SPARSE BESOV SPACE ANALYSIS OF DEEP LEARNING REPRESENTATION LAYERS IN HIGH DIMENSIONS

IDO BEN-SHAUL, SHAI DEKEL, AND OREN ELISHA

ABSTRACT. It is known that it is not possible to characterize the approximation spaces of deep learning models using classic smoothness spaces [19]. Furthermore, many problems solved by deep learning are high dimensional where classical function spaces such as the isotropic Besov spaces are somewhat inadequate. In this paper we try to shed some light on this problem by analyzing the dynamics of sparse Besov function smoothness of representations across the layers of a deep neural network, during and after training. We justify our approach by extensive experiments demonstrating that in well-performing trained networks, the sparse Besov smoothness of the training set, measured in the corresponding hidden layer feature map representation, increases from layer to layer. Our approach also serves as a unifying platform for the analysis of signal processing, classic machine learning tree-based models and deep learning.

1. INTRODUCTION

In recent years there has been growing interest and much research on the topic of the 'mathematics of deep learning'. While deep learning (DL) exhibits overwhelming effectiveness in many challenging problems in computer vision and natural language processing, the deep neural networks themselves are considered by many as mysterious black boxes. Thus, there is ongoing significant research efforts to find a theory to explain this success using statistics, dynamical systems, compositionality classes, approximation theory and more (e.g. [20, 32, 19, 29, 2]). However, most of the proposed theories do not present a unified approach to the three pillars of data processing which are: signal processing, classic machine learning models (e.g. decision trees, random forest, gradient boosting) and deep learning. Quite often data scientists struggle to understand when does deep learning provide an advantage over the simpler machine learning models (e.g. tree-based models). There is some folklore knowledge that DL does not provide any advantage for most tabular data, but this phenomena is not well understood [31]. In this work we advocate to model datasets as samples of functions and then

1991 *Mathematics Subject Classification.* 41A25,41A63,65D40,68T07.

Key words and phrases. Mathematical foundations of deep learning, representation learning, deep learning approximation spaces.

estimate the weak type smoothness of these functions using sparse tree-based Besov spaces. We also characterize this type of smoothness using an equivalent form of sparsity of wavelet decompositions of tree representations. This allows us to link the well established approximation theory of signal theory [16, 17], the theory of classical tree-based machine learning models [13] and the approximation theoretical potential of DL. We advocate that in cases where the input function has low 'weak-type' smoothness, the role of a feed-forward neural network is to perform automatic feature engineering which improves the smoothness of the representations as we go deeper through the layers of the network. In Figure 1 we see a visualization of the feature space of two famous datasets: Titanic and MNIST. The Titanic dataset is composed of tabular data where the passenger features are: age, gender, class of ticket, port of boarding, etc. and then the response variable is the binary survival outcome. The MNIST dataset is composed of 28×28 grayscale images, each containing a hand-written digit and each labeled: '0' to '9'. Although the feature spaces of the samples of the two datasets are high-dimensional, they are depicted colored with their label using a nonlinear dimension reduction algorithm. Roughly speaking, the tabular Titanic dataset has more cluster structures in the feature space, while the feature space of MNIST seems very unstructured. We argue that while DL is not needed for the Titanic dataset, the role of DL for the MNIST dataset is to unravel the feature space and improve it layer by layer, where at each layer clustering improves as shown in Figure 2. Consequently, the smoothness of the dataset improves across layers in the corresponding representation spaces.

Let us provide more details for this process: Assume we are presented with a set of gray-scale images of dimension $\sqrt{n_0} \times \sqrt{n_0}$ with L class labels. Assume further that a DL network has been successfully trained to classify these images with relatively high accuracy. This allows us to extract the representation of each image in each of the hidden layers. To create a representation at layer 0, we concatenate the $\sqrt{n_0}$ rows of pixel values of each image, to create a vector of dimension n_0. We also normalize the pixel values to the range $[0, 1]$. Since we advocate a function-theoretical approach, we transform the class labels into vector-values in the space $\mathbb{R}^L$ by assigning the l-th label to the l-th standard basis vector, so the first class to $(1, 0, \cdots)$, the second class to $(0, 1, 0, \cdots)$, etc. Thus, the images are considered as samples of a function $f_0 : [0, 1]^{n_0} \to \mathbb{R}^L$. Typically, for non-tabular data, such as in computer vision problems, there is no hope that there exists geometric clustering of the classes in this initial feature space and that f_0 has sufficient 'weak-type' smoothness. Thus, a transform into a different feature space is needed. We associate with each k-th layer of a DL network, a function $f_k : [0, 1]^{n_k} \to \mathbb{R}^L$ where the samples are vectors created by normalizing and concatenating the feature maps computed from each of the images. Interestingly enough, although the series of functions f_k are embedded in different dimensions n_k, through the simple normalizing of

FIGURE 1. Visualization of input feature spaces: MNIST dataset and Titanic dataset

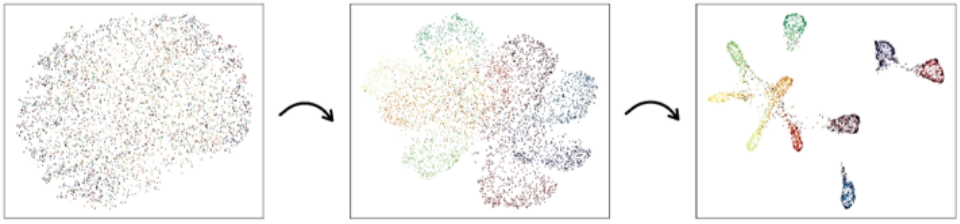

FIGURE 2. Improved clustering

the features, our method is able to assign smoothness indices to each layer that are comparable. We claim that for well performing networks, the representations in general 'improve' from layer to layer and that our method mathematically quantifies the phenomena that is rendered in Figure 2.

The paper is organized as follows: In Section 2 we present some basic DL architectures and discuss the problem of the characterization of DL approximation spaces. In Section 3 we review our sparse smoothness analysis machinery which is the Wavelet Decomposition of Random Forest (RF) [18]. In Section 4 we present the required sparse geometric function space theoretical background. Since we are comparing different representations over different spaces of different dimensions, we add to the theory presented in [18] relevant 'dimension-free' results. In Section 5 we show how to apply the

theory in practice. Specifically, we use a high dimensional sparse generalization of the equivalence of wavelet sparsity formulation and sparse Besov semi-norms. This allows us to numerically estimate a sparse Besov 'weak-type' smoothness index of a given function in any representation space (e.g hidden layer). Section 6 provides experimental results that demonstrate how our theory is able to explain empirical findings in various scenarios.

2. Deep Learning approximation models

2.1. **Deep Learning Architectures.** The two main applications in classical machine learning are regression and classification. To obtain a unified approach, we convert the classification problems into vector-valued regression problems as follows. For classification tasks, where each input is mapped into one of L classes, we assign to each class $1 \leq l \leq L$, the standard basis vector e_l. This vector valued formulation assures that there is no bias, as the distances in $\mathbb{R}^L$ between the class vector representations are equivalent.

Neural network architectures can be highly complex, but in this paper we focus on three simple models:

(i) Multilayer Perceptron (MLP) - An MLP network is a forward feed network $\tilde{f} : \mathbb{R}^n \to \mathbb{R}^L$

$$\tilde{f} = \Phi(\sigma_K \circ T_K \circ \dots \sigma_1 \circ T_1), \tag{2.1}$$

where each T_k is an affine linear transformation, $T_k : \mathbb{R}^{n_k} \to \mathbb{R}^{n_{k+1}}$, $T_k(x_k) = M_k x_k + b_k$, and the functions σ_k are pointwise nonlinear activation functions. One of the most popular nonlinear activation functions is the Rectified Linear Unit (ReLU)

$$\mathrm{ReLU}(x) := \begin{cases} 0, & -\infty < x < 0, \\ x, & 0 \leq x < \infty. \end{cases} \tag{2.2}$$

Observe that an MLP network (2.1), with σ_k =ReLU, $1 \leq k \leq K$, imposes a piecewise linear approximation over polyhedral subdomains in the input feature space.

Typically, in classification problems, the last layer is of dimension L and the function Φ in (2.1) is the softmax

$$\Phi(x_1, \cdots, x_L) = \frac{1}{\sum_{i=1}^{L} e^{x_i}} (e^{x_1}, e^{x_2}, \cdots, e^{x_L}),$$

which 'normalizes' the output of the network to be a vector whose components are in the range $[0, 1]$ and sum up to one. This design facilitates training the network to output the probabilities of a sample to belong to each of the L classes.

(ii) Convolution Neural Network (CNN) - A convolutional layer is in fact a special case of an MLP layer, where the linear transformation M_k is broken up to a finite number of compactly supported convolutional

operations on the input to the layer. This is useful if we are expecting 'local' correlations in the input and in some cases allows to reduce the number of network weights substantially. Applying convolutions requires to maintain the shape of the input signal through the layers. So for example, if the input is an image, the layers will be three dimensional, as a stack of two-dimensional 'feature maps' and the convolutions applied to the input layer k will be three dimensional. The dimension of the stack of two-dimensional 'feature maps' in the output layer $k+1$ is equal to the number of convolutions applied between the layers, where each convolution produces a 'feature map'.

(iii) Multiplicative Neural Network - Multiplicative networks provide a platform for a polynomial type of nonlinearity which has been found to be extremely powerful in the form of attention mechanisms as part of the Transformer network architectures [28]. Here we describe one useful variant which can be regarded as a multiplicative network of degree 2. Let $x_k \in \mathbb{R}^{n_k}$ be an input vector to the k-th layer. For each component $1 \leq i \leq n_{k+1}$ of the layer output $x_{k+1} \in \mathbb{R}^{n_{k+1}}$, one selects as part of the architecture two predetermined indices of the input space $1 \leq j_1(i), j_2(i) \leq n_k$, such that for $1 \leq i \leq n_{k+1}$,

$$x_{k+1}(i) = A_k(i)x_k(j_1(i))x_k(j_2(i)) + \sum_{j=1}^{n_k} M_k(i,j)x_k(j) + b_k(i),$$

where the vector $A_k \in \mathbb{R}^{n_{k+1}}$ is an additional set of network parameters. In this variant, there is no additional nonlinearity applied as part of the transformation between layers. The multiplicative structure of degree 2 is extremely useful in efficient representation of multivariate polynomials, leading to superior approximation properties over regular MLP networks. Also, one can regard one of the inputs to the multiplication, for example $x_k(j_1(i))$, as an adaptive context, or attention for the second input $x_k(j_2(i))$. This can be regarded as an adaptive generalization of MLP, where the weights depend on the input.

