Journal of Statistical Theory and Practice (2022) 16:7
https://doi.org/10.1007/s42519-021-00229-5

ORIGINAL ARTICLE

Function Approximation by Deep Neural Networks with Parameters $\{0, \pm\frac{1}{2}, \pm 1, 2\}$

Aleksandr Beknazaryan[1,2]

Accepted: 24 December 2021 / Published online: 19 January 2022

Abstract
In this paper, it is shown that C_β-smooth functions can be approximated by deep neural networks with ReLU activation function and with parameters $\{0, \pm\frac{1}{2}, \pm 1, 2\}$. The l_0 and l_1 parameter norms of considered networks are thus equivalent. The depth, the width and the number of active parameters of the constructed networks have, up to a logarithmic factor, the same dependence on the approximation error as the networks with parameters in $[-1, 1]$. In particular, this implies that the nonparametric regression estimation with constructed networks achieves, up to logarithmic factors, the same minimax convergence rates as with sparse networks with parameters in $[-1, 1]$.

Keywords Neural networks · Function approximation · Entropy · Nonparametric regression

1 Introduction

The problem of function approximation with neural networks has been of big interest in mathematical research for the last several decades. Various results have been obtained that describe the approximation rates in terms of the structures of the networks and the properties of the approximated functions. One of the most remarkable results in this direction is the universal approximation theorem, which shows that even shallow (but sufficiently wide) networks can approximate continuous functions arbitrarily well (see [9] for the overview and possible proofs of the theorem). Also, in [6] it was shown that integrable functions can be approximated by networks with fixed width. Those networks, however, may need to be very deep to attain small approximation errors.

✉ Aleksandr Beknazaryan
a.beknazaryan@utwente.nl

[1] Department of Applied Mathematics, University of Twente, Drienerlolaan 5, 7522 NB Enschede, The Netherlands

[2] Institute of Environmental and Agricultural Biology (X-BIO), University of Tyumen, Volodarskogo 6, Tyumen, Russia 625003

Yet, from a pragmatic point of view, and, in particular, in statistical applications, allowing very big number of network parameters may be impractical. The reason is that in this case controlling the complexity of approximant networks at an optimal rate becomes problematic. Complexities of classes of neural networks are usually described in terms of their covering numbers and entropies. Those two concepts also play an important role in various branches of statistics, such as regression analysis, density estimation and empirical processes (see, e.g., [1,3,8]). In particular, in regression estimation the following dichotomy usually comes up while selecting the class of functions from which the estimator will be chosen: On the one hand, the selected class of approximants should be "big" enough to be able to approximate various non-trivial functions, and on the other hand it should have "small" enough entropy to attain good learning rates. Thus, the general problem is to obtain powerful classes of functions with well-controllable entropies.

As to the powerfulness of classes of neural networks, it has recently been shown ([10,13]) that with properly chosen architecture the classes of sparse deep neural networks with ReLU activation function can well approximate smooth functions. In particular, it is shown in [13] that C_β-smooth functions on $[0,1]^d$ can be ε-approximated by deep ReLU networks with $O(\varepsilon^{-d/\beta}\log_2(1/\varepsilon))$ active (nonzero) parameters. A similar result for sparse ReLU networks with parameters in $[-1,1]$ has been obtained in [10]. The number of active parameters s in those networks is much smaller than the total number of network parameters, and the network depth L depends logarithmically on the approximation error. Boundedness of parameters of the networks constructed in [10] implies that the $\varepsilon-$entropy of the approximating networks has order $O(sL^2\log_2(1/\varepsilon))$. The main advantages of this entropy bound are its logarithmic dependence on $1/\varepsilon$, which allows to take the covering radius ε to be very small in applications, and its linear dependence on the sparsity s and quadratic dependence on the depth L, both of which, as described above, can also be taken to be small. Using this entropy bound, it is then shown in [10] that if the regression function is a composition of Hölder smooth functions, then sparse neural networks with depth $L \lesssim \log_2 n$ and the number of active parameters $s \sim n^{\frac{t}{2\beta+t}}\log_2 n$, where $\beta > 0$ and $t \geq 1$ depend on the structure and the smoothness of the regression function and attain the minimax optimal prediction error rate $n^{\frac{-2\beta}{2\beta+t}}$ (up to a logarithmic factor), where n is the sample size. It would therefore be desirable to obtain a similar entropy bound for the spaces of networks for which the above l_0 (sparsity) regularization is replaced by the better practically implementable l_1 regularization.

Networks with l_1 norm of all parameters bounded by 1 are considered in [12]. As in those networks, there are at most $1/\varepsilon^2$ parameters outside of the interval $(-\varepsilon^2, \varepsilon^2)$; an entropy bound of order $O((2/L)^{2L-1}/\varepsilon^2)$ has been obtained by taking in the covering networks the remaining parameters to be 0. This bound, however, depends polynomially on $1/\varepsilon$, and it leads to the convergence rate of order $1/\sqrt{n}$ for regression estimation with given n samples. As it is discussed in [12], the rate $1/\sqrt{n}$ is seemingly the best possible for l_1 regularized estimators. Alternative approaches of sparsifying neural networks using derivatives, iterative prunings and clipped l_1 penalties are given in [2,4,5] and [7].

