Discontinuous Neural Networks and Discontinuity Learning

Francesco Della Santa[a,b,c], Sandra Pieraccini[a,c,*]

[a] *Department of Mathematical Sciences, Politecnico di Torino, Turin, Italy*
[b] *SmartData@PoliTO center for Big Data and Machine Learning technologies*
[c] *Member of the INdAM-GNCS research group*

Abstract

In the framework of discontinuous function approximation and discontinuity interface detection, we consider an approach involving Neural Networks. In particular, we define a novel typology of Neural Network layers endowed with new learnable parameters and discontinuities in the space of the activations. These layers allow to create a new kind of Neural Networks, whose main property is to be discontinuous, able not only to approximate discontinuous functions but also to learn and detect the discontinuity interfaces. A sound theoretical analysis concerning the properties of the new discontinuous layers is performed, and some tests on discontinuous functions are proposed, in order to assess the potential of such instruments.

Keywords: Discontinuous Functions, Neural Networks, Deep Learning, Automatic Detection of Discontinuity Interface.
2020 MSC: 68T07, 65D15

1. Introduction

In this work, we introduce a new typology of layers for Neural Networks (NNs); the novelty is given by the introduction of discontinuities in the layer's characterizing function and, consequently, in the Neural Network. In the framework of NNs, discontinuities were involved in the first mathematical models of biological neurons, dating back to the 1940s [24], and the first Neural Networks proposed in the 1950s and 1960s [29, 32]. In these models, the activation functions of the NN units were mainly inspired by the mechanisms of the biological neurons and, therefore, were modeled using the Heaviside step function, or suitable variants. Then, the activation functions evolved into the continuous (and often smooth) ones used nowadays in almost all the deep learning algorithms (see [10, ch. 6.2.2, 6.3] and [4]), thanks to the advantages that they grant in

[*] Corresponding author
Email address: `sandra.pieraccini@polito.it` (Sandra Pieraccini)

adapting the parameters of a NN. To the best of authors' knowledge, recent literature does not report examples of practical use of discontinuous NN layers or discontinuous NNs; nonetheless, a renewed interest on discontinuous activation functions, at least from the theoretical point of view, is witnessed by very recent works, see e.g. [30], in which the floor function $f(x) = \lfloor x \rfloor$ is used as activation function, and [26], which uses the Heaviside function.

We are interested in introducing discontinuities in NN learning models, aiming at using feedforward NNs to detect the discontinuity interfaces while approximating discontinuous functions. The problem of detecting discontinuity interfaces is quite a challenging task, especially for functions with a high-dimensional domain. Moreover, the information can be quite relevant in several applications. To mention an example, the smoothness of the target function can critically affect the behavior of numerical methods for stochastic collocation in the framework of uncertainty quantification; thus, knowing the discontinuity interfaces and being able to partition the function domain in several regions in which the function is smooth, can be of paramount importance (see, e.g., [17] and references therein).

In the past decades, feedforward NNs have been used mainly for classification-type tasks [20, 11, 31] but they can perform very well also regression tasks, as guaranteed by the universal approximation theorems [21, 28, 18, 26]. In particular, we recall that the universal approximation theorem of Leshno et al. [21] is guaranteed also for discontinuous activation functions, while Park et al. recently showed (see [26, Th. 3]) that a NN with `relu` and Heaviside activation functions is dense in the space of continuos functions from a compact set $K \subset \mathbb{R}^n$ to $\mathbb{R}^m$.

Concerning the approximation of discontinuous functions with NNs, interesting results have been recently obtained in [27, 13, 14]. In particular in [13, 14] the problem is deeply investigated comparing deep NN estimators to other more classical methods in the task of approximating piecewise-smooth functions with singularities on smooth hypersurfaces in their domain. In particular, according to the analysis in [14], the superiority of deep NNs depends on the relative level of smoothness of the function pieces and of the discontinuity interfaces. The problem of approximation of nonsmooth functions using NNs is addressed in [23] from a different perspective. Indeed, in [23] the focus is on the use of NNs for approximating numerical solutions of partial differential equations; the work addresses in particular the case in which the PDE is defined on a polygonal or polyhedral domain possibly yielding a solution with corner and/or edge singularities. The analysis in [23] present rigorous estimates of the approximation errors also in these situations.

A precursor of the use of NN in the framework of approximation of discontinuous functions can be found in [7]. In such a paper the author, leveraging the interpretation of shallow NNs as the superposition of ridge functions, uses a general construction based on ridgelets to build shallow NNs, thus presenting a tool for approximating target functions with spatial inhomogeneities. Functions with linear discontinuity interfaces are well approximated with such an approach, but the method is yet not satisfying for curvilinear interfaces, as stated by the author.

Overall, the NN models mentioned above are characterized by good approximation skills for discontinuous functions but, on the other hand, they are not suitable nor designed to simultaneously tackle the discontinuity interface detection problem; indeed, the approximation of a discontinuous function with a NN is quite a simple task, but the function represented by the NN will actually be continuous, whenever continuous activation functions are used in the NN layers. Therefore, the aim of this work is to propose a new NN layer specifically designed to build NNs able to simultaneously perform these two tasks.

As far as the discontinuity detection problem is concerned, the main results have been proposed in the last decades, see e.g. [6, 5, 16, 33]. In [6] a polynomial annihilation edge detection method is proposed: the discontinuous interfaces of a piece-wise smooth function $f : \mathbb{R}^n \to \mathbb{R}$, $n \leq 2$, are identified through the reconstruction of the jump function, given a set of function evaluations; the method proposed in [6] is extended in [5] to higher dimensions by applying the detection method for each input dimension to a generalized polynomial chaos approximation of the target function. Nonetheless, the method in [5] suffers the curse of dimensionality, setting practical restrictions on the dimensionality that can be handled; an improvement of [5] is proposed in [16], that exploits sparse grids to develop an adaptive method that increases the possibilities to be used in higher dimensions. The method proposed in [33] is based on the approximation of the hypersurface representing the discontinuity interface with hyper-spherical coordinates, and it is well suited also for large n, but the method is designed for decting a single interface, which is assumed to satisfy the star-convexity assumption.

We remark that in general, a discontinuity interface detection method can be generalized to the case of functions $\boldsymbol{F} : \mathbb{R}^n \to \mathbb{R}^m$ with $m > 1$, through the common practice of applying m times the method for functions with codomain of dimension one.

The present work aims at building new discontinuous NNs able to approximate discontinuous functions with other discontinuous functions, whose discontinuity interfaces are relatively easy to be detected. More specifically, the new NNs are endowed with trainable discontinuity jump parameters that allow the model to learn the discontinuity interface of the target function $\boldsymbol{F} : \mathbb{R}^n \to \mathbb{R}^m$; then, analyzing the function compositions that define the NN, the discontinuity interfaces and the continuity regions of $\boldsymbol{F}$ in the domain can be characterized. The main advantages of this method are that the NN both returns an approximation of $\boldsymbol{F}$ and an approximation of its discontinuity interfaces. Moreover, by a theoretical point of view, there are no restrictions on the applicability of the method proposed, indeed: i) the method does not require assumptions about the regularity of $\boldsymbol{F}$ (at least, $\boldsymbol{F}$ must be approximable according to one of the universal approximation theorems); ii) there are no theoretical restrictions on the dimensions (there are only practical restrictions, due to the curse of dimensionality problem suffered by the regression task).

The work is organized as follows. In the next subsection, the main notations used herein are listed. In Section 2, the new discontinuous layer for NNs is presented. In Section 3, the theoretical results that characterize a discontinuous NN are described. Section 4 illustrates numerical results on some examples assessing the potential of the new discontinuous NNs. We end with some conclusions drawn in Section 5.

1.1. Notation and basic results

In this subsection we introduce some useful notations and simple results for the following sections. First, we introduce the notation adopted for the description of the inner operations taking place in a fully-connected layer of a *Multi Layer Perceptron* (MLP). Then, we introduce some notations that will be useful in Section 3 to analyze discontinuities in NNs.

1.1.1. Notation for Neural Networks

Let N be an H-layers perceptron; i.e., N is an MLP characterized by $H \in \mathbb{N}$ hidden layers. We use the following notation to describe its architecture and the related mathematical entities:

- $L_1, \dots, L_H$ denote the hidden layers; L_0 and L_{H+1} are the *input* and *output* layers, respectively;
- for each $h = 0\,, \dots, H+1$, $N_h \in \mathbb{N}$ is the number of units of layer L_h;
- for each $h = 0\,, \dots, H$, $W^{(h+1)} \in \mathbb{R}^{N_h \times N_{h+1}}$ is the matrix of weights between layers L_h and L_{h+1} and $\boldsymbol{b}^{(h+1)} \in \mathbb{R}^{N_{h+1}}$ is the vector of biases of layer L_{h+1};
- $\boldsymbol{f}_h : \mathbb{R}^{N_h} \to \mathbb{R}^{N_h}$ denotes the element-wise application of the activation function $f_h : \mathbb{R} \to \mathbb{R}$ used in layer L_h;
- for each $h = 0, \dots, H$, $\mathcal{L}_{h+1} : \mathbb{R}^{N_h} \to \mathbb{R}^{N_{h+1}}$ denotes the characterizing function of the fully-connected layer L_{h+1}, i.e. the map from the outputs of layer L_h to the ones of layer L_{h+1}, defined as

$$\mathcal{L}_{h+1}(\boldsymbol{x}^{(h)}) = \boldsymbol{f}_{h+1}\left(W^{(h+1)\,T}\boldsymbol{x}^{(h)} + \boldsymbol{b}^{(h+1)}\right)\,, \quad \text{with } \boldsymbol{x}^{(h)} \in \mathbb{R}^{N_h}\,. \quad (1)$$

- we generalize the notation used for maps between pairs of layers to sequences of layers as follows:

$$\mathcal{L}_{h_1}^{h_2} := \mathcal{L}_{h_2} \circ \mathcal{L}_{h_2-1} \circ \cdots \circ \mathcal{L}_{h_1+1} \circ \mathcal{L}_{h_1}\,,$$

  for each $1 \le h_1 < h_2 \le H+1$. Note that, following this notation, $\mathcal{L}_1^{H+1}$ is the characterizing function $\widehat{\boldsymbol{F}}$ of the perceptron N and $\mathcal{L}_{h_1}^{h_2}$ is the function characterizing a sub-NN of N given by layers $L_{h_1-1}, L_{h_1}, \dots, L_{h_2-1}, L_{h_2}$.

Remark 1.1. Equation (1) characterizes the action of the so-called fully-connected layers; however it is easy to prove that it can describe also the action of convolutional layers (e.g., see [22]) or the connection of layers that are not fully-connected, setting to zero specific elements of the weight matrix. Then, formula (1) can be used as representative of the general characterizing function of a NN layer. Analogously, almost any feedforward (i.e., non-recurrent) NN can be represented by an equivalent MLP or, at most, by a composition of MLPs; then, in this work, we use the H-layers perceptron N as representative of a generic feedforward NN.

1.1.2. Notation for hyperplanes and corresponding partitions

Let $\mathit{\Pi} = \{\Pi_1, \dots, \Pi_m\}$ be a set of hyperplanes in $\mathbb{R}^n$, each one characterized by the equation $\boldsymbol{x}^T\boldsymbol{w}_j + b_j = 0$, for $j = 1, \dots, m$. Then, we will use the following notation to denote some special subsets of $\mathbb{R}^n$ characterized by $\Pi_1, \dots, \Pi_m$:

- for each pair of disjoint subsets of hyperplanes $\{\Pi_{i_1}, \dots, \Pi_{i_s}\}, \{\Pi_{k_1}, \dots, \Pi_{k_t}\} \subset \mathit{\Pi}$ we denote by $C(\{\Pi_{i_1}, \dots, \Pi_{i_s}\}; \{\Pi_{k_1}, \dots, \Pi_{k_t}\})$ the subset of vectors $\boldsymbol{x} \in \mathbb{R}^n$ such that

$$\begin{cases} \boldsymbol{x}^T\boldsymbol{w}_i + b_i \geq 0 & \forall\ i = i_1, \dots, i_s \\ \boldsymbol{x}^T\boldsymbol{w}_k + b_k < 0 & \forall\ k = k_1, \dots, k_t \end{cases} .$$

  We observe that, if $C(\{\Pi_{i_1}, \dots, \Pi_{i_s}\}; \{\Pi_{k_1}, \dots, \Pi_{k_t}\})$ is not empty, then it is convex.

- the set $\mathit{\Pi}$ generates a partition $\mathcal{C}(\mathit{\Pi})$ of convex subsets of $\mathbb{R}^n$ defined as

$$\mathcal{C}(\mathit{\Pi}) = \left\{ C(P; P^C) \mid P \in \mathcal{P}(\mathit{\Pi}) \right\} \setminus \{\emptyset\}, \tag{2}$$

  where P^C is the complement of P in $\mathit{\Pi}$ and $\mathcal{P}(\mathit{\Pi})$ is the power set of $\mathit{\Pi}$.

Remark 1.2 (Special cases)**.** Let $\mathit{\Pi} = \{\Pi_1, \dots, \Pi_m\}$ be a set of hyperplanes in $\mathbb{R}^n$ and $\mathcal{C}(\mathit{\Pi})$ the partition (2). Then, the following special cases may occur:

1. Let $\Pi_i, \Pi_j \in \mathit{\Pi}$, $i \neq j$, be such that $\Pi_i = \Pi_j$ and $\boldsymbol{w}_i = a\boldsymbol{w}_j$, $b_i = ab_j$, for an $a \in \mathbb{R} \setminus \{0\}$. If $a < 0$, for each $P \in \mathcal{P}(\mathit{\Pi})$ such that $\Pi_i, \Pi_j \in P$, we have that the set $C(P; P^C)$ lies on the hyperplane $\Pi_i = \Pi_j$, while the set $C(P^C; P)$ is empty.
2. Let us admit as possible elements of $\mathit{\Pi}$ also the degenerate hyperplanes $\Pi_0 = \mathbb{R}^n$ and $\Pi_\emptyset = \emptyset$ defined by equations $\boldsymbol{x}^T\boldsymbol{0} + 0 = 0$ and $\boldsymbol{x}^T\boldsymbol{0} + b_\emptyset = 0$, respectively, where $b_\emptyset \neq 0$. Then, if $\Pi_0 \in \mathit{\Pi}$ and/or $\Pi_\emptyset \in \mathit{\Pi}$, we have that $\mathcal{C}(\mathit{\Pi})$ is still a partition of $\mathbb{R}^n$ in convex subsets and, in particular, $\mathcal{C}(\mathit{\Pi}) = \mathcal{C}(\mathit{\Pi} \setminus \{\Pi_0, \Pi_\emptyset\})$.

Each element $X \in \mathcal{C}(\mathit{\Pi})$ can be identified by a unique vector with elements in $\{0, 1\}$, as highlighted by the following definition.

Definition 1.1 (Region Vectors and Region Function)**.** *Let $\mathit{\Pi} = \{\Pi_1, \dots, \Pi_m\}$ be a set of m hyperplanes in $\mathbb{R}^n$, possibly including the degenerate cases Π_0 and $\Pi_\emptyset$, each one characterized by the equation $\boldsymbol{x}^T\boldsymbol{w}_j + b_j = 0$, for $j = 1, \dots, m$. Let $g : \mathbb{R}^n \to \{0, 1\}^m$ be the function*

$$g(\boldsymbol{x}) := \boldsymbol{\mathcal{H}}\left(W^T\boldsymbol{x} + \boldsymbol{b}\right), \tag{3}$$

where $\boldsymbol{\mathcal{H}}$ denotes the component-wise application of the Heaviside function

$$\mathcal{H}(x) = \begin{cases} 1 \ , & \textit{if } x \geq 0 \\ 0 \ , & \textit{otherwise} \end{cases}$$

and $W = [\boldsymbol{w}_1, \ldots, \boldsymbol{w}_m] \in \mathbb{R}^{n \times m}$ and $\boldsymbol{b} = [b_1, \ldots, b_m]^T \in \mathbb{R}^m$. The function g defined in (3) is called region function associated to $\mathit{\Pi}$, and $g(\boldsymbol{x}) \in \{0,1\}^m$ is called region vector of $\boldsymbol{x}$.

The region function g introduced in Definition 1.1 characterizes uniquely the subsets of $\mathbb{R}^n$ of the partition $\mathcal{C}(\mathit{\Pi})$, as stated in the following Lemma, whose proof is straightforward.

