2021 Graduation Thesis

Neural Network Topological Expressiveness

Understanding neural networks behavior problem-independently

February 2021

Kyushu Institute of Technology, Faculty of Information Engineering
Department of Interdisciplinary Informatics

19677504-1

Alexandre Louvet

Supervisor: Aoi Honda

Contents

List of Figures

List of Tables

1 Neural network expressiveness

1.1 Definition

Let I_n denote the n-dimensional unit cube $[0,1]^n$ and $\mathcal{F}(I_n, \mathbb{R})$ be the space of functions from I_n to $\mathbb{R}$. We want to study the density of the subsets S_f of $\mathcal{F}(I_n, \mathbb{R})$ that can be written:

$$S_f = \{G_N(x) \in \mathcal{F}(I_n, \mathbb{R}) \mid G(x) = \sum_{i=1}^{N} \alpha_i f(y_j^T x + \theta_j)\},\ N \in \mathbb{N}$$

depending on the choice of $f \in \mathcal{F}(\mathbb{R}, \mathbb{R})$. In the previous equation $y_j \in \mathbb{R}^n$ and $\alpha_j, \theta \in \mathbb{R}$, y^T is the transpose of y and $y^T x$ is the inner product of y and x.

The study of neural network expressiveness consists of the problem described above when f is a function used as an activation function for neural network. The study of density can be on the whole set $\mathcal{F}(I_n, \mathbb{R})$ or on subsets of it such as $\mathcal{C}(I_n, \mathbb{R})$ the set of continuous functions from I_n to $\mathbb{R}$.

In particular if S_f is dense in a subset $A \subseteq \mathcal{F}(I_n, \mathbb{R})$ we will say that a single-layer feed-forward neural network (Figure 1) with f as its activation function is a *universal approximator* of A. Considering that a neural network has a finite number of nodes neural network expressiveness also consists of the study of the rate of approach of the approximation, i.e. the study of

$$\lim_{N \to \infty} H(N) = \max_{h \in A}(\min_{G_n \in S_f}(\parallel G_n - h \parallel))$$

with $\parallel . \parallel$ the canonical norm on $\mathcal{F}(I_n, \mathbb{R})$

The study of that limit and especially of its asympatotic approximation gives an idea of the efficiency of the approximator, i.e. the amount of nodes to add to the network to improve the approximation.

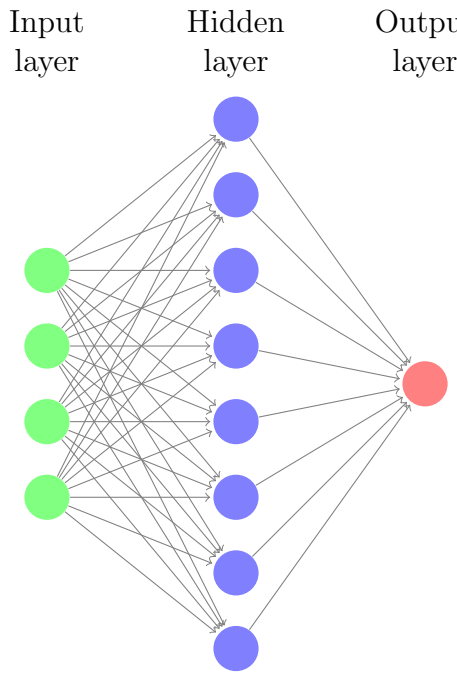


Figure 1: A single-layer feed-forward neural network with $n = 4$ and $N = 8$

1.2 Universal Approximator

In this section we will study the different subsets on which the logistic and ReLU functions act as universal approximators.

1.2.1 Sigmoidal functions

We say that a function $\sigma \in \mathcal{F}(I_n, \mathbb{R})$ is a sigmoidal function if:

$$\sigma(x) \to \begin{cases} 0 & \text{as } t \to +\infty \\ 1 & \text{as } t \to -\infty \end{cases}$$

The sigmoidal functions include the logistic function defined as:

$$f(x) = \frac{1}{1+e^{-x}}$$

widely used as an activation function for neural networks.

The first study of neural network expressiveness with sigmoidal functions dates back to [1] by G.Cybenko in 1989. He proved that S_σ for σ a sigmoidal function is dense in regards to the supremum norm in $C(I_n, \mathbb{R})$. The demonstration goes as follows.

We denote $M(I_n)$ the space of signed regular Borel measures on I_n

Definition 1 *σ is discriminatory if $\mu \in M(I_n)$ and*

$$\forall y \in \mathbb{R}^n,\ \theta \in \mathbb{R} \int_{I_n} \sigma(y^T x + \theta) d\mu(x) = 0 \implies \mu = 0$$

Theorem 1 *Let σ be a contininuous discriminatory function. Then finite sums of the form*

$$G(x) = \sum_{i=1}^{N} \alpha_i \sigma(y^T x + \theta_i)$$

are dense in $C(I_n, \mathbb{R})$

Proof: Let $S \subset C(I_n)$ be the set of the function of the form $G(x)$. S is a linear subset of $C(I_n)$. Let us show that the closure of S is $C(I_n)$.

Assume it is not the case. Then the closure of S, denoted R, is a proper subspace of $C(I_n)$. Using the Hahn-Banach theorem, there exists L a bounded linear functional on $C(I_n)$ with $L \neq 0$ and $L(R) = L(S) = 0$

Using the Riesz representation theorem, we obtain:

$$L(h) = \int_{I_n} h(x) d\mu(x)$$

for some $\mu \in M(I_n)$, for all $h \in C(I_n)$. Since $\sigma(y^T x + \theta_i) \in R$, we have

$$\forall y, \theta \int_{I_n} \sigma(y^T x + \theta) d\mu(x) = 0$$

Since σ is discriminatory, we have $\mu = 0$ and $L = 0$ follows!

Therefore the closure of S is $C(I_n)$ and by definition S is dense in $C(I_n)$ □

Now let us show that sigmoidal functions are discriminatory.

Lemma 1 *Any bounded, measurable sigmoidal function, a, is discriminatory. In particular, any continuous sigmoidal function is discriminatory.*

Proof: First note $\forall x, y, \theta, \phi$

$$\sigma_\lambda(x) = \sigma(\lambda(y^T x + \theta) + \phi) \begin{cases} \to 1 & \text{for } y^T x + \theta > 0 \text{ as } \lambda \to +\infty \\ \to 0 & \text{for } y^T x + \theta < 0 \text{ as } \lambda \to +\infty \\ = \sigma(\phi) & \text{for } y^T x + \theta = 0 \end{cases}$$

