



## Full Length Article



## A topological description of loss surfaces based on Betti Numbers

Maria Sofia Bucarelli <sup>a,\*</sup>, Giuseppe Alessio D'Inverno <sup>b,1</sup>, Monica Bianchini <sup>b</sup>, Franco Scarselli <sup>b</sup>, Fabrizio Silvestri <sup>a</sup>

<sup>a</sup> DIAG, Sapienza University of Rome, Piazzale Aldo Moro 5, Rome, 00185, Italy

<sup>b</sup> Department of Information Engineering and Mathematics, University of Siena, Via Roma 56, Siena, 53100, Italy

## ARTICLE INFO

## Keywords:

Betti numbers

Loss surface

ResNet

$\ell_2$  regularization

## ABSTRACT

In the context of deep learning models, attention has recently been paid to studying the surface of the loss function in order to better understand training with methods based on gradient descent. This search for an appropriate description, both analytical and topological, has led to numerous efforts in identifying spurious minima and characterize gradient dynamics. Our work aims to contribute to this field by providing a topological measure for evaluating loss complexity in the case of multilayer neural networks. We compare deep and shallow architectures with common sigmoidal activation functions by deriving upper and lower bounds for the complexity of their respective loss functions and revealing how that complexity is influenced by the number of hidden units, training models, and the activation function used. Additionally, we found that certain variations in the loss function or model architecture, such as adding an  $\ell_2$  regularization term or implementing skip connections in a feedforward network, do not affect loss topology in specific cases.

## 1. Introduction

In recent years, the investigation into the theoretical foundations of Machine Learning (ML) and Deep Learning (DL) models has gained increasing attention as the research community has decided to delve deeper into the reasons why these models can achieve exceptional performance in different application fields (Devlin, Chang, Lee, & Toutanova, 2018; Koren, Bell, & Volinsky, 2009; Krizhevsky, Sutskever, & Hinton, 2017; Radford, Narasimhan, Salimans, Sutskever, et al., 2018; Scarselli, Gori, Tsoi, Hagenbuchner, & Monfardini, 2008). On the one hand, as the automated decisions provided by these algorithms can have a relevant impact on people's lives, their behavior has to be aligned with the values and principles of individuals and society. This demands designing automated methods we can trust, fulfilling the requirements of fairness, robustness, privacy, and explainability (Oneto et al., 2022). On the other hand, a wide range of tools that have arisen from different areas of mathematics and statistical mechanics have been taken into account to give proper explanations to the behavior of ML and DL models according to different mathematical aspects, such as the gradient descent dynamics (Goodfellow, Vinyals, & Saxe, 2014; Maennel, Bousquet, & Gelly, 2018; Williams et al., 2019), the role of the activation functions (Ramachandran, Zoph, & Le, 2017), and

the importance of the number of layers (Bianchini & Scarselli, 2014). From the theoretical point of view, along with the aforementioned features, the characterization of the loss function to be minimized is a crucial aspect, as the whole training efficiency relies on its shape, which in turn depends on the network architecture. Several works have already dealt with the analysis of the surface of the loss function, identifying conditions for the presence (or absence) of spurious valleys in a theoretical (Venturi, Bandeira, & Bruna, 2018) or empirical-driven way (Safran & Shamir, 2018), pointing out the role of saddle points in slowing down the learning (Dauphin et al., 2014), and giving hints on the topological structure of the loss for networks with different types of activation functions (Freeman & Bruna, 2016; Nguyen & Hein, 2017).

Our contribution fits into the latter line of research, providing a characterization of the complexity of loss functions based on a topological argumentation. More precisely, given a layered neural network  $\mathcal{N}$  and a loss function  $\mathcal{L}_{\mathcal{N}}$  computed on some training data, we will measure the complexity of  $\mathcal{L}_{\mathcal{N}}$  by analyzing the topological complexity of the set  $S_{\mathcal{N}} = \{\theta | \mathcal{L}_{\mathcal{N}}(\theta) \leq c\}$ . Such an approach is natural, since  $S_{\mathcal{N}}$ , observed at each level  $c$ , provides the form of the loss function: for example, if  $\mathcal{L}_{\mathcal{N}}$  has  $k$  isolated minima, then  $S_{\mathcal{N}}$  has  $k$  disconnected regions for some small  $c$ .

\* Corresponding author.

E-mail addresses: [mariasofia.bucarelli@uniroma1.it](mailto:mariasofia.bucarelli@uniroma1.it) (M.S. Bucarelli), [dinverno@diism.unisi.it](mailto:dinverno@diism.unisi.it) (G.A. D'Inverno), [monica.bianchini@unisi.it](mailto:monica.bianchini@unisi.it) (M. Bianchini), [franco.scarselli@unisi.it](mailto:franco.scarselli@unisi.it) (F. Scarselli), [fsilvestri@diag.uniroma1.it](mailto:fsilvestri@diag.uniroma1.it) (F. Silvestri).

URL: <https://www3.diism.unisi.it/dinverno/> (G.A. D'Inverno).

<sup>1</sup> These authors contributed equally.

In the paper, we will provide a bound on the sum of Betti numbers (Bredon, 2013) of the set  $S_{\mathcal{N}}$ . In algebraic topology, Betti numbers can be exploited in order to distinguish spaces with different topological properties. More precisely, for any subset  $S \subset \mathbb{R}^n$ , there exist  $n$  Betti numbers  $b_i(S), 0 \leq i \leq n - 1$ . Intuitively, the first Betti number  $b_0(S)$  is the number of connected components of the set  $S$  that, in the case of  $S = S_{\mathcal{N}}$ , corresponds to the number of basins of attraction of the loss function. The  $i$ th Betti number  $b_i(S)$  counts the number of  $(i + 1)$ -dimensional holes in  $S$ , which provides a measure of the complexity of the error function on a given level set.

The upper bound is derived for feedforward neural networks with different numbers of layers, number of neurons per layer, and number of training samples. Moreover, we consider networks with skip connections, such as ResNet, both Binary Cross Entropy (BCE) and Mean Squared Error (MSE) loss functions, with or without regularization terms. Finally, we treat networks with Pfaffian activation functions. The class of Pfaffian maps is broad and includes most of the functions, with continuous derivatives, commonly used in Engineering and Computer Science, such as the hyperbolic tangent, the logistic sigmoid, polynomials and their compositions. Most of the elementary functions are Pfaffian, for example, the exponential and trigonometric functions. The concept of a Pfaffian function was first introduced in Khovanski (1991), where an analog of Bézout’s theorem was proved. Bézout’s classic theorem states that the number of complex solutions of a set of polynomial equations can be estimated based on their degree (to be precise, it is equal to the product of their degree). The analog of this theorem proposed in Khovanski (1991) holds for some classes of real and transcendental equations: for a wide class of real transcendental equations (including all real algebraic ones), the number of solutions of a set of  $k$  such equations in  $k$  real unknowns is finite, and can be explicitly estimated in terms of the “complexity” of the equations. In the Pfaffian setting, Gabrielov and Vorobjov introduced a suitable notion of complexity or *format* (Gabrielov & Vorobjov, 2004). We will define these concepts formally in the Preliminaries section.

It is worth mentioning that this work takes inspiration from Bianchini and Scarselli (2014), where a similar topological argumentation has been used to study the complexity of the functions realized by a neural network, showing that deep architectures can implement more complex functions than shallow ones while using the same number of parameters. While the results in Bianchini and Scarselli (2014) are about the network function, namely what a network can do, the results in this paper are about the error function, namely how hard the optimization problem is.

In this work, we attempt to provide an answer to the following questions:

1. Can a topological measure effectively assess the complexity of the loss implemented by layered neural networks?
2. How do the complexity bounds of deep and shallow neural architectures relate to the number of hidden units and the selected activation function?

#### Our contribution

This work paves the way toward a more comprehensive understanding of the landscape of loss functions in deep learning models. The sum of Betti numbers of the sublevel sets of the loss function is used to measure the complexity of the empirical risk landscape. More precisely, we derive upper bounds on the complexity of empirical risk for deep and shallow neural architectures employing the theory of Pfaffian functions. This study highlights the dependence of the error surface complexity on crucial factors such as the number of hidden units, the number of training samples, and the specific choice of the activation function.

Our main contributions are summarized below.

**Table 1**

Summary of the obtained upper bounds for  $B(S)$  as a function of different hyperparameters.  $h$  denotes the width of the layers;  $m$  is the number of samples, and  $L$  is the number of layers of the network.  $\kappa$  is a constant greater than 2. The activation function can be either the hyperbolic tangent or the sigmoid function; the loss can be either the MSE or the BCE function. When examining the impact of the number of neurons per layer,  $h$ , a qualitative difference emerges between the shallow ( $L = 2$ ) and the deep ( $L > 2$ ) networks.

Hyperparameter	Layers	Bound
$m$	any	$\kappa^{O(m^2)}$
$h$	$L = 2$	$O(h)^{O(h)}$
$h$	$L > 2$	$O(h^2)^{O(h^2)}$
$L$	any	$2^{O(L^2)} O(L^2)^{O(L)}$

- We derive the Pfaffian format of common loss functions, i.e., the Mean Square Error (MSE) loss function and the Binary Cross-Entropy (BCE) loss function, when training feedforward networks; the Pfaffian format is computed with respect to the weights and involves the knowledge of the Pfaffian format of Pfaffian activation functions.
- We provide bounds on the complexity of the loss functions, in terms of Betti numbers’s characterization. Table 1 shows the derived bound w.r.t. the number of layers, the number of neurons per layer, and the size of the training set. In particular, it is shown that the order of growth w.r.t. the number of neurons is different in networks with only one hidden layer and deeper networks. The complexity of the loss function increases super-exponentially with the number of layers or neurons per layer and exponentially with the number of training samples.
- We prove that adding a  $\ell_2$  regularization term to the loss function does not influence its topological complexity under our theoretical analysis.
- We demonstrate that incorporating skip connections (as in ResNets (He, Zhang, Ren, & Sun, 2016)) into the network does not affect the Betti numbers’ bounds.