Lemma 1.1. *Let $\mathit{\Pi}$ be a set of m hyperplanes as in Definition 1.1. Then, for each pair of vectors $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^n$ such that $\boldsymbol{x}_1 \in X_1, \boldsymbol{x}_2 \in X_2$ with $X_1, X_2 \in \mathcal{C}(\mathit{\Pi})$, it holds that $X_1 = X_2$ if and only if $g(\boldsymbol{x}_1) = g(\boldsymbol{x}_2)$.*

Due to Lemma 1.1, each $X_i \in \mathcal{C}(\mathit{\Pi})$ is uniquely identified by a vector $\boldsymbol{k}_i \in \{0,1\}^m$ such that $\boldsymbol{k}_i = g(\boldsymbol{x})$, for each $\boldsymbol{x} \in X_i$.

Definition 1.2 (Region Vectors of Subsets)**.** *Let $\mathit{\Pi}$ be as in Definition 1.1, and let g be the region function associated to $\mathit{\Pi}$. Let $\boldsymbol{k}_i \in \{0,1\}^m$ be a vector such that $\boldsymbol{k}_i = g(\boldsymbol{x})$ for each $\boldsymbol{x} \in X_i$, given a fixed $X_i \in \mathcal{C}(\mathit{\Pi})$. Then $\boldsymbol{k}_i$ is called Region Vector of X_i with respect to the hyperplanes of $\mathit{\Pi}$.*

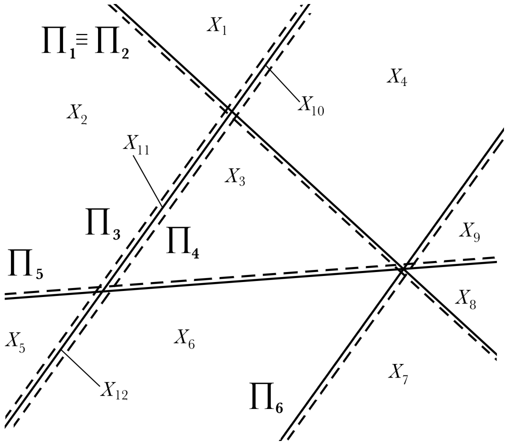
We end this section with an example of a partition $\mathcal{C}(\mathit{\Pi})$ of $\mathbb{R}^2$ and the corresponding region vectors; this example is illustrated in Figure 1. Six hyperplanes $\mathit{\Pi} = \{\Pi_1, \ldots, \Pi_6\}$ are considered, and some special cases are also included, as we have $\Pi_1 = \Pi_2$ and $\Pi_3 = -\Pi_4$; these situations correspond to the one discussed in Remark 1.2, item 1, with $a > 0$ and $a < 0$, respectively.

In this example, for each $X_i, X_j \in \mathcal{C}(\mathit{\Pi})$, we observe that the region vectors $\boldsymbol{k}_i$, $\boldsymbol{k}_j$, are strictly related to subsets connection. For example, looking at X_1 and X_3 we observe that their boundaries do intersect in the point given by the intersection $\bigcap_{k=1}^4 \Pi_k$, which are the hyperplanes identified by elements of $\boldsymbol{k}_1$ and $\boldsymbol{k}_3$ that are different. In other cases, such as X_3 and X_6 or X_1 and X_{10}, we observe that the intersection of the boundaries is *contained* in the intersection of the hyperplanes identified by the region vector elements that are different. More in general, still focusing on this example, let $\boldsymbol{k}_i$ and $\boldsymbol{k}_j$ differ for $1 \leq t \leq 6$ components of indices $\ell_1, \ldots, \ell_t$: if the shared boundary $\partial X_i \cap \partial X_j$ is not empty (the union of the closures $\bar{X}_i \cup \bar{X}_j$ is therefore a connected set), then

$$(\partial X_i \cap \partial X_j) \subseteq (\Pi_{\ell_1} \cap \cdots \cap \Pi_{\ell_t})\,;$$

otherwise, the set $\Pi_{\ell_1} \cap \cdots \cap \Pi_{\ell_t}$ does not intersect anyone of the boundaries ∂X_i, ∂X_j.

The relationship between the elements of the region vectors $\boldsymbol{k}_i, \boldsymbol{k}_j$ and connectivity of $\bar{X}_i \cup \bar{X}_j$, observed in the example of Figure 1, is generalizable to partitions in $\mathbb{R}^n$. However, the example is not intended to be exhaustive of all possible cases which may occur.



(a) Hyperplanes $\Pi_1, \ldots, \Pi_6$ and the corresponding sets $X_1, \ldots, X_{12}$.

	region vector ($\boldsymbol{k}_i$)					
reg.	Π_1	Π_2	Π_3	Π_4	Π_5	Π_6
X_1	1	1	0	1	0	1
X_2	0	0	0	1	0	1
X_3	0	0	1	0	0	1
X_4	1	1	1	0	0	1
X_5	0	0	0	1	1	1
X_6	0	0	1	0	1	1
X_7	0	0	1	0	1	0
X_8	1	1	1	0	1	0
X_9	1	1	1	0	0	0
X_{10}	1	1	1	1	0	1
X_{11}	0	0	1	1	0	1
X_{12}	0	0	1	1	1	1

(b) Region vectors of the sets $X_1, \ldots, X_{12}$ with respect to the hyperplanes $\Pi_1, \ldots, \Pi_6$.

Figure 1: Left: example of partition of $\mathbb{R}^2$ by six hyperplanes $\Pi_1, \ldots, \Pi_6$. Dotted lines denote the part of the plane with $\boldsymbol{x}^T\boldsymbol{w}_j + b_j < 0$, for each $j = 1, \ldots, 6$. Right: region vectors $\boldsymbol{k}_i \in \{0,1\}^6$ corresponding to each subset X_i, $i = 1, \ldots, 12$, of the partion.

2. Discontinuity for Neural Networks

Let N be an H-layers perceptron. We recall that the map $\mathcal{L}_{h+1}$, characterizing the transformations performed by layer L_{h+1} on the outputs of layer L_h, is defined by (1).

From this formula, it is straightforward to note that the characterizing function $\widehat{\boldsymbol{F}} = \mathcal{L}_1^{H+1} : \mathbb{R}^{N_0} \to \mathbb{R}^{N_{H+1}}$ of N is a continuous function if $\boldsymbol{f}_1, \ldots, \boldsymbol{f}_{H+1}$ are all continuous functions. In the earliest works on Neural Networks [24, 29], the first models for artificial neurons did not consider continuous activation functions but the Heaviside function (or suitable variations of it), mainly used to model the "on/off" activation of the neurons.

In the subsequent development of artificial intelligence, the Heaviside function $\mathcal{H}$ has been abandoned, as the subderivative constantly equal to zero prevents the use of the gradient descent during NN training; this phenomenon is equivalent to the asymptotic end of the so-called vanishing gradient problem (see [10, ch. 8.2.5]). In place of $\mathcal{H}$, many other continuous functions have been introduced; as a consequence, all recent NNs described in literature are characterized by continuous functions since they are given by the composition of continuous functions.

In this work, we describe a novel approach to reintroduce discontinuities in NNs in such a way that NNs can not only approximate discontinuous functions but also learn discontinuities. In the sequel, we will refer to this property of the NN as the ability of learning discontinuity interfaces, and such discontinuity interfaces will be called "learnable discontinuities".

2.1. Adding Heaviside to Activation Functions

The main idea behind learnable discontinuities for NNs is to apply the effects of a bias "outside" the activation function f only when the inputs satisfy certain conditions, for example to be not smaller than zero. Thanks to this new bias, the NN has a new trainable parameter that introduces a discontinuity in the function of the NN. The discontinuity introduced depends both on the new parameters and on the weights and biases of the NN; for this reason we will refer to *learnable discontinuities*.

Herein, in order to introduce possible discontinuites in the layers, we add to the activation functions a jump, whose size is expressed by a parameter ε. In details, we add to each element of the right-hand-side of (1) a multiple of the Heaviside function applied component-wise to $W^{(h+1)\,T}\boldsymbol{x}^{(h)} + \boldsymbol{b}^{(h+1)}$, namely we set:

$$\boldsymbol{x}^{(h+1)} = \boldsymbol{f}_{h+1}\left(W^{(h+1)\,T}\boldsymbol{x}^{(h)} + \boldsymbol{b}^{(h+1)}\right) + \boldsymbol{\varepsilon}^{(h+1)} \odot \boldsymbol{\mathcal{H}}\left(W^{(h+1)\,T}\boldsymbol{x}^{(h)} + \boldsymbol{b}^{(h+1)}\right), \tag{4}$$

where the symbol $\odot$ denotes the Hadamard (element-wise) product and $\boldsymbol{\varepsilon}^{(h+1)} \in \mathbb{R}^{N_{h+1}}$ is the vector collecting the N_{h+1} jumps introduced (see Figure 2).

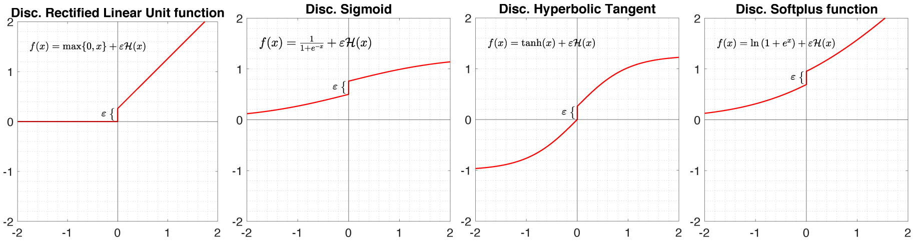


Figure 2: Examples of activation functions plus a multiple of $\mathcal{H}$.

Definition 2.1 (Discontinuous Layer)**.** *A discontinuous fully-connected layer* L *with input in* $\mathbb{R}^c$, *output in* $\mathbb{R}^d$, *and activation function* f *for the continuous part, is a layer with incoming connections and output signals defined by the characterizing function* $\mathcal{L} : \mathbb{R}^c \to \mathbb{R}^d$ *such that:*

$$\mathcal{L}(\boldsymbol{x}) = \boldsymbol{f}\left(W^T\boldsymbol{x} + \boldsymbol{b}\right) + \boldsymbol{\varepsilon} \odot \boldsymbol{\mathcal{H}}\left(W^T\boldsymbol{x} + \boldsymbol{b}\right), \tag{5}$$

where $\boldsymbol{\varepsilon} \in \mathbb{R}^d$ *is the vector of trainable discontinuity jumps and where* $\boldsymbol{f}$, W, *and* $\boldsymbol{b}$ *are the element-wise application of* f, *the weight matrix, and the bias vector, respectively.*

In the following, a discontinuous layer L *of a NN and the corresponding characterizing function* $\mathcal{L}$, *defined by* (5), *will be denoted by* δL *and* $\delta\mathcal{L}$, *respectively.*

Definition 2.2 (Discontinuous Neural Network)**.** *Let* N *be a NN with at least one discontinuous layer. Then* N *is called a discontinuous Neural Network and will be denoted by* δ*NN.*

We remark that the d parameters $\varepsilon_1, \dots, \varepsilon_d$ appearing in (5) are learnable parameters, since the derivatives of $\delta\mathcal{L}(\boldsymbol{x})$ with respect to them are not constantly equal to zero, as stated in the following Proposition.

Proposition 2.1. *Let y_j be the j-th element of $\delta\mathcal{L}(\boldsymbol{x}) =: \boldsymbol{y} \in \mathbb{R}^d$ given by (5). Then it holds $\partial\, y_j / \partial\, \varepsilon_j \not\equiv 0$ for each $j = 1, \dots, d$.*

Proof. The proof is straightforward since, for each $j = 1, \dots, d$, we have that

$$\partial\, y_j / \partial\, \varepsilon_j = \mathcal{H}\left(W_{\cdot\,,\,j}^T \boldsymbol{x} + b_j\right) ,$$

that is not constantly equal to zero. □

From the previous proposition, we can easily deduce also the following one.

Proposition 2.2. *Let N be an H-layers perceptron. Assume that for a fixed $h \in \{0, \dots, H\}$, the $(h+1)$-th layer of N is a discontinuous one (therefore, denoted by δL_{h+1}). For each $\boldsymbol{x}^{(0)} \in \mathbb{R}^{N_0}$, let y_j be the j-th element of $\boldsymbol{y} = \mathcal{L}_1^{H+1}(\boldsymbol{x}^{(0)})$ and let $\boldsymbol{x}^{(i)}$ denote the vector $\mathcal{L}_1^i(\boldsymbol{x}^{(0)}) \in \mathbb{R}^{N_i}$, for each $i = 1, \dots, H+1$. Let $\mathcal{J}$ denote the jacobian of $\mathcal{L}_{h+2}^{H+1}$.*

For each input vector $\boldsymbol{x}^{(0)}$, if the derivatives corresponding to the j-th row $J_{j\,,\,\cdot}$ of $\mathcal{J}$ exist at $\boldsymbol{x}^{(h+1)}$, if the (j,k)-th function $J_{j,k}$ of $\mathcal{J}$ is not constantly equal to zero at $\boldsymbol{x}^{(h+1)}$, and if $W_{\cdot\,,\,k}^{(h+1)\,T} \boldsymbol{x}^{(h)} + b_k^{(h+1)}$ is not constantly negative, then

$$\frac{\partial\, y_j}{\partial\, \varepsilon_k^{(h+1)}} \not\equiv 0 \,,$$

where $j \in \{1, \dots, N_{H+1}\}$, $k \in \{1, \dots, N_{h+1}\}$, and $\varepsilon_k^{(h+1)}$ is the k-th element of $\boldsymbol{\varepsilon}^{(h+1)}$ of $\delta\mathcal{L}_{h+1}$ (see (4)).

Proof. The proof is straightforward since $\boldsymbol{y} = (\mathcal{L}_{h+2}^{H+1} \circ \mathcal{L}_1^{h+1})(\boldsymbol{x}^{(0)})$. Indeed, for each $j = 1, \dots, N_{H+1}$ and $k = 1, \dots, N_{h+1}$, we have that

$$\begin{aligned}
\frac{\partial\, y_j}{\partial\, \varepsilon_k^{(h+1)}} &= \left(\nabla\left(\mathcal{L}_{h+2}^{H+1}(\boldsymbol{x}^{(h+1)})\right)_j\right)^T \cdot \nabla_{\varepsilon_k^{(h+1)}} \boldsymbol{x}^{(h+1)} = \\
&= J_{j\,,\,\cdot}^T(\boldsymbol{x}^{(h+1)}) \cdot \nabla_{\varepsilon_k^{(h+1)}} \boldsymbol{x}^{(h+1)} = \\
&= J_{j,k}(\boldsymbol{x}^{(h+1)}) \cdot \frac{\partial x_k^{(h+1)}}{\partial \varepsilon_k^{(h+1)}} = J_{j,k}(\boldsymbol{x}^{(h+1)}) \cdot \mathcal{H}\left(W_{\cdot\,,\,k}^{(h+1)\,T} \boldsymbol{x}^{(h)} + b_k^{(h+1)}\right) ,
\end{aligned}$$

and from the assumptions the thesis follows. □

Generalizing Propositions 2.1 and 2.2, we observe that the derivatives with respect to the discontinuity parameters of the loss function can't be constantly equal to zero (excluding special cases) and, therefore, the discontinuity parameters are trainable. To the best of the authors' knowledge, in literature there are no discontinuous NNs characterized by *trainable* discontinuity jumps.

Remark 2.1 (Theoretical justification for discontinuous activation functions)**.** Concerning the universal approximation properties of NNs, we recall the theoretical results of [21, 26], that involve also discontinuous activation functions. These result legitimate the development of discontinuous layers. Indeed, even if there are no clear universal approximation results for H-layers perceptrons with discontinuous activation functions, there are clues suggesting that discontinuities may enhance the approximation (e.g., see [26, Th. 3] or [30]).

Remark 2.2 (Training discontinuous Neural Networks)**.** Introducing discontinuous layers inside a NN implies that the loss function can be piece-wise continuous. Then, during the training, the loss can be non-differentiable and/or discontinuous with respect to the current weights and biases. Actually, non-smoothness is already quite frequent in the working environment of NNs and, due to the discrete representations of functions in a computational domain, the non-differentiability can be "safely disregarded" [10, ch. 6.3]. We can extend the same reasoning when a derivative is needed at a discontinuity point x_0; indeed, it is likely that the underlying value is $\widetilde{x}_0 = x_0 \pm \epsilon$, with a small $\epsilon > 0$ (say, less than the machine precision) [10, ch. 6.3]. Furthermore, we observe that derivative-based methods typically perform sufficiently well on piece-wise continuous functions, generally converging to local minima (see the case studies in [34]). Then, we can "safely disregard" the discontinuity of the activation functions during the gradient-based training. Indeed, the training methods for NNs are typically gradient-based and stochastic and, usually, local minima are sought, to avoid overfitting.
However, even if in discontinuous NNs the training ability is preserved (see the results of Section 4), the optimization methods available in the Deep Learning frameworks (e.g., TensorFlow [3]) are indeed not designed to work efficiently with discontinuous functions. The study of more efficient optimization algorithms for the training of discontinuous NNs is certainly of great interest and deserves future investigation.