2.2. **Deep Learning Approximation Spaces.** One of the main goals of approximation theory is the characterization of the approximation spaces of an approximation algorithm using function spaces [15]. To this day, the characterization of deep learning approximation is still out of reach [19]. Still, one can certainly prove Jackson-type theorems that bound the degree of approximation using smoothness norms such as Sobolev or Besov. To this end, we first recall the Sobolev space $W_p^r(\Omega)$, $1 \leq p \leq \infty$, $\Omega \subseteq \mathbb{R}^n$, consisting of functions with distributional derivatives up to order r satisfying

$$\|f\|_{W_p^r(\mathbb{R}^n)} := \sum_{|\alpha| \leq r} \|\partial^\alpha f\|_{L_p(\Omega)} < \infty.$$

The semi-norm is given by

$$|f|_{W_p^r(\mathbb{R}^n)} = |f|_{r,p} := \sum_{|\alpha|=r} \|\partial^\alpha f\|_{L_p(\Omega)}.$$

In this work we are focused on cases where the dimension n is large, since even the simplest classification network trained on a computer vision dataset consisting of very small images of size 32×32, produces inner layer dimensions of up to $n = 16,000 - 64,000$ neurons. However, when trying to approximate using a network functions from the unit ball of $W_p^r([0,1]^n)$, one encounters the 'curse of dimensionality'. That is, in the worst case, approximation with error $\varepsilon > 0$, requires a network of size $\sim \varepsilon^{-n/r}$ (e.g. there are lower bounds [34, 33]). In contrast, the following result demonstrates, that with additional mild conditions, the size of the network may depend linearly or quadratically on the dimension n (see definitions in Subsection 2.1). These results are useful in cases where $n >> r$, that is, the dimension is relatively higher than the given smoothness.

Theorem 2.1. [4] *Let $n \geq 2$, $r \in \mathbb{N}$, $f \in W_2^r(\mathbb{R}^n)$ such that $\|\hat{f}\|_{L_1(\mathbb{R}^n)} < \infty$, where $\hat{f}$ is the Fourier transform of f and $0 < \varepsilon \leq 1$. Then there exist:*

(i) *A deep MLP network $\tilde{f}_{MLP}$ with ReLU activations, of depth $O(n^2\varepsilon^{-2/r})$ and $O(n^2\varepsilon^{-(2+2/r)})$ neurons, such that*

$$\|f - \tilde{f}_{MLP}\|_{L_2([0,1]^n)} \leq c \max(|f|_{r,2}^2, \|\hat{f}\|_1)\varepsilon.$$

(ii) *A deep multiplicative network $\tilde{f}_{MUL}$ of depth $O(n\varepsilon^{-1/r})$ and $O(n\varepsilon^{-(2+1/r)})$ neurons, such that*

$$\|f - \tilde{f}_{MUL}\|_{L_2([0,1]^n)} \leq c \max(|f|_{r,2}^2, \|\hat{f}\|_1)\varepsilon.$$

Returning to our discussion of the characterization of the approximation spaces, the main difficulty lies with the inverse embedding. It is not trivial to conclude the smoothness of a given function from the behavior of the degree of DL approximation. Let us provide the well-known counter-example of the 'sawtooth' functions (see also [12] for a more in depth discussion).

Let $\Delta_1 : \mathbb{R} \to [0,1]$ be defined by

$$\Delta_1(x) := \begin{cases} 2x, & 0 \leq x < 1/2, \\ -2x + 2, & 1/2 \leq x \leq 1, \\ 0, & \text{else.} \end{cases} \tag{2.3}$$

We then define $\Delta_j := \Delta_{j-1} \circ \Delta_1$, for $j \geq 2$. The sawtooth function Δ_j has 2^{j-1} 'teeth', as depicted in Figure 3. The function Δ_1 has the following representation

$$\begin{aligned} \Delta_1(x) &:= 2(x)_+ - 2(2x-1)_+ + 2(x-1)_+ \\ &= \sigma(2x) - 2\sigma(2x-1) + \sigma(2x-2), \end{aligned}$$

where σ is the ReLU nonlinear activation. Therefore, it can be realized using a network block composed of two MLP layers

$$x \to (2x, 2x, 2x) + (0, -1, -2) \to \sigma \to (x_1, x_2, x_3) \to x_1 - 2x_2 + x_3.$$

This implies that Δ_j, $j \geq 1$, can be realized as a composition of j blocks, which is a network of depth $2j$.

We would like to quantify the 'weak-type' smoothness of the representation layers of the sawtooth family. To this end recall that for a function $f \in L_\tau(\Omega)$, $0 < \tau \leq \infty$, $h \in \mathbb{R}^n$ and $r \in \mathbb{N}$, we have the r-th order difference operator

$$\Delta_h^r(f, x) = \Delta_h^r(f, \Omega, x) := \sum_{k=0}^{r} (-1)^{r+k} \begin{pmatrix} r \\ k \end{pmatrix} f(x + kh),$$

where we assume the segment $[x, x + rh]$ is contained in Ω. Otherwise, we set $\Delta_h^r(f, \Omega, x) = 0$. The modulus of smoothness of order r is defined by

$$\omega_r(f, t)_\tau := \sup_{|h| \leq t} \|\Delta_h^r(f, \Omega, \cdot)\|_{L_\tau(\Omega)}, \quad t > 0,$$

where for $h \in \mathbb{R}^n$, $|h|$ denotes the norm of h. We also denote

$$\omega_r(f, \Omega)_\tau := \omega_r\left(f, \frac{diam(\Omega)}{r}\right)_\tau. \tag{2.4}$$

We also recall a subset of the classic Besov spaces defined by functions in $L_\tau(\Omega)$, $0 < \tau < \infty$, for which

$$|f|_{\mathcal{B}_\tau^\alpha(\Omega)} := \left(\int_0^\infty \left(t^{-\alpha} \omega_r(f, \Omega, t)_\tau\right)^\tau \frac{dt}{t}\right)^{1/\tau} < \infty, \tag{2.5}$$

with $\alpha > 0$, $r \geq \lfloor \alpha \rfloor + 1$. The (quasi-)norm is then defined by

$$\|f\|_{\mathcal{B}_\tau^\alpha(\Omega)} := \|f\|_{L_\tau(\Omega)} + |f|_{\mathcal{B}_\tau^\alpha(\Omega)}.$$

It can be shown [19] that for any $\alpha < 1/\tau$

$$|\Delta_j|_{\mathcal{B}_\tau^\alpha} \sim 2^{j\alpha}. \tag{2.6}$$

This is a pro-typical example of a function that, one the one hand, has a compositional structure, that allows it to be represented by a relatively shallow network of depth $\sim j$ and yet its Besov smoothness semi-norm is exponentially larger with j. That is, when initially inspecting the smoothness of the input function, it is not obvious that it can be approximated or, as in this special case, even realized, by a relatively shallow network.

In going further, let us denote by $E_N(f)_p$ the degree of approximation of a neural network with width of size $\sim n$ and depth of size N of a function $f \in L_p([0, 1]^n)$. Let us recall the approximation space $\mathcal{A}_\infty^\gamma(L_p)$, defined by functions in L_p for which

$$|f|_{\mathcal{A}_\infty^\gamma(L_p)} := \sup_N N^\gamma E_N(f)_p < \infty.$$

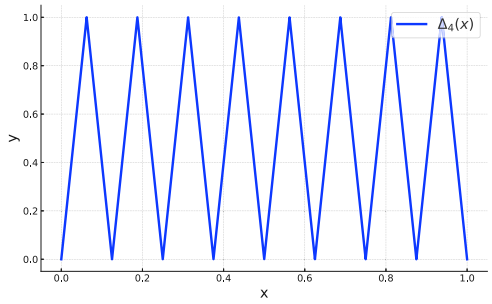

FIGURE 3. Sawtooth function Δ_4

Since Δ_j can be realized by a network of width 3 and $2j$ layers, we have that

$$E_N(\Delta_j)_p \sim \begin{cases} 1, & N < j/2, \\ 0, & N \geq j/2. \end{cases}$$

Therefore

$$|\Delta_j|_{\mathcal{A}^\gamma_\infty(L_p)} \sim j^\gamma.$$

Combining this observation with (2.6), we see that we do not have a continuous embedding of $\mathcal{B}^\alpha_\tau$ in $\mathcal{A}^\gamma_\infty(L_p)$ for any values of $\alpha, \gamma > 0$.

Yet, with the realization of $f_0 := \Delta_j$ as a neural network, we have that $|f_0|_{\mathcal{B}^\alpha_\tau} \sim 2^{j\alpha}$ and then, as we proceed trough the layers, the Besov semi-norm of the inner layer representations decreases exponentially, since $f_2 = \Delta_{j-1}$, $f_4 = \Delta_{j-2}$, etc. That is, $|f_2|_{\mathcal{B}^\alpha_\tau} \sim 2^{(j-1)\alpha}$, $|f_4|_{\mathcal{B}^\alpha_\tau} \sim 2^{(j-2)\alpha}$, etc.

Observe that one can easily construct more examples of such compositional 'sawtooth-type' functions. For example, let $1 < A < B < \infty$ and for each $1 \leq i \leq j$, pick a random parameter $a_i \in [A, B]$. Next define the piecewise linear 'tooth' with support in $[0, 1]$ realized by

$$\Delta_{a_i}(x) := a_i(x)_+ - \frac{a_i}{a_i - 1}(a_i x - 1)_+ + \frac{a_i}{a_i - 1}(x - 1)_+, \quad 1 \leq i \leq j.$$

One has $\Delta_{a_i}(a_i^{-1}) = 1$. Then, the 'sawtooth-type' function

$$\tilde{\Delta} := \Delta_{a_1} \circ \cdots \circ \Delta_{a_j},$$

also has 2^{j-1} teeth and also satisfies $|\tilde{\Delta}|_{\mathcal{B}^\alpha_\tau} \sim 2^{j\alpha}$. The sawtooth with uniform knots Δ_j corresponds to the choices $a_i = 2$, $1 \leq i \leq j$. Furthermore, $\tilde{\Delta}$ can also be realized using a network of depth $\sim j$, with the Besov semi-norm of the inner layer representation spaces also decaying exponentially. All this, with constants that further depend on A, B.

