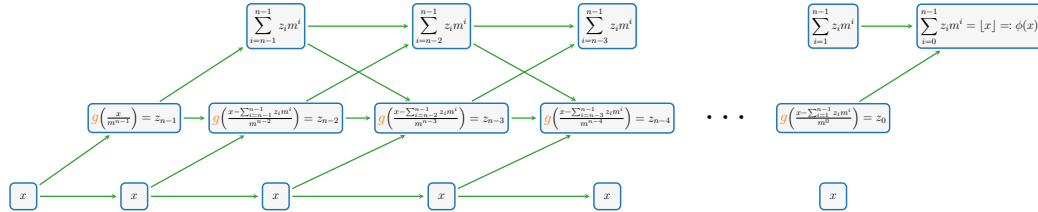


Figure 17: An illustration of the NestNet realizing ϕ . Here, g is regarded as an activation function.

836 Clearly,

$$837 \quad \phi(x) = \lfloor x \rfloor \quad \text{for any } x \in \bigcup_{\ell=0}^{m^n-1} [\ell, \ell + 1 - m^n\delta] = \bigcup_{\ell=0}^{2^{n^{s+1}}-1} [\ell, \ell + 1 - 2^{n^{s+1}}\delta].$$

838 Moreover, the fact $g \in \mathcal{NN}_s\{\widehat{n}\}$ implies that ϕ can be realized by a height- $(s+1)$ NestNet with at
839 most

$$840 \quad \underbrace{(1+1)2 + (2+1)3 + (3+1)3(n-2) + (3+1)}_{\text{outer network}} + \underbrace{\widehat{n}}_g = \widehat{n} + 12n - 7$$

841 parameters. Hence, we finish the proof of Lemma C.3. \square

842 With Lemmas C.2 and C.3 in hand, we are ready to prove Lemma C.1.

843 *Proof of Lemma C.1.* We will use the mathematical induction to prove Lemma C.1. First, we consider
844 the base case $r = 1$. By Lemma C.2, there exists a function ϕ realized by a ReLU network of width 4
845 and depth $4n - 1$ such that

$$846 \quad \phi(x) = \lfloor x \rfloor \quad \text{for any } x \in \bigcup_{\ell=0}^{2^n-1} [\ell, \ell + 1 - \delta] \subseteq \bigcup_{\ell=0}^{2^n-1} [\ell, \ell + 1 - C(r, n) \cdot \delta] \text{ with } r = 1.$$

Moreover, the network realizing ϕ has at most $(4+1)4((4n-1)+1) = 80n$ parameters, implying $\phi \in \mathcal{NN}_1\{80n\} \subseteq \mathcal{NN}_1\{(12r+68)n\}$ for $r = 1$. Thus, the base case $r = 1$ is proved.

Next, assume Lemma C.1 holds for $r = s \in \mathbb{N}^+$. We need to show it is also true for $r = s+1$. By the induction hypothesis, there exists $g \in \mathcal{NN}_s\{(12s+68)n\}$ such that

$$g(x) = \lfloor x \rfloor \quad \text{for any } x \in \bigcup_{\ell=0}^{2^{n^s}-1} [\ell, \ell+1 - C(s, n) \cdot \delta].$$

By Lemma C.3 with $\widehat{n} = (12s+68)n$ therein and setting $\widehat{\delta} = C(s, n) \cdot \delta$, there exists

$$\phi \in \mathcal{NN}_{s+1}\{\widehat{n} + 12n - 7\} \subseteq \mathcal{NN}_{s+1}\{(12s+68)n + 12n - 7\} \subseteq \mathcal{NN}_{s+1}\{(12(s+1)+68)n\}$$

such that

$$\phi(x) = \lfloor x \rfloor \quad \text{for any } x \in \bigcup_{\ell=0}^{2^{n^{s+1}}-1} [\ell, \ell+1 - 2^{n^{s+1}}\widehat{\delta}].$$

Observe that

$$2^{n^{s+1}}\widehat{\delta} = 2^{n^{s+1}}C(s, n) \cdot \delta = 2^{n^{s+1}}\left(\prod_{i=1}^s 2^{n^i}\right) \cdot \delta = \left(\prod_{i=1}^{s+1} 2^{n^i}\right) \cdot \delta = C(s+1, n) \cdot \delta.$$

It follows that

$$\phi(x) = \lfloor x \rfloor \quad \text{for any } x \in \bigcup_{\ell=0}^{2^{n^{s+1}}-1} [\ell, \ell+1 - C(s+1, n) \cdot \delta].$$

Thus, Lemma C.1 is proved for the case $r = s+1$, which means we finish the induction step. Hence, by the principle of induction, we complete the proof of Lemma C.1. \square

C.2 Detailed proof of Proposition B.1

Set $C(r, n) = \prod_{i=1}^r 2^{n^i}$ and $\widetilde{\delta} = \frac{\delta}{C(r, n)} \in (0, \frac{1}{C(r, n)})$. By Lemma C.1, there exists $\phi_0 \in \mathcal{NN}_r\{(12r+68)n\}$ such that

$$\phi_0(x) = \lfloor x \rfloor \quad \text{for any } x \in \bigcup_{\ell=0}^{2^{n^r}-1} [\ell, \ell+1 - C(r, n) \cdot \widetilde{\delta}] = \bigcup_{\ell=0}^{2^{n^r}-1} [\ell, \ell+1 - \delta].$$

It follows from $J \leq 2^{n^r}$ that

$$\phi_0(x) = \lfloor x \rfloor \quad \text{for any } x \in \bigcup_{j=0}^{J-1} [j, j+1 - \delta].$$

Set

$$\widetilde{M} = \max_{x \in [J, J+1]} |\phi_0(x)| \quad \text{and} \quad M = \frac{\widetilde{M} + J}{\delta}.$$

Then, for any $x \in [J, J+1]$, we have

$$\phi_0(x) + M\sigma(x - (J - \delta)) \geq -\widetilde{M} + M\delta = -\widetilde{M} + (\widetilde{M} + J) = J,$$

implying

$$\min \left\{ \phi_0(x) + M\sigma(x - (J - \delta)), J \right\} = J.$$

Moreover, for any $x \in \bigcup_{j=0}^{J-1} [j, j+1 - \delta]$, we have $\sigma(x - (J - \delta)) = 0$, implying

$$\min \left\{ \phi_0(x) + M\sigma(x - (J - \delta)), J \right\} = \min \left\{ \phi_0(x), J \right\} = \min \left\{ \lfloor x \rfloor, J \right\} = \lfloor x \rfloor.$$

Therefore, by defining

$$\phi(x) := \min \left\{ \phi_0(x) + M\sigma(x - (J - \delta)), J \right\} \quad \text{for any } x \in \bigcup_{j=0}^J [j, j+1 - \delta \cdot \mathbf{1}_{\{j \leq J-1\}}],$$