Remark 2.3 (Discontinuity and increased capacity)**.** We observe that the introduction of discontinuities in NN architectures increases their capacity (see [10, ch. 5.2]), adding a new set of discontinuous functions to the set of functions represented by continuous NNs, the latter being equivalent to formulation (5) with $\boldsymbol{\varepsilon} = \mathbf{0}$. Concerning this point, we remark that we refer to the layers characterized by (5) as discontinuous layers independently of the actual values learned for $\boldsymbol{\varepsilon}$.

In the next section, we analyze the properties of NNs with at least one discontinuous layer; the aim of this analysis is to understand how a discontinuous layer characterizes the discontinuities of the function $\mathcal{L}_1^{H+1} = \widehat{\boldsymbol{F}}$ of the NN.

3. Properties of Discontinuous Neural Networks

An interesting property of δNNs is that, in principle, it is possible to exactly find the discontinuities of their characterizing functions; indeed, these

NNs are piece-wise continuous functions with known analytical expression (see Corollary 3.3 in the following). This property may be useful to improve the approximations in regression problems but it can be also extremely important for problems in which the discontinuity interfaces of functions are sought, especially in high-dimensional domains (see [8], for an example of application). With this new typology of NN architectures, we propose a novel approach to the discontinuity detection problem, showing that such a kind of NNs can be potentially useful both for discontinuity function approximation and for learning discontinuities.

In this section, we introduce some statements that describe properties related to discontinuity of δNNs.

3.1. Theoretic Foundations of Discontinuous Neural Networks

The following propositions (Proposition 3.1 and Proposition 3.2) represent some basic results concerning properties of NNs characterized by discontinuous layers. In a nutshell, the propositions state that in a δNN characterized by a function $\mathcal{L}_1^{H+1}$, for each discontinuous layer δL_{h+1} of N we have that:

- the discontinuity interfaces of the map $\delta\mathcal{L}_{h+1}$ are affine hyperplanes in $\mathbb{R}^{N_h}$, characterized by the columns of $W^{(h+1)}$ and the elements of $\boldsymbol{b}^{(h+1)}$;
- the existence of discontinuity interfaces for $\delta\mathcal{L}_{h+1}$ depends on the nonzero elements of $\boldsymbol{\varepsilon}^{(h+1)}$ corresponding to non-null columns of $W^{(h+1)}$.

Moreover, assuming that N has only one discontinuous layer δL_{h+1}, a necessary condition for a point $\widehat{\boldsymbol{x}}^{(0)}$ to be a discontinuity point for $\mathcal{L}_1^{H+1}$ is that its image through the first h layers, i.e. $\widehat{\boldsymbol{x}}^{(h)} = \mathcal{L}_1^h(\widehat{\boldsymbol{x}}^{(0)})$, is a discontinuity point for the map $\delta\mathcal{L}_{h+1}$.

In view of the next results, we introduce here the following sets defined for an H-layers perceptron with at least one discontinuous layer δL_{h+1}:

- $\Pi_j^{(h+1)}$ denotes the (possibly degenerate) hyperplane of $\mathbb{R}^{N_h}$ defined by the j-th column of weights and the j-th bias of δL_{h+1}, i.e.:

$$\Pi_j^{(h+1)} := \left\{ \boldsymbol{x}^{(h)} \in \mathbb{R}^{N_h} \mid W_{\cdot\,,j}^{(h+1)\,T} \boldsymbol{x}^{(h)} + b_j^{(h+1)} = 0 \right\} ;$$

- $\mathit{\Pi}^{(h+1)}$ denotes the set of all the sets $\Pi_j^{(h+1)}$ defined by the weights and the biases of δL_{h+1}, i.e..

$$\mathit{\Pi}^{(h+1)} := \left\{ \Pi_1^{(h+1)}, \dots, \Pi_{N_{h+1}}^{(h+1)} \right\} ;$$

- $\Delta^{(h+1)}$ denotes the set of all and only the discontinuity points in $\mathbb{R}^{N_h}$ for $\delta\mathcal{L}_{h+1}$;
- Δ denotes the set of all and only the discontinuity points in $\mathbb{R}^{N_0}$ for $\mathcal{L}_1^{H+1}$ (i.e., for the NN).

- Γ_{h+1} denotes the (possibly empty) counterimage of $\Delta^{(h+1)}$ through $\mathcal{L}_1^h$, namely:

$$\Gamma_{h+1} := \left\{ \boldsymbol{x}^{(0)} \in \mathbb{R}^{N_0} \;\middle|\; \mathcal{L}_1^h(\boldsymbol{x}^{(0)}) \in \Delta^{(h+1)} \right\} =: \left(\mathcal{L}_1^h\right)^{-1}(\Delta^{(h+1)})\,, \tag{6}$$

  where we set $\mathcal{L}_1^0$ as the identity function, by convention;

- let $\delta L_{h_1+1}, \dots, \delta L_{h_M+1}$ be all and only the discontinuous layers of the H-layers perceptron. Then, we denote by Γ the union of all the counterimages $\Gamma_{h_1+1}, \dots \Gamma_{h_m+1}$:

$$\Gamma := \bigcup_{m=1}^{M} \Gamma_{h_m+1}\,. \tag{7}$$

Proposition 3.1. *Let* N *be an* H*-layers perceptron. Assume that for a fixed* $h \in \{0, \dots, H\}$*, the* $(h+1)$*-th layer is a discontinuous one. Let* $\mathcal{C}(\mathit{\Pi}^{(h+1)})$ *be the partition of* $\mathbb{R}^{N_h}$ *generated by* $\mathit{\Pi}^{(h+1)}$ *as in* (2) *and characterized by* $P \in \mathbb{N}$ *non-empty subsets such that* $\mathcal{C}(\mathit{\Pi}^{(h+1)}) = \{X_1^{(h)} \dots, X_P^{(h)}\}$*. For each* $X_p^{(h)} \in \mathcal{C}(\mathit{\Pi}^{(h+1)})$*, let* $\boldsymbol{k}_p^{(h)}$ *be the region vector of* $X_p^{(h)}$ *introduced in Definition 1.2.*

Then, the following assertions are true:

1. *for each* $\boldsymbol{x}^{(h)} \in \mathbb{R}^{N_h}$*, equation* (4) *can be rewritten as*

$$\boldsymbol{x}^{(h+1)} = \boldsymbol{f}_{h+1}\left(W^{(h+1)\,T}\boldsymbol{x}^{(h)} + \boldsymbol{b}^{(h+1)}\right) + \boldsymbol{\varepsilon}^{(h+1)} \odot \boldsymbol{k}_i^{(h)}\,, \tag{8}$$

   where $i \in \{1, \dots, P\}$ *is such that* $\boldsymbol{x}^{(h)} \in X_i^{(h)}$*;*

2. $\delta\mathcal{L}_{h+1}$ *is discontinuous at* $\widehat{\boldsymbol{x}}^{(h)} \in \mathbb{R}^{N_h}$ *if and only if exists* $j \in \{1, \dots, N_{h+1}\}$ *such that* $\widehat{\boldsymbol{x}}^{(h)} \in \Pi_j^{(h+1)}$*,* $\varepsilon_j^{(h+1)} \neq 0$*, and* $W_{\cdot\,,\,j}^{(h+1)} \neq \mathbf{0}$*. In other words:*

$$\Delta^{(h+1)} = \bigcup_{\substack{j=1,\dots,N_{h+1}\\ \varepsilon_j^{(h+1)}\neq 0\\ W_{\cdot\,,\,j}^{(h+1)}\neq\mathbf{0}}} \Pi_j^{(h+1)}. \tag{9}$$

Proof.

1. The proof is immediate as it directly follows from Definition 1.1.

2. The function f_{h+1} is continuous and $\mathcal{H}$ is discontinuous at zero. Then, for each $j = 1, \dots, N_{h+1}$, the function $\mathcal{H}(W_{\cdot\,,\,j}^{(h+1)\,T}\boldsymbol{x}^{(h)} + b_j^{(h+1)})$ is discontinuous at $\widehat{\boldsymbol{x}}^{(h)}$ if and only if $\widehat{\boldsymbol{x}}^{(h)} \in \Pi_j^{(h+1)}$ and $W_{\cdot\,,\,j}^{(h+1)} \neq \mathbf{0}$. Therefore, $\delta\mathcal{L}_{h+1}$ is discontinuous at $\widehat{\boldsymbol{x}}^{(h)}$ if and only if there exists $j \in \{1, \dots, N_{h+1}\}$ such that $\widehat{\boldsymbol{x}}^{(h)} \in \Pi_j^{(h+1)}$, $W_{\cdot\,,\,j}^{(h+1)} \neq \mathbf{0}$, and $\varepsilon_j^{(h+1)} \neq 0$.

□

In a nutshell, according to item 2 of the previous Proposition, discontinuity interfaces of $\delta\mathcal{L}_{h+1}$ are affine hyperplanes in $\mathbb{R}^{N_h}$, whose equations depend on the columns of $W^{(h+1)}$ and the elements of $\boldsymbol{b}^{(h+1)}$.

While Proposition 3.1 characterizes the discontinuity points of a layer δL_{h+1}, in Proposition 3.2 we characterize the discontinuity points of $\mathcal{L}_1^{H+1}$, assuming that δL_{h+1} is the only discontinuous layer of the δNN.

Proposition 3.2. *Under the assumptions of Proposition 3.1, assume that* N *has a unique discontinuous layer δL_{h+1} for a fixed $h \in \{0, \dots, H\}$. Let $\widehat{\boldsymbol{x}}^{(0)}$ be a given vector in $\mathbb{R}^{N_0}$. Then:*

1. *$\mathcal{L}_1^{h+1}$ is discontinuous at $\widehat{\boldsymbol{x}}^{(0)}$ if and only if $\widehat{\boldsymbol{x}}^{(0)} \in \Gamma_{h+1}$, i.e. $\mathcal{L}_1^h(\widehat{\boldsymbol{x}}^{(0)}) \in \Delta^{(h+1)}$;*

2. *if $\widehat{\boldsymbol{x}}^{(0)}$ is a discontinuity point for $\mathcal{L}_1^{H+1}$ then $\widehat{\boldsymbol{x}}^{(0)}$ is a discontinuity point for $\mathcal{L}_1^{h+1}$; i.e.:*

$$\Delta \subseteq \Gamma_{h+1}\,.$$

Proof.

1. For each $\widehat{\boldsymbol{x}}^{(0)} \in \Gamma_{h+1}$, the proof that $\widehat{\boldsymbol{x}}^{(0)}$ is a discontinuity point for $\mathcal{L}_1^{h+1}$ is straightforward, as δL_{h+1} is the only discontinuous layer of N. On the other hand, let $\widehat{\boldsymbol{x}}^{(0)} \in \mathbb{R}^{N_0}$ be a discontinuity point for $\mathcal{L}_1^{h+1}$ such that $\mathcal{L}_1^h(\widehat{\boldsymbol{x}}^{(0)}) \notin \Delta^{(h+1)}$; then, $\delta\mathcal{L}_{h+1}$ is continuous at $\mathcal{L}_1^h(\widehat{\boldsymbol{x}}^{(0)})$. But $\mathcal{L}_1^h$ is continuous and $\mathcal{L}_1^{h+1} = \delta\mathcal{L}_{h+1} \circ \mathcal{L}_1^h$; then, $\mathcal{L}_1^{h+1}$ is continuous at $\widehat{\boldsymbol{x}}^{(0)}$, which is a contradiction of the hypothesis.

2. The result is straightforward, as δL_{h+1} is the only discontinuous layer of N.

□

Proposition 3.2 can be generalized to NNs that take into account more discontinuous layers. This generalization is summarized in Theorem 3.1 in the next section.

3.2. Main Results about Discontinuous Neural Networks

The results presented in Section 3.1 describe the discontinuity behavior in a δNN and give a general idea about the potential of such a kind of instruments. Indeed, the discontinuities of a δNN can be quite well characterized.

However, the detection of all the discontinuity interfaces $\Delta \subset \mathbb{R}^{N_0}$ of a map $\mathcal{L}_1^{H+1}$ representing a δNN can be quite an hard task; even just considering a δNN with one discontinuous layer (see Proposition 3.2), the search for points in Γ_{h+1} would require to solve the nonlinear system

$$\begin{cases} W_{\cdot\,,\,j_1}^{(h+1)\,T}\mathcal{L}_1^h(\boldsymbol{x}) + b_{j_1} = 0 \\ \vdots \\ W_{\cdot\,,\,j_k}^{(h+1)\,T}\mathcal{L}_1^h(\boldsymbol{x}) + b_{j_k} = 0 \end{cases},$$

where $\varepsilon_j^{(h+1)} \neq 0$ and $W_{\cdot\,,j}^{(h+1)} \neq \mathbf{0}$ for all and only the $j \in \{j_1, \dots, j_k\} \subseteq \{1, \dots, N_{h+1}\}$.

An alternative idea, useful to avoid the difficulties related to the direct detection of the discontinuity interfaces, is to solve its complementary problem, i.e., find the continuity regions of the domain. Theorem 3.1 and Corollary 3.3 state that δNNs are, actually, piecewise continuous functions and that given a pair of points in the function domain, it can be easily detected if they belongs or not to a region described by the same continuous piece of function.

Before illustrating these results, we generalize Definition 1.1 with respect to all the discontinuous layers of a δNN.

Definition 3.1 (Generalized Region Vectors for δNN)**.** *Let* N *be an H-layers perceptron with M discontinuous layers. Let δL_{h_m+1}, for $m = 1, \dots, M$, and $0 \leq h_1 < \dots < h_M \leq H$, be the discontinuous layers. For each $m = 1, \dots, M$, let $g_m : \mathbb{R}^{N_{h_m}} \to \{0,1\}^{N_{h_m+1}}$ denote the region function corresponding to the weights and biases of δL_{h_m+1}, i.e.*

$$g_m(\boldsymbol{x}^{(h_m)}) := \boldsymbol{\mathcal{H}}\left(W^{(h_m+1)\,T} \boldsymbol{x}^{(h_m)} + \boldsymbol{b}^{(h_m+1)} \right) ,$$

for each $\boldsymbol{x}^{(h_m)} \in \mathbb{R}^{N_{h_m}}$; then, we denote by K_m the image of g_m, i.e. the set

$$K_m = \left\{ \boldsymbol{k}^{(h_m)} \in \{0,1\}^{N_{h_m+1}} \;\middle|\; \exists\ \boldsymbol{x}^{(h_m)} \in \mathbb{R}^{N_{h_m}} \ s.t.\ g_m(\boldsymbol{x}^{(h_m)}) = \boldsymbol{k}^{(h_m)} \right\} , \tag{10}$$

representing all the region vectors characterizing the sets of $\mathcal{C}(\Pi^{(h_m+1)})$ in $\mathbb{R}^{N_{h_m}}$ identified by the weights and biases of layer δL_{h_m+1}.

Setting $\delta N = \sum_{m=1}^{M} N_{h_m+1}$, let $\mathcal{G} : \mathbb{R}^{N_0} \to \{0,1\}^{\delta N}$ be defined as

$$\mathcal{G}(\boldsymbol{x}^{(0)}) = \begin{bmatrix} g_1 \circ \mathcal{L}_1^{h_1}(\boldsymbol{x}^{(0)}) \\ \vdots \\ g_M \circ \mathcal{L}_1^{h_M}(\boldsymbol{x}^{(0)}) \end{bmatrix} . \tag{11}$$

Then, $\mathcal{G}$ is called generalized region function of N *and we call generalized region vectors of* N *all the vectors $\boldsymbol{k} \in K$, where K is the image of $\mathcal{G}$:*

$$K := \left\{ \boldsymbol{k} \in \{0,1\}^{\delta N} \;\middle|\; \exists\, \boldsymbol{x}^{(0)} \in \mathbb{R}^{N_0} \ s.t.\ \mathcal{G}(\boldsymbol{x}^{(0)}) = \boldsymbol{k} \right\} . \tag{12}$$

in particular, for each $m = 1, \dots, M$, we denote by $\boldsymbol{k}^{(h_m)}$ the subvectors of $\boldsymbol{k}$ belonging to K_m and related to $\delta\mathcal{L}_{h_m+1}$ (see (10)).