In the general case, it is known that every piecewise linear function on $[0, 1]$ over N (possibly non-uniform) knots can be realized by a network of fixed width and $\sim N$ layers [12, 34]. Recall that in this work we are focused on the analysis of the dynamics of the Besov smoothness across the inner

layer representations. To this end, we now show that one can provide a relatively simple and stable NN realization of depth $\sim N$ of the linear spline such that the $\mathcal{B}^\alpha_\tau$ smoothness of the inner layer representations is guaranteed to decrease at rate $\leq c(N-k+2)^{1/\tau}$, $k = 1, ..., N+2$.

Theorem 2.2. *Let $f : [0,1] \to \mathbb{R}$ be a continuous piecewise linear function with N knots and let $\alpha < 1/\tau$. Then, there exists a the neural network realization of f of width 5 and depth $N+3$, where the Besov semi-norms of the inner layer representations $f_k : \Omega \to \mathbb{R}$, $\Omega := [0,1]^4 \times [-\|f\|_\infty, \|f\|_\infty]$, $2 \leq k \leq N+2$, are bounded by*

$$|f_k|_{\mathcal{B}^\alpha_\tau(\Omega)} \leq c(\alpha, \tau, \|f\|_\infty)(N-k+2)^{1/\tau}. \tag{2.7}$$

Proof. Let f be a piecewise linear over $[0,1]$, with knots $0 < \xi_1 < \xi_2 < \cdots < \xi_N < 1$. We augment the knots by $-0.5 =: \xi_{-1} < 0 =: \xi_0 < \xi_1 < \xi_2 < \cdots < \xi_N < \xi_{N+1} := 1 < \xi_{N+2} := 1.5$. There are many ways to realize f using a ReLU network [12, 34]. Here, we use a construction that provides us with uniformly bounded inner layer representation domains. We use the non-uniform linear B-spline representation

$$f(x) = \sum_{k=0}^{N+1} c_j H_j(x), \quad x \in [0,1],$$

where for $0 \leq k \leq N+1$

$$H_k(x) := \frac{f(\xi_k)}{\xi_k - \xi_{k-1}}(x-\xi_{k-1})_+ - \frac{f(\xi_k)(\xi_{k+1}-\xi_{k-1})}{(\xi_k-\xi_{k-1})(\xi_{k+1}-\xi_k)}(x-\xi_k)_+ + \frac{f(\xi_k)}{\xi_{k+1}-\xi_k}(x-\xi_{k+1})_+.$$

Observe that H_k has support in $[\xi_{k-1}, \xi_{k+1}]$, with two linear pieces in $[\xi_{k-1}, \xi_k]$ and $[\xi_k, \xi_{k+1}]$, interpolating $H(\xi_k) = f(\xi_k)$.

The first inner layer is produced by the transformation

$$M_1 = [1,1,1,1,0]^T, \quad b_1 = [0, -\xi_{-1}, -\xi_0, \xi_1, 0]^T,$$

followed by ReLU activation. Thus, the feature space of the first inner layer is

$$(x, (x-\xi_{-1})_+, (x-\xi_0)_+, (x-\xi_1)_+, 0),$$

which provides a representation of f using this feature space by

$$f_1(x, z_1, z_2, z_3, 0) = f(x).$$

The second layer is given by the transformation

$$M_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & \frac{f(\xi_0)}{\xi_0-\xi_{-1}} & -\frac{f(\xi_0)(\xi_1-\xi_{-1})}{(\xi_0-\xi_{-1})(\xi_1-\xi_0)} & \frac{f(\xi_0)}{\xi_1-\xi_0} & 1 \end{bmatrix}, \qquad b_2 = \begin{bmatrix} 0 \\ -\xi_0 \\ -\xi_1 \\ -\xi_2 \\ 0 \end{bmatrix},$$

followed by ReLU activation. This gives a representation space

$$(x, (x-\xi_0)_+, (x-\xi_1)_+, (x-\xi_2)_+, H_0(x)),$$

which we may parameterize as

$$(x, z_1, z_2, z_3, y),$$

to obtain

$$f_2(x, z_1, z_2, z_3, y) = y + \sum_{k=1}^{N+1} H_k(x) = f(x).$$

In general, the $k-th$ inner layer, $2 \le k \le N+3$, is produced by

$$M_k = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & \frac{f(\xi_{k-2})}{\xi_{k-2}-\xi_{k-3}} & -\frac{f(\xi_{k-2})(\xi_{k-1}-\xi_{k-3})}{(\xi_{k-2}-\xi_{k-3})(\xi_{k-1}-\xi_{k-2})} & \frac{f(\xi_{k-2})}{\xi_{k-1}-\xi_{k-2}} & 1 \end{bmatrix}, \qquad b_k = \begin{bmatrix} 0 \\ -\xi_{k-2} \\ -\xi_{k-1} \\ -\xi_k \\ 0 \end{bmatrix},$$

followed by ReLU activation. This gives a representation space

$$(x, (x-\xi_{k-1})_+, (x-\xi_k)_+, (x-\xi_{k+1})_+, \sum_{j=0}^{k-2} H_j(x)),$$

which we may use to reproduce f by

$$f_k(x, z_1, z_2, z_3, y) = y + \sum_{j=k-1}^{N+1} H_j(x) = f(x). \tag{2.8}$$

We see that the y component serves as a collocation channel through the network. This means that as we proceed trough the layers, the number of nonlinearities in the feature spaces of f_k decreases linearly with k. We claim this implies the decrease (2.7) on the bound of the $|f_k|_{\mathcal{B}^\alpha_\tau(\Omega)}$ with k. Indeed, for any $\alpha > 0$, let us choose $r \ge \max(\lfloor\alpha\rfloor + 1, 2)$. It is well known that choosing any $r \ge \lfloor\alpha\rfloor + 1$ gives an equivalent Besov norm.

We use the representation (2.8) to bound the semi-norms $|f_k|_{\mathcal{B}^\alpha_\tau}$. Denote $s_{\min} := \min_{j=-1,...,N+1}\{\xi_{j+1} - \xi_j\} > 0$, fix $2 \le k \le N+1$, let $0 < h_1 < s_{\min}/r$ and $h := (h_1, 0, 0, 0)$. The function f_k has $N-k+2$ knots in the first coordinate. Denote

$$S_h := \{x = (x_1, ..., x_5) \in \Omega : \quad [x_1, x_1 + rh_1] \quad \text{does not intersect a knot}\}.$$

Then

$$|S_h| \ge 1 - (N-k+2)rh_1.$$

Since $\Delta^r_h(f_k, x) = 0$, for $x \in S_h$ and $r \ge 2$

$$\begin{aligned} \int_\Omega |\Delta^r_h(f_k, x)|^\tau dx &= \int_{\Omega \setminus S_h} |\Delta^r_h(f_k, x)|^\tau dx \\ &\le c\|f\|^{1+\tau}_{L_\infty[0,1]}(N-k+2)h_1. \end{aligned}$$

This implies for $t < s_{\min}/r$

$$\begin{aligned}\omega_r(f_k,t)^\tau_{L_\tau(\Omega)} &= \sup_{|h|\le t} \|\Delta_h^r(f_k,\cdot)\|_\tau^\tau \\ &= \sup_{h=(h_1,0,0,0,0),|h_1|\le t} \|\Delta_h^r(f_k,\cdot)\|_\tau^\tau \\ &\le c\|f\|^\tau_{L_\infty[0,1]}(N-k+2)t.\end{aligned}$$

Using the condition $\alpha < 1/\tau$

$$\begin{aligned}\int_0^{s_{\min}/r} \left(t^{-\alpha}\omega_r(f_k,t)_{L_\tau(\Omega)}\right)^\tau \frac{dt}{t} &\le c\|f\|^{1+\tau}_{L_\infty[0,1]}(N-k+2)\int_0^{s_{\min}/r} t^{-\alpha\tau}dt \\ &\le c(\alpha,\tau,f)(N-k+2).\end{aligned}$$

For $s_{\min}/r < t \le 1$ and $\tau \le 1$ we have

$$\begin{aligned}\omega_r(f_k,t)^\tau_\tau &\le \left(\frac{r}{s_{\min}}+1\right)^r \omega_r(f_k,s_{\min}/r)^\tau_\tau \\ &\le c(\alpha,\tau,f)(N-k+2).\end{aligned}$$

Using also the fact that Ω is bounded

$$\begin{aligned}\int_{s_{\min}/r}^\infty \left(t^{-\alpha}\omega_r(f_k,t)_{L_\tau(\Omega)}\right)^\tau \frac{dt}{t} &\le c(\alpha,\tau,f)\int_{s_{\min}/r}^1 \left(t^{-\alpha}\omega_r(f_k,t)_{L_\tau(\Omega)}\right)^\tau \frac{dt}{t} \\ &\le c(\alpha,\tau,f)(N-k+2).\end{aligned}$$

We may conclude that $|f_k|_{\mathcal{B}^\alpha_\tau(\Omega)} \le c(\alpha,\tau,f)(N-k+2)^{1/\tau}$. □

In this work, we analyze the phenomena of improved smoothness in feed forward neural networks' representation layers using sparse Besov spaces that are equivalent to classic Besov spaces in lower dimensions, yet are more adequate in higher dimensions.

3. Wavelet decomposition of Random Forests

To overcome the challenge of analyzing the smoothness of high dimensional representation spaces of datasets (e.g. inner layers in DL), we apply the construction of wavelet decompositions of Random Forests [18]. The Random Forest (RF) [5, 9, 21] introduced by Breiman [6, 7] as a machine learning algorithm, is in fact a powerful adaptive sparse piecewise polynomial approximation algorithm for high dimensional problems. The forest overcomes the 'greedy' nature and high variance of a single decision tree. The geometric wavelets [13, 18] used to decompose the forest allow sparse representations and have many properties that are shared with the classical wavelets [11, 26] that are not computationally feasible in high dimensions (in dimension $n = 100,000$, there are $2^{100,000}-1$ classic tensor wavelet 'types'). When combined, the wavelet decomposition of the RF unravels the sparsity of the underlying function and establishes an order of the RF nodes from 'important' components to 'negligible' noise. Therefore, the method provides a better understanding of any constructed RF. As we shall see, our

motivation to use these wavelet decompositions is the equivalence between sparse Besov smoothness over forests and the wavelet sparsity (see (4.5)).