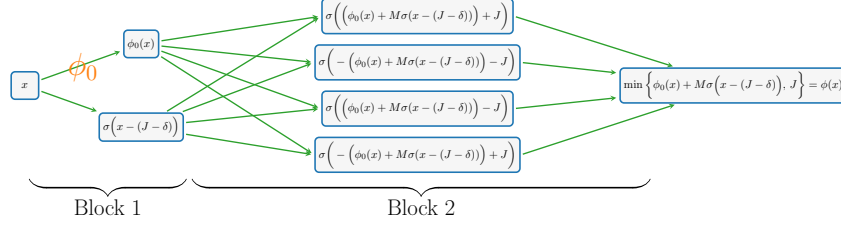


Figure 18: An illustration of the network realizing ϕ for any $x \in \bigcup_{j=0}^J [j, j+1 - \delta \cdot \mathbf{1}_{\{j \leq J-1\}}]$ based on the fact $\min\{a, b\} = \frac{1}{2}(\sigma(a+b) - \sigma(-a-b) - \sigma(a-b) - \sigma(-a+b))$.

we have

$$\phi(x) = \lfloor x \rfloor \quad \text{for any } x \in \bigcup_{j=0}^{J-1} [j, j+1 - \delta]$$

and

$$\phi(x) = J \quad \text{for any } x \in [J, J+1].$$

Moreover, ϕ can be realized by the network in Figure 18. The fact $\phi_0 \in \mathcal{NN}_r\{(12r+68)n\}$ implies that ϕ can be realized by a height- r NestNet with at most

$$\underbrace{3((12r+68)n)}_{\text{Block 1}} + \underbrace{(2+1)4 + (4+1)}_{\text{Block 2}} \leq 36(r+7)n$$

parameters. So we finish the proof of Proposition B.1.

D Proof of Proposition B.2

The key idea of proving Proposition B.2 is the bit extraction technique proposed in [6]. First, we establish several lemmas for proving Proposition B.2 and give their proofs in Section D.1 except for Lemma D.2, the proof of which is placed in Section D.3 since it is complicated. Next, we present the detailed proof of Proposition B.2 in Section D.2 based on the lemmas established in Section D.1.

D.1 Lemmas for proving Proposition B.2

To simplify the proof of Proposition B.2, we establish several lemmas as the intermediate step. We first establish a lemma to show that any continuous piecewise linear functions on \mathbb{R} can be realized by one-hidden-layer ReLU networks.

Lemma D.1. *Given any $p \in \mathbb{N}^+$, any continuous piecewise linear function on \mathbb{R} with at most p breakpoints can be realized by a one-hidden-layer ReLU network of width $p+1$.*

Proof. We will use the mathematical induction to prove Lemma D.1. First, we consider the base case $p = 1$. Suppose $f : \mathbb{R} \rightarrow \mathbb{R}$ is a continuous piecewise linear function on \mathbb{R} with at most $p = 1$ breakpoints. Then there exist $a_1, a_2, x_0 \in \mathbb{R}$ such that

$$f(x) = \begin{cases} a_1(x - x_0) + f(x_0) & \text{if } x \geq x_0 \\ a_2(x_0 - x) + f(x_0) & \text{if } x < x_0. \end{cases}$$

Thus, $f(x) = a_1\sigma(x - x_0) + a_2\sigma(x_0 - x) + f(x_0)$ for any $x \in \mathbb{R}$, implying f can be realized by a one-hidden-layer ReLU network of width $2 = p+1$ for $p = 1$. Hence, Lemma D.1 is proved for the case $p = 1$.

Now, assume Lemma D.1 holds for $p = k \in \mathbb{N}^+$, we would like to show it is also true for $p = k+1$. Suppose $f : \mathbb{R} \rightarrow \mathbb{R}$ is a continuous piecewise linear function on \mathbb{R} with at most $k+1$ breakpoints. We may assume the biggest breakpoint of f is x_0 since it is trivial for the case that f has no breakpoint. Denote the slopes of the linear pieces left and right next to x_0 by a_1 and a_2 , respectively. Define

$$\tilde{f}(x) := f(x) - (a_2 - a_1)\sigma(x - x_0) \quad \text{for any } x \in \mathbb{R}.$$

909 Then \tilde{f} has at most k breakpoints. By the induction hypothesis, \tilde{f} can be realized by a one-hidden-
 910 layer ReLU network of width $k + 1$. Thus, there exist $w_{0,j}, b_{0,j}, w_{1,j}, b_1$ for $j = 1, 2, \dots, k + 1$ such
 911 that

$$912 \quad \tilde{f}(x) = \sum_{j=1}^{k+1} w_{1,j} \sigma(w_{0,j}x + b_{0,j}) + b_1 \quad \text{for any } x \in \mathbb{R}.$$

913 Therefore, for any $x \in \mathbb{R}$, we have

$$914 \quad f(x) = (a_2 - a_1)\sigma(x - x_0) + \tilde{f}(x) = (a_2 - a_1)\sigma(x - x_0) + \sum_{j=1}^{k+1} w_{1,j} \sigma(w_{0,j}x + b_{0,j}) + b_1,$$

915 implying f can be realized by a one-hidden-layer ReLU network of width $k + 2 = (k + 1) + 1 = p + 1$
 916 for $p = k + 1$. Thus, we finish the induction process. Therefore, by the principle of induction, we
 917 complete the proof of Lemma D.1. \square

918 Next, we establish a lemma to extract the sum of n^s bits via a height- s NestNet with $\mathcal{O}(n)$ parameters.

919 **Lemma D.2.** *Given any $n, s \in \mathbb{N}^+$, there exists $\phi \in \mathcal{NN}_s\{57(s + 7)^2(n + 1)\}$ such that: For any
 920 $\theta_1, \theta_2, \dots, \theta_{n^s} \in \{0, 1\}$, we have*

$$921 \quad \phi(k + \text{bin } 0.\theta_1\theta_2\cdots\theta_{n^s}) = \sum_{\ell=1}^k \theta_\ell \quad \text{for } k = 0, 1, \dots, n^s. \quad (14)$$

922 The proof of Lemma D.2 is complicated and hence is placed in Section D.3. Then, based on
 923 Lemma D.2, we establish a new lemma, Lemma D.3 below, which is a key intermediate conclusion
 924 to prove Proposition B.2.

925 **Lemma D.3.** *Given any $n, s \in \mathbb{N}^+$ and $\theta_{i,\ell} \in \{0, 1\}$ for $i = 0, 1, \dots, n - 1$ and $\ell = 0, 1, \dots, m - 1$, where
 926 $m = n^s$, there exists $\phi \in \mathcal{NN}_s\{58(s + 7)^2(n + 1)\}$ such that*

$$927 \quad \phi(j) = \sum_{\ell=0}^k \theta_{i,\ell} \quad \text{for } j = 0, 1, \dots, nm - 1,$$

928 where (i, k) is the unique index pair satisfying $j = im + k$ with $i \in \{0, 1, \dots, n - 1\}$ and $k \in$
 929 $\{0, 1, \dots, m - 1\}$.