To combine the advantages of both l_0 and l_1 regularizations, as well as to make the networks easier to encode, we consider networks with parameters $\{0, \pm\frac{1}{2}, \pm 1, 2\}$. The l_0 and l_1 parameter regularizations of those networks can differ at most by a factor of 2, which, in particular, allows to employ all the features induced from the sparsity of networks (including their entropy bounds) while imposing l_1 constraints on their parameters. Moreover, discretization of parameters allows to calculate the exact number of networks (the 0–entropy) required to attain a given approximation rate. The latter, in turn, allows to reduce the problem of selection of the estimator of an unknown regression function to a simple and straightforward procedure of minimization over a finite set of candidates. Importantly, the depth, the width and the number of active parameters in the approximant networks are equivalent to those of networks constructed in [10]. Hence, for the considered networks the l_0 parameter regularization can be replaced by the l_1 parameter regularization, leading, up to a logarithmic factor, to the same statistical guarantees as in [10]. In our construction, the parameters ± 1 are used to add/subtract the nodes, change, if necessary, their signs and transfer them to the next layers. The parameters $\pm\frac{1}{2}$ and 2 are used to attain the values of the form $k/2^j \in [-1, 1]$, $j \in \mathbb{N}$, $k \in \mathbb{Z}$, which can get sufficiently close to any number from $[-1, 1]$. Note that this can also be done using only the parameters $\pm\frac{1}{2}$ and 1. The latter, however, would require a larger depth and a bigger number of active nodes.

Notation. The notation $|\mathbf{v}|_\infty$ is used for the l_∞ norm of a vector $\mathbf{v} \in \mathbb{R}^d$ and $\|f\|_{L^\infty[0,1]^d}$ denotes the sup norm of a function f defined on $[0, 1]^d$, $d \in \mathbb{N}$. For $x, y \in \mathbb{R}$, we denote $x \vee y := \max\{x, y\}$ and $(x)_+ := \max\{0, x\}$. Also, to make them multiplicable with preceding matrices, the vectors from $\mathbb{R}^d$, depending on the context, are considered as matrices from $\mathbb{R}^{d\times 1}$ rather than $\mathbb{R}^{1\times d}$.

2 Main Result

Consider the set of neural networks with L hidden layers and with ReLU activation function $\sigma(x) = 0 \vee x = (x)_+$ defined by

$$\mathcal{F}(L, \mathbf{p}) := \{f : [0, 1]^d \to \mathbb{R}^{p_{L+1}} \mid f(\mathbf{x}) = W_L \sigma_{\mathbf{v}_L} W_{L-1} \sigma_{\mathbf{v}_{L-1}} \dots W_1 \sigma_{\mathbf{v}_1} W_0 \mathbf{x}\},$$

where $W_i \in \mathbb{R}^{p_{i+1}\times p_i}$ are weight matrices, $i = 0, \dots, L$, $\mathbf{v}_i \in \mathbb{R}^{p_i}$ are shift vectors, $i = 1, \dots, L$, and $\mathbf{p} = (p_0, p_1, \dots, p_{L+1})$ is the width vector with $p_0 = d$. For a given shift vector $\mathbf{v} = (v_1, \dots, v_p)$ and a given input vector $\mathbf{y} = (y_1, \dots, y_p)$, the action of shifted activation function $\sigma_\mathbf{v}$ on $\mathbf{y}$ is defined as

$$\sigma_\mathbf{v}(\mathbf{y}) = \left(\sigma(y_1 - v_1), \cdots, \sigma(y_p - v_p)\right).$$

It is assumed that the network parameters, that is, the entries of weight matrices W_i and the coordinates of shift vectors $\mathbf{v}_i$, are all in $[-1, 1]$. For $s \in \mathbb{N}$, let $\mathcal{F}(L, \mathbf{p}, s)$ be the subset of $\mathcal{F}(L, \mathbf{p})$ consisting of networks with at most s nonzero parameters. In [10], Theorem 5, the following approximation of β-Hölder continuous functions belonging to the ball

$$\mathcal{C}_d^\beta(K) = \Bigg\{ f : [0,1]^d \to \mathbb{R} : \sum_{0 \le |\boldsymbol{\alpha}| < \beta} \|\partial^{\boldsymbol{\alpha}} f\|_{L^\infty[0,1]^d} + \sum_{|\boldsymbol{\alpha}| = \lfloor \beta \rfloor} \sup_{\substack{\mathbf{x},\mathbf{y} \in [0,1]^d \\ \mathbf{x} \neq \mathbf{y}}} \frac{|\partial^{\boldsymbol{\alpha}} f(\mathbf{x}) - \partial^{\boldsymbol{\alpha}} f(\mathbf{y})|}{|\mathbf{x} - \mathbf{y}|_\infty^{\beta - \lfloor \beta \rfloor}} \le K \Bigg\}$$

with networks from $\mathcal{F}(L, \mathbf{p}, s)$ is given:

Theorem 2.1 (Schmidt-Hieber, [10], Theorem 5) *For any function $f \in \mathcal{C}_d^\beta(K)$ and any integers $m \ge 1$ and $N \ge (\beta+1)^d \vee (K+1)e^d$, there exists a network $\tilde{f} \in \mathcal{F}(L, \mathbf{p}, s)$ with depth*

$$L = 8 + (m+5)(1 + \lceil \log_2(d \vee \beta) \rceil),$$

width

$$|\mathbf{p}|_\infty = 6(d + \lceil \beta \rceil)N$$

and number of nonzero parameters

$$s \le 141(d + \beta + 1)^{3+d} N(m+6),$$

such that

$$\|\tilde{f} - f\|_{L^\infty[0,1]^d} \le (2K+1)(1 + d^2 + \beta^2)6^d N 2^{-m} + K 3^\beta N^{-\frac{\beta}{d}}.$$

The proof of the theorem is based on local sparse neural network approximation of Taylor polynomials of the function f.