We begin with an overview of single trees. In statistics and machine learning the construction is called a Decision Tree or the Classification and Regression Tree (CART) [8, 1, 5, 21]. Assume we are given a real-valued function $f \in L_p([0,1]^n)$ or a discrete dataset $\{x_i \in [0,1]^n, f(x_i)\}_{i\in I}$ (the generalization to any convex bounded domain $\Omega_0 \subset \mathbb{R}^n$ is trivial). The goal is to compute an adaptive piecewise polynomial approximation of f, even for large dimensions n (potentially in the range of hundreds of thousands). To this end, we subdivide the initial domain $\Omega_0 := [0,1]^n$ into two subdomains, e.g. by intersecting it with a hyper-plane. The goal is to find a subdivision that approximately minimize a given cost function. This subdivision process then continues recursively on the subdomains until some stopping criterion is met, which in turn, determines the leaves of the tree. We now describe one instance of the cost function which is related to minimizing variance. At each stage of the subdivision process, at a certain node of the tree, the algorithm finds, for the convex domain $\Omega \subset \mathbb{R}^n$ associated with the node:

(i) A partition by an hyper-plane into two convex subdomains Ω', Ω'', $\Omega' \cup \Omega'' = \Omega$.
(ii) Two multivariate polynomials $Q_{\Omega'}, Q_{\Omega''} \in \Pi_{r-1}(\mathbb{R}^n)$, of fixed (typically low) total degree r $-$ 1.

The partition and the polynomials are chosen to minimize the following quantity

$$\|f - Q_{\Omega'}\|^p_{L_p(\Omega')} + \|f - Q_{\Omega''}\|^p_{L_p(\Omega'')}. \tag{3.1}$$

For the theory that follows, we require that the polynomials $Q_{\Omega'}, Q_{\Omega''}$ provide local near best approximation on their respective subdomains. In applications, where the dataset is discrete, consisting of feature vectors $x_i \in [0,1]^n, i \in I$, with given values $f(x_i)$, a discrete functional is minimized over all partitions $\Omega' \cup \Omega'' = \Omega$

$$\sum_{x_i\in\Omega'} |f(x_i) - Q_{\Omega'}(x_i)|^p + \sum_{x_i\in\Omega''} |f(x_i) - Q_{\Omega''}(x_i)|^p. \tag{3.2}$$

Observe that for any given subdividing hyperplane, the approximating polynomials in (3.2) can be uniquely determined for $p = 2$, by least square minimization. For the order $r = 1$, the approximating polynomials are nothing but the mean of the function values over each of the subdomains

$$Q_{\Omega'}(x) = C_{\Omega'} = \frac{1}{\#\{x_i \in \Omega'\}} \sum_{x_i\in\Omega'} f(x_i),$$
$$Q_{\Omega''}(x) = C_{\Omega''} = \frac{1}{\#\{x_i \in \Omega''\}} \sum_{x_i\in\Omega''} f(x_i).$$

In many applications of decision trees, the high-dimensionality of the data does not allow to search through all possible subdivisions. As in our experimental results, one may restrict the subdivisions to the class of hyperplanes aligned with the main axes. In contrast, there are cases where one would like to consider more advanced form of subdivisions, where they take certain hyper-surface form or even non-linear forms through kernel Support Vector Machines. Our paradigm of wavelet decompositions can support in principle all of these forms.

Random Forest (RF) is a popular machine learning tool that collects decision trees into an ensemble model. The trees are constructed independently in a diverse fashion and prediction is done by a voting mechanism among all trees. A key element [6], is that large diversity between the trees reduces the ensemble's variance. There are many RFs variations that differ in the way randomness is injected into the model, e.g bagging, random feature subset selection and the partition criterion. Bagging [7] is a method that produces partial replicates of the training data for each tree. A typical approach is to randomly select for each tree a certain percentage of the training set (e.g. 80%).

Additional methods to inject randomness can be achieved at the node partitioning level. For each node, we may restrict the partition criteria to a small random subset of the parameter values. A typical selection is to search for a partition from a random subset of $\sqrt{n}$ features [6]. This technique is also useful for reducing the amount of computations when searching the appropriate partition for each node. Bagging and random feature selections are not mutually exclusive and can be used together.

For $j = 1, ..., J$, one creates a tree $\mathcal{T}_j$, based on a subset of the data, X^j. One then provides a weight (score) w_j to the tree $\mathcal{T}_j$, based on the estimated performance of the tree, where $\sum_{j=1}^{J} w_j = 1$. In the supervised learning, one typically uses the remaining data points $x_i \notin X^j$ to evaluate the performance of $\mathcal{T}_j$. For simplicity, we will mostly consider in this paper the choice of uniform weights $w_j = 1/J$. For any point $x \in \Omega_0$, the approximation associated with the j^{th} tree, denoted by $\tilde{f}_j(x)$, is computed by finding the leaf $\Omega \in \mathcal{T}_j$ in which x is contained and then evaluating $\tilde{f}_j(x_i) := Q_\Omega(x)$, where Q_Ω is the corresponding polynomial associated with the decision node Ω. One then assigns the RF approximate value to any point $x \in \Omega_0$ by

$$\tilde{f}(x) = \sum_{j=1}^{J} w_j \tilde{f}_j(x).$$

As already discussed, in classification problems, each input training point $x_i \in \Omega_0$ is assigned with a class $Cl(x_i)$. To convert the problem to the 'functional' setting described above, one assigns to the l-th class the standard basis vector $e_l \in \mathbb{R}^L$, $1 \leq l \leq L$. Thus, we may assume that the input data is in the form

$$\{x_i, Cl(x_i)\}_{i \in I} \in \left(\mathbb{R}^n, \mathbb{R}^L\right).$$

In this case, if we choose approximation using constants ($r = 1$), then the calculated mean over any subdomain Ω is in fact a point $\vec{E}_\Omega \in \mathbb{R}^L$. Obviously, any vector value $v = (v_1, ..., v_L) \in R^L$ can be mapped back to a class, along with an estimated confidence level, by calculating $\arg\max_{i=1}^{L} v_i$.

Next, we recall the construction of a wavelet decomposition of a forest [18]. Let Ω' be a child of Ω in a tree $\mathcal{T}$, i.e. $\Omega' \subset \Omega$ and Ω' was created by a partition of Ω. Denote by $\mathbf{1}_{\Omega'}$, the indicator function over the child domain Ω', i.e. $\mathbf{1}_{\Omega'}(x) = 1$, if $x \in \Omega'$ and $\mathbf{1}_{\Omega'}(x) = 0$, if $x \notin \Omega'$. We use the polynomial approximations $Q_{\Omega'}, Q_\Omega \in \Pi_{r-1}(\mathbb{R}^n)$, computed by the local minimization (3.1) and define

$$\psi_{\Omega'}(x) := \psi_{\Omega'}(f)(x) := \mathbf{1}_{\Omega'}(x)\left(Q_{\Omega'}(x) - Q_\Omega(x)\right), \tag{3.3}$$

as the geometric wavelet associated with the subdomain Ω' and the function f, or the given discrete dataset $\{x_i, f(x_i)\}_{i\in I}$. Each wavelet $\psi_{\Omega'}$, is a 'local difference' component that belongs to the detail space between two levels in the tree, a 'low resolution' level associated with Ω and a 'high resolution' level associated with Ω'. Also, the wavelets (3.3) have the 'zero moments' property, i.e., if the response variable is sampled from a polynomial of degree $r-1$ over Ω, then our local scheme will compute $Q_{\Omega'}(x) = Q_\Omega(x) = f(x)$, $\forall x \in \Omega$, and therefore $\psi_{\Omega'} = 0$. We also define $\psi_{\Omega_0} := Q_{\Omega_0}$.

Under certain mild conditions on the tree $\mathcal{T}$ and the function f, we have by the nature of the wavelets, the 'telescopic' sum of differences

$$f = \sum_{\Omega\in\mathcal{T}} \psi_\Omega, \qquad \psi_{\Omega_0} := Q_{\Omega_0}. \tag{3.4}$$

For example, (3.4) holds in L_p-sense, $1 \le p < \infty$, if $f \in L_p(\Omega_0)$ and for any $x \in \Omega_0$ and series of domains $\Omega_l \in \mathcal{T}$, each on a level l, with $x \in \Omega_l$, we have that $\lim_{l\to\infty} \operatorname{diam}(\Omega_l) = 0$.

The norm of a wavelet supported on a child Ω' of Ω is computed by

$$\|\psi_{\Omega'}\|_p^p = \int_{\Omega'} \left(Q_{\Omega'}(x) - Q_\Omega(x)\right)^p dx.$$

For the case $r = 1$, where $Q_\Omega(x) = C_\Omega$ and $Q_{\Omega'}(x) = C_{\Omega'}$ this simplifies to

$$\|\psi_{\Omega'}\|_p^p = |C_{\Omega'} - C_\Omega|^p \left|\Omega'\right|, \tag{3.5}$$

where $|\Omega'|$ denotes the volume of Ω'. Observe that for $r = 1$, the subdivision process for partitioning a node by minimizing (3.1) is equivalent to maximizing the sum of squared norms of the wavelets that are formed in that partition (see [18]).

Recall that our approach is to convert classification problems into a 'functional' setting by assigning the class labels to vector values in $\mathbb{R}^L$. In such cases of vector-valued functions, choosing $r = 1$, the wavelet $\psi_{\Omega'} : \mathbb{R}^n \to \mathbb{R}^L$ is

$$\psi_{\Omega'}(x) = \mathbf{1}_{\Omega'}(x)\left(\vec{E}_{\Omega'} - \vec{E}_\Omega\right),$$

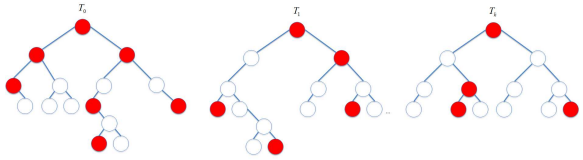

FIGURE 4. Selection of an M-term approximation from the entire forest.

and its norm is given by

$$\|\psi_{\Omega'}\|_p^p = \left\|\vec{E}_{\Omega'} - \vec{E}_{\Omega}\right\|_{l_2(\mathbb{R}^L)}^p \left|\Omega'\right|, \tag{3.6}$$

where for $\vec{v} \in \mathbb{R}^L$,$\|\vec{v}\|_{l_2} := \sqrt{\sum_{i=1}^L v_i^2}$.