The paper is organized as follows. In Section 2, we briefly review the literature on the characterization of the loss surface and the topological tools used to evaluate the computational power and generalization ability of feedforward networks. In Section 3, some notations and preliminary definitions are introduced, while in Section 4, the main results proposed in this paper are presented. All proofs are collected in the Appendix to ensure easy reading. Finally, conclusions and future perspectives are reported in Section 5.

## 2. Related work

The characterization of the loss landscape has gained more and more attention in the last few years; attempts to provide a satisfying description have been made through several approaches. In Choromanska, Henaff, Mathieu, Arous and LeCun (2015), Choromanska, LeCun and Arous (2015), the spin-glass theory is exploited to quantify, in probability, the presence of local minima and saddle points. Unfortunately, the connection between the loss function of neural networks and the Hamiltonian of the spherical spin-glass models relies on a number of possibly unrealistic assumptions, yet the empirical evidence suggests that it may exist also under mild conditions. In Dauphin et al. (2014), the authors propose a method that quickly avoids high-dimensional saddle points, unlike gradient descent and quasi-Newton methods. Based on results from statistical physics, random matrix theory, neural network theory, and empirical evidence, it is argued that the major difficulty in using local optimization methods originates from the proliferation of saddle points, instead of local minima, especially in high dimensional problems of practical interest. Saddle points are surrounded

by plateaus that can dramatically slow down the learning process. In Venturi, Bandeira, and Bruna (2019), the topological property of the loss function defined as *presence or absence of spurious valleys* (i.e., local minima) is addressed for one-hidden layer neural networks, providing the following contributions: (1) the Empirical Risk Minimization loss for any continuous activation function and the Expected Value loss with polynomial activations do not exhibit spurious valleys as long as the network is sufficiently over-parametrized; (2) for non-polynomial and non-negative activations, for any hidden width, there exists a data distribution that produces spurious valleys (with non-zero dimension), whose value is arbitrarily far from that of the global minimum; (3) finally, drawing on connections with random feature expansions, in general, their measure decreases as their width increases, even if spurious valleys may appear. This holds up to a low energy threshold, when approaching the global minimum at a speed inversely proportional to the size of the hidden layer.

Based on computer-driven empirical proof, similar results regarding the characterization of the loss surface in terms of the presence of spurious valleys are also reported in Safran and Shamir (2018). In Freeman and Bruna (2019), the loss surface is studied in terms of level sets, where it is shown that the landscape of the loss functions of deep networks with linear activation functions is significantly different from those exploiting half-rectified ones: in the absence of nonlinearities, the level sets are connected, while, in the half-rectified case, the topology is intrinsically different and clearly dependent on the interplay between data distribution and model architecture. Finally, a comprehensive research survey focused on analyzing the loss landscape can be found in Berner, Grohs, Kutyniok, and Petersen (2021).

In algebraic topology, Betti numbers are used to distinguish spaces with different topological properties. Betti numbers have been exploited either to give a topological description of the complexity of the function implemented by neural networks with Pfaffian activation functions or to describe their generalization capabilities. In Karpinski and Macintyre (1997), by such a technique, bounds on the VC dimension of feedforward neural networks are provided; this work has been later extended to Recurrent and Graph Neural Networks (Scarselli, Tsou, & Hagenbuchner, 2018). In Bianchini and Scarselli (2014), bounds on the sum of Betti numbers are provided in order to describe the complexity of the map implemented by feedforward networks with Pfaffian activation functions. In Naitzat, Zhitnikov, and Lim (2020), the relative efficacy of ReLUs over traditional sigmoidal activations is justified based on the different speeds with which they change the topology of a dataset – as it passes through the layers of a well-trained neural network<sup>2</sup> – representing two classes of objects in a binary classification problem. This dataset is viewed as a combination of two components: the first component represents the topological manifold of elements from the first class, and the second component encompasses the elements from the second class. A ReLU-activated neural network (neither a homeomorphism nor Pfaffian) can sharply reduce Betti numbers of the two components, but not a sigmoidal-activated one (which is a homomorphism). Reducing the Betti numbers means that the neural network simplifies the structure of the dataset by reducing the number of connected components, holes, or voids within the data manifold. This reduction suggests that the network is indeed simplifying or transforming the dataset topology, making it more amenable to analysis and classification. Finally, this research suggests that, when dealing with higher topological complexity data (meaning the data has more intricate or complex shapes and relationships), we need neural networks with greater depth (more layers) to adequately capture and understand these complexities.

<sup>2</sup> This corresponds to a network with perfect accuracy on its training set and a near-zero generalization error.

### 3. Preliminaries

In this section, we introduce the notation, basic concepts, and definitions that will be necessary for the subsequent description of the main results. In the following sections, we will exclusively focus on feedforward neural networks, defined as follows:

*Feedforward neural networks* — Let  $\theta = \{\tilde{W}^1, b^1, \dots, \tilde{W}^L, b^L\}$  be the set of the trainable network parameters, where  $\tilde{W}^l \in \mathbb{R}^{n_l \times n_{l-1}}$ ,  $b^l \in \mathbb{R}^{n_l \times 1}$ ,  $l = 1, \dots, L$  identifies each layer, and  $n_0, \dots, n_L \in \mathbb{N}$  denote the number of neurons per layer. In the following, we assume, without loss of generality, that the network has a single output, i.e.,  $n_L = 1$ . Note that the last assumption is unnecessary to demonstrate the Pfaffian nature of the loss function. However, its inclusion significantly simplifies the subsequent calculations involved in the analysis. The total number of parameters is  $\tilde{n} = \sum_{l=1}^L n_l(n_{l-1} + 1)$ .

Let  $x \in \mathbb{R}^{n_0}$  be the input to the neural network. The function implemented by the layered network is  $f_\theta(x) : \mathbb{R}^{n_0} \rightarrow \mathbb{R}$ , where  $f_\theta(x) = g(\tilde{W}^L \sigma(\tilde{W}^{L-1} \dots \sigma(\tilde{W}^1 x + b^1)) \dots + b^{L-1}) + b^L$ , where  $\sigma$  is the hidden layer activation function and  $g$  is the activation function of the output neuron.

To simplify the notation, we will aggregate the weights and bias from the same layer into a single matrix, the *augmented weight matrix*  $W^l = [b^l, \tilde{W}^l]$ ; moreover, we will denote by  $z^l$  the output of the  $l$ th layer and by  $a^l$  the neuron activations at the same layer. Thus, we have

$$z^0 = \begin{bmatrix} 1 \\ x \end{bmatrix}, a^l = W^l z^{l-1}, z^l = \begin{bmatrix} 1 \\ \sigma(a^l) \end{bmatrix}, \text{ for } 1 \leq l \leq L,$$

and  $f_\theta(x) = g(a^L)$ .

*Loss functions* — Let  $D = (x_i, y_i)_{i=1}^m$  be a set of training data, with  $x_i \in \mathbb{R}^{n_0}$  and  $y_i \in \mathbb{R}$ . Let  $\mathcal{L}$  be a generic loss to be minimized over the parameters  $\theta$ . The empirical risk of loss (or cost) function is defined as the average of the per-sample contributions, where each sample contribution is a measure of the error between the network output and the target value for that sample:

$$\mathcal{L}(\theta, D) = \frac{1}{m} \sum_{i=1}^m \text{loss}(f_\theta(x_i), y_i),$$

being  $y_i$  the target for the  $i$ -th pattern  $x_i$ . In this work we will study the topological complexity of the landscape of the Mean Square Error (MSE) and Binary Cross-Entropy (BCE) loss functions, which are defined as

$$\mathcal{L}_{\text{MSE}}(\theta, D) = \frac{1}{m} \sum_{i=1}^m (y_i - f_\theta(x_i))^2,$$

$$\mathcal{L}_{\text{BCE}}(\theta, D) = \sum_{i=1}^m -y_i \log(f_\theta(x_i)) - (1 - y_i) \log(1 - f_\theta(x_i)).$$

In the following, we will remove the dependency on the dataset  $D$  in the notation of empirical risk. We will focus solely on its reliance on the network parameters  $\theta$ . Indeed, we want to study the topological complexity of the sublevel set of the empirical risk as a function of the parameters of the network, considering the dataset samples as fixed. Unless specified otherwise, the notation  $\mathcal{L}(\theta)$  in the following represents the empirical risk corresponding to the loss function  $\mathcal{L}$  applied to a dataset containing  $m$  pairs  $(x_i, y_i)$ . In general, we consider the targets to be real numbers. Notice that since we compute the topological complexity of the empirical risk, the complexity obtained using the Pfaffian format will also depend on the number of samples in  $D$ . However, for classification problems where the Binary Cross-Entropy (BCE) loss is used, each target  $y_i$  is either 0 or 1.

Moreover, we consider a regularized form for the objective function that can be expressed as

$$\tilde{\mathcal{L}}(\theta) = \mathcal{L}(\theta) + \lambda \Omega(\theta),$$

where  $\Omega(\theta)$  is a regularization term, for instance,  $\Omega(\theta) = \frac{1}{2} \|w\|_2^2$  ( $\ell_2$  regularization norm).

**Pfaffian functions** — Pfaffian functions (Khovanskii, 1991) are analytic functions satisfying triangular systems of first order partial differential equations with polynomial coefficients. For this kind of function, an analogous of the Bézout theorem holds. The classical Bézout theorem states that the number of complex solutions of a set of  $k$  polynomial equations in  $k$  unknowns can be estimated in terms of their degrees (it equals the product of the degrees).

For a wide class of real transcendental equations (including all real algebraic ones), the number of solutions of a set of  $k$  such equations in  $k$  real unknowns is finite and can be explicitly estimated in terms of the “complexity” of the equations, which leads to the version of Bézout theorem for Pfaffian curves and Pfaffian manifolds (Khovanskii, 1991).

The class of Pfaffian functions includes a wide variety of known functions, e.g. the elementary functions, including exponential, logarithm, tangent, and their combinations (Gabrielov & Vorobjov, 2004). Interestingly, many common activation functions used in neural networks, such as the sigmoid function and hyperbolic tangent, can be classified as Pfaffian functions. Intuitively, a function is Pfaffian if its derivatives can be defined in terms of polynomials of the original function and/or a chain of other Pfaffian functions. Formally, we can state the following definition.