Thanks to the generalized region vectors of a given δNN, it is possible to identify a partition of the domain $\mathbb{R}^{N_0}$ such that the discontinuities of δNN are contained in the union of all the boundaries of the partition sets, while the map $\mathcal{L}_1^{H+1}$ is continuous in the interior of these sets. These results are better described in the following theorem and corollaries.

Theorem 3.1. *Let* N *be an* H*-layers perceptron with* M *discontinuous layers. Let* δL_{h_m+1}*, for* $m = 1, \dots, M$*, and* $0 \leq h_1 < \cdots < h_M \leq H$*, be the discontinuous layers. For each* $i = 1, \dots, |K|$*, consider the region vector* $\boldsymbol{k}_i \in K$ *and let* $\mathcal{K}_i$ *be defined as*

$$\mathcal{K}_i = \{\boldsymbol{x}^{(0)} \in \mathbb{R}^{N_0} \mid \mathcal{G}(\boldsymbol{x}^{(0)}) = \boldsymbol{k}_i\}. \tag{13}$$

Then, the following assertions are true:

1. *The set* Δ *of discontinuity points of* $\mathcal{L}_1^{H+1}$ *is contained in* Γ *(see (7)).*
2. $\{\mathcal{K}_1 \dots, \mathcal{K}_{|K|}\}$ *is a partition of* $\mathbb{R}^{N_0}$*;*
3. $\mathcal{L}_1^{H+1}$ *is continuous in the interior of* $\mathcal{K}_i$ *(denoted by* $\overset{\circ}{\mathcal{K}}_i$*), for each* $i = 1, \dots, |K|$*;*
4. *Let* $\widetilde{\Gamma}$ *denote the union of all the counterimages, through* $\mathcal{L}_1^{h_m}$*, of the hyperplanes* $\Pi_j^{(h_m+1)}$*, i.e.:*

$$\widetilde{\Gamma} := \bigcup_{m=1}^{M} \bigcup_{\substack{j=1,\dots,N_{h_m+1} \\ W_{\cdot\,,\,j}^{(h_m+1)} \neq \mathbf{0}}} \left(\mathcal{L}_1^{h_m}\right)^{-1} (\Pi_j^{(h_m+1)}) \tag{14}$$

Then,

$$\bigcup_{i=1}^{|K|} \partial\mathcal{K}_i = \widetilde{\Gamma} \tag{15}$$

Proof.

1. Let $\widehat{\boldsymbol{x}}^{(0)}$ be a discontinuity point for $\mathcal{L}_1^{H+1}$ such that $\widehat{\boldsymbol{x}}^{(0)} \notin \Gamma$. Then, for each $m = 1, \dots, M$, we have that $\mathcal{L}_1^{h_m+1}(\widehat{\boldsymbol{x}}^{(0)}) \notin \Delta^{(h_m+1)}$ and $\mathcal{L}_1^{h_m+1}$ is continuous at $\widehat{\boldsymbol{x}}^{(0)}$. Therefore, all the layers are continuous at $\widehat{\boldsymbol{x}}^{(0)}$ and $\mathcal{L}_1^{H+1}$ is continuous at $\widehat{\boldsymbol{x}}^{(0)}$, which is a contradiction of the hypothesis.
2. The result is a direct consequence of the definition of K and $\mathcal{K}_1 \dots, \mathcal{K}_{|K|}$.
3. Let $\widehat{\boldsymbol{x}}^{(0)}$ be an arbitrary point of $\overset{\circ}{\mathcal{K}}_i$, for a fixed $i \in \{1 \dots, |K|\}$; then, there exists an $\widehat{\epsilon} > 0$ such that, for each $0 < \epsilon \leq \widehat{\epsilon}$, the ball $B_\epsilon(\widehat{\boldsymbol{x}}^{(0)})$ contains only points $\boldsymbol{x}^{(0)}$ belonging to $\mathcal{K}_i$ and, therefore, for each $m = 1, \dots, M$ it

holds

$$\begin{aligned}
\lim_{\boldsymbol{x}^{(0)}\to\widehat{\boldsymbol{x}}^{(0)}} \mathcal{L}_1^{h_m+1}(\boldsymbol{x}^{(0)}) = & \lim_{\boldsymbol{x}^{(0)} \underset{B_\epsilon(\widehat{\boldsymbol{x}}^{(0)})}{\longrightarrow} \widehat{\boldsymbol{x}}^{(0)}} \boldsymbol{f}_{h_m+1}\left(W^{(h_m+1)\,T}\mathcal{L}_1^{h_m}(\boldsymbol{x}^{(0)}) + \boldsymbol{b}^{(h_m+1)}\right) + \\
& \lim_{\boldsymbol{x}^{(0)} \underset{B_\epsilon(\widehat{\boldsymbol{x}}^{(0)})}{\longrightarrow} \widehat{\boldsymbol{x}}^{(0)}} \boldsymbol{\varepsilon}^{(h_m+1)} \odot \boldsymbol{\mathcal{H}}\left(W^{(h_m+1)\,T}\mathcal{L}_1^{h_m}(\boldsymbol{x}^{(0)}) + \boldsymbol{b}^{(h_m+1)}\right) = \\
= & \ \boldsymbol{f}_{h_m+1}\left(W^{(h_m+1)\,T}\mathcal{L}_1^{h_m}(\widehat{\boldsymbol{x}}^{(0)}) + \boldsymbol{b}^{(h_m+1)}\right) + \\
& \ \boldsymbol{\varepsilon}^{(h_m+1)} \odot \boldsymbol{k}_i^{(h_m)} = \\
= & \ \mathcal{L}_1^{h_m+1}(\widehat{\boldsymbol{x}}^{(0)})\,,
\end{aligned}$$

where $[\boldsymbol{k}_i^{(h_1)\,T}, \dots, \boldsymbol{k}_i^{(h_M)\,T}]^T = \boldsymbol{k}_i$ is the generalized region vector of $\mathcal{K}_i$. Then $\mathcal{L}_1^{H+1}$ (and $\mathcal{L}_1^{h_m+1}$, for each $m = 1\,, \dots, M$) is continuous at $\widehat{\boldsymbol{x}}^{(0)}$; since $\widehat{\boldsymbol{x}}^{(0)}$ is an arbitrary point, $\mathcal{L}_1^{H+1}$ is continuous in $\overset{\circ}{\mathcal{K}}_i$, for each $i = 1\,, \dots, |K|$.

4. The inclusion $\widetilde{\Gamma} \subseteq \bigcup_{i=1}^{|K|} \partial\mathcal{K}_i$ is straightforward, by the definition of the sets $\mathcal{K}_1, \dots, \mathcal{K}_{|K|}$, so we are only left to prove the opposite inclusion.

   Let $\widehat{\boldsymbol{x}}^{(0)}$ be a boundary point for a given $\mathcal{K}_i$, $i \in \{1\,\dots, |K|\}$, i.e. $\widehat{\boldsymbol{x}}^{(0)} \in \partial\mathcal{K}_i$; then, for each $\epsilon > 0$, the ball $B_\epsilon(\widehat{\boldsymbol{x}}^{(0)})$ contains both an internal point $\boldsymbol{x}_{\text{in}}^{(0)} \in \mathcal{K}_i$ and an external point $\boldsymbol{x}_{\text{out}}^{(0)} \in \mathcal{K}_j$, $j \neq i$. Since these two vectors belong to two different continuity regions, they have different generalized region vectors, i.e. $\boldsymbol{k}_i := \mathcal{G}(\boldsymbol{x}_{\text{in}}^{(0)}) \neq \mathcal{G}(\boldsymbol{x}_{\text{out}}^{(0)}) =: \boldsymbol{k}_j$.

   Let $\mathcal{K}_j$, $j \neq i$, be a region that shares the boundary with $\mathcal{K}_i$ through $\widehat{\boldsymbol{x}}^{(0)} \in \partial\mathcal{K}_i$, and let $m \in \{1, \dots, M\}$ be the first index such that $\boldsymbol{k}_i^{(h_m)} \neq \boldsymbol{k}_j^{(h_m)}$, for each pair of internal and external points $\boldsymbol{x}_{\text{in}}^{(0)} \in \mathcal{K}_i$ and $\boldsymbol{x}_{\text{out}}^{(0)} \in \mathcal{K}_j$, respectively, in the ball $B_\epsilon(\widehat{\boldsymbol{x}}^{(0)})$, for each $\epsilon > 0$.

   For item 3, the sub-NN characterized by $\mathcal{L}_1^{h_m}$ is continuous on $B_\epsilon(\widehat{\boldsymbol{x}}^{(0)})$; then, the image $B_\epsilon^{(h_m)}(\widehat{\boldsymbol{x}}^{(h_m)}) := \mathcal{L}_1^{h_m}(B_\epsilon(\widehat{\boldsymbol{x}}^{(0)}))$ is a connected neighborhood of $\widehat{\boldsymbol{x}}^{(h_m)}$ in $\mathbb{R}^{N_{h_m}}$, where we denoted by $\widehat{\boldsymbol{x}}^{(h_m)}$ the image of $\widehat{\boldsymbol{x}}^{(0)}$ through $\mathcal{L}_1^{h_m}$.

   Let us denote by $\boldsymbol{x}_{\text{in}}^{(h_m)}, \boldsymbol{x}_{\text{out}}^{(h_m)} \in B_\epsilon^{(h_m)}(\widehat{\boldsymbol{x}}^{(h_m)})$ the images of $\boldsymbol{x}_{\text{in}}^{(0)}, \boldsymbol{x}_{\text{out}}^{(0)} \in B_\epsilon(\widehat{\boldsymbol{x}}^{(0)})$, respectively, through $\mathcal{L}_1^{h_m}$. Since we have $\boldsymbol{k}_i^{(h_m)} \neq \boldsymbol{k}_j^{(h_m)}$, it holds that $\boldsymbol{x}_{\text{in}}^{(h_m)}, \boldsymbol{x}_{\text{out}}^{(h_m)}$ belong to two distinct sets $X_i^{(h_m)}, X_j^{(h_m)} \in \mathcal{C}(\mathit{\Pi}^{(h_m+1)})$, respectively, where $\partial X_i^{(h_m)} \cap \partial X_j^{(h_m)} \neq \emptyset$.

   Then, for each $\epsilon > 0$, we have that $\widehat{\boldsymbol{x}}^{(h_m)} \in \partial X_i^{(h_m)} \cap \partial X_j^{(h_m)}$ and, therefore, $\widehat{\boldsymbol{x}}^{(0)} \in \widetilde{\Gamma}$ because $\widehat{\boldsymbol{x}}^{(h_m)}$ belongs to one of the hyperplanes in $\mathit{\Pi}^{(h_m+1)}$. For the generality of the choice of $\mathcal{K}_i$ and $\widehat{\boldsymbol{x}}^{(0)} \in \partial\mathcal{K}_i$, we have that $\bigcup_{i=1}^{|K|} \partial\mathcal{K}_i \subseteq \widetilde{\Gamma}$.

□

Corollary 3.2. *Let the hypotheses of Theorem 3.1 be satisfied. Then, the set Δ of all the discontinuity points of $\mathcal{L}_1^{H+1}$ is contained in $\bigcup_{i=1}^{|K|} \partial\mathcal{K}_i$ and, in particular, we have that*

$$\Delta \subseteq \Gamma \subseteq \bigcup_{i=1}^{|K|} \partial\mathcal{K}_i \,. \tag{16}$$

Proof. We observe that item 3 of Theorem 3.1 implies $\Delta \subseteq \bigcup_{i=1}^{|K|} \partial\mathcal{K}_i$ and we recall that item 1 of Theorem 3.1 implies $\Delta \subseteq \Gamma$. By construction, $\Gamma \subseteq \widetilde{\Gamma}$ and for item 4 we have $\widetilde{\Gamma} = \bigcup_{i=1}^{|K|} \partial\mathcal{K}_i$; then, (16) is proved. □

The Theorem and the Corollary above highlight the importance of the partition given by sets (13). Due to the properties illustrated in the theorem, we denote the sets of the partition by the names of *continuity regions*, through the following definition.

Definition 3.2 (Continuity Regions of a δNN)**.** *Let* N *be an H-layers perceptron with M discontinuous layers, and let $\mathcal{K}_1 \,\ldots, \mathcal{K}_{|K|} \subseteq \mathbb{R}^{N_0}$ be defined as in* (13)*. The sets $\mathcal{K}_1 \,\ldots, \mathcal{K}_{|K|}$ are called continuity regions of* N*.*

Definition 3.2 makes even more sense looking at the results illustrated in the following Corollary, immediate consequence of Theorem 3.1, which states that δNNs are piece-wise continuous functions.

Corollary 3.3. *Let the hypotheses of Theorem 3.1 be satisfied. Let $\mathcal{M}_{h_m+1}^{(i)}$ be the function*

$$\mathcal{M}_{h_m+1}^{(i)}(\boldsymbol{x}^{(h_m)}) = \boldsymbol{f}_{h+1}\left(W^{(h_m+1)\,T}\boldsymbol{x}^{(h_m)} + \boldsymbol{b}^{(h+1)}\right) + \boldsymbol{\varepsilon}^{(h_m+1)} \odot \boldsymbol{k}_i^{(h_m)} \,,$$

for $m = 1\,,\ldots, M$, where $\boldsymbol{k}_i^{(h_m)}$ is the subvector of $\boldsymbol{k}_i$ (see Definition 3.1).

Then $\mathcal{L}_1^{H+1}$ is a piecewise continuous function such that

$$\mathcal{L}_1^{H+1}(\boldsymbol{x}^{(0)}) = \begin{cases} \mathcal{F}_1(\boldsymbol{x}^{(0)})\,, & \textit{if } \ \boldsymbol{x}^{(0)} \in \mathcal{K}_1 \\ \vdots & \\ \mathcal{F}_{|K|}(\boldsymbol{x}^{(0)})\,, & \textit{if } \ \boldsymbol{x}^{(0)} \in \mathcal{K}_{|K|} \end{cases}$$

where, for each $i = 1\,,\ldots, |K|$, $\mathcal{F}_i$ is the continuous function defined by

$$\mathcal{F}_i = \mathcal{L}_{h_M+2}^{H+1} \circ \mathcal{M}_{h_M+1}^{(i)} \circ \mathcal{L}_{h_{M-1}+2}^{h_M} \circ \mathcal{M}_{h_{M-1}+1}^{(i)} \circ \cdots \circ \mathcal{M}_{h_1+1}^{(i)} \circ \mathcal{L}_1^{h_1} \,.$$

We conclude this section making a resume and some observations concerning the outcomes of Theorem 3.1 and Corollary 3.3 that are useful for practical applications. First, the results state that it is possible to identify the points where the map $\mathcal{L}_1^{H+1}$ of the δNN is certainly continuous; these regions of the domain are the open sets $(\mathcal{K}_i \setminus \partial\mathcal{K}_i)$, i.e. the interior of the continuity regions $\mathcal{K}_i$.

As a consequence, we have the precise indication that the discontinuity interfaces of $\mathcal{L}_1^{H+1}$ are contained in the union of the boundaries $\partial\mathcal{K}_i$. Furthermore, since the continuity regions are characterized by the generalized region vectors and the set K of these vectors is contained in $K_1 \times \cdots \times K_M$ (see (12)), we can control the number of continuity regions through the following proposition.

Proposition 3.3 (Maximum number of continuity regions)**.** *Let the hypotheses of Theorem 3.1 be satisfied. Then, the number $|K|$ of continuity regions for* N *is such that*

$$|K| \leq 2^{\sum_{m=1}^{M} N_{h_m+1}} .$$

The continuity regions $\mathcal{K}_i$ of a δNN are a bit more complex than their simpler counterparts in the discontinuous layers, which are sets $X_i^{(h)}$ identified by hyperplanes and region vectors (see Section 1.1.2). Indeed, sets $\mathcal{K}_i$ are not necessarily convex nor connected sets in $\mathbb{R}^{N_0}$, due to the function compositions that take place inside the δNN; this is also the reason for the difficulties in directly computing the discontinuity interfaces, even if theoretically possible. On the other hand, given a set $X^{(0)} \subseteq \mathbb{R}^{N_0}$, the computation of the generalized region vector $\mathcal{G}(\boldsymbol{x}^{(0)})$ for each $\boldsymbol{x}^{(0)} \in X^{(0)}$ is extremely easy and fast since, by definition, $\mathcal{G}$ is characterized by sub-networks of the δNN considered (see (11)). Therefore, in practice, the regions $\mathcal{K}_i$ can be deduced by sampling a large enough set of points in the domain and then computing the generalized region vector for all such points.