Using any given weights assigned to the trees, we obtain a wavelet representation of the entire RF

$$\tilde{f}(x) = \sum_{j=1}^{J} \sum_{\Omega \in \mathcal{T}_j} w_j \psi_{\Omega}(x). \tag{3.7}$$

The theory tells us that, just as in the classical case [14], sparse approximation is achieved by reordering the wavelet components based on their norm [18]

$$w_{j(\Omega_{k_1})} \left\|\psi_{\Omega_{k_1}}\right\|_p \geq w_{j(\Omega_{k_2})} \left\|\psi_{\Omega_{k_3}}\right\|_p \geq \cdots, \tag{3.8}$$

with the notation $\Omega \in \mathcal{T}_j \Rightarrow j(\Omega) = j$. Thus, the adaptive M-term approximation of a RF is

$$f_M(x) := \sum_{m=1}^{M} w_{j(\Omega_{k_m})} \psi_{\Omega_{k_m}}(x). \tag{3.9}$$

Observe that, contrary to most existing tree pruning techniques [23], where each tree is pruned separately, the above approximation process applies a 'global' pruning strategy where the significant components can come from any node of any of the trees at any level. For simplicity, one could choose $w_j = 1/J$, and obtain

$$f_M(x) = \frac{1}{J} \sum_{m=1}^{M} \psi_{\Omega_{k_m}}(x). \tag{3.10}$$

Figure 4 depicts an M-term (3.10) selected from an RF ensemble. The red colored nodes illustrate the selection of the M wavelets with the highest norm values from the entire forest. Observe that they can be selected from any tree at any level, with no connectivity restrictions.

4. Geometric multivariate function space theory

An important research area of approximation theory, pioneered by Pencho Petrushev, is the characterization of adaptive geometric approximation algorithms by generalizations of the classic isotropic Besov spaces to sparse Besov-type spaces [10, 13, 22] that are more adequate for unstructured data in high-dimensions. We begin by defining the 'weak-type' sparse Besov smoothness of a function, subject to the geometry of a single (possibly adaptive) tree

Definition 4.1. For $\alpha < 1/\tau$, $r \geq 1$, $f \in L_\tau(\Omega_0)$, $\Omega_0 \subset \mathbb{R}^n$, and tree $\mathcal{T}$ over Ω_0, we define the associated sparse tree Besov smoothness in $\mathcal{B}_\tau^{\alpha,r}(\mathcal{T})$, $r \in \mathrm{N}$, by

$$|f|_{\mathcal{B}_\tau^{\alpha,r}(\mathcal{T})} := \left(\sum_{\Omega \in \mathcal{T}} \left(|\Omega|^{-\alpha} \, \omega_r(f,\Omega)_\tau \right)^\tau \right)^{1/\tau}, \tag{4.1}$$

where $|\Omega|$ denotes the volume of Ω.

In cases where we wish to approximate in the p-norm, $0 < p < \infty$ we shall typically set $1/\tau := \alpha + 1/p$. The higher the index α for which (4.1) is finite, the smoother the function is. Also, the above definition generalizes the classical function space theory of Besov spaces, where the tree partitions are non-adaptive. In fact, classical Besov spaces are a special case, where the tree is constructed (non-adaptively) by partitioning over dyadic knots such that at levels which are multiples of n one obtains dyadic cubes. For these special dyadic trees $\mathcal{T}_D$, one has $\mathcal{B}_\tau^\alpha(\mathcal{T}_D) \sim \mathcal{B}_\tau^{\alpha n}$, where the latter space is the classic Besov semi-norm defined by (2.5).

We note that we do not impose the condition $r > \alpha$ for (4.1) as we have done for (2.5). In fact, as we shall see below, for most part we will use $r = 1$. We remind the reader that in certain configurations of classical Besov spaces $\mathcal{B}_\tau^{\alpha,r}$, where $r < \alpha$ and $1 \leq \tau \leq \infty$, we get a trivial space [15] (e.g. polynomials of degree $r-1$ when Ω is a segment and $\{0\}$ for $\Omega = \mathbb{R}$). However, this is not the case here, since whenever $\alpha \geq 1$, we immediately have $\tau < 1$. For example, any $f(x) := \mathbf{1}_{\tilde{\Omega}}(x)$, where $\tilde{\Omega} \subset \mathbb{R}^n$ is a compact domain with a smooth boundary, satisfies $f \in \mathcal{B}_\tau^{\alpha,r}(\mathcal{T}_D)$, for any $r \geq 1$ and $\alpha < 1/\tau$.

For a given forest $\mathcal{F} = \{\mathcal{T}_j\}_{j=1}^J$ and weights $w_j = 1/J$, the α sparse Besov semi-norm associated with the forest is

$$|f|_{\mathcal{B}_\tau^{\alpha,r}(\mathcal{F})} := \frac{1}{J} \left(\sum_{j=1}^J |f|^\tau_{\mathcal{B}_\tau^{\alpha,r}(\mathcal{T}_j)} \right)^{1/\tau}. \tag{4.2}$$

Definition 4.2. Given a (possibly adaptive) forest representation, we define the sparse Besov smoothness index of f by the maximal index α for which (4.2) is finite.

Remark It is known that different geometric approximation schemes are characterized by different flavors of Besov-type smoothness. In this work, for example, all of our experimental results compute smoothness of representations using partitions along the main n axes. This restriction may lead, in general, to potentially lower Besov smoothness of the underlying function and lower sparsity of the wavelet representation. Yet, the theoretical definitions and results of this paper can also apply to more generalized schemes where, for example, tree partitions are performed using arbitrary hyper-planes. In such a case, the smoothness index of a given function may increase.

Next, for a given tree $\mathcal{T}$ and parameter $0 < \tau < p$, we denote the τ-sparsity of the tree by

$$N_\tau(f,\mathcal{T}) = \left(\sum_{\Omega \neq \Omega_0, \Omega \in \mathcal{T}} \|\psi_\Omega\|_p^\tau \right)^{1/\tau}. \tag{4.3}$$

Let us further denote the τ-sparsity of a forest $\mathcal{F}$, by

$$\begin{aligned} N_\tau(f,\mathcal{F}) &:= \frac{1}{J} \left(\sum_{j=1}^{J} \sum_{\Omega \neq \Omega_0, \Omega \in \mathcal{T}_j} \|\psi_\Omega\|_p^\tau \right)^{1/\tau} \\ &= \frac{1}{J} \left(\sum_{j=1}^{J} N_\tau(f,\mathcal{T}_j)^\tau \right)^{1/\tau}. \end{aligned}$$

In the setting of a single tree constructed to represent a real-valued function and under mild conditions on the partitions (see remark after (3.4) and condition (4.7)), the theory of [13] proves the equivalence

$$|f|_{\mathcal{B}_\tau^{\alpha,r}(\mathcal{T})} \sim N_\tau(f,\mathcal{T}). \tag{4.4}$$

Here, we assume $1/\tau := \alpha + 1/p$ and that the wavelets are local differences of near-best local polynomial approximations from the space of polynomials of degree $r-1$ (see (3.1)). This implies that there are constants $0 < C_1 < C_2 < \infty$, that depend on parameters such as α, p, n, r and ρ in condition (4.7) below, such that

$$C_1 |f|_{\mathcal{B}_\tau^{\alpha,r}(\mathcal{T})} \leq N_\tau(f,\mathcal{T}) \leq C_2 |f|_{\mathcal{B}_\tau^{\alpha,r}(\mathcal{T})}.$$

Therefore, we also have for the forest model

$$|f|_{\mathcal{B}_\tau^{\alpha,r}(\mathcal{F})} \sim N_\tau(f,\mathcal{F}). \tag{4.5}$$

Next, we present a simple invariance property of the smoothness analysis under higher dimension embedding.

Lemma 4.3. *Let $f : [0,1]^n \to \mathbb{R}^L$, $f \in L_p([0,1]^n)$ and let $\mathcal{F}$ be a forest approximation of f. For any $m \geq 0$, let $\tilde{x} = (x, 0, ..., 0) \in [0,1]^{n+m}$, $x \in [0,1]^n$. Let us further define $\tilde{f}(\tilde{x}) := f(x)$. Next, denote by $\tilde{\mathcal{F}}$ a forest defined over $[0,1]^{n+m}$ which is the natural extension of $\mathcal{F}$, using the same*

trees with same partitions over the first n coordinates. Then, for $r = 1$ and any $\tau > 0$,

$$N_\tau\left(\tilde{f}, \tilde{\mathcal{F}}\right) = N_\tau\left(f, \mathcal{F}\right).$$

Proof. Let $\Omega' \in \mathcal{F}$ be one the domains of the trees of $\mathcal{F}$, with wavelet of the type

$$\psi_{\Omega'}(x) = \mathbf{1}_{\Omega'}(x)\left(\vec{E}_{\Omega'} - \vec{E}_{\Omega}\right).$$

Recall that $N_\tau\left(f, \mathcal{F}\right)$ is the l_τ norm of the sequence of the wavelet norms given by (3.6).

Now, for each domain $\Omega' \in \mathcal{F}$ and the corresponding domain $\tilde{\Omega}' \in \tilde{\mathcal{F}}$, using the normalization of the feature spaces ensures that

$$\left|\Omega'\right| = \left|\Omega'\right| \times \left|[0,1]^m\right| = \left|\tilde{\Omega}'\right|. \tag{4.6}$$

Since the vector means remain unchanged under the higher dimensional embedding $\vec{E}_{\Omega'} = \vec{E}_{\tilde{\Omega}'}$, we have using (4.6) and (3.6)

$$\left\|\psi_{\Omega'}\right\|_{L_p([0,1]^n)} = \left\|\psi_{\tilde{\Omega}'}\right\|_{L_p([0,1]^{n+m})}.$$

This gives $N_\tau\left(\tilde{f}, \tilde{\mathcal{F}}\right) = N_\tau\left(f, \mathcal{F}\right)$. □

Remark Note that the above invariance property also holds if the additional m redundant features are intertwined with the n significant features that are 'detected' and used for tree subdivisions during the construction of the random forest.