Our goal is to attain an identical approximation rate for networks with parameters in $\{0, \pm\frac{1}{2}, \pm 1, 2\}$. In our construction, we will omit the shift vectors (by adding a coordinate 1 to the input vector $\mathbf{x}$) and will consider the networks of the form

$$\Big\{ f : [0,1]^d \to \mathbb{R} \mid \ f(\mathbf{x}) = W_L \circ \sigma \circ W_{L-1} \circ \sigma \circ \ldots \circ \sigma \circ W_0(1, \mathbf{x}) \Big\} \qquad (1)$$

with weight matrices $W_i \in \mathbb{R}^{p_i \times p_{i+1}}$, $i = 0, \ldots, L$, and with width vector $\mathbf{p} = (p_0, p_1, \ldots, p_{L+1})$, $p_0 = d$. In this case, the ReLU activation function $\sigma(x)$ acts coordinate-wise on the input vectors. Let $\widetilde{\mathcal{F}}(L, \mathbf{p})$ be the set of networks of the form (1) with parameters in $\{0, \pm\frac{1}{2}, \pm 1, 2\}$. For $s \in \mathbb{N}$, let $\widetilde{\mathcal{F}}(L, \mathbf{p}, s)$ be the subset of $\widetilde{\mathcal{F}}(L, \mathbf{p})$ with at most s nonzero parameters. We then have the following.

Theorem 2.2 *For any function $f \in \mathcal{C}_d^\beta(K)$ and any integers $m \geq 1$ and $N \geq (\beta + 1)^d \vee (K+1)e^d$, there exists a network $\tilde{f} \in \widetilde{\mathcal{F}}(\tilde{L}, \tilde{\mathbf{p}}, \tilde{s})$ with depth*

$$\tilde{L} \leq 4\Delta + 2L,$$

width

$$|\tilde{\mathbf{p}}|_\infty \leq 2(1 + d + R + \Delta) \vee 2^d |\mathbf{p}|_\infty$$

and number of nonzero parameters

$$\tilde{s} \leq (1 + d + R + \Delta)\tilde{L} + 2^d s,$$

such that

$$\|\tilde{f} - f\|_{L^\infty[0,1]^d} \leq (2K+1)(1 + d^2 + \beta^2)12^d N 2^{-m} + (K+1)3^\beta N^{-\frac{\beta}{d}},$$

where $\Delta \leq 2\log_2(N^{\beta+d} K e^d)$, $R \leq (2\beta)^d N$ and $L, \mathbf{p}$ and s are the same as in Theorem 2.1.

Let us now compare the above two theorems. First, the approximation errors in those theorems differ by a constant factor depending only on the input dimension d. (Note that the values of β, d and K are assumed to be fixed.) The depths and the number of nonzero parameters of the networks presented in Theorems 2.1 and 2.2 differ at most by $\log_2 N$ multiplied by a constant depending on β, d and K, and the maximal widths of those networks differ at most by a constant factor $C(\beta, d, K)$. Thus, the architecture and the number of active parameters of network given in Theorem 2.2 have, up to a logarithmic factor, the same dependence on the approximation error as the network given in Theorem 2.1.

Application to nonparametric regression Consider a nonparametric regression model

$$Y_i = f_0(\mathbf{X}_i) + \epsilon_i,$$

where $f_0 : [0,1]^d \to [-F, F]$ is the unknown regression function that needs to be recovered from n observed iid pairs $(\mathbf{X}_i, Y_i), i = 1, \ldots, n$. The standard normal noise variables ϵ_i are assumed to be independent of $\mathbf{X}_i$. For a set of functions $\mathcal{F}$ from $[0,1]^d$ to $[-F, F]$ and for an estimator $\hat{f} \in \mathcal{F}$ of f_0, define

$$\Delta_n = \Delta_n(\hat{f}, f_0, \mathcal{F}) = \mathbb{E}_{f_0}\left[\frac{1}{n}\sum_{i=1}^n (Y_i - \hat{f}(\mathbf{X}_i))^2 - \inf_{f \in \mathcal{F}} \frac{1}{n}\sum_{i=1}^n (Y_i - f(\mathbf{X}_i))^2\right].$$

The subscript f_0 indicates that the expectation is taken over the training data generated by our regression model and $\Delta_n(\hat{f}, f_0, \mathcal{F})$ measures how close the estimator $\hat{f}$ is to the empirical risk minimizer. Let also

$$R(\hat{f}, f_0) = \mathbb{E}_{f_0}[(\hat{f}(\mathbf{X}) - f_0(\mathbf{X}))^2]$$

be the prediction error of the estimator $\hat{f} \in \mathcal{F}$, where $\mathbf{X} \overset{\mathcal{D}}{=} \mathbf{X}_1$ is independent of the sample $(\mathbf{X}_i, Y_i)$. Prediction errors are assessed by oracle inequalities, which, in turn, are usually given in terms of below-defined covering numbers and the entropies of the function class $\mathcal{F}$ from which the estimator is chosen.

Definition 2.1 For $\delta \geq 0$, the covering number $\mathcal{N}(\delta, \mathcal{F}, \|\cdot\|_\infty)$ of radius δ of the set of functions $\mathcal{F}$ taken with respect to the $\|\cdot\|_\infty$ distance of functions on $[0,1]^d$ is the minimal number $N \in \mathbb{N}$ such that there exist $f_1, \dots, f_N$ from $[0,1]^d$ to $\mathbb{R}$ with the property that for any $f \in \mathcal{F}$ there is some $k \in \{1, \dots, N\}$ such that

$$\|f - f_k\|_{L^\infty[0,1]^d} \leq \delta.$$

The number $\log_2 \mathcal{N}(\delta, \mathcal{F}, \|\cdot\|_\infty)$ is then called a δ-entropy of the set $\mathcal{F}$.