930 *Proof.* We first construct a network to extract the unique index pair (i, k) from $j \in \{0, 1, \dots, nm - 1\}$
 931 with the following condition

$$932 \quad j = im + k \quad \text{with } i \in \{0, 1, \dots, n - 1\} \text{ and } k \in \{0, 1, \dots, m - 1\}.$$

933 There exists a continuous piecewise linear function ϕ_1 with $2n$ breakpoints such that

$$934 \quad \phi_1(x) = \lfloor x \rfloor \quad \text{for any } x \in \bigcup_{\ell=0}^{n-1} [\ell, \ell + 1 - \delta] \text{ with } \delta = \frac{1}{2m}.$$

935 By Lemma D.1, ϕ_1 can be realized by a one-hidden-layer ReLU network of width $2n + 1$. Moreover,
 936 for any $j \in \{0, 1, \dots, nm - 1\}$, we have

$$937 \quad \phi_1\left(\frac{j}{m}\right) = \lfloor \frac{j}{m} \rfloor = i \quad \text{and} \quad j - m\phi_1\left(\frac{j}{m}\right) = j - mi = k,$$

938 where (i, k) is the unique index pair satisfying $j = im + k$ with $i \in \{0, 1, \dots, n - 1\}$ and $k \in$
 939 $\{0, 1, \dots, m - 1\}$. By defining

$$940 \quad \Phi_1(x) := \begin{bmatrix} \phi_1\left(\frac{x}{m}\right) \\ x - m\phi_1\left(\frac{x}{m}\right) \end{bmatrix} \quad \text{for any } x \geq 0,$$

941 we have

$$942 \quad \Phi_1(j) = \begin{bmatrix} \phi_1\left(\frac{j}{m}\right) \\ j - m\phi_1\left(\frac{j}{m}\right) \end{bmatrix} = \begin{bmatrix} i \\ k \end{bmatrix} \quad \text{for } j = 0, 1, \dots, nm - 1,$$

943 where (i, k) is the unique index pair satisfying $j = im + k$ with $i \in \{0, 1, \dots, n - 1\}$ and $k \in \{0, 1, \dots, m -$
 944 $1\}$. Moreover, Φ_1 can be realized by a one-hidden-layer ReLU network of width $2(2n + 1) + 1 = 4n + 3$.

945 Hence, the network realizing Φ_1 has at most $(1 + 1)(4n + 3) + ((4n + 3) + 1)2 = 16n + 14$ parameters.

946 Define

$$947 \quad z_i := \text{bin} 0.\theta_{i,0}\theta_{i,1}\cdots\theta_{i,m-1} \quad \text{for } i = 0, 1, \dots, n-1.$$

948 There exists a continuous piecewise linear function $\tilde{\phi}_2$ with n breakpoints such that

$$949 \quad \tilde{\phi}_2(i) = z_i \quad \text{for } i = 0, 1, \dots, n-1.$$

950 By Lemma D.1, $\tilde{\phi}_2$ can be realized by a one-hidden-layer ReLU network of width $n+1$.

951 By Lemma D.2, there exists $\phi_3 \in \mathcal{NN}_s\{57(s+7)^2(n+1)\}$ such that: For any $\xi_1, \xi_2, \dots, \xi_{n^s} \in \{0, 1\}$,
952 we have

$$953 \quad \phi_3(k + \text{bin} 0.\xi_1\xi_2\cdots\xi_{n^s}) = \sum_{\ell=1}^k \xi_\ell \quad \text{for } k = 1, 2, \dots, n^s.$$

954 It follows from $m = n^s$ that, for any $\xi_0, \xi_1, \dots, \xi_{m-1} \in \{0, 1\}$, we have

$$955 \quad \phi_3(k + \text{bin} 0.\xi_0\xi_1\cdots\xi_{m-1}) = \sum_{\ell=1}^k \xi_{\ell-1} = \sum_{\ell=0}^{k-1} \xi_\ell \quad \text{for } k = 1, 2, \dots, m,$$

956 implying

$$957 \quad \phi_3(k + 1 + \text{bin} 0.\xi_0\xi_1\cdots\xi_{m-1}) = \sum_{\ell=0}^k \xi_\ell \quad \text{for } k = 0, 1, \dots, m-1.$$

958 Then, for $i = 0, 1, \dots, n-1$ and $k = 0, 1, \dots, m-1$, we have

$$959 \quad \phi_3(k + 1 + \tilde{\phi}_2(i)) = \phi_2(k + 1 + z_i) = \phi_3(k + 1 + \text{bin} 0.\theta_{i,0}\theta_{i,1}\cdots\theta_{i,m-1}) = \sum_{\ell=0}^k \theta_{i,\ell}.$$

960 By defining

$$961 \quad \phi_2(x, y) := y + 1 + \tilde{\phi}_2(x) \quad \text{for any } x, y \in [0, \infty)$$

962 and $\phi := \phi_3 \circ \phi_2 \circ \Phi_1$, we have

$$963 \quad \phi(j) = \phi_3 \circ \phi_2 \circ \Phi_1(j) = \phi_3 \circ \phi_2(i, k) = \phi_3(k + 1 + \tilde{\phi}_2(i)) = \sum_{\ell=0}^k \theta_{i,\ell}$$

964 for $j = 0, 1, \dots, nm-1$, where (i, k) is the unique index pair satisfying $j = im + k$ with $i \in$
965 $\{0, 1, \dots, n-1\}$ and $k \in \{0, 1, \dots, m-1\}$.

966 It remains to estimate the number of parameters in the NestNet realizing $\phi = \phi_3 \circ \phi_2 \circ \Phi_1$. Observe
967 that ϕ_2 can be realized by a one-hidden-layer ReLU network of width $(n+1) + 1 = n+2$. Then, the
968 network realizing ϕ_2 has at most $(2+1)(n+2) + ((n+2)+1) = 4n+9$ parameters. Therefore, ϕ
969 can be realized by a height- s NestNet with at most

$$970 \quad \underbrace{(16n+14)}_{\Phi_1} + \underbrace{(4n+9)}_{\phi_2} + \underbrace{57(s+7)^2(n+1)}_{\phi_3} \leq 58(s+7)^2(n+1)$$

971 parameters, which means we complete the proof of Lemma D.3. \square

972 D.2 Detailed proof of Proposition B.2

973 We may assume $J = mn = n^{s+1}$ with $m = n^s$ since we can set $y_{J-1} = y_J = \dots = y_{mn-1}$ if $J < mn$.

974 Define

$$975 \quad a_j := \lfloor y_j / \varepsilon \rfloor \quad \text{for } j = 0, 1, \dots, nm-1.$$

976 Our goal is to construct a function ϕ such that $\phi(j) = a_j \varepsilon$ for $j = 0, 1, \dots, nm-1$.

977 For $i = 0, 1, \dots, n-1$, we define

$$978 \quad b_{i,\ell} = \begin{cases} 0 & \text{for } \ell = 0 \\ a_{im+\ell} - a_{im+\ell-1} & \text{for } \ell = 1, 2, \dots, m-1. \end{cases}$$