In the setting in which we wish to apply our function theoretical approach, we are comparing smoothness of representations over different layers of DL networks. This implies that we are analyzing and comparing the smoothness of a set of functions f_k, each over a different representation space of a different dimension n_k. This is, in some sense, non-standard in function space theory, where the space, or at least the dimension, over which the functions have their domain is typically fixed. Specifically, observe that the equivalence (4.5) depends on the dimension n of the feature space. To this end, we add to the theory 'dimension-free' analysis for the case $r = 1$.

We begin with a Jackson-type estimate for the degree of the adaptive wavelet forest approximation, which we keep 'dimension free' for the case $r = 1$. To this end, let $\mathcal{F} = \{\mathcal{T}_j\}_{j=1}^J$ be a forest. Assume there exists a constant $0 < \rho < 1$, such that for any domain $\Omega \in \mathcal{F}$ on a level l and any domain $\Omega' \in \mathcal{F}$, on the level $l+1$, with $\Omega \cap \Omega' \neq \emptyset$, we have

$$\left|\Omega'\right| \leq \rho\left|\Omega\right|. \tag{4.7}$$

For any $r \geq 1$, denote formally $f = \sum_{\Omega \in \mathcal{F}} w_{j(\Omega)}\psi_\Omega$, and assume that $N_\tau(f, \mathcal{F}) < \infty$, where

$$\frac{1}{\tau} = \alpha + \frac{1}{p}.$$

Under these conditions, it is proved in [18] that $f \in L_p$, and that the following Jackson estimate holds for the wavelet forest M-term approximation (3.9)

$$\sigma_M(f) := \|f - f_M\|_p \leq C\,(p,\alpha,\rho,r,n)\,JM^{-\alpha}N_\tau(f,\mathcal{F}). \tag{4.8}$$

Here, we observe that for the case $r = 1$, one can remove the dependence of the constant on the dimension

Theorem 4.4. *Under the above conditions on $\mathcal{F}$, for $r = 1$ and the M-term approximation (3.9) we have*

$$\sigma_M(f) := \|f - f_M\|_p \leq C\,(p,\alpha,\rho)\,JM^{-\alpha}N_\tau(f,\mathcal{F}). \tag{4.9}$$

Proof. We essentially follow the proof in [18]. To see (4.9) we observe that the dimension n comes into play in the Nikolskii-type estimate for bounded convex domains $\Omega \subset \mathbb{R}^n$, and $r \geq 1$

$$\|\psi_\Omega\|_\infty \leq c(p,n,r)|\Omega|^{-1/p}\|\psi_\Omega\|_p\,.$$

However, for the special case of $r = 1$ this actually simplifies to

$$\|\psi_\Omega\|_\infty = |\Omega|^{-1/p}\|\psi_\Omega\|_p\,.$$

□

Using the Jackson estimate (4.8) and the equivalence (4.5), we get for any $r \geq 1$

$$\sigma_M(f) \leq C\,(p,\alpha,\rho,n)\,JM^{-\alpha}\,|f|_{\mathcal{B}_\tau^{\alpha,r}(\mathcal{F})}\,,$$

which is not a 'dimension-free' Jackson estimate, as the one will show below for $r = 1$ (see (4.14)).

Next, to allow our smoothness analysis to be 'dimension free' we modify the modulus of smoothness (2.4) for $r = 1$ and use the following form of 'averaged modulus'

Definition 4.5. For a function $f : \Omega \to \mathbb{R}^L$ we define

$$w_1(f,\Omega)_\tau := \left(\int_\Omega \left\|f(x) - \vec{E}_\Omega\right\|_{l_2(\mathbb{R}^L)}^\tau dx\right)^{1/\tau}, \tag{4.10}$$

where $\vec{E}_\Omega$ is the average of f over Ω.

It is well known that averaged forms of the modulus are equivalent to the form (2.4), but with constants that depend on the dimension. However, replacing (2.4) with (4.10) allows us to produce 'dimension-free' analysis. We use (4.10) to define

$$|f|_{\tilde{\mathcal{B}}_\tau^\alpha(\mathcal{T})} := \left(\sum_{\Omega\in\mathcal{T}}\left(|\Omega|^{-\alpha}w_1(f,\Omega)_\tau\right)^\tau\right)^{1/\tau},$$

$$|f|_{\tilde{\mathcal{B}}_\tau^\alpha(\mathcal{F})} := \frac{1}{J}\left(\sum_{j=1}^J |f|^\tau_{\tilde{\mathcal{B}}_\tau^\alpha(\mathcal{T}_j)}\right)^{1/\tau}.$$

We can now show

Theorem 4.6. *Let $f : \Omega_0 \to \mathbb{R}^L$. Then the following equivalence holds for the case $r = 1$,*

$$|f|_{\tilde{\mathcal{B}}^{\alpha}_{\tau}(\mathcal{F})} \sim N_{\tau}(f, \mathcal{F}), \tag{4.11}$$

where $1/\tau = \alpha + 1/p$, and the constants of equivalence depend on α, τ, ρ, but not n.

Proof. Obviously, it is sufficient to prove the equivalence for a single tree $\mathcal{T}$. Observe that condition (4.7) also implies that for any $\Omega' \in \mathcal{T}$, with parent Ω, we also have

$$|\Omega| \leq (1-\rho)^{-1} \left|\Omega'\right|.$$

We use this as well as (3.6) to prove the first direction of the equivalence as follows

$$\begin{aligned}
N_{\tau}(f,\mathcal{T})^{\tau} &= \sum_{\Omega' \neq \Omega_0, \Omega' \in \mathcal{T}} \|\psi_{\Omega'}\|_p^{\tau} \\
&= \sum_{\substack{\Omega' \neq \Omega_0, \Omega' \in \mathcal{T}, \\ \Omega \text{ parent of } \Omega'}} \left(\left|\Omega'\right|^{1/p} \left\| \vec{E}_{\Omega'} - \vec{E}_{\Omega} \right\|_{l_2(\mathbb{R}^L)} \right)^{\tau} \\
&= \sum_{\Omega' \neq \Omega_0, \Omega' \in \mathcal{T}} \left(\left|\Omega'\right|^{1/p - 1/\tau} \|\psi_{\Omega'}\|_{\tau} \right)^{\tau} \\
&\leq c(\tau) \sum_{\substack{\Omega' \neq \Omega_0, \Omega' \in \mathcal{T} \\ ,\Omega \text{ parent of } \Omega'}} \left\{ \left(\left|\Omega'\right|^{-\alpha} \left\| \left\| f(\cdot) - \vec{E}_{\Omega'} \right\|_{l_2(\mathbb{R}^L)} \right\|_{L_{\tau}(\Omega')} \right)^{\tau} \right. \\
&\qquad \left. + \left(\left|\Omega'\right|^{-\alpha} \left\| \left\| f(\cdot) - \vec{E}_{\Omega} \right\|_{l_2(\mathbb{R}^L)} \right\|_{L_{\tau}(\Omega')} \right)^{\tau} \right\} \\
&\leq c(\tau,\rho,\alpha) \sum_{\Omega \in \mathcal{T}} \left(|\Omega|^{-\alpha} \left\| \left\| f(\cdot) - \vec{E}_{\Omega} \right\|_{l_2(\mathbb{R}^L)} \right\|_{L_{\tau}(\Omega)} \right)^{\tau} \\
&= c(\tau,\rho,\alpha) \sum_{\Omega \in \mathcal{T}} \left(|\Omega|^{-\alpha} w_1(f,\Omega)_{\tau} \right)^{\tau} \\
&= c\, |f|^{\tau}_{\tilde{\mathcal{B}}^{\alpha}_{\tau}}.
\end{aligned}$$

We now prove the other direction. We assume $0 < \tau \leq 1$ (the case $1 < \tau < \infty$ is similar). For any $\Omega \in \mathcal{T}$ we have

$$w_1(f,\Omega)_{\tau}^{\tau} \leq \sum_{\Omega' \in \mathcal{T}, \Omega' \subset \Omega} \|\psi_{\Omega'}\|_{\tau}^{\tau}, \tag{4.12}$$

by the following estimates

$$\begin{aligned} w_1\left(f,\Omega\right)_\tau^\tau &= \int_\Omega \left\| \sum_{\Omega'\in\mathcal{T}} \psi_{\Omega'}\left(x\right) - \vec{E}_\Omega \right\|_{l_2(\mathbb{R}^L)}^\tau dx \\ &= \int_\Omega \left\| \sum_{\Omega'\in\mathcal{T}} \psi_{\Omega'}\left(x\right) - \sum_{\Omega'\in\mathcal{T},\Omega\subseteq\Omega'} \psi_{\Omega'}\left(x\right) \right\|_{l_2(\mathbb{R}^L)}^\tau dx \\ &= \int_\Omega \left\| \sum_{\Omega'\in\mathcal{T},\Omega'\subset\Omega} \psi_{\Omega'}\left(x\right) \right\|_{l_2(\mathbb{R}^L)}^\tau dx \\ &\leq \sum_{\Omega'\in\mathcal{T},\Omega'\subset\Omega} \left\|\psi_{\Omega'}\right\|_\tau^\tau. \end{aligned}$$

Also, observe that by condition (4.7), for any $\Omega'\in\mathcal{T}$

$$\sum_{\Omega\in\mathcal{T},\Omega'\subset\Omega} \left(\frac{|\Omega'|}{|\Omega|}\right)^{\alpha\tau} \leq \sum_{k=1}^{\infty} \rho^{k\alpha\tau} \leq c\left(\rho,\alpha,\tau\right). \tag{4.13}$$