Thus $\sigma_\lambda(x)$ converges pointwise and boundedly to:

$$\gamma(x) \begin{cases} = 1 & \text{for } y^T x + \theta > 0 \\ = 0 & \text{for } y^T x + \theta < 0 \\ = \sigma(\phi) & \text{for } y^T x + \theta = 0 \end{cases}$$

as $\lambda \to +\infty$

Let $\Pi_{y,\theta} = \{x \mid y^T x + \theta = 0\}$ and let $H_{y,\theta} = \{x \mid y^T x + \theta > 0\}$. Lesbegue bounded convergence theorem gives us:

$$\begin{aligned} 0 &= \int_{I_n} \sigma_\lambda(x) d\mu(x) \\ &= \int_{I_n} \gamma(x) d\mu(x) \\ &= \sigma(\phi)\mu(\Pi_{y,\theta}) + \mu(H_{y,\theta}) \end{aligned}$$

for all ϕ, θ, y

Fix y, we write

$$F(h) = \textstyle\int_{I_n} h(y^T x) d\mu(x)$$

Note that F is a bounded function on $L^\infty(\mathbb{R})$ since μ is a signed mesure. By chosing h as the indicator function on $[\theta, \infty[$, we have:

$$F(h) = \textstyle\int_{I_n} h(y^T x) d\mu(x) = \mu(\Pi_{y,-\theta}) + \mu(H_{y,-\theta}) = 0$$

By linearity $F(h) = 0$ for indicator functions on any interval and hence for any simple function (sum of indicator functions). Since simple functions are dense in $L^\infty(\mathbb{R}), F = 0$. In particular it is true for the bounded functions $s(u) = sin(m.u)$ and $c(u) = cos(m.u)$. It gives:

$$F(s + ic) = \textstyle\int_{I_n} \cos(m^T x) + i \sin(m^T x) d\mu(x) = \int_{I_n} \exp(im^T x) d\mu(x) = 0$$

for all m. Therefore the fourier transform of μ is 0 and μ must be 0. Hence σ is discriminatory. □

This proves that any function of $C(I_n, \mathbb{R})$ can be approximated by a single-layer network with sigmoidal functions as activation function.

In 1991, Hornik extended in [2] Cybenko's theorem giving the proof for bounded non-constant functions of $F(I_n, \mathbb{R})$. The proof is very similar to the original one.

1.2.2 ReLU functions

ReLU functions stands for Rectified Linear Units, there are formed of two pieces of linear functions. They gained interest recently by showing convincing results in a lot of different applications.

In 2016 Arora et al. [3] showed a version of the universal approximation theorem for piecewise linear function that includes ReLU functions.

Definition 2 *We say a function $f : \mathbb{R}^n \to \mathbb{R}$ is continuous piecewise linear (PWL) if there exists a finite set of polyhedra whose union is $\mathbb{R}^n$, and f is affine linear over each polyhedron (implies continuity because affine regions are closed and cover $\mathbb{R}^n$)*

Proposition 1 *Every function in $L^q(\mathbb{R}^n), (1 \leq q \leq \infty)$ can be arbitrarily approximated in the L^q norm (which for a function f is given by $||f||_q = (\int |f|^q)^{1/q}$) by a ReLU Deep Neural Network.*

Proof:
We know that any ReLU DNN represents a PWL function, let's prove the converse.

By theorem 1 in [4] any piecewise linear function $f : \mathbb{R}^n \to \mathbb{R}$, can be written:

$$f = \sum_{j=1}^{p} s_j(\max_{i \in S_j} l_i)$$

with l_i $(i \leq i \leq k)$ linear functions, S_i $(1 \leq i \leq p) \subseteq \{1, ..., k\}$ with $\forall i, |S_i| \leq n + 1$ and $\forall j \in \{1, ..., p\}, s_j \in \{-1, +1\}$

It means that any PWL convex function can be represented as a linear combination of at most $n + 1$ affine pieces, i.e. a ReLU DNN with size $n + 1$.

Let $p \in \mathbb{N}^*$, let $f \in L^p(\mathbb{R}^n)$, consider the function sequence:

$$f_n(x) = (x - \tfrac{k}{n})f(\tfrac{k}{n}) + (1 - x + \tfrac{k}{n})f(\tfrac{k+1}{n}) \text{ with } \tfrac{k}{n} \leq x < \tfrac{k+1}{n}, n \geq 1$$

$f_n \xrightarrow[n\to\infty]{} f$, and $\forall n$, f_n is a PWL continous function. Therefore the continuous PWL functions are dense in $L^p(\mathbb{R}^n)$.

Since any PWL function $\mathbb{R}^n \to \mathbb{R}$ is representable by a ReLU DNN, unniversal approximation theorem follows from the fact that the family of continuous PWL functions is dense in any $L^p(\mathbb{R}^n)$ space for $1 \leq p \leq \infty$.

□

In this article, Arora et al. also gave a higher bound for the depth of the network. They showed that the network required for any function $f \in L^q(\mathbb{R}^n)$ is of at most depth $\lceil log_2(n + 1) \rceil$.

In 2018, Hannin & Selke [5] showed that the minimal width a ReLU neural network must have in order to approximate any function $f : \mathbb{R}^n \to \mathbb{R}^m$ a depth $d \geq n + m$ giving a lower bound to the width of the ReLU network.

2 Mathematical preliminaries

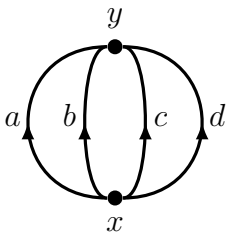
2.1 Homology

The study of homology is part of the field of algebraic topology. We will first introduce the concept of homology, then take a look at the theoretical concept behind it by introducing Δ-complexes: a primitive topological structure that allows the study of homology. Then with the introduction of simplicial homology we will complete the presentation of the fundamental objects used in our research in the study of homology. We will then reconsider these tools in the context of our research. This part heavily relies on the book *Algebraic Topolgy* by A. Hatcher [6] and the class given by P. Albin at University of Illinois [7] with the same book as class material.

2.1.1 The idea of homology

In algebraic topology, the most fundamental object to study is the homotopy, that roughly studies "paths". This study leads to the definition of the fundamental group $\pi_1(X)$ that gives information on the topological stucture of a space X for low-dimensional space. However the fundamental group $\pi_1(X)$ can not for instance distinguish the different spheres $\mathcal{S}^n, n \geq 2$. A workaround for that problem is to generalize the notion of homotopy to higher dimensions with different groups $\pi_n(X)$ generalizing the concept of homotopy in higher dimension.

However the homotopy groups suffers a serious drawback in their difficult computationality for high dimensions. Even for simple spaces like spheres, the calculation of $\pi_i(\mathcal{S}^n)$ turns out to be very difficult for $i > n$. That is why homology exists as a computable alternative homotopy.



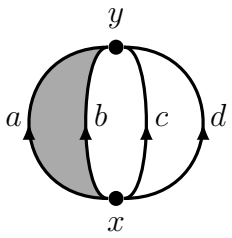
Consider the space X_1 in the figure drawn above. We can see two points x and y as well as four edges a, b, c and d. Consider the loops formed by travelling along the edges, for instance the path ab^{-1} is a loop with x as the basepoint. Something to consider is that the loop $b^{-1}a$ is basically the same loop but starting from a different basepoint, y in that case. By abelianizing, we can consider cycles instead of loops without any basepoint.

The abelian groups having only one operation, we will then switch to additive notations to remain in accordance with Hatcher's notations. With that new notation we

can write equalities such as $(a-c)+(b-d)=(a-d)+(b-c)$. This is justified by the fact that from an algebraic point of view, there is no difference between these two cycles.