**Definition 3.1.** A Pfaffian chain of order  $\ell \geq 0$  and degree  $\alpha \geq 1$ , on an open domain  $U \subseteq \mathbb{R}^n$ , is a sequence of real analytic functions  $f_1, f_2, \dots, f_\ell$ , defined on  $U$ , satisfying the differential equations

$$\frac{\partial f_i}{\partial x_j}(x) = P_{ij}(x, f_1(x), \dots, f_i(x)),$$

for  $1 \leq i, j \leq \ell$  and  $x = (x_1, \dots, x_n) \in U$ . Here,  $P_{ij}(x, y_1, \dots, y_i)$  are polynomials in the  $n + i$  variables  $x_1, \dots, x_n, y_1, \dots, y_i$  of degree not exceeding  $\alpha$ .

**Definition 3.2.** Let  $(f_1, \dots, f_\ell)$  be a Pfaffian chain of length  $\ell$  and degree  $\alpha$ , and let  $U$  be its domain. A function  $f$  defined on  $U$  is called a Pfaffian function of degree  $\beta$  in the chain  $(f_1, \dots, f_\ell)$  if there exists a polynomial  $P$  in  $n + \ell$  variables, of degree at most  $\beta$ , such that  $f(x) = P(x, f_1(x), \dots, f_\ell(x))$ ,  $\forall x \in U$ . The triple  $(\alpha, \beta, \ell)$  is called the format of  $f$ .

The polynomial  $P_{ij}$  itself may explicitly depend on  $x$ . Moreover, even if  $P_{ij}$  does not have a direct dependence on  $x$ , it can still depend on  $x$  indirectly through the function  $f_1(x)$ . We say that the polynomial  $P_{ij}$  depends directly on  $x$  if there are occurrences of  $x$  that are not of the type  $f_i(x)$ . For example, consider the function  $f(x) = \arctan(x)$ , which falls into the second category of functions described above since it can be represented by the chain  $(f_1, f_2)$ , where  $f_2(x) = \arctan(x)$  and  $f_1(x) = (1 + x^2)^{-1}$ . In this case,  $\frac{\partial f_2}{\partial x} = P_1(x, f_1) = f_1(x)$ , and  $\frac{\partial f_1}{\partial x} = P_2(x, f_1) = x f_1(x)^2$ , with  $P_2$  that explicitly depend on  $x$ . On the other hand, the tanh function is a Pfaffian function falling under the first category. It can be represented by the chain containing only  $f_1(x) = \tanh(x)$ , and  $\frac{\partial f_1(x)}{\partial x} = P_1(x) = 1 - f_1(x)^2$ . In this case, there is no explicit dependence on  $x$  in the polynomial  $P_1$ . This distinction will be crucial to assess the right computations in the statement of Theorem 4.1.

Let us now introduce the notion of Pfaffian variety and semi-Pfaffian variety.

**Definition 3.3.** The set  $V \subset U$  is a Pfaffian variety if there are Pfaffian functions  $p_1, \dots, p_r$  in the chain  $(f_1, \dots, f_\ell)$  such that  $V = \{x \in U : p_1(x) = \dots = p_r(x) = 0\}$ .

**Definition 3.4.** A basic semi-variety  $S$  on the variety  $V$  is a subset of  $V$  defined by a set of sign conditions (inequalities or equalities) based on the Pfaffian functions  $p_1, \dots, p_s$  in the chain  $(f_1, \dots, f_\ell)$  such that  $S = \{x \in V : p_1(x)\varepsilon_1 0 \wedge \dots \wedge p_s(x)\varepsilon_s 0\}$ , where  $\varepsilon_1, \dots, \varepsilon_s$  are any comparison operator among  $\{<; >; \leq; \geq; =\}$ .

As outlined in the Introduction, our focus in this work is directed towards exploring the complexity of the topological space defined by the sublevel set of the empirical risk related to the loss function  $\mathcal{L}$ , namely  $S = \{\theta : \mathcal{L}(\theta) \leq c\}$ . This set represents the collection of parameter values  $\theta$  for which the empirical risk of the loss function is less than or equal to a constant  $c$ . When the empirical risk of the loss function is a Pfaffian function with respect to the parameters of the network, this set is exactly a Pfaffian semi-variety. Level curves or basins of attractions can be often described in terms of Pfaffian varieties, whose complexity can be measured through the characterization of their Betti numbers (Bianchini & Scarselli, 2014) or counting directly the number of connected components (Gabrielov & Vorobjov, 2004).

**Betti numbers** — Betti numbers are topological objects that can be used to describe the complexity of topological spaces. More formally, the  $i$ th Betti number of a space  $X$  is defined as the rank of the (finitely generated)  $i$ th singular homology group of  $X$ . Roughly speaking, it counts the number of  $i$ th dimensional holes of a space  $X^3$  and captures a topological notion of complexity that can be used to compare subspaces of  $\mathbb{R}^d$ . The reader can refer to algebraic topology textbooks for a more comprehensive introduction to homology (Bredon, 2013; Hatcher, 2002).

Informally, Betti numbers quantify the number of “holes” of various dimensions in a topological space. Each Betti number,  $b_i$ , represents the number of  $i$ -dimensional “independent” holes or cycles that cannot be continuously deformed into each other. For instance, the 0-th Betti number,  $b_0$ , counts the number of connected components in the space. The first Betti number,  $b_1$ , counts the number of independent loops or one-dimensional holes. Higher Betti numbers, such as  $b_2, b_3$ , and so on, count holes of increasing dimensionality. To give some examples, a circle has one connected component ( $b_0 = 1$ ) and one hole of dimension one ( $b_1 = 1$ ). On the contrary, a 2-dimensional unit sphere,  $\{x \in \mathbb{R}^2 : \|x\|_2 = 1\}$ , has one connected component ( $b_0 = 1$ ) and no holes ( $b_1 = 0$ ) but has one two-dimensional cavity ( $b_2 = 1$ ). A torus, like a doughnut, has one connected component ( $b_0 = 1$ ) two independent holes ( $b_1 = 2$ ), and one two-dimensional cavity ( $b_2 = 1$ ). This distinction can be understood visually: a sphere is a solid shape with no internal holes, while a torus has a hole in its center and an additional loop around the hole.

When applied to sub-level sets of the loss function  $\{\theta : \mathcal{L}(\theta) \leq c\}$ , Betti numbers offer insights into the complexity of its loss landscape  $\mathcal{L}$ , such as the presence of multiple local minima and regions of attraction. In other words, Betti numbers can be considered a tool to analyze the configuration of critical points in the optimization function landscape.

The following result connects the theory of Pfaffian functions and Betti numbers; in particular, it gives a bound on the Betti numbers for varieties defined by equations, including Pfaffian functions.

**Theorem 3.5** (Sum of the Betti Numbers for a Pfaffian Variety (Zell, 1999)). Let  $S$  be a compact semi-Pfaffian variety in  $U \subset \mathbb{R}^n$ , given on a compact Pfaffian variety  $V$ , of dimension  $n'$ , defined by  $s$  sign conditions of Pfaffian functions. If all the functions defining  $S$  have complexity at most  $(\alpha, \beta, \ell)$ , then the sum of the Betti numbers of  $S$  is

$$B(S) \in s^{n'} 2^{(\ell(\ell-1))/2} O((\tilde{n}\beta + \min(\tilde{n}, \ell)\alpha)^{\tilde{n}+\ell}). \quad (1)$$

In this paper, the theorem is applied to the Pfaffian variety  $S_N = \{\theta \in U, \text{s.t. } \mathcal{L}(\theta) \leq c\}$  as determined by the unique sign condition  $\mathcal{L}(\theta) \leq c$  — for a given threshold  $c$ . This way, we obtain a bound on the sum of the Betti numbers of  $S_N$ . Therefore, we must first demonstrate that the loss function is a Pfaffian function and compute its format, which can be achieved by writing the loss derivatives in terms of the network parameters and a Pfaffian chain. We will make use of the chain rule of Backpropagation to write the derivatives of the loss function.

<sup>3</sup> A  $i$ th dimensional hole is a  $i$ -dimensional cycle that is not a boundary of a  $(i + 1)$ -dimensional manifold.

The constraints on the compactness of  $U$  and  $V$  can be removed without affecting the bounds, as shown in Zell (2003). In our case, we consider  $U = \mathbb{R}^n$ , because the parameters have  $\mathbb{R}^n$  as their domain. Additionally, the semi-Pfaffian variety is defined by a single sign condition, so  $s = 1$ .

#### 4. Main results

In this section, we present our theoretical analysis of the topology of the loss function landscape. We measure the complexity of the loss function  $\mathcal{L}$  by bounding the sum of Betti Number of the sub-level sets  $S = \{\theta : \mathcal{L}(\theta) \leq c\}$ . We assume that the network is a feedforward with a Pfaffian activation function  $\sigma$  having format  $(\alpha_\sigma, \beta_\sigma, \ell_\sigma)$ . Our analysis is structured around the following two steps:

- We prove that the loss function is Pfaffian and determine its format, which depends on both the format of the activation function and the meta-parameters of the network (Theorems 4.2 and 4.1);
- Utilizing Theorem 3.5 we establish a bound on the Betti number of  $S$ . The formula allows to deduce how the complexity of  $\mathcal{L}$  depends on the network meta-parameters (Theorem 4.5).

We start computing the formats of the MSE and the BCE loss functions. The formats are provided showing the dependency on the format of  $\sigma$ , the number of layers, and the number of neurons per layer.

**Theorem 4.1 (MSE Loss).** Let  $\sigma : \mathbb{R} \rightarrow \mathbb{R}$  be a function for which exists a Pfaffian chain  $(\sigma_1, \dots, \sigma_{\ell_\sigma})$  and  $\ell_\sigma + 1$  polynomials  $Q$  and  $P_i$ ,  $1 \leq i \leq \ell_\sigma$  of degree  $\beta_\sigma$  and  $\alpha_\sigma$ , respectively s.t.  $\sigma$  is Pfaffian with format  $(\alpha_\sigma, \beta_\sigma, \ell_\sigma)$ .