The following oracle-type inequality is obtained in [10], Lemma 4:

Lemma 2.1 *For any* $\delta \in (0, 1]$

$$R(\hat{f}, f_0) \leq 4\Bigg[\inf_{f\in\mathcal{F}} \mathbb{E}[(f(\mathbf{X}) - f_0(\mathbf{X}))^2] \\ + F^2 \frac{18\log_2 \mathcal{N}(\delta, \mathcal{F}, \|\cdot\|_\infty) + 72}{n} + 32\delta F + \Delta_n\Bigg],$$

where $\mathcal{N}(\delta, \mathcal{F}, \|\cdot\|_\infty)$ *is the covering number of* $\mathcal{F}$ *of radius* δ.

Assume now that the unknown regression function $f_0 : [0,1]^d \to [-F, F]$ belongs to the class $\mathcal{C}_d^\beta(K)$ with $F \geq \max(K, 1)$. Taking in Theorem 2.2 $r = d, m = \lceil \log_2 n \rceil$ and $N = n^{\frac{d}{2\beta+d}}$, we get the existence of a network $\tilde{f}_n \in \widetilde{\mathcal{F}}(\tilde{L}_n, \tilde{\mathbf{p}}_n, \tilde{s}_n)$ with $\tilde{L}_n \leq c\log_2 n$, $|\tilde{\mathbf{p}}_n|_\infty \leq cn^{\frac{d}{2\beta+d}}$ and $\tilde{s}_n \leq cn^{\frac{d}{2\beta+d}} \log_2 n$ such that

$$\|\tilde{f}_n - f_0\|^2_{L^\infty[0,1]^d} \leq cn^{\frac{-2\beta}{2\beta+d}}, \tag{2}$$

where $c = c(\beta, d, F)$ is some constant. In order to apply Lemma 2.1, it remains to estimate the covering number $\mathcal{N}(\delta, \widetilde{\mathcal{F}}(\tilde{L}_n, \tilde{\mathbf{p}}_n, \tilde{s}_n), \|\cdot\|_\infty)$. Note, however, that since the parameters of networks from $\widetilde{\mathcal{F}}(\tilde{L}_n, \tilde{\mathbf{p}}_n, \tilde{s}_n)$ belong to the discrete set $\{0, \pm\frac{1}{2}, \pm 1, 2\}$, we can calculate the exact number of networks from $\widetilde{\mathcal{F}}(\tilde{L}_n, \tilde{\mathbf{p}}_n, \tilde{s}_n)$, or, in other words, we can upper-bound the covering number of radius $\delta = 0$. Indeed, as there are at most $(\tilde{L}_n + 1)|\tilde{\mathbf{p}}_n|^2_\infty$ parameters in the networks from $\widetilde{\mathcal{F}}(\tilde{L}_n, \tilde{\mathbf{p}}_n, \tilde{s}_n)$, then for a given s there are at most $\binom{(\tilde{L}_n + 1)|\tilde{\mathbf{p}}_n|^2_\infty}{s}$ ways to choose s nonzero parameters. As the nonzero parameters can take one of the 5 values $\{\pm\frac{1}{2}, \pm 1, 2\}$, the total number of networks from $\widetilde{\mathcal{F}}(\tilde{L}_n, \tilde{\mathbf{p}}_n, \tilde{s}_n)$ is bounded by

$$\sum_{s \le \tilde{s}_n} \left(5(\tilde{L}_n + 1)|\tilde{\mathbf{p}}_n|_\infty^2\right)^s \le \left(5(\tilde{L}_n + 1)|\tilde{\mathbf{p}}_n|_\infty^2\right)^{\tilde{s}_n+1}.$$

Together with (2) and Lemma 2.1, for the empirical risk minimizer

$$\hat{f}_n \in \underset{f \in \widetilde{\mathcal{F}}(\tilde{L}_n, \tilde{\mathbf{p}}_n, \tilde{s}_n)}{\arg\min} \sum_{i=1}^n (Y_i - f(\mathbf{X}_i))^2$$

we get an existence of a constant $C = C(\beta, d, F)$ such that

$$R(\hat{f}_n, f_0) \le C n^{\frac{-2\beta}{2\beta+d}} \log_2^2 n \tag{3}$$

which coincides, up to a logarithmic factor, with the minimax estimation rate $n^{\frac{-2\beta}{2\beta+d}}$ of the prediction error for β-smooth functions.

Remark 2.1 Note that if s and s_p denote, respectively, the l_0 and l_p parameter norms of a network with parameters $\{0, \pm\frac{1}{2}, \pm 1, 2\}$, then $s^{1/p}/2 \le s_p \le 2s^{1/p}$, $p > 0$. Therefore, in the above application the same convergence rate as in (3) can be attained by replacing $\widetilde{\mathcal{F}}(\tilde{L}_n, \tilde{\mathbf{p}}_n, \tilde{s}_n)$ with the subclass of $\widetilde{\mathcal{F}}(\tilde{L}_n, \tilde{\mathbf{p}}_n)$ consisting of networks with l_p parameter norms bounded by $\tilde{s}_n^{1/p}$. In particular, taking $p = 1$ we get that both in Theorem 2.2 and in the application above the same approximation and prediction rates can be obtained by replacing the sparsity constraint with the l_1 network parameter regularization.