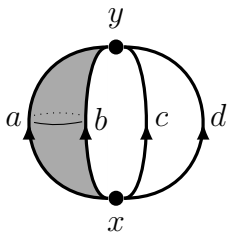
Let us call any linear combination of edges, *chains*. Then the condition for a chain $ka+lb+mc+nd$ with $k,l,m,n \in \mathbb{N}$ to be a cycle is $k+l+m+n=0$.

Now let us start to formalize these concepts. Consider C_0 the free abelian group spanned by the vertices x, y and let C_1 be the free abelian group spanned by the edges a, b, c, d. One can define an homomorphism $\delta : C_1 \to C_0$ that sends any basis element of C_1 to $x-y$. Thus we have $\delta(ka+lb+mc+nd) = (k+l+m+n)a-(k+l+m+n)y$ and $ker(\delta) = \{$ cycles of X_1 $\}$. $\{a-b, b-c, c-d\}$ forms a basis for this kernel, i.e. any cycle is a linear combination of the most obvious cycles of X_1, that conveys the information that X_1 has three visibles "holes" formed by the space between its four edges.



Now fill the hole between edges a and b with a 2-cell, namely A to create a new space X_2. Considering it as being oriented clockwise, its boundary is $a-b$. The cycle $a-b$ is homotopic to a point since it can now be contracted by sliding over A, it no longer encloses a hole in G. That suggests that homology should consider only the quotient group of what we found previously by $a-b$. In this quotient group the cycles $a-c$ and $b-c$ are equivalent. Which is consistent with the fact that they are homotopic in X_2.

Algebraically we can now consider a pair of homomorphism δ_1, δ_2 such that $C_2 \xrightarrow{\delta_2} C_1 \xrightarrow{\delta_1} C_0$ where C_2 is the cyclic group spanned by A and $\delta_2(A) = a-b$. The quotient group we are interested in is $ker(\delta_1)/Im(\delta_2)$, i.e. the 1-dimensional cycles modulo those that are boundaries. We will call it the homology group $H_1(X_2)$. It can also be computed in X_1 by considering C_2 to be zero since there are no 2-cells in X_1. $H_1(X_2)$ now only admits two generators $b-c$ and $c-d$ expressing the geometric fact that the number of holes reduced from 3 to 2.



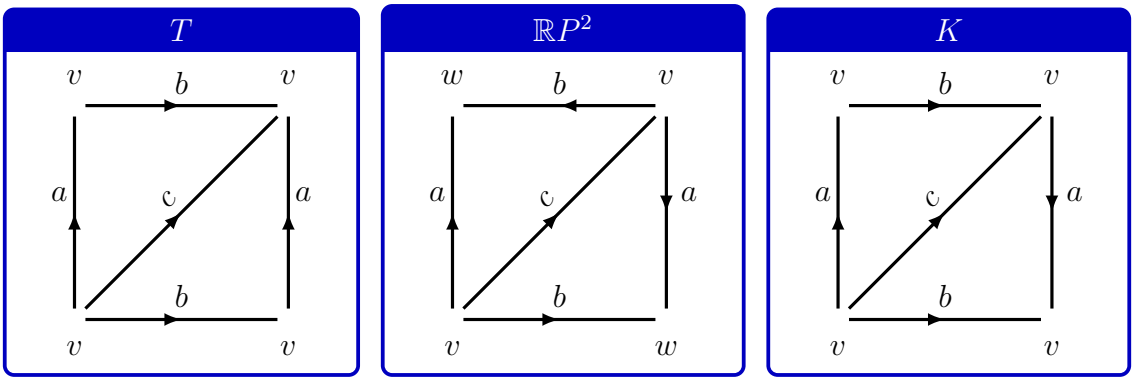
Suppose we attach another 2-cell between a and b to create X_3, namely B. C_2 now consists of linear combinations of A and B and $\delta_2(A) = \delta_2(B) = a - b$. On the one hand, $H_1(X_3) = H_1(X_2) \approx \mathbb{Z} \times \mathbb{Z}$, but on the other hand $ker(\delta_2) \neq 0$ and $A - B$ is its generator, hence we have $H_2(X_3) = ker(\delta_2) \approx \mathbb{Z}$. Topologically the cycle $A - B$ is equivalent to a sphere and it detects the presence of a "hole" enclosed by this sphere rather than a circle.

The pattern we can see appear with these examples is rather clear. The n-cell complexes of X forms free abelian groups $C_n(X)$ and one can define homomorphisms $\delta_n : C_n \to C_{n-1}$ to define homology groups $H_n(X) = ker(\delta_n)/Im(\delta_{n+1})$. The only problem now is to define δ_n for any n. If it is simple for small n (head minus tail for a vertex), it becomes rather complicated when n grows and more complex polyhedral cells appear in X. The most efficient approach is to decompose polyhedra into simplices which allows simple orientation and boundary computation. For that purpose we will define nice cellular complexes that will be the fundamental element for homology computation in the homology theories presented after.

2.1.2 Δ-complexes

The most important theory of algebraic topology is called singular homology, since it is technically complicated we will use a simpler version of it called simplicial homology. The natural definition spaces of simplicial homology is called Δ-complexes that we will introduce in this part.

The projective plane, the klein bottle and the torus can be obtained from squares by identifying edges and giving them orientation as draw below.



In fact any closed surface can be constructed this way, i.e. by cutting a polygon along its diagonals and identifying pairs of edges. The idea between Δ-complexes

is to generalize this idea to any dimension. The n-dimensional analog of the triangle is called the n-simplex.This is the smallest convex set in a Euclidean space $\mathbb{R}^m$ containing $n+1$ points $v_0, ..., v_n$ that do not lie in a hyperplane of dimension less than n. The points v_i are the vertices of the simplex and the simplex is denoted $[v_0, ..., v_n]$. The standard n-simplex is:

$$\Delta^n = \{(t_0, ...t_n) \in \mathbb{R}^{n+1} \mid \textstyle\sum_i t_i = 1 \text{ and } t_i \geq 0 \text{ for all } i\}$$

Its vertices are the unit vectors along the coordinate axes.

It is important to keep track of an ordering for the vertices and an n-simplex will always be considered with an ordering for its vertices. That will allow us to give an orientation for its edges $[v_i v_j]$ with respect to the ordering of the indices. It also determines a canonical linear homeomorphism from the standard n-simplex Δ^n into any other n-simplex preserving the order of the vertices: $(t_0, ..., t_n) \mapsto \sum_i t_i v_i$. The coefficients are the barycentric coordinates of the points $\sum_i t_i v_i$ in $[v_0, ..., v_n]$.

If we delete one of the vertices of an n-simplex $[v_0, ..., v_n]$, the remaining n vertices span an (n-1)-simplex, called face of $[v_0, ..., v_n]$ that will conventionnaly keep the same orientation as in $[v_0, ..., v_n]$. The union of all the face of Δ^n is the boundary of Δ^n, noted $\delta\Delta^n$. The open simplex $\mathring{\Delta}^n$ is $\Delta^n - \delta\Delta^n$, is the interior of Δ^n.