Since the continuity regions can be non-convex and/or disconnected, the actual significance of Proposition 3.3 is to define an upper bound to the number of continuous functions that define the equation of $\mathcal{L}_1^{H+1}$ through (3.3).

4. Learning Discontinuities: Numerical Experiments

In the previous sections we have focused on the analysis of the function $\mathcal{L}_1^{H+1}$ characterizing a (trained) δNN, highlighting properties related to the discontinuous layers present in the network. In this section we show experimental evidence about the ability of a δNN to learn discontinuity interfaces, given a target function to approximate. To this aim, we perform the following experiments:

1. We start considering four test cases given by discontinuous functions displaying different kinds of discontinuity interfaces with increasing complexity (Section 4.1). Several δNN architectures are tested; since the main target is to show the viability of δNNs for discontinuous function regression and discontinuity interfaces detection, we do not focus on the problem of finding the best hyper-parameters or architectures for regression performance, but we focus on the parameters and characteristics related to the discontinuous layers of the δNNs, aiming to highlight the actual sensitivity of the δNN approximation performances with respect to the new discontinuous layers.

From these experiments we observe interesting relationships between a good discontinuity interface detection and the number/position of discontinuous layers in the NN (Section 4.1.2). Then, we define a clustering algorithm for the continuity regions of δNNs to reduce the overestimation of the actual continuity regions and improve their detection (Section 4.1.3).

2. We continue with an experiment in which the test function is obtained from real data (Section 4.2). In particular, we consider the phase transition phenomenon of acetone. Representing the acetone's density ρ as a function of temperature T and pressure p, a discontinuity interface is present and corresponds to the so-called equilibrium points between gas and liquid states. Based on the observations of previous experiments, we train a δNN and analyze the results using the clustering algorithm introduced in Section 4.1.3.

3. We conclude with a more general simulation study (Section 4.3). We generate 50 random discontinuous functions and we compare the continuity region detection abilities of a given δNN and a well-assessed discontinuity detection method proposed in literature. Each random function is given by two functions $g_1, g_2 : D \subset \mathbb{R}^2 \to \mathbb{R}$, based on 2-dimensional Legendre polynomials, separated by a discontinuity interface p, based on a 1-dimensional Legendre polynomial.

4.1. Initial Test Cases

In all the test cases here considered, the underlying function is a scalar function with domain in $\mathbb{R}^2$. In all the cases, we consider the functions restricted to the region $D = [-2, 2] \times [-2, 2]$. The test functions used are the following.

1. **Test 1**. We consider the function $g_\ell : D \to \mathbb{R}$ defined as

$$g_\ell(\boldsymbol{x}) = \begin{cases} 2\sin(1.25\pi||\boldsymbol{x}||) + 4 & \text{if } \; x_2 \leq 2x_1 \\ 2\sin(0.75\pi||\boldsymbol{x}||) & \text{otherwise} \end{cases} \;, \tag{17}$$

   The discontinuity interface is the line $\ell : x_2 = 2x_1$ that halves the square domain D (see Figure 3-(a)).

2. **Test 2**. As a second test function, we consider $g_s : D \to \mathbb{R}$ defined as

$$g_s(\boldsymbol{x}) = \begin{cases} -2x_1^2 + 6\,, & \text{if } \; \boldsymbol{x} \in [-1, 1] \times [\,0, +\infty) \\ 4e^{(1-x_1^2)/2}\,, & \text{otherwise} \end{cases} \;, \tag{18}$$

   The discontinuity interface is the segment $s := \{\boldsymbol{x}(\lambda) = \lambda[-1, 0]^T + (1 - \lambda)[1, 0]^T,\ 0 \leq \lambda \leq 1\}$ (see Figure 3-(b)). Then, in this case, the discontinuity is a sort of straight "rip" for the graph of the function.

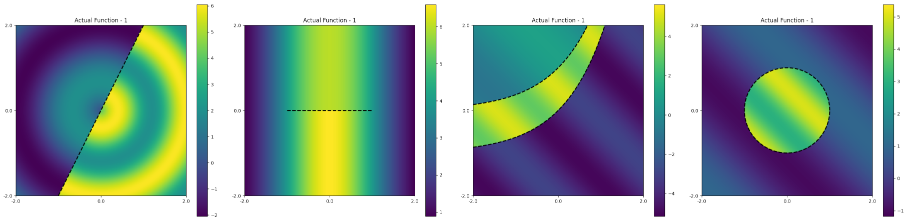
3. **Test 3**. The third test function we consider is $g_\eta : D \to \mathbb{R}$, defined as

$$g_\eta(\boldsymbol{x}) = \begin{cases} \sin\left(0.4\pi(x_1 + x_2)\right)\,, & \text{if } \; x_2 \geq e^{x_1} \\ \sin\left(0.7\pi(x_1 + x_2)\right) - 4\,, & \text{if } \; x_2 < e^{x_1} - 1 \\ \sin\left(\pi(x_1 + x_2)\right) + 4\,, & \text{otherwise} \end{cases}\,, \tag{19}$$

   The discontinuity interfaces are two curves, η_1 and η_2, that split the domain in three regions (see Figure 3-(c)). In particular, $\eta_1 : x_2 = e^{x_1}$ and $\eta_2 : x_2 = e^{x_1} - 1$.

4. **Test 4**. As a last example we consider a function $g_\gamma : D \to \mathbb{R}$ characterized by a discontinuity interface that is a closed curve, the circumference $\gamma : ||\boldsymbol{x}||^2 = 1$. The function g_γ (see Figure 3-(d)) is defined as

$$g_\gamma(\boldsymbol{x}) = \begin{cases} \sin(\pi(x_1 + x_2)) + 4\,, & \text{if } \; ||\boldsymbol{x}||^2 \leq 1 \\ \sin\left(0.4\pi(x_1 + x_2)\right)\,, & \text{otherwise} \end{cases}\,. \tag{20}$$



(a) g_ℓ (b) g_s (c) g_η (d) g_γ

Figure 3: Top view of the functions g_ℓ, g_s, g_η, and g_γ (see equations (17)-(20), respectively). The dotted curves correspond to the discontinuity interfaces.

4.1.1. Architectures and Performance Measures

In the numerical experiments, we use discontinuous H-layers perceptrons. We consider three archetypes of δNN architectures, varying the depth, the number of discontinuous layers, and the size and position of these layers. The number of units in the fully connected (i.e., non-discontinuous) hidden layers is fixed to 128, while we let d denote the number of units in each discontinuous layer.

The three architecture archetypes we consider are:

1. *Architectures with one discontinuous layer only.* We consider a unique discontinuous layer with d units, which is the h-th inner layer, with $h \in \{1, \dots, H\}$ (see Figure 4-(a)). We let $\mathcal{A}^d_{h,H}$ denote such an architecture.

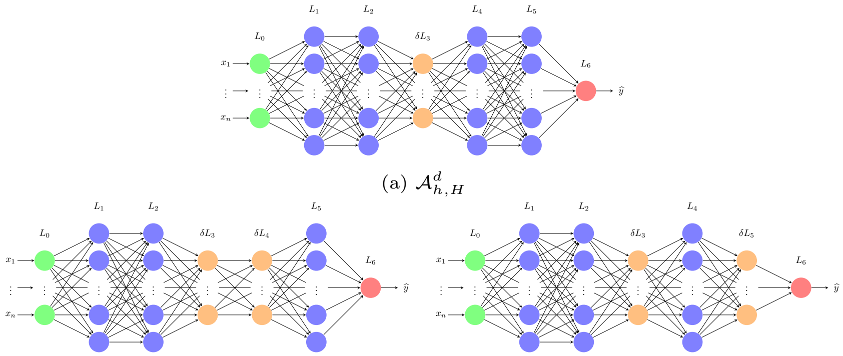
2. *Architectures with two discontinuous layers.*

   2.1. We consider two consecutive discontinuos layers with d units each, which are the h-th and $(h+1)$-th inner layers, for $h \in \{1, \dots, H-1\}$ (see Figure 4-(b)). We let $\mathcal{B}^d_{h,H}$ denote such an architecture.

2.2. We consider two discontinuos layers with d units each, separated by a fully-connected layer; they are the h-th and $(h+2)$-th inner layers, with $h \in \{1, \dots, H-2\}$ (see Figure 4-(c)). We let $\mathcal{C}^d_{h,H}$ denote such an architecture.

In our tests we considered: architecture $\mathcal{A}^d_{h,H}$ with $H = 5$, $h = 1, \dots, 5$ and $d = 2, 4, 8$; architecture $\mathcal{B}^d_{h,H}$ with $H = 5, 7$, $h = 1, \dots, H-1$ and $d = 2, 4, 8$; architecture $\mathcal{C}^d_{h,H}$ with $H = 5, 7$, $h = 1, \dots, H-2$ and $d = 2, 4, 8$. As a whole, we tested 69 architectures.

The activation function for all the hidden layers of the δNNs is the `elu` activation function [9], chosen after a preliminary investigation, while in the output layer the linear activation function is used. The depth H for the NNs has been chosen to analyze the effects of the discontinuous layers with respect to their position in the network, while guaranteeing a good approximation of the target function, and trying to avoid the so-called degradation problem [12]. The size of the discontinuous layers d has been chosen to keep limited the maximum number of continuity regions (see Proposition 3.3) to ease the analysis.



(a) $\mathcal{A}^d_{h,H}$

(b) $\mathcal{B}^d_{h,H}$

(c) $\mathcal{C}^d_{h,H}$

Figure 4: The three archetypes $\mathcal{A}^d_{h,H}$, $\mathcal{B}^d_{h,H}$, $\mathcal{C}^d_{h,H}$ (example with $h = 3$ and $H = 5$). Discontinuous layers are represented by orange units, fully-connected hidden layers by purple units, input layers by green units, and output layers by red units.

All the networks are trained with the same training options and configurations:

- **Dataset:** for each testcase $g = g_\ell, g_s, g_\eta, g_\gamma$, the dataset $\mathcal{D}$ is made of 10 000 pairs $(\boldsymbol{x}_i, y_i)$, with $\boldsymbol{x}_i$ randomly sampled with uniform distribution from the domain of the target function and $y_i = g(\boldsymbol{x}_i)$. Then, $\mathcal{D}$ is randomly split into the training set (5 600 pairs), validation set (1 400 pairs) and test set (3 000 pairs);

- **Data preprocessing:** standardization (see [25]) of the input data with respect to the training set;

- **Training options:** Mean Square Error (MSE) loss function, Adam optimizer (learning rate $\epsilon = 10^{-3}$, decay rates $\beta_1 = 0.9$ and $\beta_2 = 0.999$; see [19]) with learning rate reduction on plateaus (factor 0.75, patience 50 epochs), mini-batch size of 64 samples, 5 000 maximum number of epochs, early stopping with best-weights restoration (patience of 250 epochs) as regularizer.

The performance measure that we adopt to evaluate the results on the test set is the Mean Absolute Error (MAE), considering that the values of all the test functions are approximately between -2 and 6:

$$\mathrm{MAE}(\delta\mathrm{NN}, \mathcal{P}) := \frac{1}{|\mathcal{P}|} \sum_{(\boldsymbol{x}_i, y_i) \in \mathcal{P}} |y_i - \widehat{y}_i| \,,$$

where $\mathcal{P}$ is the test set and $\widehat{y}_i$ denotes the output prediction of the δNN for $\boldsymbol{x}_i$.

4.1.2. Performance Analysis for the Test Functions

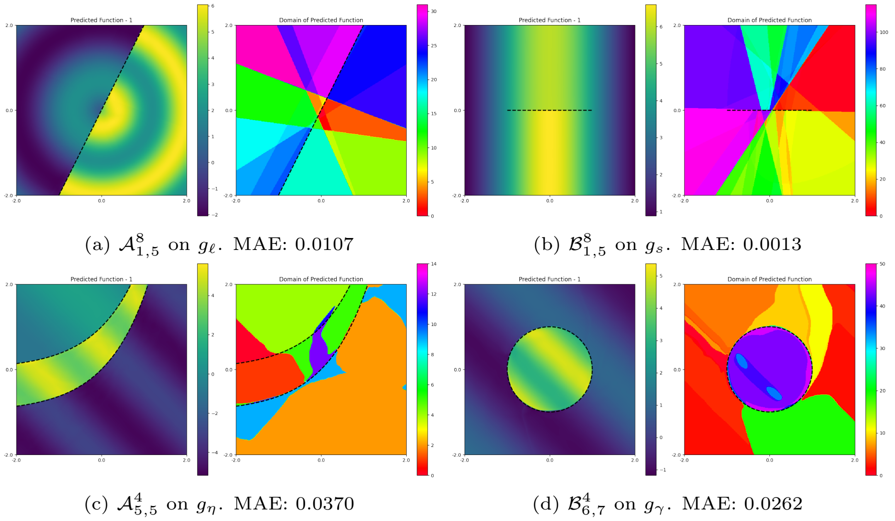
We start our analysis verifying the performance of the δNN in the regression task. Upon training all the NNs, with respect to all the test cases, we analyze the MAE of the models on the test set. The errors, reported in Table 1, prove that δNNs behave quite well in the regression task, and discontinuous layers do not hinder the regression abilities of Neural Networks.

MAE	g_ℓ	g_s	g_η	g_γ
mean	0.0906	0.0218	0.1868	0.0605
std	0.1597	0.1363	0.2706	0.0557
median	0.0274	0.0029	0.0695	0.0427

Table 1: Statistics of the MAEs over all the δNNs on the test sets of the functions $g_\ell, g_s, g_\eta, g_\gamma$.

Once the good approximation abilities of the δNNs are verified, we focus the analysis on the ability to identify the continuity regions. We recall that the computation of the generalized region vectors for an arbitrary set of points in the domain is extremely easy and fast since, by definition, $\mathcal{G}$ is characterized by subnetworks of the δNN (see (11)). In a nutshell, when we compute the prediction $\mathcal{L}_1^{H+1}(\boldsymbol{x}^{(0)})$ for a generic vector $\boldsymbol{x}^{(0)} \in \mathbb{R}^{N_0}$, we can easily obtain from the NN, at no additional cost, also the intermediate values $\mathcal{L}_1^h(\boldsymbol{x}^{(0)})$ needed in (11), returned by any arbitrary hidden layer L_h; the computation of $\mathcal{G}$ is therefore and extremely easy and fast task, for an arbitrary batch of vectors.

In Figure 5 we report some examples of results of the regression and interface detection tasks obtained on all the tests considered. For each subfigure the left plot reports the approximation of the corresponding function, and the right panel reports the continuity regions identified by the δNN. These last figures are obtained by picking a large enough number of points in the region D, and



(a) $\mathcal{A}^8_{1,5}$ on g_ℓ. MAE: 0.0107

(b) $\mathcal{B}^8_{1,5}$ on g_s. MAE: 0.0013

(c) $\mathcal{A}^4_{5,5}$ on g_η. MAE: 0.0370

(d) $\mathcal{B}^4_{6,7}$ on g_γ. MAE: 0.0262

Figure 5: Approximation of $g_\ell, g_s, g_\eta, g_\gamma$ with the δNNs $\mathcal{A}^8_{1,5}$, $\mathcal{B}^8_{1,5}$, $\mathcal{A}^4_{5,5}$, $\mathcal{B}^4_{6,7}$, respectively. For each subfigure we have the δNN function values (left) and the continuity regions (right). In the plots of the continuity regions, each region is identified by a different color, according to the colorbar on the right.

computing the corresponding region vector. Points with the same region vector are labeled with the same color. Note that according to Proposition 3.3 the number of continuity regions can be up to $2^{\delta N}$, being δN the total number of units in the discontinuous layers.

In general, we observe a very good approximation of both the test function and the actual continuity regions by the δNNs. In Figure 5, we show the results obtained for each test function on a selected δNN. Looking at the continuity regions of the δNNs, we observe the following phenomena.

Boundaries of the continuity regions. If the first hidden layer is discontinuous, then there are necessarily some continuity regions with straight boundaries. This is due to the fact that the boundaries of the continuity regions correspond to the counterimages through $\mathcal{L}_1^h$ of the hyperplanes $\Pi_j^{(h+1)}$ introduced by the discontinuous layers (see Theorem 3.1, item 4). When the first layer is a discontinuous one, since by convention $\mathcal{L}_1^0$ is the identity function (see (6)), the continuity region boundaries identified by the hyperplanes $\Pi_j^{(1)}$ are the hyperplanes themselves. In particular, if δL_1 is the only discontinuous layer, then the continuity regions have *only* linear boundaries. This phenomenon is depicted in Figure 5: figure top, left is obtained with a unique discontinuous layer which is the first hidden layer, whereas figure top, right is obtained with two discontinuous layers, which are the first and second hidden layers. Note that in the second case the boundaries separating the continuity regions identified are either lines or mildly curvilinear lines.