3 Proofs

One of the ways to approximate functions by neural networks is based on the neural network approximation of local Taylor polynomials of those functions (see, e.g., [10, 13]). Thus, in this procedure, approximation of the product xy given the input (x, y) becomes crucial. The latter is usually done by representing the product xy as a linear combination of functions that can be approximated by neural network-implementable functions. For example, the approximation algorithm presented in [10] is based on the approximation of a function $g(x) = x(1 - x)$, which then leads to an approximation of the product

$$xy = g\left(\frac{x - y + 1}{2}\right) - g\left(\frac{x + y}{2}\right) + \frac{x + y}{2} - \frac{1}{4}. \tag{4}$$

The key observation is that the function $g(x)$ can be approximated by combinations of triangle waves and the latter can be easily implemented by neural networks with

ReLU activation function. In the proof of Theorem 2.1, neural network approximation of function $(x, y) \mapsto xy$ is followed by approximation of the product $(x_1, \dots, x_r) \mapsto \prod_{j=1}^{r} x_j$ which then leads to approximation of monomials of degree up to β. The result then follows by local approximation of Taylor polynomials of f. Below, we show that all those approximations can also be performed using only the parameters $\{0, \pm\frac{1}{2}, \pm 1, 2\}$. As it is formalized in (1), in our constructions we add a coordinate 1 to the input vector to omit the shift vectors. To check the equivalence of those two approaches, suppose that a given hidden layer of a network is determined by a weight matrix $W \in \mathbb{R}^{d_1 \times d_2}$ and a shift vector $\mathbf{v} \in \mathbb{R}^{d_1}$ and let $\widetilde{W} \in \mathbb{R}^{d_1 \times d_2 + 1}$ be the matrix obtained by appending the matrix W to the column $-\mathbf{v}$: $\widetilde{W} = (-\mathbf{v}, W)$. Then, for a given input vector $\mathbf{y} \in \mathbb{R}^{d_2}$ we have that $\sigma \widetilde{W}(1, \mathbf{y}) = \sigma_{\mathbf{v}} W \mathbf{y}$ from which the desired equivalence follows.

We start our constructions with Lemma 3.1 which shows that networks with parameters $\{0, \pm\frac{1}{2}, \pm 1\}$, depth $2m + 4$ and width 9 can approximate the product xy exponentially fast in m:

Lemma 3.1 *For any positive integer m, there exists a network* $\mathrm{Mult}_m \in \widetilde{\mathcal{F}}(2m+4, \mathbf{p})$*, with* $p_0 = 3$, $p_{L+1} = 1$ *and* $|\mathbf{p}|_\infty = 9$*, such that*

$$|\mathrm{Mult}_m(1, x, y) - xy| \leq 2^{-m}, \quad \text{for all } x, y \in [0, 1]. \tag{5}$$

Proof Consider the functions $T^k : [0, 2^{2-2k}] \to [0, 2^{-2k}]$, $k \in \mathbb{N}$, defined by

$$T^k(x) := (x/2)_+ - (x - 2^{1-2k})_+ = T_+(x) - T_-^k(x), \tag{6}$$

where $T_+(x) := (x/2)_+$ and $T_-^k(x) := (x - 2^{1-2k})_+$. In [11], Lemma A.1, it is shown that for the functions $R^k : [0, 1] \to [0, 2^{-2k}]$,

$$R^k = T^k \circ T^{k-1} \circ \dots \circ T^1, \tag{7}$$

and for any positive integer m,

$$|g(x) - \sum_{k=1}^{m} R^k(x)| \leq 2^{-m}, \quad x \in [0, 1], \tag{8}$$

where $g(x) = x(1 - x)$. Taking into account (4) and (8), we need to construct a network that computes

$$(1, x, y) \mapsto \left(\sum_{k=1}^{m+1} R^k\left(\frac{x - y + 1}{2}\right) - \sum_{k=1}^{m+1} R^k\left(\frac{x + y}{2}\right) + \frac{x + y}{2} - \frac{1}{4} \right)_+ \wedge 1. \tag{9}$$

Let us first construct a network N_m with depth $2m$, width 4 and weights $\{0, \pm\frac{1}{2}, \pm 1\}$ that computes

$$(1/4, T_+(u), h(u), T_-^1(u)) \mapsto \sum_{k=1}^{m+1} R^k(u) + h(u), \quad u \in [0,1].$$

For this goal, we modify the network presented in [11], Fig. 2, to assure that the parameters are all in $\{0, \pm\frac{1}{2}, \pm 1\}$. More explicitly, denote

$$A := \begin{pmatrix} \frac{1}{2} & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & -\frac{1}{2} \\ 0 & 1 & 1 & -1 \\ -\frac{1}{2} & 1 & 0 & -1 \end{pmatrix}$$

and

$$B := \begin{pmatrix} \frac{1}{2} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Then,

$$N_m = \sigma \circ (0 \quad 1 \quad 1 \quad -1) \circ \sigma \circ B \circ \sigma \circ A \circ \ldots \circ \sigma \circ B \circ \sigma \circ A \circ \sigma \circ B \circ \sigma \circ A,$$

where each of the mutually succeeding matrices A and B appears in the above representation m times. Using parameters $\{0, \pm\frac{1}{2}, \pm 1\}$, for a given input $(1, x, y)$ the first two layers of the network Mult_m compute the vector

$$\left(1, \frac{1}{4}, T_+\left(\frac{x-y+1}{2}\right), \left(\frac{x+y}{2}\right)_+, T_-^1\left(\frac{x-y+1}{2}\right), \frac{1}{4}, T_+\left(\frac{x+y}{2}\right), \frac{1}{4}, T_-^1\left(\frac{x+y}{2}\right)\right).$$

(Note that as in our construction we omit shift vectors, throughout the whole construction we will keep the first coordinate equal to 1.) We then apply the network N_m to the first and last four coordinates of the above vector that follow the first coordinate 1. We thus obtain a network with $2m+2$ hidden layers and of width 9 that computes

$$(1, x, y) \mapsto \left(1, \sum_{k=1}^{m+1} R^k\left(\frac{x-y+1}{2}\right) + \frac{x+y}{2}, \sum_{k=1}^{m+1} R^k\left(\frac{x+y}{2}\right) + \frac{1}{4}\right). \quad (10)$$