A Δ-complex on a topological space X is a collection of maps $\sigma_\alpha : \Delta^n \to X$ with the following properties:

(i) The restriction $\sigma_\alpha \mid \mathring{\Delta}^n$ is injective, and each point of X is in the image of exactly one such restriction $\sigma_\alpha \mid \mathring{\Delta}^n$.

(ii) Each restriction if σ_α to a face of Δ^n is one if the maps $\sigma_\beta : \Delta^{n-1} \to X$ (identifying the face of Δ^n with the face of Δ^{n-1} using the canonical linear homeomorphism between them and preserving the vertices ordering).

(iii) A set $A \subset X$ is open if and only if $\sigma_\alpha^{-1}(A)$ is open in Δ^n for each σ_α.

The earlier decompositions of the torus, projective plane, and Klein bottle into two triangles, three edges, and one or two vertices allow well-definess of associated Δ-complex structures with a total of six σ_α's for the torus and Klein bottle and seven for the projective plane. The orientation of edges are compatible with an ordering of the vertices (example below). This ordering determines the maps σ_α.

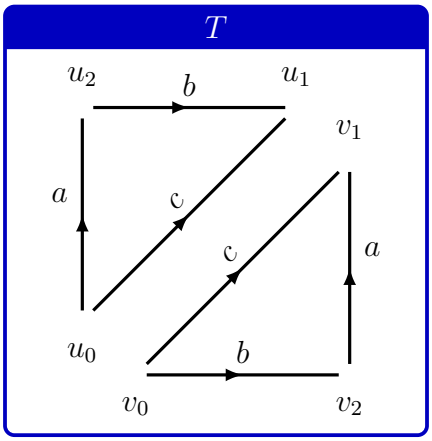
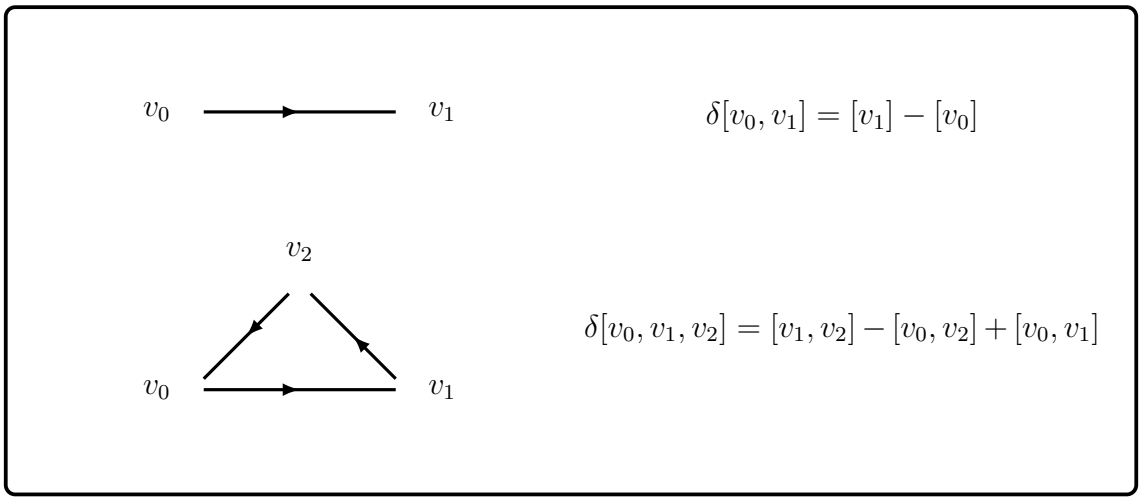


Figure 2: The δ-complex structure of the torus T

2.1.3 Simplicial homology

Let us now define the simplicial homology groups of a Δ-complex X. Let $\Delta_n(X)$ be the free abelian group with basis the open n-simplices e^n_α of X. The elements of $\Delta_n(X)$ are called n-chains and are of the form $\sum_\alpha n_\alpha e^n_\alpha$ with $n_\alpha \in \mathbb{Z}$. Equivalently, this could be written as $\sum_\alpha n_\alpha \sigma_\alpha$ where $\sigma_\alpha : \Delta^n \to X$ is the characteristic map of e^n_α with image the closure of e^n_α. Such a sum can be considered as a finite collection, or "chain", of n-simplices in X.

The boundary of the n-simplex $[v_0, ..., v_n]$ consists of the various $(n-1)$-dimensional simplices $[v_0, ..., \hat{v_i}, ..., v_n]$ (indicates all $v_k, 1 \leq k \leq n$, but v_i). We might wish that the boundary of $[v_0, ..., v_n]$ be the $(n-1)$-chain formed by the sum of the faces $[v_0, ..., \hat{v_i}, ..., v_n]$. It turns out to be betters to insert sign to take into account orientations. It allow to coherently orient all the faces of the simplex. That is why we let the boundary be $\sum_i (-1)^i [v_0, ..., \hat{v_i}, ..., v_n]$.



We define a boundary homomorphism for a general Δ-complex X, $\delta_n : \Delta_n(X) \to$

$\Delta_{n-1}(X)$ by specifying its values on the basis elements:

$$\delta_n(\sigma_\alpha) = \sum_i (-1)^i \sigma_\alpha \mid [v_0, ..., \hat{v_i}, ..., v_n]$$

The image set of δ_n is $\Delta_{n-1}(X)$ since each restriction $\sigma_\alpha \mid [v_0, ..., \hat{v_i}, ..., v_n]$ is the caracteristic map of an $(n-1)$-simplex of X.

Lemma 2 *The composition* $\Delta_n(X) \xrightarrow{\delta_n} \Delta_{n-1}(X) \xrightarrow{\delta_{n-1}} \Delta_{n-2}(X)$ *is zero.*

PROOF:
We have $\delta_n(\sigma) = \sum_i (-1)^i \sigma \mid [v_0, ..., \hat{v_i}, ..., v_n]$, hence

$$\begin{aligned}\delta_{n-1}\delta_n(\sigma) = &\sum_{j<i} (-1)^i (-1)^j \sigma \mid [v_0, ..., \hat{v_j}, ..., \hat{v_j}, ..., v_n] \\ &+ \sum_{j>i} (-1)^i (-1)^{j-1} \sigma \mid [v_0, ..., \hat{v_i}, ..., \hat{v_j}, ..., v_n] = 0.\end{aligned}$$

The last equality comes from the switch of i and j in the second sum.

□

We end up with a sequence of homomorphisms of abelian groups:

$$... \to C_{n+1} \xrightarrow{\delta_{n+1}} C_n \xrightarrow{\delta_n} C_{n-1} \to ... \to C_1 \xrightarrow{\delta_1} C_0 \xrightarrow{\delta_0} 0$$

with $\delta_n \delta_{n+1} = 0$ for each n (cf Lemma 2).We call such a sequence a chain complex. The equation $\delta_n \delta_{n+1} = 0$ is equivalent to the inclusion Im $\delta_{n+1} \subset$ Ker δ_n. We can define the n^{th} homology group of the chain complex to be the quotient group $H_n =$ Ker δ_n/Im δ_{n+1}. The elements of Ker δ_n are called cycles and elements of Im δ_{n+1} are called boundaries. Elements of H_n are cosets of Im δ_{n+1} called homology classes. Two cycles representing the same homology class are said to be homologous, it means their difference is a boundary.