On the other hand, the more the discontinuous layers are located toward the output layer, the more the counter-images of the corresponding hyperplanes are obtained from the application of nonlinear functions, and the more the boundaries of the continuity regions can have a curved shape. This behaviour is still depicted in Figure 5: the bottom figures are obtained with discontinuos layers which are either the last one (bottom left case) or the last two (bottom right case): in both cases, highly curved boundaries are obtained.

Trade-off between approximation and discontinuity detection. The more the discontinuous layers are located toward the output layer, the more they are forced to focus on learning the discontinuity jumps of the target function. Indeed, a discontinuous layer near to the input can mainly focus on learning the discontinuity interface while spending not much effort on the precise values of the jump parameters ε, since the following layers can enlarge/shorten these jumps to reach appropriate values for the approximation task. Then, a discontinuous layer followed by few layers is forced also to learn the jumps, having less help available from the following layers.

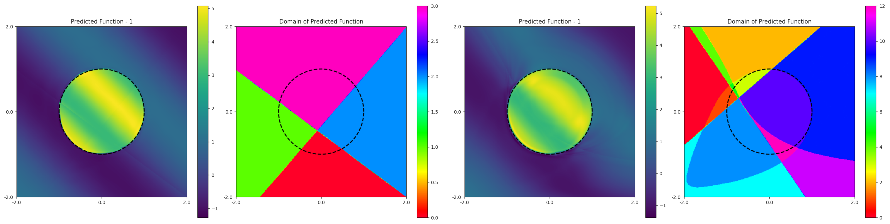
In few cases, we observe that the efforts of the δNN is mostly spent in learning the discontinuity interfaces, harming the function approximation. However, this mainly happens with discontinuous layers with few units (i.e., $d = 2$) that are not near to the input layer. This rare problems clearly depend on the dimensionality reduction inside the NN (from $\mathbb{R}^{128}$ to $\mathbb{R}^2$) that occurs toward the end of the network.

Since the δNNs can learn much more discontinuity interfaces than the ones actually characterizing the test functions, we observe that the models use the discontinuous layers also trying to adjust and improve the regression. In particular, we observe that in many cases the boundaries of the continuity regions partially "follow" the level curves of the target function but discontinuities are almost imperceptible.

From all the previous observations, the following indications can be deduced. If we are looking for discontinuities with an almost linear discontinuity interface, it is preferred to introduce discontinuous layers near to the input layer, whereas if we are looking for discontinuities with highly curvilinear interfaces, it is preferred to introduce discontinuous layers near to the output layer. Moreover, too small discontinuous layers should be avoided, and also the use of a discontinuous layer as first hidden layer should be avoided, unless it is *a priori* known that the discontinuity interface is a straight line or segment.

We remark that in case of an incorrect choice of the discontinuous layers, we observe that the approximation abilities very rarely are compromised even if the discontinuity interfaces are not learned. Typical examples are the ones illustrated in Figure 6, where two δNN approximating the function g_γ (see Figure 3-(d)) have continuity regions with boundaries that are "not-enough" curvilinear to approximate the circumference γ. Following these indications, the architectures used in Figure 5 are the best δNNs among the ones with

$h < \lceil H/2 \rceil$ for g_ℓ and g_s and among the ones with $h \geq \lceil H/2 \rceil$ for g_η and g_γ, where we recall that h is the index of the first discontinuous layer in the NN architecture and H is the depth of the NN.



(a) $\mathcal{A}^2_{1,5}$

(b) $\mathcal{C}^2_{1,5}$

Figure 6: Approximation of g_γ with the δNNs $\mathcal{A}^2_{1,5}$ and $\mathcal{C}^2_{1,5}$. For each subfigure we have the δNN function values (left) and the continuity regions (right). In the plots of the continuity regions, each region is identified by a different color, according to the colorbar on the right.

4.1.3. Continuity Region Clustering

In the previous section, we observed that most of the trained δNNs approximate with a good quality level both the functions and their continuity regions of the test functions. Nonetheless, in most of the cases, the continuity regions of the test functions are approximated by the δNNs with many continuity regions that are separated by negligible/inexistent discontinuities; in practice the method in its basic form largely overestimates the number of continuity regions. For example, in Figure 5-(b), we see more than 100 continuity regions but, clearly, the regions in the bottom half of the square can be considered as one single continuity region, as no discontinuities are perceived in the left plot; the same applies to the top half of the square. Then, to identify the actual continuity regions of the test functions, we introduce a method to group the continuity regions of the δNNs if the discontinuities on the boundaries are negligible/inexistent.

The development of such a kind of method is by far not trivial. From Proposition 3.1 we know that if a discontinuity jump parameter is zero, the δNN does not introduce a discontinuity on the corresponding continuity region boundary. Nonetheless, small values of the discontinuity jump parameters not necessarily correspond to negligible discontinuity jumps; indeed, as previously observed, a jump introduced by a discontinuous layer can be enlarged/shortened by the following layers. Then, we developed a clusterization method based both on the values of the discontinuity jump parameters and on the action they play inside the δNN.

Let N be a discontinuous H-layers perceptron defined as in Theorem 3.1, i.e. with M discontinuous layers $\delta L_{h_1+1}, \dots, \delta L_{h_M+1}$ such that $0 \leq h_1 < \cdots < h_M \leq H$. We recall that N_{h_m+1} is the number of units of the discontinuous layer δL_{h_m+1} and that δN denote the total number of units in discontinuos layers, namely $\delta N := \sum_{m=1}^{M} N_{h_m+1}$; we also recall that the generalized vector function is $\mathcal{G} : \mathbb{R}^{N_0} \to \{0,1\}^{\delta N}$. In view of the formal description of the clusterization method, we introduce here the following further notation:

- for each $\{i_1, \dots, i_k\} \subseteq \{1, \dots, \delta N\}$, we denote by $\mathcal{G}|_{i_1,\dots,i_k}$ the vector valued function whose elements are elements $i_1, \dots, i_k$ of $\mathcal{G}$ (see Definition 3.1);
- after introducing a global indexing for the discontinuity jump parameters, now labeled ε_i for $i = 1, \dots, \delta N$, we denote by $\mathcal{L}_1^{H+1}|_{\varepsilon_i=0}$ the characterizing function of the δNN obtained from N by setting to zero the discontinuity jump parameter ε_i;
- for any finite set of vectors $X = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_q\} \subset \mathbb{R}^{N_0}$, we denote by $\mathcal{L}_1^{H+1}(X)$ the matrix in $\mathbb{R}^{N_{H+1} \times q}$ defined as

  $$\mathcal{L}_1^{H+1}(X) := \left[\mathcal{L}_1^{H+1}(\boldsymbol{x}_1) \dots \mathcal{L}_1^{H+1}(\boldsymbol{x}_q)\right] ;$$

  we adopt the same convention for $\mathcal{L}_1^{H+1}|_{\varepsilon_i=0}(X)$.

We can now describe the clusterization method for the continuity regions of N. The proposed method defines new regions by suitably merging some continuity regions of the δNN, leveraging the function $\mathcal{G}|_{i_1,\dots,i_k}$. To understand the merging procedure, consider for example the function $\mathcal{G}|_i$, restricted to the index $i \in \{1, \dots, \delta N\}$ only: this restriction is blind to all the interfaces not related to ε_i and therefore only two regions in $\mathbb{R}^{N_0}$ are retained, which are, respectively: i) the union of all the continuity regions $\widehat{C}$ such that $\left(\mathcal{G}(\boldsymbol{x}^{(0)})\right)_i = \mathcal{G}|_i(\boldsymbol{x}^{(0)}) = 0$, for all $\boldsymbol{x}^{(0)} \in \widehat{C}$; ii) the union of all the continuity regions $\widetilde{C}$ such that $\left(\mathcal{G}(\boldsymbol{x}^{(0)})\right)_i = \mathcal{G}|_i(\boldsymbol{x}^{(0)}) = 1$ for all $\boldsymbol{x}^{(0)} \in \widetilde{C}$.

The number $k \in \mathbb{N}$ of discontinuity interfaces can be arbitrarily fixed. The clustering is based both on the values of the discontinuity jump parameters ε_i and on the action they play inside the δNN. Indeed, we introduce for each ε_i a rank value which depends not only on the size of ε_i itself, but also on its effect on the δNN; the latter dependence is obtained measuring the difference between $\mathcal{L}_1^{H+1}$ and $\mathcal{L}_1^{H+1}|_{\varepsilon_i=0}$, i.e. switching-off ε_i. Formally, let $X = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_q\} \subset \mathbb{R}^{N_0}$ be a finite set of vectors. Let $||\cdot||$ be a norm on $\mathbb{R}^{N_{H+1} \times q}$ and let $k \in \mathbb{N}$, $k \leq \delta N$, be fixed. For each $i = 1, \dots, \delta N$, we compute the rank value

$$\rho_i(X) := |\varepsilon_i| \cdot ||\mathcal{L}_1^{H+1}(X) - \mathcal{L}_1^{H+1}|_{\varepsilon_i=0}(X)|| \, . \tag{21}$$

In a nutshell, $\rho_i(X)$ measures how much ε_i contributes to the outputs of the δNN, weighted by the value of ε_i itself. The higher $\rho_i(X)$ is, the more likely the discontinuity interface corresponding to ε_i approximates a real discontinuity interface of the target function. Then, retaining only the discontinuity interfaces corresponding to the k highest rank values only, we can merge the continuity regions that are unlikely to be separated by a discontinuity interface of the target function. Following this idea, the rank values are sorted in descending order, and continuity regions separated by hyperplanes corresponding to parameters ε_i with the smallest rank values are merged, in such a way that we end with a fixed number of discontinuity interfaces.

The above procedure can be sketched in the following algorithm.

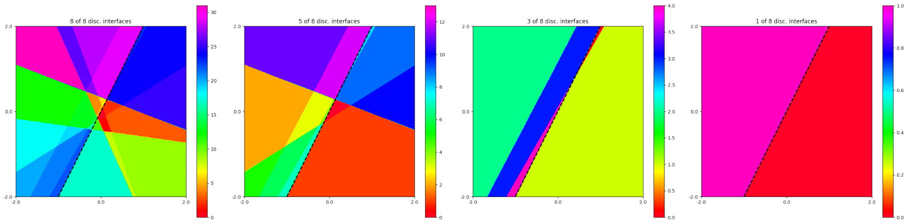
Algorithm 4.1 (Clusterization Method for Continuity Regions of a δNN)**.** *Let* $\mathcal{N}$ *be a discontinuous H-layers perceptron defined as in Theorem 3.1 and let $k \in \mathbb{N}$, $k \leq \delta N$, be the number of indices with respect to which I want to perform the continuity region clustering of* $\mathcal{N}$*. Then:*

1. *For each $i = 1, \ldots, \delta N$, compute the rank value $\rho_i(X)$ as in (21);*
2. *sort the rank values in descending order: $\rho_{i_1}(X) \geq \cdots \geq \rho_{i_{\delta N}}(X)$. Let $i_1, \ldots, i_k$ be the indices corresponding to the largest rank values;*
3. *Compute the new regions with respect to $\mathcal{G}|_{i_1,\ldots,i_k}$.*

Algorithm 4.1 represents a first attempt to build an effective method to identify the continuity regions of a target unknown function using a δNN. Testing it on the best δNNs selected for the test functions (see Figure 5), we observe extremely good results. In this tests we use the infinity norm for the rank values computations (see (21)), as a preliminary analysis showed better clustering performance with respect to the ℓ_1 and ℓ_2 norms.

As far as functions g_ℓ, g_s, and g_η are concerned, regions returned by Algorithm 4.1 follow very well the actual discontinuity interfaces of the test function, for each $1 \leq k \leq \delta N$ (see Figures 7-9). The only exception is the case of g_γ, in which the method is not able to catch the circumference for $k = 1$ (see Figure 10), and $k = 3$ is needed to reach the target. However, we observe that the method is able to detect the circumference γ with other non-optimal δNNs (e.g., see Figure 11); this phenomenon is still under investigation and may suggest that a basic error-based criterion not necessarily select the δNN that best identify the discontinuity interfaces of the target function.

Nevertheless, δNNs proved to have the potential for being a new useful tool for the discontinuity detection problem.

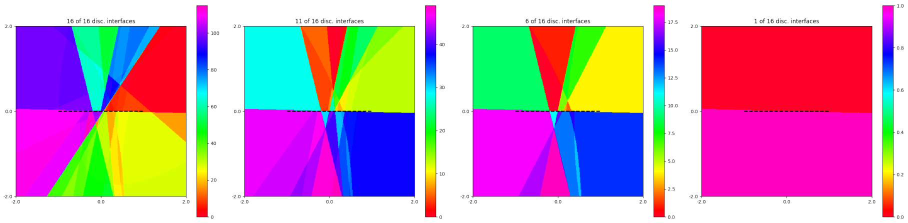


(a) $k = \delta N = 8$ (b) $k = 5$ (c) $k = 3$ (d) $k = 1$

Figure 7: Test 1. Regions returned by Algorithm 4.1 and $\mathcal{A}^{8}_{1,5}$

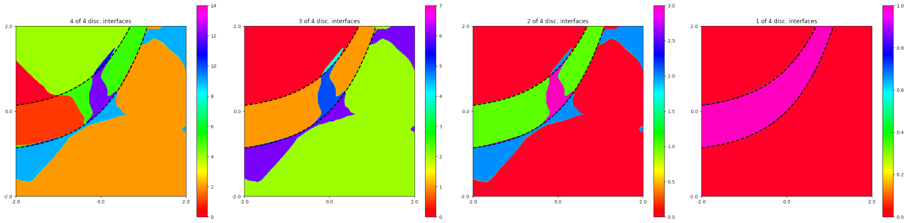
4.2. Real Data Test Case

The tests reported in the previous subsection are based on the synthetic functions $g_\ell, g_s, g_\eta, g_\gamma$; here we train a δNN to approximate a discontinuous function obtained from real data, considering the phase transition phenomenon of a given medium. For any medium, the state of matter can be identified by



(a) $k = \delta N = 16$ (b) $k = 11$ (c) $k = 6$ (d) $k = 1$

Figure 8: Test 2. Regions returned by Algorithm 4.1 and $\mathcal{B}^8_{1,5}$



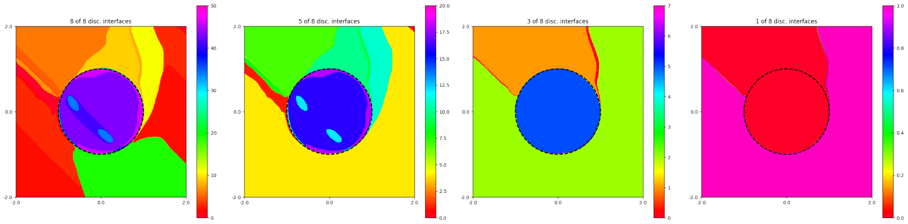
(a) $k = \delta N = 4$ (b) $k = 3$ (c) $k = 2$ (d) $k = 1$

Figure 9: Test 3. Regions returned by Algorithm 4.1 and $\mathcal{A}^4_{5,5}$

some parameters, such as the density ρ. However, the density of a medium depends on the temperature T and the pressure p; i.e., for each medium there exists a function g such that

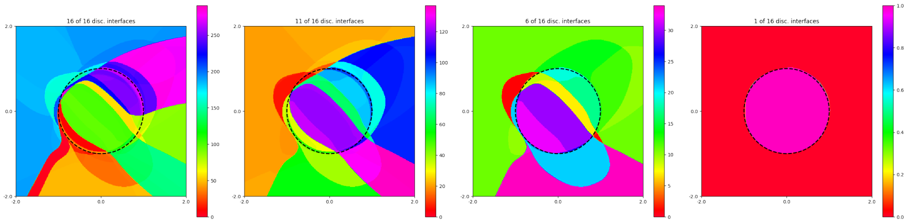
$$\rho = g(T, p)\,. \tag{22}$$

For a wide range of media, experimental density measurements show that the function g is characterized by discontinuity interfaces that separate two or more regions of the (T, p) plane representing different states of matter; in particular, the points along these discontinuity interfaces are defined as equilibrium points with respect to the states of matter they separate. On the other hand, not necessarily a phase transition takes place through a discontinuity jump of the density and can happen continuously, crossing a region of so-called supercritical



(a) $k = \delta N = 8$ (b) $k = 5$ (c) $k = 3$ (d) $k = 1$

Figure 10: Test 4. Regions returned by Algorithm 4.1 and $\mathcal{B}^4_{6,7}$

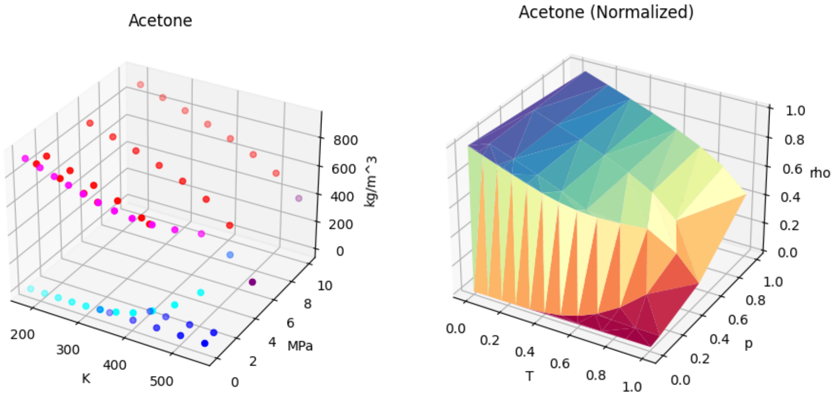


(a) $k = \delta N = 16$ (b) $k = 11$ (c) $k = 6$ (d) $k = 1$

Figure 11: Test 4. Regions returned by Algorithm 4.1 and $\mathcal{B}^8_{4,7}$

points of the (T, p) plane. This classification of phase transition phenomena is called Ehrenfest classification [15].