Moreover, let  $g : \mathbb{R} \rightarrow \mathbb{R}$  be a function for which there exists a Pfaffian chain  $(g_1, \dots, g_{\ell_g})$  and  $\ell_g + 1$  polynomials,  $Q_g$  and  $P_g^i$ ,  $1 \leq i \leq \ell_g$  of degree  $\beta_g$  and  $\alpha_g$ , respectively, s.t.  $g$  is Pfaffian with format  $(\alpha_g, \beta_g, \ell_g)$ .

Let  $f(\theta, x)$  be the function implemented by a neural network with parameters  $\theta \in \mathbb{R}^{\bar{n}}$ , input  $x \in \mathbb{R}^{n_0}$ ,  $L$  layers, and an activation function  $\sigma$  for all layers except the last. The last layer can either have an activation function  $g$  or be linear. Then, the MSE Loss function is Pfaffian with format

$$\left( (\text{degree}(\sigma') + 1)(L - 2) + \text{degree}(\sigma'), 2(\beta_\sigma + 1), m\ell_\sigma \sum_{k=1}^{L-1} n_k \right),$$

when the last layer is linear. Here:

$$\text{degree}(\sigma') = \begin{cases} \beta_\sigma + \alpha_\sigma - 1 & \text{case 1} \\ \beta_\sigma + \alpha_\sigma - 1 + \alpha_\sigma(\beta_\sigma + 1) & \text{case 2} \end{cases}$$

where case 1 refers to  $P_i^\sigma(a, \sigma_1(a), \dots, \sigma_{\ell_\sigma}(a))$  not depending explicitly on  $a$ , namely occurrences of  $a$  appear that are not of the type  $\sigma_i(a)$ ; case 2 refers to the case where they depend on it. Moreover, we assume that the polynomials  $P_g^i(a, g_1(a), \dots, g_{\ell_g}(a))$  do not depend explicitly on  $a$ .

The format of the chain becomes

$$\left( (\text{degree}(\sigma') + 1)(L - 2) + \text{degree}(\sigma') + \text{degree}(g') + 1, 2\beta_g, m(\ell_\sigma \sum_{k=1}^{L-1} n_k + \ell_g) \right)$$

if the nonlinearity  $g$  is applied also to the last layer. The definition of  $\text{degree}(g')$  is analogous to that of  $\text{degree}(\sigma')$ .

For the BCE Loss function, we only explore the case where the last layer contains a non-linearity, namely  $f_\theta(x) = g(a^L)$  since BCE loss is commonly used in binary classification problems where the output is a probability of the input belonging to one of two classes. In such problems, the last layer of the model typically uses a sigmoidal activation function to ensure that the output is between 0 and 1, representing the probability of the input belonging to a certain class.

**Theorem 4.2 (BCE Loss).** Let the hypothesis of Theorem 4.1 hold. If the activation function  $g$  is also used in the last layer, the BCE Loss function is Pfaffian with format

$$\left( (L - 2)(\text{degree}(\sigma') + 1) + \text{degree}(\sigma') + \text{degree}(g') + 3, 1, m(\ell_\sigma \sum_{k=1}^{L-1} n_k + \ell_g + 4) \right).$$

If the last activation function is the sigmoid function, the BCE Loss function has the format:

$$\left( (L - 2)(\text{degree}(\sigma') + 1) + \text{degree}(\sigma') + 3, 1, m(\ell_\sigma \sum_{k=1}^{L-1} n_k + 1) + 1 \right).$$

Theorems 4.1 and 4.2 hold in general for any Pfaffian activation function and any layer widths  $(n_0, n_1, \dots, n_L)$  and their proofs can be found in Appendix A.2. Intuitively, proofs are carried out by means of chain rule computation, as the derivatives of the loss function  $\mathcal{L}$  with respect to layer  $l$  depend on the derivatives of  $\mathcal{L}$  computed with respect to layer  $l + 1$ . This computation can be recursively extended up to layer 1. Thus, to characterize the Pfaffian chain of  $\mathcal{L}$  and the derivatives of the functions in the Pfaffian chain, we need to include in the chain all the derivatives of the loss function with respect to each layer computed on each input sample. In addition, we have to take into account the length of the Pfaffian chain of the considered activation function.

In the following, we present the outcomes for the particular scenario in which the activation function of the intermediate layers is either a sigmoid or a hyperbolic tangent. Additionally, for the sake of simplicity, we assume that all hidden layers share the same width, denoted as  $h$ .

**Corollary 4.3.** Consider a feedforward perceptron network where all hidden layers have the same width  $h$ . The activation function of intermediate layers can be either the hyperbolic tangent ( $\tanh$ ) or the sigmoid function ( $\text{logsig}$ ), and the loss function is the Mean Squared Error. In this setting, the following holds:

- when the last layer of the network is linear, the Pfaffian format of the loss function is given by

$$(3(L - 2), 4, m(L - 1)h);$$

- when the last layer is passed through a sigmoid activation function, the Pfaffian format of the loss function is:

$$(3(L - 2) + 5, 2, m(h(L - 1) + 1))$$

We can state an analogous result for the BCE loss function.

**Corollary 4.4.** For a feedforward perceptron network where all the hidden layers have the same width  $h$ , trained using BCE loss function and a non-linear last layer, the following holds:

- if the non-linearity used is the sigmoid function, the Pfaffian format of the loss function is

$$(3(L - 2) + 5, 1, m((L - 1)h + 1) + 1)$$

- if the non-linearity used is  $\tanh(x)$ , the Pfaffian format of the loss function is

$$(3(L - 2) + 7, 1, m((L - 1)h + 5)).$$

We now state the main results of our work.

The presented theorem investigates the dependence of the sum of Betti numbers associated with the semi-Pfaffian variety  $S_{\mathcal{N}}$ , which represents the parameter set where the loss is non-negative, on factors such as the number of samples, network width, and network depth. Using Corollaries 4.3 and 4.4, we can derive the bounds on the Betti numbers of the Pfaffian semi-variety for both MSE and BCE loss functions.

**Theorem 4.5.** Let us consider a deep feedforward perceptron network  $\mathcal{N}$  with  $L \geq 3$  layers all having width  $h$ . The activation function can be either the hyperbolic tangent or the sigmoid function, the loss is either the MSE or the BCE function, and the last layer is either linear (only for MSE) or nonlinear. Moreover, let us denote by  $S$  the semi-Pfaffian variety given by the set of parameters where the loss function is non-negative, i.e.  $S = S_{\mathcal{N}} = \{\theta | \mathcal{L}(\theta) \leq c\}$  for a threshold  $c \in \mathbb{R}^+$ . Then, the sum of the Betti numbers  $B(S)$  is bounded as follows.

- With respect to the number of samples  $m$ , fixing  $h$  and  $L$  as constants, we have that  $B(S) \in \kappa_1^{O(m^2)}$ , where  $\kappa_1$  is a constant greater than 2.
- With respect to  $h$ , we have  $B(S) \in O(h^2)^{O(h^2)}$ .
- Finally, with respect to  $L$ ,  $B(S) \in 2^{O(L^2)}O(L^2)^{O(L)}$ .

On the other hand, in the case of a shallow network with one hidden layer, i.e.,  $L = 2$ , the following results hold.

- With respect to  $m$ , the bound is the same we obtained for deep networks,  $B(S) \in \kappa_2^{O(m^2)}$ , where  $\kappa_2$  is a constant greater than 2.
- With respect to  $h$ , we have  $B(S) \in O(h)^{O(h)}$ .

The proof with detailed computations can be found in [Appendix A.5](#).

It is worth emphasizing that the bounds concerning  $h$  and  $L$  offer an insight into the relationship between the topological complexity of the loss landscape and the total number of parameters  $\tilde{n}$ . In the first case, we explore the effect of changing the network width by fixing the depth. In the second case, we investigate the impact of altering network depth  $L$  by performing the converse.

A first major remark from [Theorem 4.5](#) is that the upper bound on the Betti numbers associated to the loss function is only exponential in the number of samples  $m$ , while it is superexponential in the number of neurons  $h$  or in the number of layers  $L$ . Intuitively, the take-home message is that the topological complexity of the loss function is less conditioned by the number of samples than by the number of parameters.

Unsurprisingly, [Theorem 4.5](#) also suggests that the complexity of the loss landscape with respect to deep networks increases with the number of neurons  $h$  at a much faster pace than the one with respect to the shallow networks, going from a dependence of the type  $O(h^2)^{O(h^2)}$  to a dependence of the type  $O(h)^{O(h)}$ . Such a difference in behavior is coherent with results present in literature ([Li, Xu, Taylor, Studer, & Goldstein, 2018](#)), where it is proven that for sufficiently deep networks the loss landscapes quickly move from being nearly convex to being highly chaotic.

#### 4.1. Regularization terms and residual connections

##### 4.1.1. The role of regularization

One could be interested in seeing how the introduction of regularization terms influences our analysis. In particular, we consider the scenario in which an  $\ell_2$  regularization term,  $\Omega(\theta) = \sum_i \|\tilde{W}_i\|^2$ , polynomial in the parameters  $\theta$ , is added to the loss function. This term only affects the term  $\beta$  of the format of the Pfaffian function  $\tilde{\mathcal{L}}(\theta) = \mathcal{L}(\theta) + \lambda\Omega(\theta)$ , as it affects the degree of the polynomial with respect to the weights, by adding a monomial term of degree 2; nevertheless, the bound on the Betti numbers is not affected by it. This can be easily derived using the computations carried out in the Appendix for the proof of [Theorem 4.5](#). The  $\ell_2$  regularization does not promote sparsity (which is instead promoted by the  $\ell_1$  regularization) ([Hastie, Tibshirani, Friedman, & Friedman, 2009](#)), but it affects the magnitude of the weights. This could suggest that the regularization term may provide a scaling of the loss function, and therefore, all local minima may still be present; however, it is possible that our theoretical bounds may not be able to catch the influence of the regularization term in the loss landscape.