In the case of $C_n = \Delta_n(X)$, the homology group Ker δ_n/Im δ_{n+1} will be denoted $H_n^\Delta(X)$ and called the n^{th} simplicial homology group of X.

Example: Let us take $X = T$, the torus with the Δ-complex structure pictured in Figure 2. For recall, the structure has one vertex v, three edges a, b, c and two 2-simplices U and V. $\delta_1 = v - v = 0$ so $H_0^\Delta(X) \approx \mathbb{Z}$. Since $\delta_2 U = a + b - c = \delta_2 L$ and $\{a, b, a+b-c\}$ is a basis for $\Delta_1(T)$, $H_1^\Delta(X) = \frac{\delta_1(X)}{\delta_2(U)}$, it follows that $H_1^\Delta(X) \approx \mathbb{Z} \oplus \mathbb{Z}$ with basis the homology classes $[a]$ and $[b]$. Since there are no 3-simplices, $H_2^\Delta(T)$ is equal to Kerδ_2, which is infinite cyclic generated by $U - L$ since $\delta(pU + qL) = (p+q)(a+b-c) = 0$ iff $p = -q$.

Therefore:

$$H_n^{\Delta}(X) \approx \begin{cases} \mathbb{Z} \oplus \mathbb{Z} & \text{for } n = 1 \\ \mathbb{Z} & \text{for } n = 0, 2 \\ 0 & \text{for } n \geq 3 \end{cases}$$

Proposition 2 *For $n \in \mathbb{N}^*, H_n^{\Delta}(\mathcal{S}^n) \approx \mathbb{Z}$ with $\mathcal{S}^n$ the n-dimensional sphere.*

PROOF: A Δ-complex structure can be obtained on $\mathcal{S}^n$ by taking two copies of Δ^n and identifying boundaries using the identity map. Using previous notations, we have Ker δ_n an infinite cycle generated by $U - L$. Therefore $H_n^{\Delta}(\mathcal{S}^n) \approx \mathbb{Z}$. □

The simplicial homology is defined for simplical complexes. These are the Δ-complexed whose simplices are uniquely determined by their vertices. Each n-simplex has $n + 1$ distinct vertices and no other n-simplex has the same set of vertices. A simplicial complex can be described as a set X_0 of vertices and sets X_n of n-simplices which are $(n + 1)$-elements subsets of X_0. It is required that each $(k + 1)$-elements subset of the vertices of an n-simplex in X_n is a k-simplex in X_k.

Compared with Δ-complexes, simplicial complexes require more computation. For example a simplicial complex structure for the torus would require 14 triangles, 21 edges and 7 vertices. It can be shown that any Δ-complex can be subdivided in a simplical complex.

It is worth mentioning that simplicial homology has flaws such as the one mentioned before. That is why in the current theory of homology, simplicial homology has been replaced with a more complete theory of homologies named singular homology. The improvements of simplicial homology brought by singular are not useful for our research, that is why we will not be discussing this theory in this paper. The theory is however thoroughly explained in [6].

2.1.4 Homology in our research

Our research is concerned with the capacity of neural networks to properly express, after training, different difficulty of problems depending on their width. There would be different approaches to chosing a way to measure this difficulty that would relate to different applied problems. The first approach would be to look at the dimension of the problem.

For instance the MNIST problem, aims to classify 28x28 pixels images of handwritten digits in 10 classes representing the 10 digits (see Figure 3). For instance in 2003 in [8], a team of researcher used a simple neural network of 2 layers with respectively 100 and 10 hidden units to classify the MNIST dataset (they used convolutional layers before inputing data in their network making the input dimension even greater) to obtain an accuracy of 98.81%. One could expect that solving the same problem of image classification for instance and using different image size and depth of networks would be an interesting way to study the problem of expressiveness.

Figure 3: Some images of the MNIST dataset

However two major problems occur in that case. First how to choose the classification problem with the certainty that the problem itself is not going to be a problem simpler than it looks? This relates to an hypothesis called the manifold hypothesis. The manifold hypothesis holds roughly as follows: "real-world high-dimensional data lie on low-dimensional manifolds embedded within the high-dimensional space". Meaning that complex data that needs a lot of parameters to be perfectly described are caracterized by few simple parameters up to continuous deformations as defined by topology. Considering this hypothesis that has not been disproved, a certain image recognition problem such as MNIST could be relying on the separation of a space of smaller dimension that the 784 dimensions space that you would expect from 28x28 pixels images.

The second problem is the problem of thresholding for accuracy since it is almost impossible to reach 100%. One would have to chose a value over which one considers the problem as correctly classified and below which it is considered not classified. That choice would be rather arbitrary and it would be hard to evaluate the value of the results as far as they might change radically with a slight change in the choice of the threshold.

This point is non-specific to the choice of accuracy as a measure since in any case the training of a neural network is a process with a random component involved and independently of the measure choice, the problem of training artifacts and imprecions might arise. Therefore one should try to chose a measure that will reduce as much as possible this incontrollable component in the research.

For that purpose, we have chosen to use homology of the data as our measure. Using homology garantees an independence from the problem chosen since homologies can take different forms due to its topological nature. Moreover if the manifold hypothesis happends to be true, the study of homology in rather small dimension and how neural networks of different depth perform on it might have applications even in high dimension problems. Unlike accuracy the perfect classification in terms of homology is something reachable since homology is studied without regard to

continuous deformations ($\approx$ topologically), errors and imprecision will mostly be ignored.

2.2 Topological Data Analysis

Topological Data Analysis, also called TDA, is an approach to the analysis of datasets using techniques from topology. The idea behind it is to provide with a framework to rigorously study the "shape" of a dataset. The main tool is persistent homology, that allows the computation of homology of a point cloud dataset. This tool will be presented in detail after a general presentation of the field of TDA. This chapter is based on the paper of L. Wasserman [9].

2.2.1 Introduction to TDA method's

Topological Data Analysis definition is somewhat vague and if it is very clear that persistent homology belong to this field, the exact set of methods, theories and results that belongs to this field is quite imprecise. According to this imprecision we chose to give the following definition to TDA: It's a "large class of data analysis method that uses the notion of shape and connectivity" [9]. In his paper Wasserman details different approaches to this question of shape and conenctivity study starting with density clusters that we will only briefly cover since these methods are of no use for the purpose of our research. We will then in this section present the ideas and tools behind the study of low-dimension support for high dimension sets and their study with TDA's tools. That second part is here in support of the manifold hypothesis discussed briefly in **Section 2.1.4**.

Density Clustering In his paper, Wasserman covers three types of clustering. More clustering methods could be considered as part of TDA but for explanation purposes of what TDA is, the first two methods will be sufficient as they will introduce paradigms we will find in persistent homology.

The first method introduced is level set clustering. The idea behind this name is the study of the density of random samples regarding a threshold. Formally consider $X_1, ..., X_n$ a random sample of a distribution P with density p where $X_i \in \mathcal{X} \subset \mathbb{R}^d$.