979 Since $|y_j - y_{j-1}| \leq \varepsilon$ for all j , we have $|a_j - a_{j-1}| \leq 1$. It follows that $b_{i,\ell} \in \{-1, 0, 1\}$ for $i =$
980 $0, 1, \dots, n-1$ and $\ell = 0, 1, \dots, m-1$. Hence, there exist $c_{i,\ell} \in \{0, 1\}$ and $d_{i,\ell} \in \{0, 1\}$ such that

$$981 \quad b_{i,\ell} = c_{i,\ell} - d_{i,\ell} \quad \text{for } i = 0, 1, \dots, n-1 \text{ and } \ell = 0, 1, \dots, m-1.$$

982 Since any $j \in \{0, 1, \dots, nm - 1\}$ can be uniquely indexed as $j = im + k$ with $i \in \{0, 1, \dots, n - 1\}$ and
 983 $k \in \{0, 1, \dots, m - 1\}$, we have

$$\begin{aligned} a_j = a_{im+k} &= a_{im} + \sum_{\ell=1}^k (a_{im+\ell} - a_{im+\ell-1}) = a_{im} + \sum_{\ell=1}^k b_{i,\ell} = a_{im} + \sum_{\ell=0}^k b_{i,\ell} \\ &= a_{im} + \sum_{\ell=0}^k c_{i,\ell} - \sum_{\ell=0}^k d_{i,\ell}. \end{aligned}$$

985 There exists a continuous piecewise linear function ϕ_1 with $2n$ breakpoints such that

$$986 \quad \phi_1(x) = a_{im} \quad \text{for any } x \in [im, im + m - 1] \text{ and } i = 0, 1, \dots, n - 1.$$

987 Then, we have

$$988 \quad \phi_1(j) = a_{im} \quad \text{for } j = 0, 1, \dots, nm - 1,$$

989 where (i, k) is the unique index pair satisfying $j = im + k$ with $i \in \{0, 1, \dots, n - 1\}$ and $k \in$
 990 $\{0, 1, \dots, m - 1\}$. By Lemma D.1, ϕ_1 can be realized by a one-hidden-layer ReLU network of width
 991 $2n + 1$.

992 By Lemma D.3, there exist $\phi_2, \phi_3 \in \mathcal{NN}_s\{58(s + 7)^2(n + 1)\}$ such that

$$993 \quad \phi_2(j) = \sum_{\ell=0}^k c_{i,\ell} \quad \text{and} \quad \phi_3(j) = \sum_{\ell=0}^k d_{i,\ell} \quad \text{for } j = 0, 1, \dots, nm - 1,$$

994 where (i, k) is the unique index pair satisfying $j = im + k$ with $i \in \{0, 1, \dots, n - 1\}$ and $k \in$
 995 $\{0, 1, \dots, m - 1\}$.

996 Hence, by indexing $j \in \{0, 1, \dots, nm - 1\}$ as $j = im + k$ for $i \in \{0, 1, \dots, n - 1\}$ and $k \in \{0, 1, \dots, m - 1\}$,
 997 we have

$$998 \quad a_j = a_{im} + \sum_{\ell=0}^k c_{i,\ell} - \sum_{\ell=0}^k d_{i,\ell} = \phi_1(j) + \phi_2(j) - \phi_3(j).$$

999 By defining

$$1000 \quad \tilde{\phi}(x) := (\phi_1(x) + \phi_2(x) + \phi_3(x))\varepsilon \quad \text{for any } x \in \mathbb{R},$$

1001 we have $\tilde{\phi}(j) = a_j\varepsilon$ for $j = 0, 1, \dots, nm - 1$ and $\tilde{\phi}$ can be realized by the height- s NestNet in Figure 19.

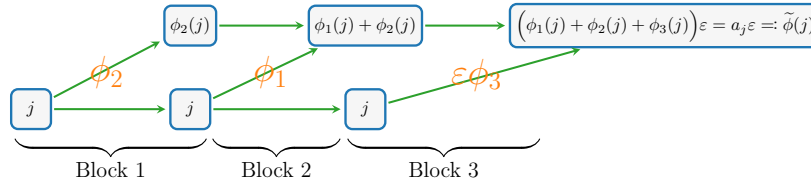


Figure 19: An illustration of the NestNet realizing $\tilde{\phi}$ for $j = 0, 1, \dots, J - 1$.

1002 In Figure 19, Block 1 or 3 has at most

$$1003 \quad 3(58(s + 7)^2(n + 1)) = 174(s + 7)^2(n + 1)$$

1004 parameters; Block 2 is of width $(2n + 1) + 2 = 2n + 3$ and depth 1, and hence has at most

$$1005 \quad (2 + 1)(2n + 3) + ((2n + 3) + 1)2 = 10n + 17$$

1006 parameters. Then, $\tilde{\phi}$ can be realized by a height- s ReLU NestNet with at most

$$1007 \quad 2(174(s + 7)^2(n + 1)) + 10n + 17 = 349(s + 7)^2(n + 1)$$

1008 parameters. Note that $\tilde{\phi}$ may not be bounded. Thus, we define

$$1009 \quad \psi(x) := \min\{\sigma(x), M\} \quad \text{for any } x \in \mathbb{R},$$

1010 where

$$1011 \quad M = \max\{y_j : j = 0, 1, \dots, nm - 1\}.$$

1012 Then, the desired function ϕ can be define via $\phi := \psi \circ \tilde{\phi}$. Clearly,

$$1013 \quad 0 \leq \phi(x) \leq M = \max\{y_j : j = 0, 1, \dots, J - 1\} \quad \text{for any } x \in \mathbb{R}.$$

1014 It follows from $0 \leq a_j \varepsilon = \lfloor y_j / \varepsilon \rfloor \varepsilon \leq y_j \leq M$ for $j = 0, 1, \dots, J - 1$ that

$$1015 \quad \phi(j) = \psi \circ \tilde{\phi}(j) = \psi(a_j \varepsilon) = \min\{\sigma(a_j \varepsilon), M\} = a_j \varepsilon,$$

1016 implying

$$1017 \quad |\phi(j) - y_j| = |a_j \varepsilon - y_j| = \left| \lfloor y_j / \varepsilon \rfloor \varepsilon - y_j \right| = \left| \lfloor y_j / \varepsilon \rfloor - y_j / \varepsilon \right| \varepsilon \leq \varepsilon.$$

1018 It remains to show that ϕ can be realized by a height- s ReLU NestNet with the desired size. Clearly,
1019 ψ can be realized by the network in Figure 20, which is of width 4 and depth 2.