##### 4.1.2. Residual connections

Skip connections or residual connections, are widely employed in neural networks to alleviate the vanishing gradient problem and enhance information flow across layers. The ResNet model popularized this architectural design ([He et al., 2016](#)) and has since then been adopted in various network architectures.

Introducing a residual term at each layer in a neural network, thus creating a Residual Neural Network (ResNet), does not impact the bounds provided in our analysis. It does not affect either the number of functions required in the chain or the maximum degree

of the polynomials. The addition of skip connections combines the output of a previous layer with that of a subsequent layer through summation. Regarding our analysis, the essential factors, such as the degree of polynomials and the length of the Pfaffian chain, remain the same. The only change lies in the specific terms to be included within the chain (see [Appendix A.6](#) for more details). Consequently, it implies that utilizing a ResNet rather than a conventional feedforward neural network does not alter the topology of the loss function or its optimization process. Instead, the primary advantage lies in enhancing the network's expressive capacity.

Skip connections allow gradients to flow more easily during back-propagation, facilitating the training of deeper networks. It has been observed that skip connections promote flat minimizers and prevent the transition to chaotic behavior ([Li et al., 2018](#)). Our current theoretical framework is limited in capturing the reduced complexity of the loss landscape induced by skip connections. Specifically, our theory provides an upper bound on the number of minima, whilst lacking a lower bound, which implies that our estimate may exceed the actual value. Moreover, the analysis in [Li et al. \(2018\)](#) focuses solely on the minima, while our proposed bound encompasses the sum of all non-zero Betti numbers. Finally, it is worth observing that the optimization algorithms do not explore the whole space but only a part where the complexity of the function might be lower. Therefore, regularization and skip connections could be mechanisms for which only submanifolds are explored by the optimization algorithm, and such behavior may not be captured by what the global bound suggests.

## 5. Conclusions

In our investigation, we determined that when employing a Pfaffian non-linearity, both the MSE and BCE loss functions can be represented as Pfaffian functions. Subsequently, we analyzed the respective Pfaffian chains obtained in each case. Specifically, we examined the differences in the complexity and performance of the Pfaffian chains resulting from the use of the two loss functions.

When studying the complexity of the loss landscape, a superexponential dependency on the network parameters has been found; interestingly, a qualitative difference can be highlighted between the shallow and the deep case, as we focus on the impact of the number of neurons  $h$ . Indeed, the superexponential dependency in  $h$  encountered for  $L = 2$  changes in  $h^2$  as the network becomes deeper.

This result is aligned with the general intuition and previous works in literature ([Bianchini & Scarselli, 2014](#)). In any case, the asymptotic analysis shows that the sum of Betti numbers has an exponential dependency on the square of the number of samples  $m$ .

It is worth underlying the characterization of the topological complexity we derived for loss functions with an additional  $\ell_2$  term; the presence of a regularization term does not have an impact in the bounds derived with our analysis, pointing to a different role of the regularization itself in the training of the network, e.g., the optimization process. Nevertheless, there remains space for deeper analysis since these boundaries are not tight.

Bounds provided by the sum of Betti numbers are not tight; our analysis suggests a qualitative interpretation more than a quantitative one. Obtaining a bound on the number of connected components  $B_0(S)$  rather than  $B(S)$  would give a more accurate characterization of the topology of the loss landscape; this is a perspective to be considered for future works.

To the best of our knowledge, we are not aware of methods that may allow to effectively count the local minima in an error function. Recognizing local minima and distinguishing them from saddle points and strange behaviors of learning algorithms is difficult. Hence, it is difficult to measure the impact of local minima more generally. Such a fact is often mentioned in the literature (see e.g., [Dauphin et al. \(2014\)](#)), where the only available studies involve very simplified architectures and/or algorithms (e.g., [Choromanska, Henaff et al. \(2015\)](#), [Venturi](#)

et al. (2019)). For this reason, a future matter of research could be the design of an effective experimental framework to measure the number of local minima in the loss surface.

Our analysis does not cover loss landscapes for networks with piecewise polynomial functions, e.g. ReLU networks, as these functions are not Pfaffian. However, we are considering expanding our study to include this function class in the future. One potential approach to reach such a goal might involve methodologies similar to those exploited in bounding the VC-dimension for piecewise polynomial functions (see Bartlett, Harvey, Liaw, and Mehrabian (2019), Bartlett, Maierov, and Meir (1998)). In those works, the bounds on the VC dimension are obtained by studying the number of connected components defined by the function realized by the neural network w.r.t. the weights.

Moreover, it would be interesting to connect our results with the *mode connectivity* framework, specifically through the lens of Morse theory (Akhtiamov & Thomson, 2023), which provides a characterization of the set of configurations of a neural network with respect to a fixed empirical loss value. Indeed, improving our analysis with Morse theory could help to define the connectivity maps at many dimension levels, leading to a better quantification of each single Betti number.

Finally, it would be worth investigating the connection between the *stability* of residual connections in neural networks and the characterization of the loss landscape, e.g., the effects of BIBO-stability and zero-stability on the bound of Betti numbers (Chen, Jin, & Shang, 2022).

#### CRedit authorship contribution statement

**Maria Sofia Bucarelli:** Writing – review & editing, Writing – original draft, Methodology, Investigation, Formal analysis, Conceptualization. **Giuseppe Alessio D’Inverno:** Writing – review & editing, Writing – original draft, Methodology, Investigation, Formal analysis, Conceptualization. **Monica Bianchini:** Writing – review & editing, Supervision. **Franco Scarselli:** Writing – review & editing, Supervision, Investigation, Conceptualization. **Fabrizio Silvestri:** Writing – review & editing, Supervision.

#### Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Giuseppe Alessio D’Inverno reports financial support was provided by Francesco Severi National Institute of Higher Mathematics National Group of Scientific Calculations. Monica Bianchini reports financial support was provided by Ministry of Education and Merit. Maria Sofia Bucarelli reports financial support was provided by Ministry of Education and Merit. Fabrizio Silvestri reports financial support was provided by Ministry of Education and Merit. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

#### Data availability

No data was used for the research described in the article.

#### Acknowledgments

Giuseppe Alessio D’Inverno is member of the Gruppo Nazionale Calcolo Scientifico - Istituto Nazionale di Alta Matematica (GNCS-INDAM). Monica Bianchini is partially supported by the PNRR Project, Italy “THE - Tuscany Health Ecosystem”, CUP: B83C22003920001. Maria Sofia Bucarelli and Fabrizio Silvestri were partially supported by projects “FAIR”, Italy (CUP: B53C22003980006) and “SERICS”, Italy (CUP: B53C22003950001) under the MUR National Recovery

and Resilience Plan funded by the European Union - NextGenerationEU. Supported also by EC H2020RIA project “SoBigData++”, Italy (871042) funded by the European Union. Fabrizio Silvestri was also supported by the project NEREO (Neural Reasoning over Open Data) project funded by the Italian Ministry of Education and Research (PRIN), Italy Grant no. 2022A6FHA.

The authors would like to extend their gratitude to Ben Cullen for carefully reading this manuscript and providing a valuable revision.

## Appendix

### A.1. Making derivatives explicit using Backpropagation

Let  $\mathcal{L}(\theta), D = \frac{1}{m} \sum_{i=1}^m \text{loss}(f_\theta(x_i), y_i)$  be a generic loss. We aim to determine the gradient for a given input–output pair  $(x_i, y_i)$ , with respect to the weight variables  $w_{jk}^l$  (connecting the  $j$ th neuron of layer  $l - 1$ , with the  $k$ th neuron of layer  $l$ ), which are elements of the augmented weight matrix  $W^l$ . The gradient components  $\frac{\partial \mathcal{L}}{\partial w_{jk}^l}$  can be calculated through the chain rule. The Backpropagation algorithm provides an efficient method of spreading the error contribution back through the layers for updating weights.

Let us define  $\delta_k^l = \frac{\partial \mathcal{L}}{\partial a_k^l}$ , for  $l = 1, \dots, L$ , as the derivative of the cost function with respect to the activation  $a_k^l$  of the  $k$ th neuron of layer  $l$ . Then

$$\frac{\partial \mathcal{L}}{\partial w_{i,j}^l} = \delta_j^l z_i^{l-1},$$

which represents a polynomial function in  $\delta_j^l, z_i^{l-1}$ , so that all  $\delta_j^l, z_i^{l-1}$  and their derivatives belong to the Pfaffian chain describing  $\mathcal{L}$ . Moreover, by the chain rule, we have that

$$\delta_j^l = \frac{\partial \mathcal{L}}{\partial a_j^l} = \sum_{k=0}^{n_{l+1}} \frac{\partial \mathcal{L}}{\partial a_k^{l+1}} \frac{\partial a_k^{l+1}}{\partial a_j^l} = \sum_{k=0}^{n_{l+1}} \delta_k^{l+1} \frac{\partial a_k^{l+1}}{\partial a_j^l} = \sum_{k=0}^{n_{l+1}} \delta_k^{l+1} w_{k,j}^{l+1} \sigma'(a_j^l),$$

which means that  $\delta_j^l$  is polynomial in all  $n_{l+1}, \delta_i^{l+1}$  and  $\sigma'(a_j^l)$ , so that we have also to include all  $\delta_j^{l+1}$  and all  $\sigma'(a_j^l)$  and their derivatives in the Pfaffian chain.

Summing up, fixing an input  $x_i, y_i$  and proceeding backward through the layers, we can derive that

$$\frac{\partial \mathcal{L}}{\partial w_{i,j}^l} = \text{poly}(\delta_1^L, \dots, \delta_{n_L}^L, \sigma'(a_1^L), \dots, \sigma'(a_{n_L}^L), \dots, \sigma'(a_1^{l+1}), \dots, \sigma'(a_{n_{l+1}}^{l+1}), \sigma'(a_j^l), \sigma'(a_i^{l-1})). \quad (\text{A.1})$$

**Remark A.1.** Notice that if  $\sigma$  is Pfaffian. It follows that the derivative  $\sigma'$  is polynomial in the factor of the chain, and the degree of the polynomial is at most  $\alpha_\sigma$ , while the degree of  $\sigma$  in its chain is  $\beta_\sigma$ .