Due to the discontinuous behavior of (22), we perform a new numerical test that involves δNNs and the acetone medium. Specifically, we train a δNN to approximate the acetone's density function (from now on labeled g_a) and its discontinuity interface, starting from the data reported in [1] with pressure smaller than or equal to 10MPa (see Figure 12-(a)).



(a) (b)

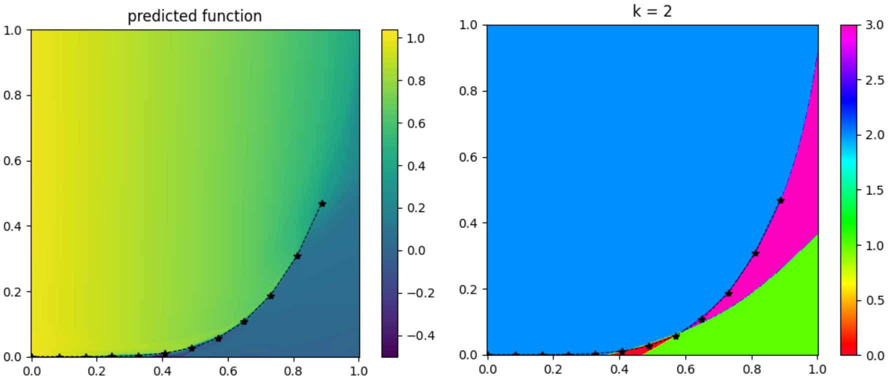
Figure 12: (a): (T, p, ρ) points of acetone taken from [1]; red dots correspond to liquid state, magenta to liquid state at equilibrium, blue to gas state, cyan to gas state at equilibrium, and purple to supercritical points. (b): Acetone's density function $\widetilde{g}_a$ reconstructed from the experimental measures reported in subfigure (a); temperature, pressure, and density are normalized.

Looking at these data, we observe that the discontinuity interface described by the equilibrium points is a nonlinear "rip". Then, following the indications at the end of Section 4.1.2, we avoid to train a large set of δNNs and we only train a δNN with architecture $\mathcal{B}^8_{3,5}$, as it is expected to fit sufficiently well this type of discontinuity interface. The training of the δNN is characterized by the same hyper-parameters and training options described in Section 4.1.1, where the dataset $\mathcal{D}$ of pairs $((T_i, p_i), \rho_i)$ is generated using the piece-wise linear density function $\widetilde{g}_a$ reconstructed from the experimental measures (see Figure 12-(b)).

The trained δNN shows very good approximation results of both the function (see Table 2 and Figure 13-(a)) and the continuity regions (Figure 13-(b)). Indeed, the MAE on the test set is approximately $5 \cdot 10^{-3}$ and, using the clustering algorithm of Section 4.1.3, we observe a quite accurate detection of the nonlinear rip of g_a. In details, the clustering algorithm for the continuity regions of $\mathcal{B}^8_{3,5}$ is applied starting from the $\delta N = 16$ discontinuity interfaces of the δNN, using the infinity norm for the computation of the rank values. Then, for each $1 < k \leq \delta N$, the clustering algorithm shows that the discontinuity interface of g_a is approximated very well by the ones of the δNN (see Figure 13-(b), for the case $k = 2$).

	mean (MAE)	std	median
Absolute Error	0.0048	0.0192	0.0015

Table 2: $\mathcal{B}^8_{3,5}$. Statistics of the absolute error for the test set of acetone's density function.



(a)

(b)

Figure 13: (a): approximation of g_a returned by $\mathcal{B}^8_{3,5}$; (b): continuity regions of $\mathcal{B}^8_{3,5}$ after the clustering algorithm, $k = 2$. In both panels, black markers and dashed lines denote the discontinuity interface defined by the equilibrium points.

4.3. Simulation Study

We conclude the numerical experiments performing a simulation study that involves 50 synthetic discontinuous functions $g : D \to \mathbb{R}$, with $D = [-2, 2] \times [-2, 2]$, in order to make a quantitative analysis of the continuity region detection ability of the δNNs, considering as a reference method the one described in [5], that is characterized by the following steps. In the one-dimensional case, letting $[g]$ denote the jump function of the target function g (i.e., $[g](x) = 0$ if g is continuous in x, otherwise $[g](x)$ is equal to the discontinuity jump), for any fixed $x \in \mathbb{R}$ a high order polynomial approximation $L_m g(x)$ of $[g](x)$ is built, using function evaluations on a local stencil made of $m+1$ points near x; in order to reduce misinterpretations possibly induced by oscillations which may occur for large values of m, the minmod limiter is also applied. The key property of such approximation is that it converges rapidly to $[g](x)$, with a rate depending on m and on the local smoothness of g; namely, away from discontinuities, we have $L_m g(x) \simeq \mathcal{O}(h^{\min(m,k)})$, where k is such that $g \in C^k$ in a neighborhood of x, and h is the grid-size; close to discontinuities, we have $L_m g(x) \simeq [g](x) + \mathcal{O}(h)$. Hence, once the polynomial approximation of $[g]$ is built, this can be used to detect presence of jumps by analyzing its departure from 0: introducing a tolerance $\tau > 0$, we assume the presence of a discontinuity interface between two points if the absolute value of the approximated jump in the midpoint is greater than τ. For a function $g : \mathbb{R}^n \to \mathbb{R}$, the one-dimensional method is applied to each one of the n dimensions.

4.3.1. Generation of Random Discontinuous Functions

Let P_n denote the Legendre polynomial of degree n; the 50 random discontinuous functions considered herein are piece-wise smooth functions $g_\pi : D \to \mathbb{R}$ defined as

$$g_\pi(\boldsymbol{x}) = \begin{cases} g_1(\boldsymbol{x}) \,, & \text{if } \pi(x_1) - x_2 \geq 0 \\ g_2(\boldsymbol{x}) \,, & \text{otherwise} \end{cases} \,, \tag{23}$$

where:

- the functions g_1, g_2 are defined as

  $$g_j(\boldsymbol{x}) = \sum_{h+k \leq 4} c_{hk} P_h\left(\frac{x_1+2}{4}\right) P_k\left(\frac{x_2+2}{4}\right) \,, \quad \forall\, j = 1, 2 \,, \tag{24}$$

  with c_{hk} random coefficients sampled from the normal distribution $\mathcal{N}(0, 10)$;

- the function π is defined as

  $$\pi(x) = C \frac{q(x)}{\max_{-2 \leq y \leq 2} |q(y)|} \,, \tag{25}$$

  with

  $$q(x) = \sum_{h=0}^{4} c_h P_h\left(\frac{x+2}{4}\right) \,, \tag{26}$$

  $C > 0$, and c_h random coefficients sampled from the normal distribution $\mathcal{N}(0, 10)$; namely, π is the polynomial q rescaled to a maximum absolute value equal to C. In our experiments, we used $C = 1.75$. See Figure 14 for some examples of test functions.

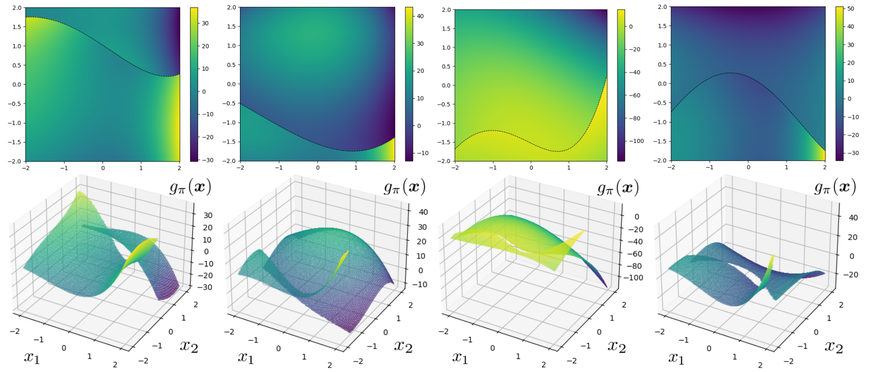


Figure 14: Top view (top row) and 3D view (bottom row) of the first four random functions generated for the simulation study. The dotted curves in the top views correspond to the discontinuity interfaces given by the random polynomials π.

4.3.2. Quantification of the Continuity Region Detection Ability

According to the indications given at the end of Section 4.1.2, for this study we only train a δNN with architecture $\mathcal{B}^8_{3,5}$, as already made in Section 4.2, for each one of the 50 functions; the training options and the hyper-parameters of the δNN are the same described in Section 4.1.1. Since all the problems are characterized by two continuity regions, we use the clustering algorithm to identify the highest-rank parameter ε_i of each δNN, that identifies the two "main" continuity regions of the NN's characterizing function; namely, we run the clustering algorithm with $k = 1$.

For each test function, we also run the discontinuity interface detection method of [5] in order to classify the 2500 points of the 50-by-50 regular grid in D, generated by the cartesian product of 50 equally spaced points on each axis, including the extrema of the interval $[-2, 2]$. Specifically, we use the method in [5] to approximate the jump function in the midpoints between any pair of neighboring points in the grid. Then, we perform the domain classification algorithm in [5] for the points of the regular grid, with respect to two continuity regions.

In brief, the domain classification method in [5] can be summarized by the following procedure:

1. starting from $(x_1^{(0)}, x_2^{(0)})$, move along the two domain dimensions and visit the points of the grid. The movement is blocked if a discontinuity jump is detected in the midpoint along the movement direction;

2. when it is no more possible to visit other grid points, classify the visited points as belonging to "region 1". Then, classify the non-visited grid points as members of "region 2".

We run the method in [5] with $m = 4$ and tolerance values $\tau = 10^{-t}$, $t = 1, \ldots, 4$. We classify the points of the regular grid also by means of the continuity regions identified by the δNNs. Then, we compare these classifications, measuring the accuracy and the weighted F_1-score [2] with respect to the real continuity regions of the functions. Here, we briefly recall the definition of these two values.

- **Accuracy:** let $\mathcal{C} = \{C_1, \dots, C_M\}$ be a set of M classes associated to the vectors of a finite set $\{\boldsymbol{x}_1, \dots, \boldsymbol{x}_N\} := \mathcal{X} \subset \mathbb{R}^n$. For each $i = 1, \dots, N$, we denote by $c_i \in \mathcal{C}$ the true class of $\boldsymbol{x}_i$ and by $\widehat{c_i} \in \mathcal{C}$ the class predicted by a given model $\widehat{C}$ for $\boldsymbol{x}_i$. Then, the accuracy of $\widehat{C}$ is the ratio between the number of elements in $\mathcal{X}$ whose class has been correctly predicted, over the cardinality of $\mathcal{X}$ itself, i.e.:

$$\mathrm{acc}_{\widehat{C}}(\mathcal{X}) := \frac{|\{\boldsymbol{x_i} \in \mathcal{X} \,|\, c_i = \widehat{c_i}\}|}{N} \,.$$

  The accuracy of a model is often reported as a percentage.

- F_1**-score:** let us consider the sets $\mathcal{X}$ and $\mathcal{C}$, and the model $\widehat{C}$ as defined in the previous item. If the classes are not uniformly distributed among the elements of $\mathcal{X}$, the accuracy can be not enough to describe the classification performances of $\widehat{C}$; e.g., if 90% of the elements of $\mathcal{X}$ belong to class $C_1 \in \mathcal{C}$ and $\widehat{C}$ predicts constantly the class C_1 for any input, then the accuracy of the model on $\mathcal{X}$ would be 90%. Therefore, other two values are typically introduced to describe the performances of the model $\widehat{C}$: the precision and the recall, with respect to each class in $\mathcal{C}$. In a nutshell, the precision of $\widehat{C}$ with respect to class $C_j \in \mathcal{C}$ is the ratio between the number of correct predictions of C_j over the number of all the elements predicted as belonging to class C_j, i.e.:

$$\mathrm{prec}_{\widehat{C}}(C_j, \mathcal{X}) := \frac{|\{\boldsymbol{x_i} \in \mathcal{X} \,|\, c_i = C_j = \widehat{c_i}\}|}{|\{\boldsymbol{x_i} \in \mathcal{X} \,|\, \widehat{c_i} = C_j\}|} \,;$$

  on the other hand, the recall of $\widehat{C}$ with respect to class $C_j \in \mathcal{C}$ is the ratio between the number of correct predictions of C_j over the number of all the elements really belonging to class C_j, i.e.:

$$\mathrm{rec}_{\widehat{C}}(C_j, \mathcal{X}) := \frac{|\{\boldsymbol{x_i} \in \mathcal{X} \,|\, c_i = C_j = \widehat{c_i}\}|}{|\{\boldsymbol{x_i} \in \mathcal{X} \,|\, c_i = C_j\}|} \,.$$

  Then, in order to aggregate the precision and recall values, the F_1-score of $\widehat{C}$ with respect to class C_j is introduced and defined as the harmonic mean of these values, i.e.:

$$F_{1\widehat{C}}(C_j, \widehat{X}) := 2\,\frac{\mathrm{prec}_{\widehat{C}}(C_j, \mathcal{X}) \cdot \mathrm{rec}_{\widehat{C}}(C_j, \mathcal{X})}{\mathrm{prec}_{\widehat{C}}(C_j, \mathcal{X}) + \mathrm{rec}_{\widehat{C}}(C_j, \mathcal{X})} \,.$$

  In the end, the weighted F_1-score of $\widehat{C}$ with respect to $\mathcal{X}$ is the weighted mean of the classes' F_1-scores, where the weights are the number of elements really belonging to the classes in $\mathcal{X}$.

	Accuracy				
	[5], $t = 4$	[5], $t = 3$	[5], $t = 2$	[5], $t = 1$	$\mathcal{B}^8_{3,5}$
mean	0.8559	0.8973	0.8777	0.8576	0.9103
std	0.2070	0.1645	0.1720	0.1789	0.1329
median	0.9872	0.9944	0.9942	0.9948	0.9720

Table 3: Statistics over the 50 random test functions of the accuracy w.r.t. the continuity region classification performed by method [5] for several tolerance values $\tau = 10^{-t}$, and $\mathcal{B}^8_{3,5}$

	F_1-score				
	[5], $t = 4$	[5], $t = 3$	[5], $t = 2$	[5], $t = 1$	$\mathcal{B}^8_{3,5}$
mean	0.8210	0.8598	0.8324	0.8039	0.9040
std	0.2562	0.2257	0.2363	0.2462	0.1489
median	0.9873	0.9944	0.9942	0.9948	0.9722

Table 4: Statistics over the 50 random test functions of the F_1-score w.r.t. the continuity region classification performed by method [5] for several tolerance values $\tau = 10^{-t}$, and $\mathcal{B}^8_{3,5}$

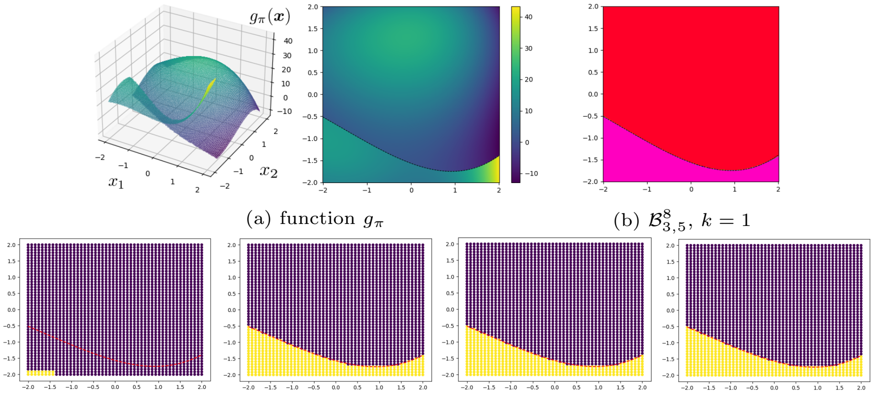
Looking at the results in Table 3 and Table 4, which report statistics on the accuracy and the F_1-score, we clearly observe the good performances of the δNN, also with respect to method [5]. Indeed, the δNN has both an average accuracy and an average F_1-score larger than the method in [5], and with a smaller standard deviation; only the median accuracy and median F_1-score are slightly larger for [5] than for the δNN.