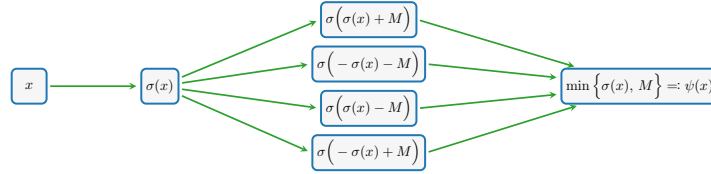


Figure 20: An illustration of the network realizing ψ based on the fact $\min\{a, b\} = \frac{1}{2}(\sigma(a + b) - \sigma(-a - b) - \sigma(a - b) - \sigma(-a + b))$.

1020 Therefore, ϕ can be realized by a height- s ReLU NestNet with at most

$$1021 \quad 349(s + 7)^2(n + 1) + (4 + 1)4(2 + 1) \leq 350(s + 7)^2(n + 1)$$

1022 parameters. Hence, we finish the proof of Proposition B.2.

1023 D.3 Proof of Lemma D.2 for Proposition B.2

1024 We will use the mathematical induction to prove Lemma D.2. To this end, we introduce two lemmas
1025 for the base case and the induction step.

1026 **Lemma D.4.** *Given any $n \in \mathbb{N}^+$, there exists a function ϕ realized by a ReLU network with $128n + 294$*
1027 *parameters such that: For any $\theta_1, \theta_2, \dots, \theta_n \in \{0, 1\}$, we have*

$$1028 \quad \phi(k + \text{bin } 0.\theta_1\theta_2\cdots\theta_n) = \sum_{\ell=1}^k \theta_\ell \quad \text{for } k = 0, 1, \dots, n. \quad (15)$$

1029 **Lemma D.5.** *Given any $n, r, \widehat{n} \in \mathbb{N}^+$, if $g \in \mathcal{NN}_r\{\widehat{n}\}$ satisfying*

$$1030 \quad g(p + \text{bin } 0.\xi_1\xi_2\cdots\xi_{n^r}) = \sum_{j=1}^p \xi_j \quad \text{for any } \xi_1, \xi_2, \dots, \xi_{n^r} \in \{0, 1\} \text{ and } p = 0, 1, \dots, n^r, \quad (16)$$

1031 *then there exists $\phi \in \mathcal{NN}_{r+1}\{\widehat{n} + 114(r + 7)(n + 1)\}$ such that: For any $\theta_1, \theta_2, \dots, \theta_{n^{r+1}} \in \{0, 1\}$, we*
1032 *have*

$$1033 \quad \phi(k + \text{bin } 0.\theta_1\theta_2\cdots\theta_{n^{r+1}}) = \sum_{\ell=1}^k \theta_\ell \quad \text{for } k = 0, 1, \dots, n^{r+1}.$$

1034 The proofs of Lemmas D.4 and D.5 can be found in Sections D.3.1 and D.3.2, respectively. We
1035 remark that the function ϕ in Lemma D.5 is independent of $\theta_1, \theta_2, \dots, \theta_{nm}$. The proof of Lemma D.2
1036 mainly relies on Lemma D.4 and repeated applications of Lemma D.5. The details can be found
1037 below.

1038 *Proof of Lemma D.2.* We will use the mathematical induction to prove Lemma D.2. First, let us
 1039 consider the base case $s = 1$. By Lemma D.4, there exists a function realized by a ReLU network
 1040 with $128n + 294$ parameters such that: For any $\theta_1, \theta_2, \dots, \theta_n \in \{0, 1\}$, we have

$$1041 \quad \phi(k + \text{bin } 0.\theta_1\theta_2\cdots\theta_n) = \sum_{\ell=1}^k \theta_\ell \quad \text{for } k = 0, 1, \dots, n.$$

1042 That means Equation (14) holds for $s = 1$. Moreover, ϕ can also be regarded as a height-1 ReLU
 1043 NestNet with $128n + 294 \leq 57(s + 7)^2(n + 1)$ parameters for $s = 1$, which means Lemma D.2 is
 1044 proved for the case $s = 1$.

1045 Next, assume Lemma D.2 holds for $s = r \in \mathbb{N}^+$. We need to show that it is also true for $s = r + 1$ by
 1046 applying Lemma D.5. By the induction hypothesis, there exists

$$1047 \quad g \in \mathcal{NN}_r \{57(r + 7)^2(n + 1)\}$$

1048 such that: For any $\xi_1, \xi_2, \dots, \xi_{n^r} \in \{0, 1\}$, we have

$$1049 \quad g(k + \text{bin } 0.\xi_1\xi_2\cdots\xi_{n^r}) = \sum_{\ell=1}^k \theta_\ell \quad \text{for } k = 0, 1, \dots, n^r.$$

1050 It follows from $m = n^r$ that

$$1051 \quad g(p + \text{bin } 0.\xi_1\xi_2\cdots\xi_m) = \sum_{j=1}^p \xi_j \quad \text{for any } \xi_1, \xi_2, \dots, \xi_m \in \{0, 1\} \text{ and } p = 0, 1, \dots, m,$$

1052 which means g satisfies Equation (16). Then, by Lemma D.5 with $m = n^r$ and $\widehat{n} = 57(r + 7)^2(n + 1)$
 1053 therein, there exists

$$1054 \quad \phi \in \mathcal{NN}_{r+1} \{\widehat{n} + 114(r + 7)(n + 1)\}$$

1055 such that: For any $\theta_1, \theta_2, \dots, \theta_{nm} \in \{0, 1\}$, we have

$$1056 \quad \phi(k + \text{bin } 0.\theta_1\theta_2\cdots\theta_{nm}) = \sum_{\ell=1}^k \theta_\ell \quad \text{for } k = 0, 1, \dots, nm.$$

1057 It follows from $m = n^r$ that, for any $\theta_1, \theta_2, \dots, \theta_{n^{r+1}} \in \{0, 1\}$, we have

$$1058 \quad \phi(k + \text{bin } 0.\theta_1\theta_2\cdots\theta_{n^{r+1}}) = \sum_{\ell=1}^k \theta_\ell \quad \text{for } k = 0, 1, \dots, n^{r+1},$$

1059 which means Equation (14) holds for $s = r + 1$. Moreover, we have

$$\begin{aligned} \widehat{n} + 114(r + 7)(n + 1) &= 57(r + 7)^2(n + 1) + 114(r + 7)(n + 1) \\ &= 57(n + 1)((r + 7)^2 + 2(r + 7)) \\ &\leq 57(n + 1)((r + 7) + 1)^2 = 57((r + 1) + 7)^2(n + 1). \end{aligned}$$

1061 This implies that

$$1062 \quad \phi \in \mathcal{NN}_{r+1} \{\widehat{n} + 114(r + 7)(n + 1)\} \subseteq \mathcal{NN}_{r+1} \{57((r + 1) + 7)^2(n + 1)\}.$$

1063 Thus, we prove Lemma D.2 for the case $s = r + 1$, which means we finish the induction step. Hence,
 1064 by the principle of induction, we complete the proof of Lemma D.2. \square

1065 D.3.1 Proof of Lemma D.4 for Lemma D.2

1066 To simplify the proof of Lemma D.4, we introduce the following lemma.

1067 **Lemma D.6.** *Given any $n \in \mathbb{N}^+$, there exists a function ϕ realized by a ReLU network of width 7 and*
 1068 *depth $2n + 1$ such that: For any $\theta_1, \theta_2, \dots, \theta_n \in \{0, 1\}$, we have*

$$1069 \quad \phi(\text{bin } 0.\theta_1\theta_2\cdots\theta_n, k) = \sum_{\ell=1}^k \theta_\ell \quad \text{for } k = 0, 1, \dots, n.$$

Lemma D.6 is the Lemma 3.5 of [35]. The detailed proof can be found therein. With Lemma D.6 in hand, we are ready to prove Lemma D.4.