For $t \geq 0$ we define:

$$L_t = \{x : p(x) > t\}$$

The density clusters at level t, denoted C_t, are the connected components of L_t. One can define an estimation of L_t, denoted $\hat{L}_t$ as follows:

$$\widehat{L}_t = \{x : \widehat{p}(x) > t\}$$

where $\widehat{p}(x)$ is an estimator for the density p, such as the kernel density:

$$\widehat{p}_h(x) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h^d}K(\frac{||x-X_i||}{h})$$

where $h > 0$ is the bandwidth and K the kernel.

To find the clusters, one need to find the connected components of $\widehat{L}_t$. Let $I_t = \{i : \widehat{p_h}(X_i) > t\}$. Create a graph with vertices $(X_i : i \in I_t)$. Now put an edge between two vertices X_i and X_j if $||X_i - X_j|| \leq \varepsilon$ where $\varepsilon > 0$ is a parameter ($\varepsilon = 2h$ works well in practice).

It appears the choice of t is crucial in the level set clustering method and there are different possibilities to chose it. First one can chose some prescribed fraction $1 - \beta$ of the total mass. Another idea is to look at clusters at all levels t. This will lead us to the idea of density trees.

The idea of density trees is to create a concept to study level set clustering for all t. That concept will give us both a representation of the structure of the data as well as a way to compare its topological similarity with another set.

The set of all density clusters $\mathcal{C}$ has a tree structure, indeed if $A, B \in \mathcal{C}$ then either $A \subset B$ or $B \subset A$ or $A \cap B = \emptyset$. The tree shows the number of level sets at some level t. The number of branches of the tree correponds to the number of connected components. Two density trees have the same "shape" if their tree structure is the same. Given a tree T_p the distance on the tree is defined as:

$$d_{T_p}(x, y) = |p(x) + p(y) - 2m_p(x, y)|$$

where

$$m_p(x, y) = sup\{t : \exists C \in C_i \mid x, y \in C\}$$

is called the merge height. For any two clusters $C_1, C_2 \in T_p$, we define $\lambda_1 = \{t : C_1 \in C_t\}$ and λ_2 analogously. We then define the tree distance function on T_p as:

$$d_{T_p}(C_1, C_2) = \lambda_1 + \lambda_2 - 2m_p(C_1, C_2)$$

where

$$m_p(C_1, C_2) = sup\{\lambda \in \mathbb{R} : \exists C \in T_p \mid C_1, C_2 \subset C\}$$

Since d_{T_p} defines a distance on the tree, it induces a topology on T_p. Given two densities p and q we can say that T_p is equivalent to T_q, denoted $T_p \approx T_q$ if there exists an homeomorphism between the two trees.

Low dimensional subsets The point of study of low dimensional subset relates to the manifold hypothesis and more generally the study of the support of the distribution P when it is only a set S of dimension r with $r < d$ (see Figure 4). Sometimes the support of P is of dimension exactly d but there exists a set S of dimension $r < d$ which has a high concentration of mass (see Figure 5).

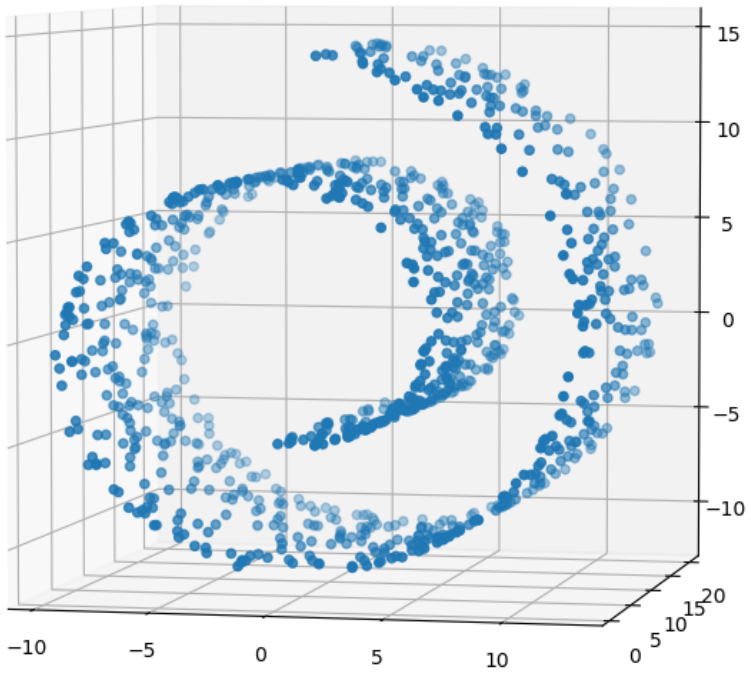


Figure 4: A plot of the swiss roll dataset. We have $d = 3$ but the support of the data S is of dimension $r = 2$

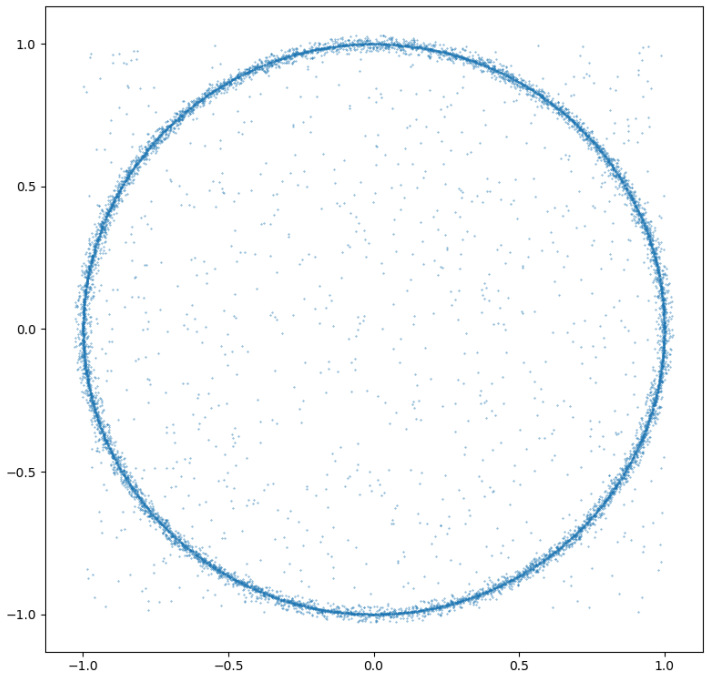


Figure 5: On this plot we have a dataset of dimension 2 but there is a support of dimension 1 that concentrates a big part of the mass

An estimator of S is $\widehat{S} = \bigcup_{i=1}^{n} B(X_i, \varepsilon_n)$. The estimator $\widehat{S}$ is d-dimensional but converges to S. The Hausdorff distance $H(A, B)$ between two sets A and B is:

$$H(A, B) = \inf\{\varepsilon : A \subset B \oplus \varepsilon \text{ and } B \subset A \oplus \varepsilon\}$$

where

$$A \oplus \varepsilon = \bigcup_{x \in A} \mathcal{B}(x, \varepsilon)$$

with $\mathcal{B}(x, \varepsilon)$ the ball of radius ε centered on x. Assume S to be r-dimensional and P to have its mass spread all over S. Formally: $\exists c > 0 \mid \forall x \in S, \varepsilon > 0, P(B(x, \varepsilon)) \geq c\varepsilon^r$ and the number of balls of size ε required to cover S is $C(\frac{1}{\varepsilon})^r$. Then:

$P(H(\widehat{S}, S) > 2\varepsilon)$
$= P$(there is no ball in $S \oplus \varepsilon$ with no sample point inside)
$\leq C\varepsilon^{-r} \max_k (1 - P(X_1 \in B_k))^n$
$\leq C\varepsilon^{-r}(1 - \min_k P(X_1 \in B_k))^n$
$\leq C\varepsilon^{-r} e^{-nc\varepsilon^r}$ (using $P(X \in B_k) \geq c\varepsilon^r$ and $(1-x)^n \leq exp(-nx)$ for $0 \leq x \leq 1$)
which converges.