Consequently, being  $\frac{\partial \mathcal{L}}{\partial w_{i,j}^l} = \delta_j^l z_i^{l-1}$ , we have:

$$\frac{\partial \mathcal{L}}{\partial w_{i,j}^l} = \text{poly}(\delta_1^L, \dots, \delta_{n_L}^L, \sigma_1(a_1^L), \dots, \sigma_{\ell_\sigma}(a_1^L), \dots, \sigma_1(a_{n_L}^L), \dots, \sigma_{\ell_\sigma}(a_{n_L}^L) \dots, \sigma_1(a_1^{l+1}), \dots, \sigma_{\ell_\sigma}(a_1^{l+1}), \dots, \sigma_1(a_{n_{l+1}}^{l+1}), \dots, \sigma_{\ell_\sigma}(a_{n_{l+1}}^{l+1}), \dots, \sigma_1(a_j^l), \dots, \sigma_{\ell_\sigma}(a_j^l), \sigma_1(a_i^{l-1}), \dots, \sigma_{\ell_\sigma}(a_i^{l-1})). \quad (\text{A.2})$$

$$\dots, \sigma_1(a_1^{l+1}), \dots, \sigma_{\ell_\sigma}(a_1^{l+1}), \dots, \sigma_1(a_{n_{l+1}}^{l+1}), \dots, \sigma_{\ell_\sigma}(a_{n_{l+1}}^{l+1}), \dots, \sigma_1(a_j^l), \dots, \sigma_{\ell_\sigma}(a_j^l), \sigma_1(a_i^{l-1}), \dots, \sigma_{\ell_\sigma}(a_i^{l-1}). \quad (\text{A.3})$$

$$\dots, \sigma_1(a_j^l), \dots, \sigma_{\ell_\sigma}(a_j^l), \sigma_1(a_i^{l-1}), \dots, \sigma_{\ell_\sigma}(a_i^{l-1}). \quad (\text{A.4})$$

### A.2. Proof of Theorems 4.1 and 4.2

#### A.2.1. Preliminaries

We want to prove that the MSE loss function and the BCE loss functions are Pfaffian functions with respect to the parameters of the network, in the hypothesis that the non-linearities  $\sigma$  and  $g$  are Pfaffian. To do so, we need to find a Pfaffian chain so that the loss function can be written as a polynomial in that chain, and we need to compute the

degree of this polynomial in the parameters and the maximum degree of the derivatives of the functions in the chain with respect to the parameters of the network.

Notice that in this particular case of the MSE,

$$loss_{MSE}(f(x_i), y_i) = \frac{1}{2}(f(\theta, x_i) - y_i)^2.$$

meaning that if  $f$  is a Pfaffian function of a given format  $(\alpha_f, \beta_f, \ell_f)$ , the loss is a Pfaffian function with respect to the same chain with format  $(\alpha_f, 2\beta_f, \ell_f)$ .

In the case of the BCE loss function,

$$loss_{BCE}(f(x_i), y_i) = -y_i \log(f_\theta(x_i)) - (1 - y_i) \log(1 - f_\theta(x_i)).$$

always assuming that  $f$  is a Pfaffian function of a given format  $(\text{degree}(f')-1, \beta_f, \ell_f)$  we can consider two possible chains. The first one is the chain in which we add to the chain of  $f$  the functions  $\log(f_\theta(x_i))$  and  $\log(1 - f_\theta(x_i))$  and their derivatives meaning that the format of the chain becomes  $(\max\{\text{degree}(f') + 2, \alpha_f\}, 1, \ell_f + 4)$ , where  $\text{degree}(f')$  is the degree of  $f'$  as polynomial in the chain, we will specify which chain in the various cases. In case we have a sigmoid as the final activation function, we could consider a different chain in which we include the loss in the chain; in this case, as we will see in ?? to obtain a Pfaffian chain, we also need to include the sigmoid function applied to  $f$ ,  $\sigma(f(x))$  so the format becomes  $(\text{degree}(f') + 2, 1, \ell_f + 4)$ .

To determine the degree of  $f'$  we will need to compute the degree of  $\sigma'$ . This will be useful for all the different cases considered, so we are doing it in this section.

If  $y_i = \sigma_i(a)$ :

$$\frac{d\sigma(a)}{da} \Big|_{a=a'_h} = \frac{dQ(y_1, \dots, y_\ell)}{da} \Big|_{a=a'_h} = \left( \sum_{s=1}^{\ell} \frac{\partial Q(y_1, \dots, y_\ell)}{\partial y_s} P_u(a, y_1, \dots, y_u) \right) \Big|_{\substack{1 \leq u \leq \ell \\ y_u = \sigma_u(a'_h) \\ a = a'_h}}$$

$\frac{\partial Q(y_1, \dots, y_\ell)}{\partial y_s}$  has degree  $\beta_\sigma - 1$  and  $P_u(a, y_1, \dots, y_u)$  has degree  $\alpha_\sigma$  in  $\sigma_1(a), \dots, \sigma_\ell(a)$ . In conclusion  $\frac{d\sigma(a)}{da} \Big|_{a=a'_h}$  is a polynomial of degree  $\beta_\sigma + \alpha_\sigma - 1$  if,  $\forall i, P_i$  does not depend directly on  $a$ , and it is  $\beta_\sigma + \alpha_\sigma - 1 + \alpha_\sigma(\beta_\sigma + 1)$  in the general case. Indeed if  $P_i$  depends on  $a$ , we have that  $P(a'_h, \sigma_1(a'_h), \dots, \sigma_h(a'_h))$  has degree  $\alpha_\sigma(\beta_\sigma + 1)$ , since the degree of  $a'_h$  is  $\beta_\sigma + 1$  in the Pfaffian chain. Notice that  $a'_h = W^t \sigma(a^{L-1})$  and  $\sigma(a^{L-1})$  has degree  $\beta_\sigma$ .

Summarizing :

$$\text{degree}(\sigma') = \begin{cases} \beta_\sigma + \alpha_\sigma - 1 & \text{case 1} \\ \beta_\sigma + \alpha_\sigma - 1 + \alpha_\sigma(\beta_\sigma + 1) & \text{case 2} \end{cases} \quad (\text{A.5})$$

where case 1 refers to the case where  $P_i^j(a, \sigma_1(a), \dots, \sigma_{\ell_\sigma}(a))$  do not depend explicitly  $a$  and case 2 where they depend on it.

### A.2.2. MSE loss function

**Proof of Theorem 4.1.** In our hypothesis  $n_L = 1$ , so  $\delta^L, a^L$  and  $f_\theta(x)$  are scalars. Depending on whether the last layer is linear or the non-linearity  $g$  is applied, we have that

$$f_\theta(x) = \begin{cases} a^L \\ g(a^L). \end{cases}$$

**Linear last layer.** In the case of linear activation, given that  $a^L = W^L \sigma(a^{L-1})$  and that  $\sigma$  is Pfaffian with respect to the chain  $\sigma_1, \dots, \sigma_{\ell_\sigma}$  we obtain that  $f_\theta(x) = a^L$  is polynomial in the functions following chain

$$(((\sigma_k(a'_i))_{k=1, \dots, \ell_\sigma})_{i=1, \dots, n_j})_{j=1, \dots, L-1}$$

The degree of the Pfaffian function  $f$  in this chain is  $\beta_f = \beta_\sigma + 1$ ; the maximum degree of the derivatives depends on the degree of  $\sigma'$  in (A.5) and is given by the chain rule, the worst case is given by

deriving of the term of the vector  $\sigma(a^{L-1})$  with respect to one of the weights of the first layer. In this case, applying the chain rule and going backward layer by layer, we obtain that every step, we multiply the weight of one layer and the  $\sigma'$  it follows that the degree of  $\frac{\partial \sigma(a^{L-1})}{\partial w_{i,j}^1}$  is  $(\text{degree}(\sigma') + 1)(L - 2) + \text{degree}(\sigma')$

Taking into account what was said in Appendix A.2.1, we obtain that for a single input point,  $x$ , we obtained that the MSE loss with linear activation for the last layer is Pfaffian with respect to the following chain of length  $\ell_\sigma \sum_{k=1}^{L-1} n_k$ :

$$(((\sigma_k(a'_i))_{k=1, \dots, \ell_\sigma})_{i=1, \dots, n_j})_{j=1, \dots, L-1} \quad (\text{A.6})$$

The order of the chain is given, going from the inner cycle to the outer cycle.

Considering that we have to consider all the input points, the length of the chain becomes  $m \ell_\sigma \sum_{k=1}^{L-1} n_k$ . The format of the chain of the MSE loss function is therefore

$$((\text{degree}(\sigma') + 1)(L - 2) + \text{degree}(\sigma'), 2(\beta_\sigma + 1), m \ell_\sigma \sum_{k=1}^{L-1} n_k) \quad (\text{A.7})$$

**Non-linear last layer with non-linearity  $g$ .** In this case, we need to add to the chain described in Eq. (A.6) the terms  $(g_k(a^L))_{k=1, \dots, \ell_g}$ . The final chain will be:

$$[(((\sigma_k(a'_i))_{k=1, \dots, \ell_\sigma})_{i=1, \dots, n_j})_{j=1, \dots, L-1}, (g_k(a^L))_{k=1, \dots, \ell_g}] \quad (\text{A.8})$$

The degree of the function  $f$  in this chain is given by  $\beta_g$ , the maximum degree of the derivatives is the degree of  $\frac{\partial g(a^L)}{\partial w_{i,j}^1}$  and is  $(\text{degree}(\sigma') + 1)(L - 2) + \text{degree}(\sigma') + \text{degree}(g') + 1$ .