We apply (4.12) and (4.13) to conclude

$$\begin{aligned} |f|_{\tilde{\mathcal{B}}_\tau^\alpha(\mathcal{T})}^\tau &\leq \sum_{\Omega\in\mathcal{T}} |\Omega|^{-\alpha\tau} \sum_{\Omega'\in\mathcal{T},\Omega'\subset\Omega} \left\|\psi_{\Omega'}\right\|_\tau^\tau \\ &= \sum_{\Omega\in\mathcal{T}} \sum_{\Omega'\in\mathcal{T},\Omega'\subset\Omega} \left(\frac{|\Omega'|}{|\Omega|}\right)^{\alpha\tau} \left(\left|\Omega'\right|^{-\alpha} \left\|\psi_{\Omega'}\right\|_\tau\right)^\tau \\ &= \sum_{\Omega'\neq\Omega_0,\Omega'\in\mathcal{T}} \left(\left|\Omega'\right|^{-\alpha} \left\|\psi_{\Omega'}\right\|_\tau\right)^\tau \sum_{\Omega\in\mathcal{T},\Omega'\subset\Omega} \left(\frac{|\Omega'|}{|\Omega|}\right)^{\alpha\tau} \\ &\leq c\left(\alpha,\tau,\rho\right) \sum_{\Omega'\neq\Omega_0,\Omega'\in\mathcal{T}} \left(\left|\Omega'\right|^{-\alpha} \left|\Omega'\right|^{1/\tau-1/p} \left\|\psi_{\Omega'}\right\|_p\right)^\tau \\ &= c\left(\alpha,\tau,\rho\right) \sum_{\Omega'\neq\Omega_0,\Omega'\in\mathcal{T}} \left\|\psi_{\Omega'}\right\|_p^\tau \\ &= cN_\tau(f,\mathcal{T})^\tau. \end{aligned}$$

□

The equivalence (4.11) together with (4.9) imply that for $r=1$ we do have a 'dimension-free' Jackson estimate

$$\sigma_M(f) \leq C\left(p,\alpha,\rho\right) JM^{-\alpha} \left|f\right|_{\tilde{\mathcal{B}}_\tau^\alpha(\mathcal{F})}. \tag{4.14}$$

5. Smoothness analysis of the representation layers in deep learning networks

We now explain how the theory presented in Section 4 is used to estimate the 'weak-type' smoothness of a given function in a given representation

layer. Recall from the introduction that we create a representation of images at layer 0 by concatenating the $\sqrt{n_0}$ rows of pixel values of each grayscale image, to create a vector of dimension n_0 (or $3 \times n_0$ for a color image). We also normalize the pixel values to the range $[0, 1]$. We then transform the class labels into vector-values in the space $\mathbb{R}^L$ by assigning each label l, the standard basis vector $e_l \in \mathbb{R}^L$ (see Section 3). Thus, the images are considered as samples of a function $f_0 : [0, 1]^{n_0} \to \mathbb{R}^L$. In the same manner, we associate with each k-th layer of a DL network, a function $f_k : [0, 1]^{n_k} \to \mathbb{R}^L$, where n_k is the number of features/neurons at the k-th layer. The samples of f_k are obtained by applying the network on the original images up the given k-th layer. For example, in a convolution layer, we capture the representations after the cycle of convolution, non-linearity and pooling. We then extract vectors created by normalizing and concatenating the feature map values corresponding to the images. Recall that although the functions $\{f_k\}$ are embedded in different dimensions $\{n_k\}$, through the simple normalizing of the features, our method is able to assign smoothness indices to each layer that are comparable.

Next we describe how we estimate the smoothness of each function f_k using the method of [3]. We compute a random forest $\mathcal{F}_k$ over the samples of f_k with the choice $r = 1$ and then apply the wavelet decomposition of the RF (see Section 3). We work under the assumption that at each layer, we have a sufficient number of samples relative to k-th feature space dimension n_k and that we construct a forest $\mathcal{F}_k$ which is diverse enough to capture the 'geometry' of f_k.

With $p = 2$, for each k, the goal is to numerically estimate

$$\tau_k^* := \inf_{0<\tau<2} \{\tau | \quad N_\tau(f_k, \mathcal{F}_k) < \infty\}.$$

Based on the either the equivalence (4.5) or the dimension free equivalence (4.11) for $r = 1$, this gives an estimate for critical (optimal) smoothness of each f_k

$$\alpha_k^* = \frac{1}{\tau_k^*} - \frac{1}{2}. \tag{5.1}$$

It is obvious that for any $f \in L_2([0, 1]^n)$ and random forest $\mathcal{F}$ built adaptively to fit it, the univariate function $N_\tau(f, \mathcal{F})$, as function of τ is monotonically non-increasing. Yet, in discrete setting, there is no value $0 < \tau < 2$, however small, for which $N_\tau(f, \mathcal{F}) = \infty$. So the estimate of a 'critical' τ^* is a numeric estimate for where the derivative (as a function of τ)

$$\frac{d}{d\tau} N_\tau(f, \mathcal{F}) = \frac{1}{J} \frac{d}{d\tau} \left(\sum_{j=1}^{J} \sum_{\Omega \neq \Omega_0, \Omega \in \mathcal{T}_j} \|\psi_\Omega\|_2^\tau \right)^{1/\tau},$$

crosses some lower threshold.

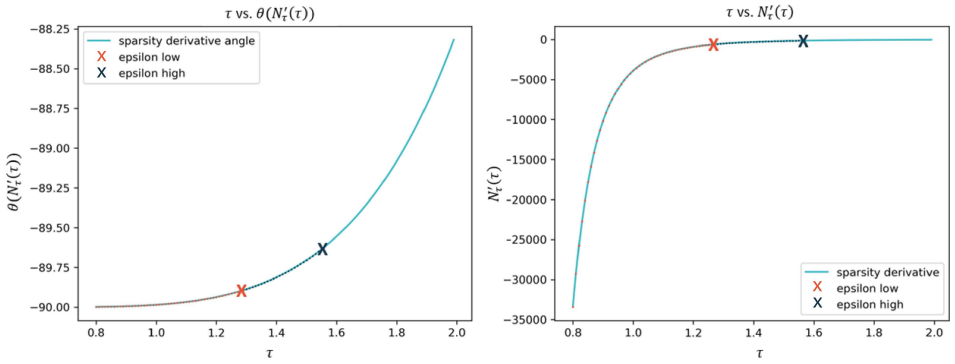

FIGURE 5. Example of plots of N'_τ and $\theta(N'_\tau)$

First, we compute a series of samples $N_{\tau_i}(f, \mathcal{F})$, for a set of discrete evenly spaced samples $\{\tau_i\}_{i=1}^M$, $0 < \tau_i < 2$. We then approximate, using the samples, the numerical derivatives

$$N'_i \approx \frac{d}{d\tau} N_{\tau_i}(f, \mathcal{F}), \quad 1 \le i \le M.$$

We use the angles of the numerical derivatives

$$\theta_i := \arctan(N'_i),$$

to estimate the transition index τ^* which is associated with an infinite derivative, or equivalently, an angle of $-\pi/2$. To this end, we use two meta parameters: $\varepsilon_{\text{low}}, \varepsilon_{\text{high}}$, and define

$$S := \{\tau_i : -\frac{\pi}{2} + \varepsilon_{\text{low}} \le \theta_i \le -\frac{\pi}{2} + \varepsilon_{\text{high}}\}.$$

We then approximate the transition index by

$$\tau^* \approx -\varepsilon_{\text{low}} + \frac{1}{|S|} \sum_{\tau_i \in S} \tau_i.$$

A demonstration is shown in Figure 5.

6. Experimental Results: Smoothness analysis across DL representation layers

Our goal is to show how the theory laid out in this paper enables to look into the 'black box' of a neural network and analyze its performance. For all the experiments presented in this section 3 different neural networks were trained using the same training set, each time with different initialization seeds. Then, for each of the 3 networks, we applied the numerical computation of the representation layers' smoothness (5.1) using wavelet decompositions of RFs of three trees each, with maximal depth of 15 layers. The hyper parameters of Section 5 were selected as $\varepsilon_{\text{low}} = 0.1$, $\varepsilon_{\text{high}} = 0.4$. The smoothness estimates for each layer were averaged over the 3 trained

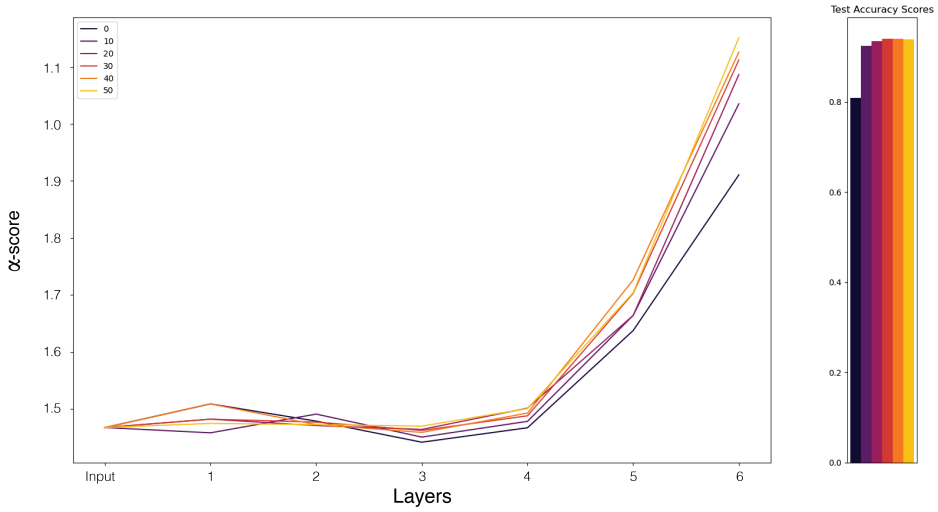

FIGURE 6. Smoothness improvement through intermediate layers and the training phase

models. It is important to note that in the results below the function space smoothness estimates are performed using only the training dataset. Then, testing of the accuracy of the models is performed using out of bag testing dataset. Nevertheless, as we shall see below, there is a strong correlation between the rate of improvement of smoothness across the model's intermediate layers and its performance in the testing phase.

6.1. **Analysis of a classic well-performing convolutional neural network.** We start off with a simple example of a relatively simple dataset and an appropriate and well functioning neural network architecture. In this example we use MNIST-1D [27], a well known univariate equivalent of the classic MNIST image dataset. In MNIST-1D, as opposed to MNIST, the signals are intertwined in the input space, and are then much less separable. We apply to it a classic convolutional network with 7 layers (see Subsection 2.1). The question is, how does the representation smoothness improve across layers and throughout the training phase? To this end we monitor for each k-th layer, the smoothness α_k^* (see (5.1)) and record it every 10 epochs of the training phase. The results are shown in figure 6. While obviously the smoothness of the input dataset at layer 0 is constant throughout the training, it is clear that the smoothness improves as we progress forward through the intermediate layers and through the training. Also notice the correlation with the testing result depicted on the right hand side, although the smoothness of the intermediate layers is numerically estimated using only the training data.