Note that we only verify one of the two inclusions for the Hausdorff distance since the other one is always respected.

Now that we can find an estimation for a manifold that would concentrate all or most of a distribution. We try to estimate the topology of a manifold. That will be done in details with persistent homology however let us tackle the topic already. To estimate the topology of a set, one can try to find an estimator of the topology that is topologically similar to the studied set. Studying topological similarity comes down to the study of the existence of homeomorphism i.e. of bi-continuous maps. Markov [10] showed that in general the question is undecidable for dimension greater than four. That is why instead we determine if two sets are homologically equivalent.

A result in topology and statistics showed that:

$$\widehat{S} = \bigcup_{i=1}^{n} \mathcal{B}(X_i, \varepsilon)$$

has the homology of S with high probability as long as S has a positive reach and ε remains relatively small compared to it. The reach r of a set S is the largest real number such that any point within distance r of S has a unique projection on S.

2.2.2 Persistent homology

Persistent homology is the computationable side of homology theory. Persistent homology is the name given to an algorithm aiming to compute the topological features of data. We have discussed in detail homology from a mathematical point of view

in **Section 2.1** and have shown that for $n \geq 2$, an n-dimensional sphere has for homology group: $H_n^{\Delta}(\mathcal{S}^n) \approx \mathbb{Z}$. To simplify vocabulary and notation we will be using betti numbers. Betti numbers of an n-dimensional topological space X counts the ranks of the groups each $(H_k^{\Delta}(X))_{0 \leq k \leq n}$ is topologically equivalent to. In the case of the torus T, the betti number are $(1, 2, 1, 0)$. Recalling the formula to compute $H_k^{\Delta}(X) = \frac{\text{Ker}\delta_n}{\text{Im}\delta_{n+1}}$, since $C_{n+1} = \emptyset$, $\delta_{n+1} = 0$ and hence any value $(H_k^{\Delta}(X))_{k>n} = 0$. Therefore, it is conventionally admitted not to write betti numbers above that value.

To properly define persistent homology one must first define the distance function between a set S and a point x. It is written as:

$$d_s(x) = \inf_{y \in S} ||x - y||$$

The lower level sets of the distance function are

$$L_\varepsilon = \{x : d_S(x) \leq \varepsilon\}$$

It is equivalent to writing:

$$L_\varepsilon = \bigcup_{x \in S} \mathcal{B}(x, \varepsilon)$$

As ε increases, L_ε changes. Different topological features (represented by the homology groups and the betti numbers) will appear and disappear. Consider $\mathcal{S}^1$, at first when $\varepsilon < 1$, we have the betti numbers being $(0, 1, 0)$ and once ε reaches 1, the betti numbers become $(1, 0, 0)$ (cf Figure 6).

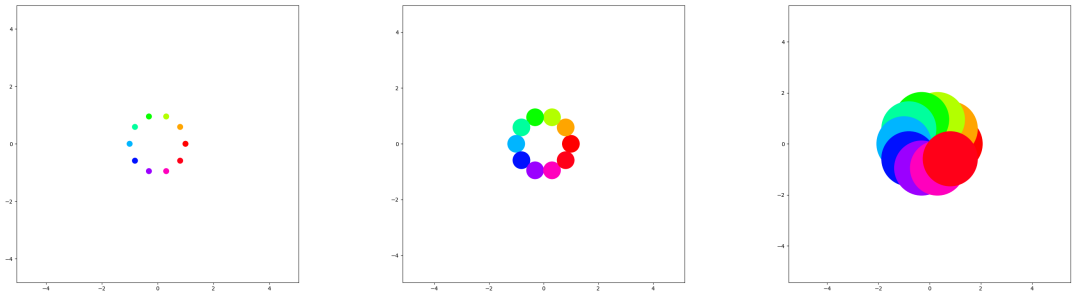


Figure 6: Three plots for a dataset with data on $\mathcal{S}^1$. The betti numbers are from left to right: $(10, 0, 0)$, $(0, 1, 0)$, $(1, 0, 0)$

Considering the evolution of L_ε, one can draw a persistence diagram (cf Figure 7) and a barcode diagram (cf Figure 8) that are equivalent. The barcode diagram represents the time of birth and death of each topological feature with a bar for each persistent generator. You will note that the full circle obtained from the merge of all balls around each point is considered to be the obtained from one of the original generator hence its big persistence. The persistence diagram represents with a point