Finally, the last two layers of Mult_m compute $(1, u, v) \mapsto (1-(1-(u-v))_+)_+$ applied to the vector obtained in (10). (Note that this computation only requires parameters 0 and ± 1.) We thus get a network Mult_m computing (9), and the inequality (5) follows by combining (4) and (8). □

Lemma 3.2 *For any positive integer m, there exists a network $\mathrm{Mult}_m^r \in \widetilde{\mathcal{F}}(L, \boldsymbol{p})$, with $L = (2m+5)\lceil \log_2 r \rceil$, $p_0 = r+1$, $p_{L+1} = 1$ and $|\mathbf{p}|_\infty \le 9r$, such that*

$$|\mathrm{Mult}_m^r(1, \mathbf{x}) - \prod_{i=1}^{r} x_i| \le r^2 2^{-m}, \quad \textit{for all } \mathbf{x} = (x_1 \ \ldots \ x_r) \in [0,1]^r.$$

Proof In order to approximate the product $\prod_{i=1}^r x_i$, we first pair the neighboring entries to get the triples $(1, x_k, x_{k+1})$ and apply the previous lemma to each of those triples to obtain the values $\mathrm{Mult}_m(1, x_k, x_{k+1})$. We repeat this procedure $q := \lceil \log_2 r \rceil$ times, until there is only one entry left. As pairing the entries requires only parameters 0 and 1, then it follows from the previous lemma that the entries of the constructed network are in $\{0, \pm\frac{1}{2}, \pm 1\}$. Using Lemma 3.1 and applying the inequality $|\mathrm{Mult}_m(1,x,y) - tz| \le 2^{-m} + |x-z| + |y-t|$, $x, y, z, t \in [0,1]$, q times we get $|\mathrm{Mult}_m^r(1,\mathbf{x}) - \prod_{i=1}^r x_i| \le 3^{q-1}2^{-m} \le r^2 2^{-m}$. □

For $\gamma > 0$, let $C_{d,\gamma}$ denote the number of d-dimensional monomials $\mathbf{x}^{\boldsymbol{\alpha}}$ with degree $|\boldsymbol{\alpha}| < \gamma$. Note that $C_{d,\gamma} < (\gamma+1)^d$. From Lemma 3.2, it follows that using weights $\{0, \pm\frac{1}{2}, \pm 1\}$, we can simultaneously approximate monomials up to degree γ:

Lemma 3.3 *There exists a network $\mathrm{Mon}_{m,\gamma}^d \in \widetilde{\mathcal{F}}(L, \mathbf{p})$ with $L \le (2m+5)\lceil \log_2(\gamma \vee 1) \rceil + 1$, $p_0 = d+1$, $p_{L+1} = C_{d,\gamma}$ and $|\mathbf{p}|_\infty \le 9\lceil\gamma\rceil C_{d,\gamma}$ such that*

$$\left| \mathrm{Mon}_{m,\gamma}^d(1,\mathbf{x}) - (\mathbf{x}^{\boldsymbol{\alpha}})_{|\boldsymbol{\alpha}|<\gamma} \right|_\infty \le \gamma^2 2^{-m}, \quad \mathbf{x} \in [0,1]^d.$$

Proof For $\mathbf{x} \in [0,1]^d$ and for a given exponent vector $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_d)$ with $|\boldsymbol{\alpha}| < \gamma$, define an $(|\boldsymbol{\alpha}|+1)$-dimensional vector

$$\tilde{\mathbf{x}}_{\boldsymbol{\alpha}} := (1, \underbrace{x_1, \ldots, x_1}_{\alpha_1}, \ldots, \underbrace{x_d, \ldots, x_d}_{\alpha_d}).$$

Using only the parameters 0 and 1 and having width at most $\lceil\gamma\rceil C_{d,\gamma}$, the first hidden layer of the network $\mathrm{Mon}_{m,\gamma}^d \in \widetilde{\mathcal{F}}(L, \mathbf{p})$ computes the vector

$$(1, \mathbf{x}) \mapsto (1, \tilde{\mathbf{x}}_{\boldsymbol{\alpha}_2}, \ldots, \tilde{\mathbf{x}}_{\boldsymbol{\alpha}_{C_{d,\gamma}}}), \quad \mathbf{x} \in [0,1]^d, \tag{11}$$

where the first coordinate 1 of the computed vector is the value of the monomial $\mathbf{x}^{\boldsymbol{\alpha}_1}$ corresponding to the zero exponent vector $\boldsymbol{\alpha}_1$ and $\boldsymbol{\alpha}_2, \ldots, \boldsymbol{\alpha}_{C_{d,\gamma}}$ are all the exponent vectors with $0 < |\boldsymbol{\alpha}_i| < \gamma$, $i = 2, \ldots, C_{d,\gamma}$. By Lemma 3.2, for each $i = 2, \ldots, C_{d,\gamma}$, there is a network $\mathrm{Mult}_m^{\boldsymbol{\alpha}_i} \in \widetilde{\mathcal{F}}((2m+5)\lceil \log_2 \gamma \rceil, 9\lceil\gamma\rceil)$ such that

$$|\mathrm{Mult}_m^{\boldsymbol{\alpha}_i}(\tilde{\mathbf{x}}_{\boldsymbol{\alpha}_i}) - \mathbf{x}^{\boldsymbol{\alpha}_i}| \le \gamma^2 2^{-m}, \quad \mathbf{x} \in [0,1]^d.$$

The network $\mathrm{Mon}^d_{m,\gamma}$ is obtained by applying in parallel the networks $\mathrm{Mult}^{\boldsymbol{\alpha}_i}_m$ to the components of the output vector (11) obtained in the first step while leaving its first coordinate unchanged. □

Note that the depths and the widths of networks presented in Lemmas 3.1, 3.2 and 3.3 that, respectively, approximate the products xy and $\prod_{i=1}^r x_i$ and the monomials up to degree γ are at most twice as large as the depths and widths of corresponding networks constructed in Lemmas A.2, A.3 and A.4 in [11]. Thus, by enlarging the network sizes at most by a constant factor of 2 and using the parameters $\{0, \pm\frac{1}{2}, \pm 1, 2\}$ we can achieve the same rates of approximations of monomials as with the networks with parameters in $[-1, 1]$.