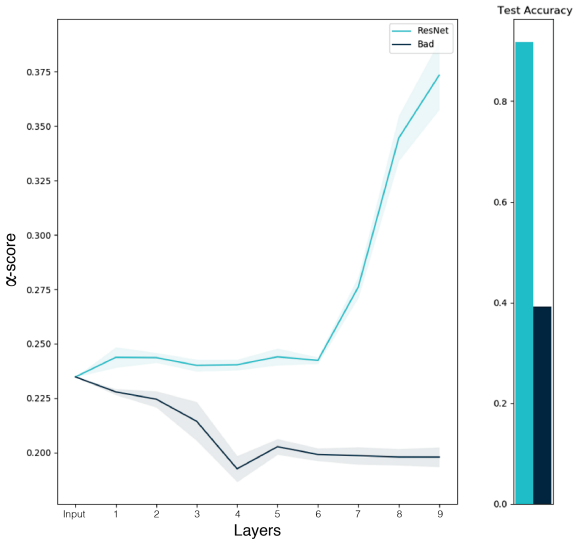

FIGURE 7. Analysis of Fashion-MNIST classification models: a 'good' ResNet architecture and a 'bad' alternating architecture.

6.2. **Analysis of a 'problematic' neural network architecture.** In most cases, where there is a problem with the chosen network architecture, we can only hopelessly observe the bad testing result. This is especially frustrating in cases where the problematic architecture is still able to overfit the training data and produce a low training loss. To demonstrate this, we use the well known Fashion-MNIST image dataset and create two types of classification models. The first model is based on a 'sensible' architecture of convolutional residual networks [24]. For the second model, we deliberately construct an architecture that does not make sense from a computer-theoretical perspective. Specifically, we construct a network where the types of layers alternate between convolutional and MLP. Thus, every convolutional layer is followed by reshaping of the 3D feature space (sequence of 2D feature maps) into a vector and then an MLP layer. In Figure 7 we see the analysis of the two models. We see that the 'bad' architecture does not support improvement of the intermediate layers which leads to low testing performance.

6.3. **Analysis of performance of activation functions.** In Subsection 2.1 we reviewed the role of the nonlinear activation functions and provided some examples. Here we use again the MNIST-1D dataset and fix a convolutional neural network as the classification model, where the only hyperparameter we change for each run is the activation function. Suppose we are

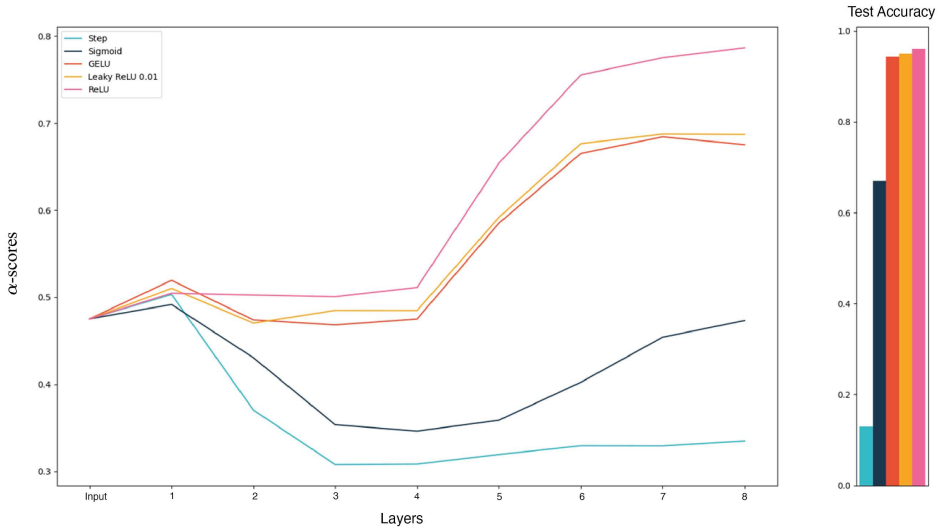

FIGURE 8. Analysis of MNIST-1D classification models: Different nonlinear activation functions.

trying to create a new activation function, by using a simple step function:

$$\sigma_H(x) = \begin{cases} 1, & \text{if } x > 0. \\ 0, & \text{otherwise.} \end{cases}$$

At face value, this seems a potentially problematic activation function since it does not output any magnitude values, only simple activation of '0' or '1'. Other nonlinearities we analyze here are the ReLU (2.2) its variants Leaky ReLU and GELU and the Sigmoid. Figure 8 shows the dominance of the ReLU activation over other activations. The ReLU variants - Leaky ReLU and GELU are relatively close in performance. The Sigmoid is indeed far lower in terms of α^*, and we see a dip during train. Again, we see a clear correlation of the architectural smoothness space analysis using the training data and the testing performance of the different activations.

Acknowledgment We deeply thank the referee for comments that resulted in a significantly improved paper.

REFERENCES

[1] E. Alpaydin, *Introduction to machine learning*, MIT Press, 2004.
[2] Y. Bengio, A. Courville and P. Vincenty, *Representation Learning: A Review and New Perspectives*, IEEE Transactions on Pattern Analysis and Machine Intelligence 8 (2013), 1798–1828.
[3] I. Ben-Shaul and S. Dekel, *Sparsity-Probe: analysis tool for deep learning models*, https://arxiv.org/abs/2105.06849.
[4] I. Ben-Shaul, T. Galanti and S. Dekel, *Exploring the approximation capabilities of multiplicative neural networks for smooth functions*, Transactions of Machine Learning 2023.

[5] G. Biau and E. Scornet, *A random forest guided tour*, TEST 25 (2016), 197–227.
[6] L Breiman, *Random forests*, Machine Learning 45 (2001), 5–32.
[7] L. Breiman, *Bagging predictors*, Machine Learning 24 (1996), 123–140.
[8] L. Breiman, J. Friedman, C. Stone and R. Olshen, *Classification and Regression Trees*, Chapman and Hall/CRC, 1984.
[9] A. Criminisi, J. Shotton and E. Konukoglu, *Forests for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning*, Microsoft Research technical report, report TR-2011-114, 2011.
[10] W. Dahmen, S. Dekel and P. Petrushev, *Two-level-split decomposition of anisotropic Besov spaces*, Constructive approximation 31 (2001), 149-194.
[11] I. Daubechies, *Ten lectures on wavelets*, CBMS-NSF Regional Conference Series in Applied Mathematics,1992.
[12] I. Dubechies, R. DeVore, S. Foucart, B. Hanin and G. Petrova, *Nonlinear approximation and (deep) ReLU networks*, Constructive approximation 55 (2022), 127-172.
[13] S. Dekel and D. Leviatan, *Adaptive multivariate approximation using binary space partitions and geometric wavelets*, SIAM Journal on Numerical Analysis 43 (2005), 707–732.
[14] R. DeVore, *Nonlinear approximation*, Acta Numerica 7 (1998), 51–150.
[15] R. DeVore and G. Lorentz, *Constructive approximation*, Springer Science and Business, 1993.
[16] R. DeVore, B. Jawerth and B. Lucier, *Image compression through wavelet transform coding*, IEEE transactions on information theory 38 (1992), 719-746.
[17] D.Donoho, M. Vetterli, R DeVore and I. Daubechies *Data compression and harmonic analysis*, IEEE Transactions on information theory 44 (1998), 2435-2476.
[18] O. Elisha and S. Dekel, *Wavelet decompositions of Random Forests - smoothness analysis,sparse approximation and applications*, Journal of machine learning research 17 (2016), 1-38.
[19] R. Gribonval, G. Kutyniok, M. Nielsen and F. Voigtlaender, *Approximation spaces of deep neural networks*, Constructive Approximation 55 (2022), 259-367.
[20] P. Grohs and G. Kutyniok (ed.), *Mathematical aspects of deep learning*, Cambridge University press, 2022.
[21] T. Hastie, R. Tibshirani and J. Friedman, *The elements of statistical learning*, Springer, 2009.
[22] B. Karaivanov and P. Petrushev, *Nonlinear piecewise polynomial approximation beyond Besov spaces*, Applied and computational harmonic analysis 15 (2003), 177-223.
[23] V. Kulkarni and P. Sinha, *Pruning of Random Forest classifiers: A survey and future directions*, In International Conference on data science and engineering, 64-68, 2012.
[24] K. He, X. Zhang, S. Ren and Jian Sun, *Deep Residual Learning for Image Recognition*, IEEE Conference on Computer Vision and Pattern Recognition 2016, 770-778.
[25] W. Loh, *Classification and regression trees*, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 1 (2011), 14-23.
[26] S. Mallat, *A Wavelet tour of signal processing, 3rd edition (the sparse way)*, Acadmic Press, 2009.
[27] Sam Greydanus, *Scaling down Deep Learning*, arXiv:2011.14439, 2020.
[28] M. Phuong and M. Hutter, *Formal algorithms for transformers*, DeepMind report, arXiv:2207.09238 (2022).
[29] T. Poggio, H. Mhaskar, L. Rosasco, B. Miranda and Q. Liao, *Why and when can deep-but not shallow-networks avoid the curves of dimensionality: a review*, International Journal of Automation and computing 14 (2017), 503-519.
[30] P. Salembier and L. Garrido, Binary partition tree as an efficient representation for image processing, segmentation, and information retrieval, *IEEE transactions on image processing*9:561-576, 2000.

[31] R. Schwartz-Ziv and A. Armon, *Tabular data:deep learning is not all you need*, Information Fusion 81 (2022), 84-90.
[32] R. Schwartz-Ziv and N. Tishbi, *Opening the black box of Deep Neural Networks via Information*, http://arxiv.org/abs/1703.00810.
[33] J. Siegel, *Optimal Approximation Rates for Deep ReLU Neural Networks on Sobolev and Besov Spaces*, JMLR 24 (2023), 152.
[34] D. Yarotsky, *Error bounds for approximations with deep ReLU networks*, Neural Networks 94 (2017), 103-114.

(I. Ben-Shaul) School of mathematical sciences, Tel-Aviv University
E-mail address: ido.benshaul@gmail.com

(S. Dekel) School of mathematical sciences, Tel-Aviv University
E-mail address: shaidekel6@gmail.com

(O. Elisha) Microsoft Research, Israel
E-mail address: Oren.Elisha@microsoft.com