Proof of Lemma D.4. By Lemma D.6, there exists a function ϕ_0 realized by a ReLU network of width 7 and depth $2n + 1$ such that: For any $\theta_1, \theta_2, \dots, \theta_n \in \{0, 1\}$, we have

$$\phi_0(\text{bin } 0.\theta_1\theta_2\cdots\theta_n, k) = \sum_{\ell=1}^k \theta_\ell \quad \text{for } k = 1, 2, \dots, n.$$

The equation above is not true for $k = 0$. We will construct ϕ_2 such that

$$\phi_2(\text{bin } 0.\theta_1\theta_2\cdots\theta_n, k) = \sum_{\ell=1}^k \theta_\ell \quad \text{for } k = 0, 1, \dots, n.$$

To this end, we first set

$$M = \max \{|\phi_0(x, y)| : x \in [0, 1], y \in [0, n]\}$$

and define

$$\phi_1(x, y) := \min \{M + \phi_0(x, y), 2My\} \quad \text{for any } x \in [0, 1] \text{ and } y \in [0, n].$$

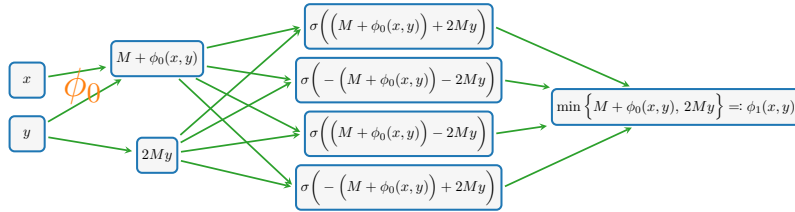


Figure 21: An illustration of the network realizing ϕ_1 for any $x \in [0, 1]$ and $y \in [0, n]$ based on the fact $\min\{a, b\} = \frac{1}{2}(\sigma(a + b) - \sigma(-a - b) - \sigma(a - b) - \sigma(-a + b))$.

As we can see from Figure 21, ϕ_1 can be realized by a ReLU network of width $\max\{7, 4\} = 7$ and depth $(2n + 1) + 2 = 2n + 3$. Moreover, we have

$$\begin{aligned} \phi_1(\text{bin } 0.\theta_1\theta_2\cdots\theta_n, k) &= \min \{M + \phi_0(\text{bin } 0.\theta_1\theta_2\cdots\theta_n, k), 2Mk\} \\ &= \begin{cases} M + \sum_{\ell=1}^k \theta_\ell & \text{for } k = 1, 2, \dots, n \\ 0 & \text{for } k = 0. \end{cases} \end{aligned}$$

Define

$$\phi_2(x, y) := \sigma(\phi_1(x, y) - M) \quad \text{for any } x \in [0, 1] \text{ and } y \in [0, \infty).$$

Then, ϕ_2 can be realized by a ReLU network of width 7 and depth $(2n + 3) + 1 = 2n + 4$. Moreover, we have

$$\begin{aligned} \phi_2(\text{bin } 0.\theta_1\theta_2\cdots\theta_n, k) &= \sigma(\phi_1(\text{bin } 0.\theta_1\theta_2\cdots\theta_n, k) - M) \\ &= \begin{cases} \sigma(\sum_{\ell=1}^k \theta_\ell) = \sum_{\ell=1}^k \theta_\ell & \text{for } k = 1, 2, \dots, n \\ \sigma(-M) = 0 & \text{for } k = 0. \end{cases} \end{aligned}$$

That is,

$$\phi_2(\text{bin } 0.\theta_1\theta_2\cdots\theta_n, k) = \sum_{\ell=1}^k \theta_\ell \quad \text{for } k = 0, 1, \dots, n.$$

Next, we will construct Ψ to extract k and $\text{bin } 0.\theta_1\theta_2\cdots\theta_n$ from $k + \text{bin } 0.\theta_1\theta_2\cdots\theta_n$. It is easy to construct a continuous piecewise linear function $\psi : \mathbb{R} \rightarrow \mathbb{R}$ with $2n$ breakpoints satisfying

$$\psi(x) = \lfloor x \rfloor \quad \text{for any } x \in \bigcup_{\ell=0}^{n-1} [\ell, \ell + 1 - \delta] \text{ with } \delta = 2^{-n}.$$

1094 By Lemma D.1 with $p = 2n$ therein, ψ can be realized by a one-hidden-layer ReLU network of width
 1095 $2n + 1$. By defining

$$1096 \quad \Psi(x) := \begin{bmatrix} x - \psi(x) \\ \psi(x) \end{bmatrix} = \begin{bmatrix} \sigma(x) - \psi(x) \\ \psi(x) \end{bmatrix} \quad \text{for any } x \in [0, \infty).$$

1097 Then, Ψ can be realized by a one-hidden-layer ReLU network of width $1 + 2(2n + 1) = 4n + 3$. That
 1098 means, the network realizing Ψ has at most

$$1099 \quad (1 + 1)(4n + 3) + ((4n + 3) + 1)2 = 16n + 14$$

1100 parameters. Moreover, for any $\theta_1, \theta_2, \dots, \theta_n \in \{0, 1\}$ and $k = 0, 1, \dots, n$, we have

$$1101 \quad \psi(k + \text{bin } 0.\theta_1\theta_2\cdots\theta_n) = \lfloor k + \text{bin } 0.\theta_1\theta_2\cdots\theta_n \rfloor = k,$$

1102 implying

$$1103 \quad \begin{aligned} \Psi(k + \text{bin } 0.\theta_1\theta_2\cdots\theta_n) &= \begin{bmatrix} k + \text{bin } 0.\theta_1\theta_2\cdots\theta_n - \psi(k + \text{bin } 0.\theta_1\theta_2\cdots\theta_n) \\ \psi(k + \text{bin } 0.\theta_1\theta_2\cdots\theta_n) \end{bmatrix} \\ &= \begin{bmatrix} \text{bin } 0.\theta_1\theta_2\cdots\theta_n \\ k \end{bmatrix}. \end{aligned}$$