We now present the final stage of the approximation, that is, the local approximation of Taylor polynomials of f.

Proof of Theorem 2.2 For a given N, let $\tilde{N} \geq N$ be the smallest integer with $\tilde{N} = (2^\nu + 1)^d$ for some $\nu \in \mathbb{N}$. Note that $\tilde{N}/2^d \leq N \leq \tilde{N}$. We are going to apply Theorem 2.1 with N in the condition of that theorem replaced by $\tilde{N}$. For $\mathbf{a} \in [0, 1]^d$, let

$$P^\beta_{\mathbf{a}} f(\mathbf{x}) = \sum_{0 \leq |\boldsymbol{\alpha}| < \beta} (\partial^{\boldsymbol{\alpha}} f)(\mathbf{a}) \frac{(\mathbf{x} - \mathbf{a})^{\boldsymbol{\alpha}}}{\boldsymbol{\alpha}!} := \sum_{0 \leq |\boldsymbol{\gamma}| < \beta} c_{\mathbf{a},\boldsymbol{\gamma}} \mathbf{x}^{\boldsymbol{\gamma}} \tag{12}$$

be the partial sum of Taylor series of f around $\mathbf{a}$. Choose M to be the largest integer such that $(M+1)^d \leq \tilde{N}$, that is, $M = 2^\nu$, and consider the set of $(M+1)^d$ grid points $\mathbf{D}(M) := \{\mathbf{x}_{\boldsymbol{\ell}} = (\ell_j/M)_{j=1,\ldots,d} : \boldsymbol{\ell} = (\ell_1, \ldots, \ell_d) \in \{0, 1, \ldots, M\}^d\}$. Denoting $\mathbf{x}_{\boldsymbol{\ell}} = (x_1^{\boldsymbol{\ell}}, \ldots, x_d^{\boldsymbol{\ell}})$, it is shown in [11], Lemma B.1, that

$$\|P^\beta f - f\|_{L^\infty[0,1]^d} \leq K M^{-\beta}, \tag{13}$$

where

$$P^\beta f(\mathbf{x}) = \sum_{\mathbf{x}_{\boldsymbol{\ell}} \in \mathbf{D}(M)} P^\beta_{\mathbf{x}_{\boldsymbol{\ell}}} f(\mathbf{x}) \prod_{j=1}^d (1 - M|x_j - x_j^{\boldsymbol{\ell}}|)_+.$$

As in our construction we only use parameters $\{0, \pm\frac{1}{2}, \pm 1, 2\}$, we need to modify the coefficients given in (12) to make them implementable by those parameters. Denote $B := \lfloor 2Ke^d \rfloor$ and let $b \in \mathbb{N}$ be the smallest integer with $2^b \geq BM^\beta(\beta+1)^d$. As $|c_{\mathbf{a},\boldsymbol{\gamma}}| < B$ ([11], eq. 34), then for each $c_{\mathbf{a},\boldsymbol{\gamma}}$ there is an integer $k \in [-2^b, 2^b]$ with $c_{\mathbf{a},\boldsymbol{\gamma}} \in [\frac{k}{2^b}B, \frac{k+1}{2^b}B)$. Denote then $\tilde{c}_{\mathbf{a},\boldsymbol{\gamma}} = \frac{k}{2^b}B$ and define

$$\tilde{P}^\beta_{\mathbf{a}} f(\mathbf{x}) = \sum_{0 \leq |\boldsymbol{\gamma}| < \beta} \tilde{c}_{\mathbf{a},\boldsymbol{\gamma}} \mathbf{x}^{\boldsymbol{\gamma}}.$$

As the number of monomials of degree up to β is bounded by $(\beta+1)^d$, then

$$\|P_{\mathbf{a}}^{\beta} f - \tilde{P}_{\mathbf{a}}^{\beta} f\|_{L^\infty[0,1]^d} \le (\beta+1)^d \frac{B}{2^b} \le M^{-\beta}.$$

Also, as

$$\sum_{\mathbf{x}_{\boldsymbol{\ell}} \in \mathbf{D}(M)} \prod_{j=1}^{d} (1 - M|x_j - x_j^{\boldsymbol{\ell}}|)_+ = 1,$$

then

$$\|P^{\beta} f(\mathbf{x}) - \sum_{\mathbf{x}_{\boldsymbol{\ell}} \in \mathbf{D}(M)} \tilde{P}_{\mathbf{x}_{\boldsymbol{\ell}}}^{\beta} f(\mathbf{x}) \prod_{j=1}^{d} (1 - M|x_j - x_j^{\boldsymbol{\ell}}|)_+ \|_{L^\infty[0,1]^d} \le M^{-\beta}.$$