The lower performance of method [5], especially with respect to the F_1-score, depends on the fact that this method typically returns only very good or very bad domain classifications. Indeed, even only one missed jump (along one of the two domain dimensions) may lead [5] to a total misclassification of one of the two regions; this misclassification is more frequent for large values of τ. It is worth noting that in most cases missed jumps occur, independently of the tolerance value, at points of the discontinuity interface π where the two functions g_1 and g_2 take the same value (i.e., the jump is zero at that point). On the other hand, if false discontinuity points are detected, undesired discontinuity interfaces are built, which let only partially identify one of the two regions during the domain classification algorithm [5]; this misclassification is more frequent when τ is small.

In both cases, typically we have an almost total miscalssification of one continuity region from [5]. On the other hand, the δNNs very rarely misclassify *an entire* region, and for this reason the performance statistics of $\mathcal{B}^8_{3,5}$ are characterized by a higher mean and a smaller standard deviation, especially for the

F_1-score.

On the other hand, the possible difficulties of the δNN are likely to be the same already observed in Section 4.1; i.e., the architecture $\mathcal{B}^8_{3,5}$ is not always the best one for the test function g_π and, therefore, the discontinuous layers sometimes focus too much on the function approximation, rather than on learning the discontinuity interface π.



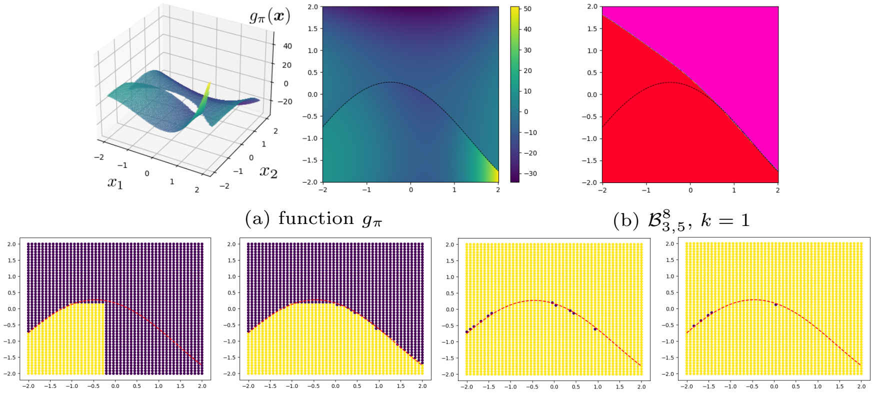
(a) function g_π

(b) $\mathcal{B}^8_{3,5}$, $k = 1$

(c) [5], $\tau = 10^{-4}, \dots, 10^{-1}$

Figure 15: Classification example 1. (a) 3D and top view of g_π; (b) domain classification with the δNN $\mathcal{B}^8_{3,5}$ (regions identified are colored in red and magenta); (c) domain classification with [5], $\tau = 10^{-4}, \dots, 10^{-1}$, left to right (regions identified are colored in yellow and purple). The dotted line is the actual discontinuity interface π.

We conclude this simulation study showing some examples of misclassification for [5], $\mathcal{B}^8_{3,5}$, or both (Figures 15-17). In Figure 15 we depict a situation in which both $\mathcal{B}^8_{3,5}$ and [5] classify almost perfectly the continuity regions of g_π. The only exception is [5], $\tau = 10^{-4}$, where a too small tolerance causes an overidentification of discontinuity points and the discontinuity regions are not properly identified (see the leftmost picture of subfigure (c)). In Figure 16 both $\mathcal{B}^8_{3,5}$ and [5] show difficulties in the domain classification and only [5], $\tau = 10^{-3}$, is able to make an almost perfect classification. In particular, $\tau = 10^{-2}$ and $\tau = 10^{-1}$ are too large and small jumps at the discontinuity interface π are not detected (see the two rightmost pictures of subfigure (c)). On the other hand, a too small tolerance, such as $\tau = 10^{-4}$, once again yields an overidentification of discontinuities and a consequent misclassification (see the leftmost picture of subfigure (c)). Finally, in Figure 17 only $\mathcal{B}^8_{3,5}$ is able to make a good domain classification. The difficulties of [5] in this case depend on the fact that there is a point on the discontinuity interface π where g_1 and g_2 take the same value, and the approximated jump function is zero; then, a jump is never detected near this point. Therefore, almost all the domain is classified as belonging to one region (see the three pictures on the right of subfigure (c)). Only the tolerance value $\tau = 10^{-4}$ can mitigate the effect of this situation, because it combines the problem here illustrated with the problem related to a small tolerance (see the

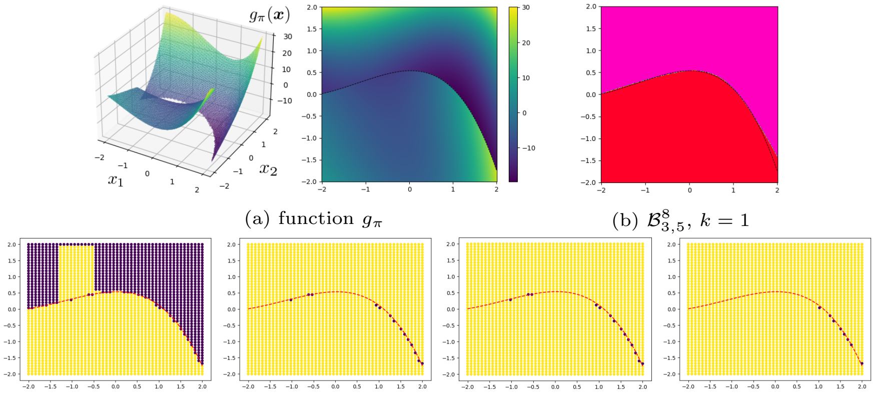


(a) function g_π

(b) $\mathcal{B}^8_{3,5}$, $k = 1$

(c) [5], $\tau = 10^{-4}, \ldots, 10^{-1}$

Figure 16: Classification example 2. (a) 3D and top view of g_π; (b) domain classification with the δNN $\mathcal{B}^8_{3,5}$ (regions identified are colored in red and magenta); (c) domain classification with [5], $\tau = 10^{-4}, \ldots, 10^{-1}$, left to right (regions identified are colored in yellow and purple). The dotted line is the actual discontinuity interface π.



(a) function g_π

(b) $\mathcal{B}^8_{3,5}$, $k = 1$

(c) [5], $\tau = 10^{-4}, \ldots, 10^{-1}$

Figure 17: Classification example 3. (a) 3D and top view of g_π; (b) domain classification with the δNN $\mathcal{B}^8_{3,5}$ (regions identified are colored in red and magenta); (c) domain classification with [5], $\tau = 10^{-4}, \ldots, 10^{-1}$, left to right (regions identified are colored in yellow and purple). The dotted line is the actual discontinuity interface π.

leftmost picture in subfigure (c)), but the classification is still incorrect.

5. Conclusions

We have presented a novel typology of layers for Neural Network models, characterized by a discontinuous map where the discontinuity action is obtained adding a vector of multiples of the Heaviside function applied to input signals of the layer (see (4)); consequently, the function $\mathcal{L}_1^{H+1}$ of a NN with at least one of these discontinuous layers could be discontinuous, too. Denoting by δNN such a NN, we analyzed and studied the theoretical properties that characterize their maps $\mathcal{L}_1^{H+1}$. Some useful results have been proven (see Section 3), concerning the discontinuities introduced in the δNNs.

We have also performed numerical experiments to show possible applications of δNNs to discontinuous functions. We started considering four different examples with increasing complexity to analyze the sensitivity of the new NN models in both approximating the discontinuous functions and detecting their discontinuity interfaces using the continuity regions. Extremely interesting results have been obtained, showing a reat deal of potential for the new δNNs; indeed, the δNNs proved to have a remarkable ability in detecting the actual discontinuity interfaces of the approximated function, without compromising the function approximation ability typical of the NNs. Since in its basic form the method proposed overestimate the number of continuity regions, we also propose a method for clustering the continuity regions of the δNN in order to have a more precise identification of the actual continuity regions of the original function.

Given the analyses of the δNNs on the four examples, we extended the numerical experiments to a similar problem based on real data arising from a phase transition phenomenon. In particular, according to the observations on the previous experiments, we select only one δNN architecture to approximate the target function. Also in this case, the new discontinuous NN model shows both good regression skills and good continuity region detection abilities. In the end, we presented a simulation study that involved 50 random discontinuous functions, where a comparison of the continuity region detection abilities of δNNs are compared with the ones of a well-assessed method. The results of the study are extremely interesting, showing that the detection performances of the δNNs are very promising.

Data Availability Statement

The code used to implement the Discontinuous Neural Networks is available at https://github.com/Fra0013To/deltaNN (accessed on 11 August 2022).

Acknowledgements

Research performed in the framework of the Italian MIUR Award "Dipartimento di Eccellenza 2018-2022" to the Department of Mathematical Sciences, Politecnico di Torino, CUP: E11G18000350001. The research has also been partially supported by INdAM-GNCS, by the Italian MIUR PRIN project 201752HKH8_003, and by the SmartData@PoliTO center for Big Data and Machine Learning technologies. F.D.S. acknowledges the support from INdAM - GNCS "Young Researchers Funding 2022-2023" Project, CUP: E55F22000270-001.

References

[1] *Acetone - density and specific weight*. https://www.engineeringtoolbox.com/acetone-2-propanone-density-specific-weight-temperature-pressure-d_2038.html. Accessed on: 2022-03-11.

[2] *Scikit-learn: F1-score function*. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html. Accessed on: 2022-04-04.

[3] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, *TensorFlow: Large-scale machine learning on heterogeneous systems*, 2015. Software available from tensorflow.org.

[4] A. Apicella, F. Donnarumma, F. Isgrò, and R. Prevete, *A survey on modern trainable activation functions*, arXiv, (2020).

[5] R. Archibald, A. Gelb, R. Saxena, and D. Xiu, *Discontinuity detection in multivariate space for stochastic simulations*, Journal of Computational Physics, 228 (2009), pp. 2676–2689.

[6] R. Archibald, A. Gelb, and J. Yoon, *Polynomial fitting for edge detection in irregularly sampled signals and images*, SIAM Journal on Numerical Analysis, 43 (2005), pp. 259–279.

[7] E. J. Candès, *Ridgelets: Estimating with ridge functions*, Annals of Statistics, 31 (2003), pp. 1561–1599.

[8] C. Canuto, S. Pieraccini, and D. Xiu, *Uncertainty quantification of discontinuous outputs via a non-intrusive bifidelity strategy*, Journal of Computational Physics, 398 (2019).

[9] D. Clevert, T. Unterthiner, and S. Hochreiter, *Fast and accurate deep network learning by exponential linear units (elus)*, in 4th International Conference on Learning Representations, ICLR 2016, San Juan,

Puerto Rico, May 2-4, 2016, Conference Track Proceedings, Y. Bengio and Y. LeCun, eds., 2016.

[10] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016. www.deeplearningbook.org.

[11] I. J. Goodfellow, M. Mirza, A. Courville, and Y. Bengio, *Multi-prediction deep Boltzmann machines*, Advances in Neural Information Processing Systems, (2013), pp. 1–9.

[12] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016-Decem (2016), pp. 770–778.

[13] M. Imaizumi and K. Fukumizu, *Deep Neural Networks Learn Non-Smooth Functions Effectively*, Proceedings of Machine Learning Research, 89 (2019), pp. 869–878.

[14] M. Imaizumi and K. Fukumizu, *Advantage of deep neural networks for estimating functions with singularity on curves.* arXiv preprint arXiv:2011.02256, 2020.

[15] G. Jaeger, *The Ehrenfest Classification of Phase Transitions: Introduction and Evolution*, Archive for History of Exact Sciences, 53 (1998), pp. 51–81.

[16] J. D. Jakeman, R. Archibald, and D. Xiu, *Characterization of discontinuities in high-dimensional stochastic problems on adaptive sparse grids*, Journal of Computational Physics, 230 (2011), pp. 3977–3997.

[17] J. D. Jakeman, A. Narayan, and D. Xiu, *Minimal multi-element stochastic collocation for uncertainty quantification of discontinuous functions*, Journal of Computational Physics, 242 (2013), pp. 790 – 808.

[18] P. Kidger and T. Lyons, *Universal Approximation with Deep Narrow Networks*, in Proceedings of Thirty Third Conference on Learning Theory, J. Abernethy and S. Agarwal, eds., vol. 125 of Proceedings of Machine Learning Research, PMLR, 09–12 Jul 2020, pp. 2306–2327.

[19] D. P. Kingma and J. L. Ba, *Adam: A method for stochastic optimization*, 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings, (2015), pp. 1–15.

[20] A. Krizhevsky, I. Sutskever, and G. Hinton, *ImageNet Classification with Deep Convolutional Neural Networks*, Advances in Neural Information Processing Systems, (2012).

[21] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken, *Multilayer feed-forward networks with a nonpolynomial activation function can approximate any function*, Neural Networks, 6 (1993), pp. 861–867.

[22] W. Ma and J. Lu, *An Equivalence of Fully Connected Layer and Convolutional Layer*, Tech. Rep. 3, 2017.

[23] C. Marcati, J. A. Opschoor, P. C. Petersen, and C. Schwab, *Exponential relu neural network approximation rates for point and edge singularities.* arXiv preprint arXiv:2010.12217, 2020.

[24] W. S. McCulloch and W. H. Pitts, *A Logical Calculus of the Ideas Immanent in Nervous Activity*, Bulletin of Mathematical Biophysics, 5 (1943), pp. 115–133.

[25] N. M. Nawi, W. H. Atomi, and M. Rehman, *The Effect of Data Preprocessing on Optimized Training of Artificial Neural Networks*, Procedia Technology, 11 (2013), pp. 32–39.

[26] S. Park, C. Yun, J. Lee, and J. Shin, *Minimum width for universal approximation*, in International Conference on Learning Representations, 2021.

[27] P. Petersen and F. Voigtlaender, *Optimal approximation of piecewise smooth functions using deep ReLU neural networks*, Neural Networks, 108 (2018), pp. 296–330.

[28] A. Pinkus, *Approximation theory of the MLP model in neural networks*, Acta Numerica, 8 (1999), pp. 143–195.

[29] F. Rosenblatt, *The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain*, Psychological Review, 65 (1958), pp. 386–408.

[30] Z. Shen, H. Yang, and S. Zhang, *Deep Network With Approximation Error Being Reciprocal of Width to Power of Square Root of Depth*, Neural Computation, 33 (2021), pp. 1005–1036.

[31] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, *DeepFace: Closing the Gap to Human-Level Performance in Face Verification*, Computer Vision Foundation, (2014).

[32] B. Widrow and M. E. Hoff, *Adaptive switching circuits*, 1960 IRE WESCON Convention Record, 4 (1960), pp. 96–104.

[33] G. Zhang, C. G. Webster, M. Gunzbuerger, and J. Burkardt, *Hyperspherical sparse approximation techniques for high-dimensional discontinuity detection*, SIAM Review, 58 (2016), pp. 517–551.

[34] S. Zhang, X. Zou, J. Ahlquist, I. M. Navon, and J. G. Sela, *Use of differentiable and nondifferentiable optimization algorithms for variational data assimilation with discontinuous cost functions*, Monthly Weather Review, 128 (2000), pp. 4031–4044.