1104 Finally, the desired function ϕ can be defined via $\phi := \phi_2 \circ \Psi$. Clearly, the network realizing ϕ_2 is of
 1105 width 7 and depth $2n + 4$, and hence has at most

$$1106 \quad (7 + 1)7((2n + 4) + 1) = 56(2n + 5)$$

1107 parameters, implying ϕ can be realized by a ReLU network with at most

$$1108 \quad 56(2n + 5) + (16n + 14) = 128n + 294$$

1109 parameters. Moreover, for any $\theta_1, \theta_2, \dots, \theta_n \in \{0, 1\}$ and $k = 0, 1, \dots, n$, we have

$$1110 \quad \begin{aligned} \phi(k + \text{bin } 0.\theta_1\theta_2\cdots\theta_n) &= \phi_2 \circ \Psi(k + \text{bin } 0.\theta_1\theta_2\cdots\theta_n) \\ &= \phi_2(\text{bin } 0.\theta_1\theta_2\cdots\theta_n, k) = \sum_{\ell=1}^k \theta_\ell. \end{aligned}$$

1111 Thus, we finish the proof of Lemma D.4. □

1112 D.3.2 Proof of Lemma D.5 for Lemma D.2

1113 The key idea of proving Lemma D.5 is to construct a network with n blocks, each of which extracts
 1114 the sum of n^r bits via g . Then the whole network can extract the sum of n^{r+1} bits as we expect.

1115 To simplify our notation, we set $m = n^r$. Given any nm binary bits $\theta_\ell \in \{0, 1\}$ for $\ell = 1, 2, \dots, nm$,
 1116 we divide these nm bits into n classes according to their indices, where the i -th class is composed
 1117 of m bits $\theta_{im+1}, \dots, \theta_{i(m+m)}$ for $i = 0, 1, \dots, n - 1$. We will show how to extract the m bits of the i -th
 1118 class, stored in $\text{bin } 0.\theta_{im+1}\cdots\theta_{i(m+m)}$.

1119 First, let us show how to construct a network to extract k and $\text{bin } 0.\theta_1\theta_2\cdots\theta_{nm}$ from $k + 0.\theta_1\theta_2\cdots\theta_{nm}$.
 1120 By setting $\tilde{n} = 2n$ and Proposition B.1 with $J = 2^{\tilde{n}^r}$ therein, there exists

$$1121 \quad \tilde{g} \in \mathcal{NN}_r\{36(r + 7)\tilde{n}\} = \mathcal{NN}_r\{36(r + 7)(2n)\} = \mathcal{NN}_r\{72(r + 7)n\}$$

1122 such that

$$1123 \quad \tilde{g}(x) = \lfloor x \rfloor \quad \text{for any } x \in \bigcup_{\ell=0}^{J-1} [\ell, \ell + 1 - \delta].$$

1124 Observe that

$$1125 \quad J - 1 = 2^{\tilde{n}^r} - 1 = 2^{(2n)^r} - 1 \geq 2^{2(n^r)} - 1 = 2^{2m} - 1 = 4^m - 1 \geq m^2 \geq nm.$$

1126 It follows from $\text{bin } 0.\theta_1\theta_2\cdots\theta_{nm} \leq 1 - 2^{-nm} = 1 - \delta$ that

$$1127 \quad k + \text{bin } 0.\theta_1\theta_2\cdots\theta_{nm} \in \bigcup_{\ell=0}^{nm} [\ell, \ell + 1 - \delta] \subseteq \bigcup_{\ell=0}^{J-1} [\ell, \ell + 1 - \delta]$$

1128 for $k = 0, 1, \dots, nm$. Thus, we have

$$1129 \quad \tilde{g}(k + \text{bin} 0.\theta_1\theta_2\cdots\theta_{nm}) = k \quad \text{for } k = 0, 1, \dots, nm. \quad (17)$$

1130 It is easy to verify that

$$1131 \quad 2^m \cdot \text{bin} 0.\theta_{im+1}\cdots\theta_{nm} \in \bigcup_{\ell=0}^{2^m-1} [\ell, \ell + 1 - \delta] \quad \text{for } i = 0, 1, \dots, n-1.$$

1132 Since $2^m - 1 = 2^{n^r} - 1 \leq 2^{(2n)^r} - 1 = J - 1$, we have

$$1133 \quad \tilde{g}(2^m \cdot \text{bin} 0.\theta_{im+1}\cdots\theta_{nm}) = \lfloor 2^m \cdot \text{bin} 0.\theta_{im+1}\cdots\theta_{nm} \rfloor \quad \text{for } i = 0, 1, \dots, n-1.$$

1134 Therefore, for $i = 0, 1, \dots, n-1$, we have

$$1135 \quad \text{bin} 0.\theta_{im+1}\cdots\theta_{im+m} = \frac{\lfloor 2^m \cdot \text{bin} 0.\theta_{im+1}\cdots\theta_{nm} \rfloor}{2^m} = \frac{\tilde{g}(2^m \cdot \text{bin} 0.\theta_{im+1}\cdots\theta_{nm})}{2^m}$$

1136 and

$$1137 \quad \begin{aligned} \text{bin} 0.\theta_{(i+1)m+1}\cdots\theta_{nm} &= 2^m \left(\text{bin} 0.\theta_{im+1}\cdots\theta_{nm} - \text{bin} 0.\theta_{im+1}\cdots\theta_{im+m} \right) \\ &= 2^m \left(\text{bin} 0.\theta_{im+1}\cdots\theta_{nm} - \frac{\tilde{g}(2^m \cdot \text{bin} 0.\theta_{im+1}\cdots\theta_{nm})}{2^m} \right). \end{aligned}$$

1138 By defining

$$1139 \quad \phi_1(x) := \frac{\tilde{g}(2^m x)}{2^m} \quad \text{and} \quad \phi_2(x) := 2^m \left(x - \frac{\tilde{g}(2^m x)}{2^m} \right) = \left(\sigma(x) - \frac{\tilde{g}(2^m x)}{2^m} \right) \quad \text{for } x \geq 0,$$

1140 we have

$$1141 \quad \text{bin} 0.\theta_{im+1}\cdots\theta_{im+m} = \phi_1(\text{bin} 0.\theta_{im+1}\cdots\theta_{nm}) \quad (18)$$

1142 and

$$1143 \quad \text{bin} 0.\theta_{(i+1)m+1}\cdots\theta_{nm} = \phi_2(\text{bin} 0.\theta_{im+1}\cdots\theta_{nm}) \quad (19)$$

1144 for any $i \in \{0, 1, \dots, n-1\}$. Moreover, ϕ_1 can be realized by a one-hidden-layer \tilde{g} -activated network
1145 of width 1; ϕ_2 can be realized by a one-hidden-layer (σ, \tilde{g}) -activated network of width 2.