Thus, defining

$$\tilde{P}^{\beta} f(\mathbf{x}) = \sum_{\mathbf{x}_{\boldsymbol{\ell}} \in \mathbf{D}(M)} \tilde{P}_{\mathbf{x}_{\boldsymbol{\ell}}}^{\beta} f(\mathbf{x}) \prod_{j=1}^{d} (1 - M|x_j - x_j^{\boldsymbol{\ell}}|)_+,$$

we get that

$$\|\tilde{P}^{\beta} f - f\|_{L^\infty[0,1]^d} \le (K+1) M^{-\beta}. \tag{14}$$

In the proof of Theorem 2.1, the neural network approximation of the function $(x_1, \dots, x_r) \mapsto \prod_{j=1}^{r} x_j$ is first constructed followed by approximation of monomials of degree up to β. The result then follows by approximating the function $P^{\beta} f(\mathbf{x})$ and applying (13). In the latter approximation, the set of parameters not belonging to $\{0, \pm\frac{1}{2}, \pm 1\}$ consists of:

- shift coordinates j/M, $j = 1, \dots, M-1$, (the grid points);
- at most $(\beta(M+1))^d$ weight matrix entries of the form $c_{\mathbf{x}_{\boldsymbol{\ell}}, \boldsymbol{\gamma}}/B$, where $c_{\mathbf{x}_{\boldsymbol{\ell}}, \boldsymbol{\gamma}}$ are coefficients of the polynomial $P_{\mathbf{x}_{\boldsymbol{\ell}}}^{\beta} f(\mathbf{x})$, $\mathbf{x}_{\boldsymbol{\ell}} \in \mathbf{D}(M)$, $|\boldsymbol{\gamma}| < \beta$;
- a shift coordinate $1/(2M^d)$ (used to scale the output entries).

Note that the above list gives at most $D := M + (\beta(M+1))^d$ different parameters. Taking into account (14), we can use $\tilde{P}^{\beta} f$ instead of $P^{\beta} f$ to approximate f. Thus, we can replace the entries $c_{\mathbf{x}_{\boldsymbol{\ell}}, \boldsymbol{\gamma}}/B$ by the entries $\tilde{c}_{\mathbf{x}_{\boldsymbol{\ell}}, \boldsymbol{\gamma}}/B = \frac{k}{2^b}$, where k is some integer from $[-2^b, 2^b]$. Also, as $M = 2^{\nu}$, then denoting $\Delta = \max\{\nu d + 1; b\}$ we need to obtain D parameters from the set $\mathcal{S} = \{\frac{k}{2^{\Delta}}, k \in \mathbb{Z} \cap (0, 2^{\Delta}]\}$. As any natural number can be represented as a sum of powers of 2, then for any $y_1, \dots, y_D \in \mathbb{Z} \cap (0, 2^{\Delta}]$ we can compute

$$(1, x_1, \dots, x_d) \mapsto (1, x_1, \dots, x_d, y_1, \dots, y_D)$$

with parameters from $\{0, 1, 2\}$ using at most Δ hidden layers. The number of active parameters required for this computation is bounded by $(1 + d + D + \Delta)\Delta$. Hence, for any $z_1, \ldots, z_D \in \mathcal{S}$, we can compute

$$(1, x_1, \ldots, x_d) \mapsto (1, x_1, \ldots, x_d, z_1, \ldots, z_D)$$

with 2Δ hidden layers and $2(1 + d + D + \Delta)\Delta$ active parameters. Applying Theorem 2.1, we get the existence of a network $\tilde{f} \in \widetilde{\mathcal{F}}(\tilde{L}, \tilde{\mathbf{p}}, \tilde{s})$ with the desired architecture and sparsity. $\square$

Acknowledgements The author has been supported by the NWO Vidi Grant: "*Statistical foundation for multilayer neural networks*." The author would like to thank Johannes Schmidt-Hieber and Hailin Sang for useful comments and suggestions. The author is also grateful to the Editor and the Reviewers for constructive and valuable comments that helped to improve the manuscript.

References

1. Birgé L (1987) Estimating a density under order restrictions: nonasymptotic minimax risk. Ann Stat 15:995–1012
2. Gale T, Elsen E, Hooker S The state of sparsity in deep neural networks, https://arxiv.org/pdf/1902.09574.pdf
3. Györfi L, Kohler M, Krzyzak A, Walk H (2006) A distribution-free theory of nonparametric regression. Springer, Berlin
4. Han S, Pool J, Tran J, Dally W (2015) Learning both weights and connections for efficient neural network. In: Cortes C, Lawrence ND, Lee DD, Sugiyama M, Garnett R (eds.) Advances in neural information processing systems. Vol. 28, pp. 1135-1143
5. Hassibi B, Stork DG (1993) Second order derivatives for network pruning: optimal brain surgeon. In: Hanson SJ, Cowan JD, Giles CL (eds). Advances in Neural Information Processing Systems, vol 5 pp. 164-171
6. Lu Z, Pu H, Wang F, Hu Z, Wang L (2017) The expressive power of neural networks: a view from the width. Adv Neural Inform Process Syst, pp. 6231-6239
7. Ohn I, Kim Y Nonconvex sparse regularization for deep neural networks and its optimality, https://arxiv.org/pdf/2003.11769.pdf
8. Pollard D (1984) Convergence of stochastic processes. Springer, Berlin
9. Scarselli F, Tsoi AC (1998) Universal approximation using feedforward neural networks: a survey of some existing methods, and some new results. Neural Netw 11:15–37
10. Schmidt-Hieber J (2020) Nonparametric regression using deep neural networks with ReLU activation function. Ann Stat 48(4):1875–1897
11. Schmidt-Hieber J Supplement to "Nonparametric regression using deep neural networks with ReLU activation function", https://arxiv.org/pdf/1708.06633.pdf
12. Taheri M, Xie F, Lederer J (2021) Statistical guarantees for regularized neural networks. Neural Netw 142:148–161
13. Yarotsky D (2017) Error bounds for approximations with deep ReLU networks. Neural Netw 94:103–114