Using the argument we used before for  $\sigma'$ , we have that the degree of  $g'(a)$  is  $\beta_g + \alpha_g - 1$  if,  $\forall i, P_i^j$  does not depend directly on  $a$ , and it is  $\beta_g + \alpha_g - 1 + \alpha_g(\beta_g + 1)$  in the general case.

$$\text{degree}(g') = \begin{cases} \beta_g + \alpha_g - 1 & \text{case 1} \\ \beta_g + \alpha_g - 1 + \alpha_g(\beta_g + 1) & \text{case 2} \end{cases} \quad (\text{A.9})$$

Summarizing the format of the chain in the case of non-linear activation for the last layer is

$$((\text{degree}(\sigma') + 1)(L - 2) + \text{degree}(\sigma') + \text{degree}(g') + 1, 2\beta_g, m(\ell_\sigma \sum_{k=1}^{L-1} n_k + \ell_g)) \quad \square \quad (\text{A.10})$$

### A.2.3. BCE loss function

**Proof of Theorem 4.2.** We study the two cases when the intermediate activation  $g$  is the sigmoid and when it is a different function.

**Non-linear activation  $g$  different from the sigmoid function.** In this case, the Pfaffian chain for the loss function will be the chain described in (A.8) to which we add the following terms  $\frac{1}{g(a^L)}, \log(g(a^L)), \frac{1}{1-g(a^L)}, \log(1 - g(a^L))$ . The length of the chain will be  $\sum_{k=1}^{L-1} n_k + \ell_g + 4$ . The degree of the loss function with respect to this chain is 1, and the maximum degree of the derivatives is given by the degree of the following derivative  $\frac{\partial 1/g(a^L)}{\partial w_{i,j}^1}$  that is  $(L - 2)(\text{degree}(\sigma') + 1) + \text{degree}(\sigma') + \text{degree}(g') + 3$ .

It follows that the format of the chain for the BC loss function with sigmoid activation for the last layer is

$$((L - 2)(\text{degree}(\sigma') + 1) + \text{degree}(\sigma') + \text{degree}(g') + 3, 1, m(\ell_\sigma \sum_{k=1}^{L-1} n_k + \ell_g + 4))$$

**Non-linear activation  $g$  is the sigmoid function.** In this case, we want to include the loss in the chain and use the trick of backpropagation introduced in Appendix A.2.1 to be sure that its derivatives are polynomial in the chain. Eq. (A.1) shows that the derivative of the loss with respect to the parameters is poly in  $\delta_i^L$  and in the terms of the chain (A.8), in this case with  $g$  equal to the sigmoid function.



If we consider only a single sample  $x$  and the output of the network  $f_\theta(x) = g(a^L)$ . We have that

$$\frac{\partial \text{loss}_{\text{BCE}}(y, f_\theta(x))}{\partial a^L} = g'(a^L) \frac{1}{g(a^L)(1-g(a^L))} (y_i - f_\theta(x)) = \sum_{i=1}^m \frac{g'(a^L)}{g(a^L)(1-g(a^L))} (y_i - g(a^L)). \quad (\text{A.11})$$

If the non-linearity  $g$  is the sigmoid function ( $g(x) = \frac{1}{1+e^{-x}}$ ), the term  $\frac{g'(a^L)}{g(a^L)(1-g(a^L))}$  becomes 1, and we do not need to deal with it and the degree of  $\delta^L$  is the degree of  $g(a^L)$ , that is  $\beta_g$  that for the sigmoid function is 1. We recall that the sigmoid function is Pfaffian with format  $(2, 1, 1)$ .

It follows that the format of the chain for the BC loss function with sigmoid activation for the last layer is

$$((L-2)(\text{degree}(\sigma') + 1) + \text{degree}(\sigma') + \text{degree}(g') + 1, 1, m(\ell_\sigma \sum_{k=1}^{L-1} n_k + 1) + 1)$$

The last +1 in the length is given by the fact that we are also adding the loss function computed on the input dataset to the chain. Moreover, we can compute  $\text{degree}(g')$  that in this case is 2, notice that this is smaller than the worst case proposed in (A.9) that would be equal to 4.

The final format of the chain will be

$$((L-2)(\text{degree}(\sigma') + 1) + \text{degree}(\sigma') + 3, 1, m(\ell_\sigma \sum_{k=1}^{L-1} n_k + 1) + 1) \quad \square$$

### A.3. Proof of Corollary 4.3

**Proof.** It is enough to remark that the format of the hyperbolic tangent and the sigmoid function is  $(\alpha_\sigma, \beta_\sigma, \ell_\sigma) = (2, 1, 1)$ . Substituting these values in Theorem 4.1 leads straightforwardly to the statement.  $\square$

### A.4. Proof of Corollary 4.4

**Proof.** The result for the hyperbolic tangent activation function can be obtained by applying Theorem 4.2 with its corresponding format of  $(2, 1, 1)$ . On the other hand, for the sigmoid function, we have that  $\sigma'(x) = \sigma(x)(1 - \sigma(x))$ . This allows us to obtain a linear dependency on  $\sigma(x)$  in the derivative of the BCE loss function. Indeed,

$$\begin{aligned} \frac{\partial \mathcal{L}(f)}{\partial \theta} &= -\frac{\partial}{\partial \theta} (y \log(\sigma(f(\theta))) + (1-y) \log(1 - \sigma(f(\theta)))) \\ &= -\left[ \frac{y}{\sigma(f(\theta))} \sigma'(f(\theta)) \frac{\partial f(\theta)}{\partial \theta} - \frac{1-y}{1-\sigma(f(\theta))} \sigma'(f(\theta)) \frac{\partial f(\theta)}{\partial \theta} \right] \\ &= -y(1 - \sigma(f(\theta))) \frac{\partial f(\theta)}{\partial \theta} + (1-y)\sigma(f(\theta)) \frac{\partial f(\theta)}{\partial \theta} \\ &= (\sigma(f(\theta)) - y) \frac{\partial f(\theta)}{\partial \theta} \quad \square \end{aligned}$$

### A.5. Proof of Theorem 4.5

**Proof.** We can use Theorem 3.5 with  $U = \mathbb{R}^{\tilde{n}}$ , with  $\tilde{n} = h(n_0 + 1) + h(h+1)(L-2) + h + 1 = h^2(L-2) + h(n_0 + L) + 1$  the total number of parameters of the network.

In this case the term  $s^{m'}$  in Eq. (1) can be ignored since  $s = 1$ .

**Bounds for MSE loss function: deep case.** For  $L \geq 3$ :

- if the last layer has a linear activation, we have that

$$B(S) \in 2^{[m(L-1)h(m(L-1)h-1)]/2} \mathcal{O}(f(n_0, h, L, m)^{h^2(L-2)+h(L+n_0+m(L-1))+1}) \quad (\text{A.12})$$

with

$$f(n_0, h, L, m) = 4[h^2(L-2) + h(L+n_0) + 1] + 3(L-2) \cdot \min(h^2(L-2) + h(L+n_0) + 1, m(L-1)h)$$

If  $m \gg h$  it becomes

$$B(S) \in 2^{(m(L-1)h(m(L-1)h-1))/2} \mathcal{O}(4[h^2(L-2) + h(L+n_0) + 1] + 3(L-2)(h^2(L-2) + h(L+n_0) + 1))^{h^2(L-2)+h(L+n_0+m(L-1))+1}$$

If  $h \gg m$ , and we consider  $L$  and  $m$  as constant it becomes

$$B(S) \in 2^{(m(L-1)h(m(L-1)h))/2} \mathcal{O}(4[h^2(L-2) + h(L+n_0) + 1]3(L-2)hm(L-1))^{h^2(L-2)+h(L+n_0+m(L-1))+1}$$

It is important to note that the overparametrized regime falls within this scenario.

- if the last layer has a nonlinear activation, we have that:

$$B(S) \in 2^{[(m(h(L-1)+1))(m(h(L-1)+1)+1)]/2} \cdot \mathcal{O}(g(n_0, h, L, m))^{h^2(L-2)+h(L+n_0+m(L-1))+2} \quad (\text{A.13})$$

with

$$g(n_0, h, L, m) = 2[h^2(L-2) + h(L+n_0) + 1] + (3(L-2) + 5) \cdot \min(h^2(L-2) + h(L+n_0) + 1, m(h(L-1) + 1))$$

If  $m \gg h$  it becomes

$$B(S) \in 2^{[(m(h(L-1)+1))(m(h(L-1)+1)+1)]/2} \mathcal{O}(2[h^2(L-2) + h(L+n_0) + 1] + (3(L-2) + 5)(h^2(L-2) + h(L+n_0) + 1))^{h^2(L-2)+h(L+n_0+m(L-1))+2}$$

If  $h \gg m$ , and we consider  $L$  and  $m$  as constant it becomes

$$B(S) \in 2^{[(m(h(L-1)+1))(m(h(L-1)+1)+1)]/2} \mathcal{O}(2[h^2(L-2) + h(L+n_0) + 1] + (3(L-2) + 5)m(h(L-1) + 1))^{h^2(L-2)+h(L+n_0+m(L-1))+2}$$

In both cases, we can see that, as a function of the number of samples  $m$ , fixing  $h$  and  $L$  as constants, we have that  $B(S) \in c^{O(m^2)}$ , where  $c$  is a constant greater than 2. As a function of  $h$ , considering the other variables as constants  $B(S) \in O(h^2)^{O(h^2)}$ . Eventually, as a function of  $L$ , considering  $m$  and  $h$  as constants,  $B(S) \in 2^{O(L^2)} O(L^2)^{O(L)}$ .

**Bounds for MSE loss function: shallow case.**

In the case of  $L = 2$ , all the terms in which  $L - 2$  occurs vanish. Therefore,

- if the last layer has a linear activation, Eq. (A.12) simplifies in the following way:

$$B(S) \in 2^{[mh(mh-1)]/2} \mathcal{O}([4h(2+n_0) + 4]^{h(2+n_0+m)+1})$$

- if the last layer has a nonlinear activation since the number of samples is usually larger than the input dimension,  $\min(h(2+n_0) + 1, 2mh) = h(2+n_0) + 1$ . This simplifies Eq. (A.13) in the following way:

$$B(S) \in 2^{[2mh(2mh-1)]/2} \mathcal{O}([9h(2+n_0) + 9]^{h(2+n_0+2m)+1})$$

As a function of  $m$  the results are the same as we obtained for deep networks,  $B(S) \in c^{O(m^2)}$ . Conversely, as a function of  $h$ , the dependency changes and we obtain  $B(S) \in O(h)^{O(h)}$ .