1146 Define

$$1147 \quad \phi_{3,i}(x) := \min\{\sigma(x - im), m\} \quad \text{for any } x \in \mathbb{R} \text{ and } i = 0, 1, \dots, n-1.$$

1148 For any $k \in \{1, 2, \dots, nm\}$, there exist $k_1 \in \{0, 1, \dots, n-1\}$ and $k_2 \in \{1, 2, \dots, m\}$ such that $k =$
1149 $k_1m + k_2$. Then we have

$$1150 \quad \phi_{3,i}(k) = \min\{\sigma(k - im), m\} = \begin{cases} m & \text{if } i \leq k_1 - 1 \\ k_2 & \text{if } i = k_1 \\ 0 & \text{if } i \geq k_1 + 1. \end{cases} \quad (20)$$

1151 Observe that

$$1152 \quad \begin{aligned} \{1, 2, \dots, k\} &= \{1, 2, \dots, k_1m + k_2\} \\ &= \left(\bigcup_{i=1}^{k_1-1} \{im + j : j = 1, 2, \dots, m\} \right) \cup \{k_1m + j : j = 1, 2, \dots, k_2\}. \end{aligned}$$

1153 It follows that

$$1154 \quad \begin{aligned} \sum_{\ell=1}^k \theta_\ell &= \sum_{\ell=1}^{k_1m+k_2} \theta_\ell = \sum_{i=0}^{k_1-1} \left(\sum_{j=1}^m \theta_{im+j} \right) + \sum_{j=1}^{k_2} \theta_{k_1m+j} + 0 \\ &= \sum_{i=0}^{k_1-1} \left(\sum_{j=1}^m \theta_{im+j} \right) + \sum_{i=k_1}^{k_1} \left(\sum_{j=1}^{k_2} \theta_{im+j} \right) + \sum_{i=k_1+1}^{n-1} \left(\sum_{j=1}^0 \theta_{im+j} \right) \\ &= \sum_{i=0}^{k_1-1} \left(\sum_{j=1}^{\phi_{3,i}(k)} \theta_{im+j} \right) + \sum_{i=k_1}^{k_1} \left(\sum_{j=1}^{\phi_{3,i}(k)} \theta_{im+j} \right) + \sum_{i=k_1+1}^{n-1} \left(\sum_{j=1}^{\phi_{3,i}(k)} \theta_{im+j} \right) \\ &= \sum_{i=0}^{n-1} \left(\sum_{j=1}^{\phi_{3,i}(k)} \theta_{im+j} \right) \end{aligned} \quad (21)$$

for $k \in \{1, 2, \dots, nm\}$, where the second to last equality comes from Equation (20). It is easy to verify that Equation (21) also holds for $k = 0$, i.e.,

$$\sum_{\ell=1}^0 \theta_{\ell} = 0 = \sum_{i=0}^{n-1} \left(\sum_{j=1}^0 \theta_{im+j} \right) = \sum_{i=0}^{n-1} \left(\sum_{j=1}^{\phi_{3,i}(0)} \theta_{im+j} \right).$$

Therefore, we have

$$\sum_{\ell=1}^k \theta_{\ell} = \sum_{i=0}^{n-1} \left(\sum_{j=1}^{\phi_{3,i}(k)} \theta_{im+j} \right) \quad \text{for any } k \in \{0, 1, \dots, nm\}. \quad (22)$$

Fix $i \in \{0, 1, \dots, n-1\}$. By setting $p = \phi_{3,i}(k) \in \{0, 1, \dots, m\}$ and $\xi_j = \theta_{im+j}$ for $j = 1, 2, \dots, m$ in Equation (16), we have

$$g(\phi_{3,i}(k) + \text{bin } 0.\theta_{im+1}\theta_{im+2}\dots\theta_{im+m}) = \sum_{j=1}^{\phi_{3,i}(k)} \theta_{im+j}. \quad (23)$$

With Equations (17), (18), (19), (22), and (23) in hand, we are ready to construct the desired function ϕ , which can be realized by the NestNet in Figure 22. Clearly, we have

$$\phi(k + \text{bin } 0.\theta_1\dots\theta_{nm}) = \sum_{\ell=1}^k \theta_{\ell} \quad \text{for } k = 0, 1, \dots, nm.$$

Note that $nm = n \cdot n^r = n^{r+1}$. Then we have

$$\phi(k + \text{bin } 0.\theta_1\dots\theta_{n^{r+1}}) = \sum_{\ell=1}^k \theta_{\ell} \quad \text{for } k = 0, 1, \dots, n^{r+1}.$$

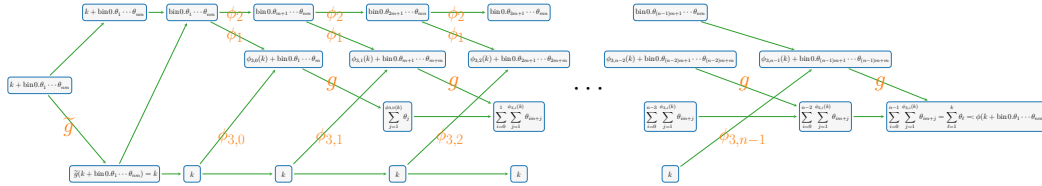


Figure 22: An illustration of the NestNet realizing ϕ based on Equations (17), (18), (19), (22), and (23). Here, g and \tilde{g} are regarded as activation functions.

It remains to estimate the number of parameters in the NestNet realizing ϕ . Recall that ϕ_1 can be realized by a one-hidden-layer \tilde{g} -activated network of width 1 and ϕ_2 can be realized by a one-hidden-layer (σ, \tilde{g}) -activated network of width 2.

Observe that

$$\min\{a, b\} = \frac{1}{2}(\sigma(a+b) - \sigma(-a-b) - \sigma(a-b) - \sigma(-a+b)) \quad \text{for any } a, b \in \mathbb{R}.$$

As we can see from Figure 23, $\phi_{3,i}$ can be realized by a σ -activated network of width 4 and depth 2 for each $i \in \{0, 1, \dots, n-1\}$.

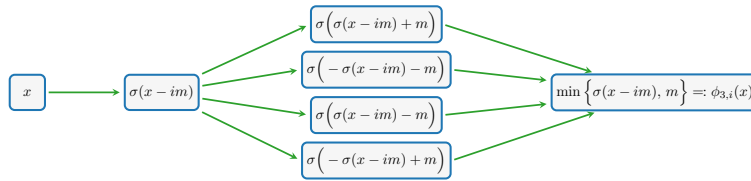


Figure 23: An illustration of $\phi_{3,i}$ for each $i \in \{0, 1, \dots, n-1\}$.

Thus, the network in Figure 22 can be regarded as a (σ, g, \tilde{g}) -activated network of width $2 + 1 + 1 + 1 + 4 + 1 = 10$ and depth $2 + (2 + 1)n = 3n + 2$. Recall that $g \in \mathcal{NN}_r\{\hat{n}\}$ and $\tilde{g} \in \mathcal{NN}_r\{72(r+7)n\}$. This implies that ϕ can be realized by a height- $(r+1)$ NestNet with at most

$$\underbrace{(10+1)10((3n+2)+1)}_{\text{outer network}} + \underbrace{\hat{n}}_g + \underbrace{72(r+7)n}_{\tilde{g}} \leq \hat{n} + 114(r+7)(n+1)$$

parameters, which means we finish the proof of Lemma D.5.