**Bounds for BCE loss function: deep case.** For  $L \geq 3$ :

- if the last layer has a sigmoid activation function, we have that

$$2^{[(m(L-1)h+1)+1)(m(L-1)h+1)]/2} \cdot \mathcal{O}(g(n_0, h, m, L))^{h(m(L-1)+n_0+2+(h+1)(L-2))+m+2}$$

with

$$\begin{aligned} g(n_0, h, m, L) &= \\ &= h^2(L-2) + h(n_0 + L) + 1 + [3(L-2) + 5] \cdot \\ &\cdot \min(h^2(L-2) + h(n_0 + L) + 1, [m((L-1)h + 1) + 1]) \end{aligned}$$

If  $m \gg h$ ,  $g(n_0, h, m, L)$  becomes

$$g(n_0, h, m, L) =$$

$$= h^2(L-2) + h(n_0 + L) + 1 + [3(L-2) + 5](h^2(L-2) + h(n_0 + L) + 1)$$

On the other side, if  $h \gg m$  and we consider  $m$  and  $L$  as constant, we have that

$$g(n_0, h, m, L) =$$

$$= h^2(L-2) + h(n_0 + L) + 1 + [3(L-2) + 5][m((L-1)h + 1) + 1].$$

**Bounds for BCE loss function: shallow case.** For  $L = 2$ :

- if the last layer has a non-linear activation function, we have that

$$2^{[(m(h+1)+1)(m(h+1))]/2} O(g(n_0, h, m)^{h(m+n_0+2)+m+2}) \quad (\text{A.14})$$

with  $g(n_0, h, m) = h(n_0 + 2) + 1 + 5 \min(h(n_0 + 2) + 1, m(h + 1) + 1)$

If  $m \gg h$ ,  $g(n_0, h, m, L)$  becomes

$$g(n_0, h, m) = h(n_0 + 2) + 1 + 5(h(n_0 + 2) + 1).$$

On the other side, if  $h \gg m$  and we consider  $m$  and  $L$  as constant, we have that

$$g(n_0, h, m, L) = h(n_0 + 2) + 1 + 5[m(h + 1) + 1].$$

Resulting in asymptotic bounds equal to those derived previously for the MSE loss with non-linear activation. The same holds with similar computations in case the last activation function is the hyperbolic tangent.

We remark that for the BCE loss, our focus is primarily on studying the case where the last layer of the neural network has a non-linear activation function since it is commonly used for binary classification tasks.  $\square$

#### A.6. Residual connections

Without loss of generality, we can consider the case in which we utilize skip connections that provide the previous layer's output, through summation, as an additional input to the subsequent layer. In this case, denoting by  $z_i$  the output of the  $i$ th layer, if we consider the case, we have that

$$\begin{aligned} z_1 &= \sigma(W^1 x) \\ z_2 &= \sigma(W^2 z_1) + z_1 \\ z_3 &= \sigma(W^3 z_2) + z_2 \\ &\vdots \\ z_l &= \sigma(W^l z_{l-1}) + z_{l-1} \\ &\vdots \end{aligned}$$

If we want to obtain the derivative of  $z_l$  with respect to a parameter of the network  $w^k = W_i^k$  with  $k < l$  and  $i \in \{1, \dots, n_k\}$ , we have that

$$\frac{\partial z_l}{\partial w^k} = W^l \sigma'(W^l z_{l-1}) \frac{\partial z_l - 1}{\partial w^k} + \frac{\partial z_l - 1}{\partial w^k}$$

We observe that the derivative  $\frac{\partial z_l - 1}{\partial w^k}$  appears twice in the equation. However, due to the multiplication with other terms in the first term, it only affects the degree of the polynomial in the first term of the sum. Therefore, we can disregard the second term in the equation when interested in computing the format of the Pfaffian chain. This reasoning can be extended to all layers, demonstrating that the degrees of the derivatives will remain unchanged, just as we computed for simple feedforward perceptron neural networks. Similarly, the length of the chain remains unaltered. In this case, as well, the functions to be included in the chain are the outputs of the layers, which are exactly the same in number as those in feedforward networks, albeit with different forms.

## References

- Akhtiamov, D., & Thomson, M. (2023). Connectedness of loss landscapes via the lens of morse theory. In *NeurIPS workshop on symmetry and geometry in neural representations* (pp. 171–181). PMLR.
- Bartlett, P. L., Harvey, N., Liaw, C., & Mehrabian, A. (2019). Nearly-tight VC-dimension and pseudodimension bounds for piecewise linear neural networks. *Journal of Machine Learning Research*, 20(63), 1–17.
- Bartlett, P., Maierov, V., & Meir, R. (1998). Almost linear VC dimension bounds for piecewise polynomial networks. *Advances in Neural Information Processing Systems*, 11.
- Berner, J., Grohs, P., Kutyniok, G., & Petersen, P. (2021). The modern mathematics of deep learning. arXiv preprint arXiv:2105.04026.
- Bianchini, M., & Scarselli, F. (2014). On the complexity of neural network classifiers: A comparison between shallow and deep architectures. *IEEE Transactions on Neural Networks and Learning Systems*, 25(8), 1553–1565.
- Bredon, G. E. (2013). vol. 139, *Topology and geometry*. Springer Science & Business Media.
- Chen, L., Jin, L., & Shang, M. (2022). Zero stability well predicts performance of convolutional neural networks. Vol. 36, In *Proceedings of the AAAI conference on artificial intelligence* (6), (pp. 6268–6277).
- Choromanska, A., Henaff, M., Mathieu, M., Arous, G. B., & LeCun, Y. (2015). The loss surfaces of multilayer networks. In *Artificial intelligence and statistics* (pp. 192–204). PMLR.
- Choromanska, A., LeCun, Y., & Arous, G. B. (2015). Open problem: The landscape of the loss surfaces of multilayer networks. In *Conference on learning theory* (pp. 1756–1760). PMLR.
- Dauphin, Y. N., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S., & Bengio, Y. (2014). Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *Advances in Neural Information Processing Systems*, 27.
- Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.
- Freeman, C. D., & Bruna, J. (2016). Topology and geometry of half-rectified network optimization. arXiv preprint arXiv:1611.01540.
- Freeman, C. D., & Bruna, J. (2019). Topology and geometry of half-rectified network optimization. In *5th international conference on learning representations, ICLR 2017*.
- Gabrielov, A., & Vorobjov, N. (2004). Complexity of computations with pfaffian and noetherian functions. Vol. 137, In *Normal forms, bifurcations and finiteness problems in differential equations* (pp. 211–250). NATO Science Series II.
- Goodfellow, I. J., Vinyals, O., & Saxe, A. M. (2014). Qualitatively characterizing neural network optimization problems. arXiv preprint arXiv:1412.6544.
- Hastie, T., Tibshirani, R., Friedman, J. H., & Friedman, J. H. (2009). vol. 2, *The elements of statistical learning: data mining, inference, and prediction*. Springer.
- Hatcher, A. (2002). *Algebraic topology*. Press, Cambridge.
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).
- Karpinski, M., & Macintyre, A. (1997). Polynomial bounds for VC dimension of sigmoidal and general Pfaffian neural networks. *Journal of Computer and System Sciences*, 54(1), 169–176.
- Khovanskii, A. G. (1991). vol. 88, *Fewnomials*. American Mathematical Soc.
- Khovanskii, A. G. (1991). *Fewnomials – translations of mathematical monographs 88*. American Mathematical Society.
- Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, 42(8), 30–37.
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6), 84–90.
- Li, H., Xu, Z., Taylor, G., Studer, C., & Goldstein, T. (2018). Visualizing the loss landscape of neural nets. *Advances in Neural Information Processing Systems*, 31.
- Maennel, H., Bousquet, O., & Gelly, S. (2018). Gradient descent quantizes ReLU network features. arXiv preprint arXiv:1803.08367.
- Naizat, G., Zhitnikov, A., & Lim, L.-H. (2020). Topology of deep neural networks. *Journal of Machine Learning Research*, 21(1), 7503–7542.
- Nguyen, Q., & Hein, M. (2017). The loss surface of deep and wide neural networks. In *International conference on machine learning* (pp. 2603–2612). PMLR.
- Oneto, L., Navarin, N., Biggio, B., Errica, F., Micheli, A., Scarselli, F., et al. (2022). Towards learning trustworthily, automatically, and with guarantees on graphs: An overview. *Neurocomputing*, 493, 217–243.
- Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. (2018). Improving language understanding by generative pre-training.
- Ramachandran, P., Zoph, B., & Le, Q. V. (2017). Searching for activation functions. arXiv preprint arXiv:1710.05941.
- Safran, I., & Shamir, O. (2018). Spurious local minima are common in two-layer ReLU neural networks. In *International conference on machine learning* (pp. 4433–4441). PMLR.
- Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., & Monfardini, G. (2008). The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1), 61–80.

- Scarselli, F., Tsoi, A. C., & Hagenbuchner, M. (2018). The Vapnik-Chervonenkis dimension of graph and recursive neural networks. *Neural Networks*, *108*, 248–259.
- Venturi, L., Bandeira, A. S., & Bruna, J. (2018). Spurious valleys in two-layer neural network optimization landscapes. arXiv preprint arXiv:1802.06384.
- Venturi, L., Bandeira, A. S., & Bruna, J. (2019). Spurious valleys in one-hidden-layer neural network optimization landscapes. *Journal of Machine Learning Research*, *20*, 133.
- Williams, F., Trager, M., Panozzo, D., Silva, C., Zorin, D., & Bruna, J. (2019). Gradient dynamics of shallow univariate ReLU networks. *Advances in Neural Information Processing Systems*, *32*, 8378–8387.
- Zell, T. (1999). Betti numbers of semi-pfaffian sets. *Journal of Pure and Applied Algebra*, *139*(1), 323–338.
- Zell, T. P. (2003). *Quantitative study of semi-pfaffian sets* (Ph.D. thesis), Purdue University.