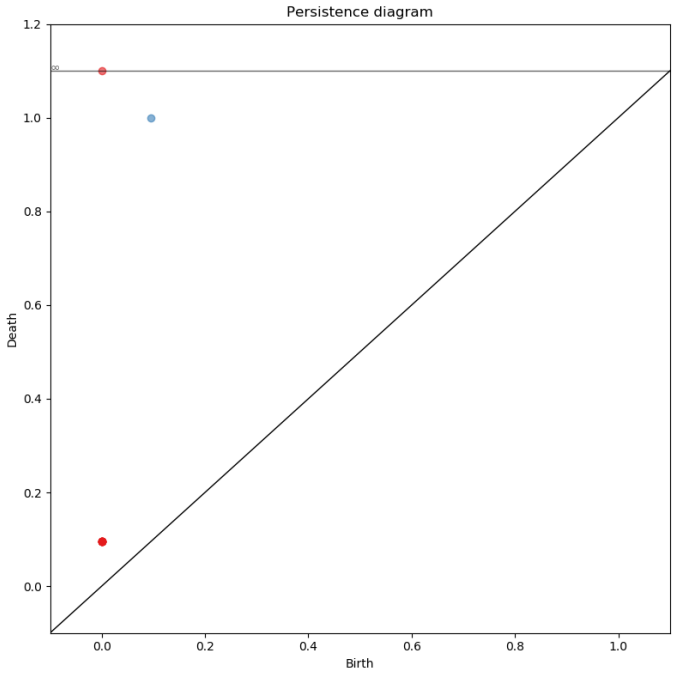


Figure 7: The persistence diagram of dataset from Figure 6

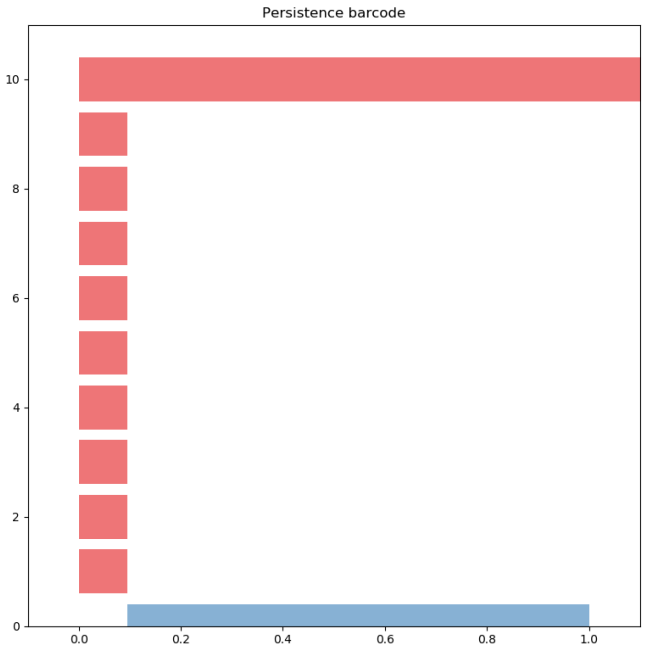


Figure 8: The barcode diagram of dataset from Figure 6

the birth and death of each topological feature, considering the symetry of our figure, there are 9 points on coordinate $\approx (0, 0.1)$.

Note that an infinite is defined on the right of the barcode and the top of the y-axis of the persistence diagram. It corresponds to the value of $\varepsilon_\infty = \max_{x,y \in S} ||x - y||$. Equivalently $\forall x, y \in \mathcal{S}, x \in \mathcal{B}(y, \varepsilon_\infty)$ and $y \in \mathcal{B}(x, \varepsilon_\infty)$. To the different persistence, we define the bottleneck distance. Given two persistence diagram D_1, D_2:

$$\delta_\infty(D_1, D_2) = \inf_\gamma \sup_{z \in D_1} ||z - \gamma(z)||_\infty$$

with γ covering the set of bijections between D_1 and D_2. The image behind it is to overlay the two diagrams and look at how to match the points as well as possible to minimize the distance between them.

To compute the dimension of the structure persistent homology one uses Δ-complexes by using an object called the $\check{C}$ech complex C_ε. It is defined as follows. All singletons are included in C_ε, there are 0-dimensional simplices. All pairs of points X_i, X_j such that $||X_i - X_j|| \leq \varepsilon$ are included in C_ε, these are 1-dimensional simplices. Each triplet X_i, X_j, X_k such that $\mathcal{B}(X_i, \varepsilon/2) \cap \mathcal{B}(X_j, \varepsilon/2) \cap \mathcal{B}(X_k, \varepsilon/2) \neq \emptyset$ is included, these are 2-dimensional simplices. And so on.

A more computationally efficient way to compute complexes is called the Vectoris-Rips complex V_ϵ and is defined as follows. A simplex is included in V_ϵ if each pair of vertices is no more than ε apart. It can be shown that the persistent homology defined by V_ε approximates the persistent homology defined by C_ε .

2.3 Knot theory

Knot theory is a part of topology concerned with the study of mathematical knots. They are inspired by real life knots that can be done on ropes or shoelaces but differs in the sense that the ends are joined together so that it can not be undone. There are multiple ways to describe knots and we will cover some of them. One of the key concepts of knot theory is equivalence. Two knots are considered equivalent if they can be transformed in each other without cutting the rope open. We will formalize these notions in this part relying on the class given by Pr. Anthony Bosman at Andrew University [11].

2.3.1 Fundamental concepts

Definition 3 *A knot is an embedding of a circle in 3-dimensional euclidian space* $\mathbb{R}^3$

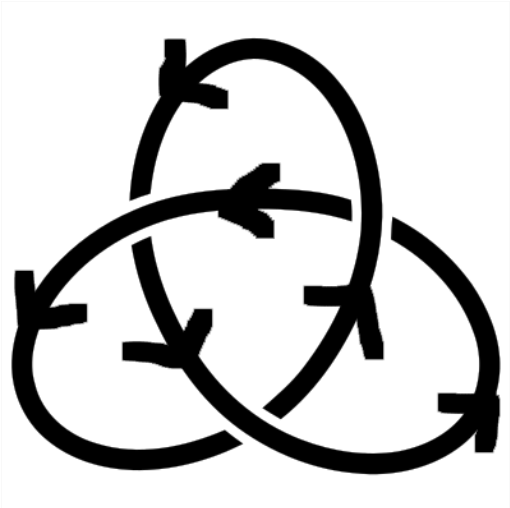
A knot is bounded by all of its homeomorphism since we are studying it from a topological point of view. Two mathematical knots are equivalent if they are ambient isotopic (can be deformed into the other via a deformation of $\mathbb{R}^3$ upon itself).

Definition 4 *let N and M be manifolds and g and h be embeddings of N and M. A continuous map:*

$$F : M \times [0, 1] \to M$$

is defined to be an ambient isotopy taking g to h if F_0 is the identity and each map F_t is an homeomorphism from M to itself, and $F_1 \circ g = h$.

Knot representation might for further explained purpose be oriented giving an orientation to each of its crossings. That result with two types of crossings left-handed crossing and right-handed crossings as shown in Figure 11



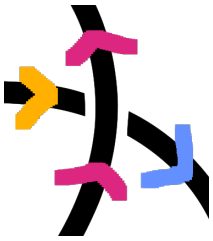


Figure 9: Left-handed crossing

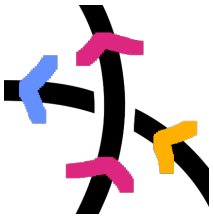


Figure 10: Right-handed crossing

Figure 11: On the left an oriented trefoil, on the right a left-handed knot and a right-handed knot

That is to say there is a map F with a parameter t ranging from 0 to 1, sending for $t = 0$, h to itself and for $t = 1$, h to g.

In 1927, Reidemeister showed that ambient isotopy was equivalent to a set of three moves that can be performed on knots [12]. The three moves are as follow:

- Type I: Twist

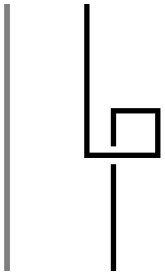


Figure 12: The Reidemeister twist transformation

- Type II: Poke

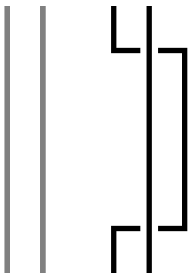


Figure 13: The Reidemeister poke transformation

- Type III: Slide

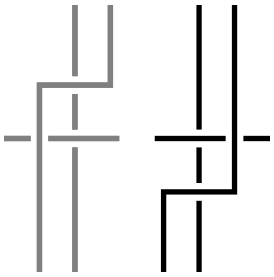


Figure 14: The Reidemeister slide transformation

i.e. two knots are equivalent if they can transformed into each others using only these three moves.

Knot are composed of crossings and components. The crossings are the point where the embedding overlaps itself and the components are the parts of the embedding between the crossings. We will later number the components and crossings to perform computations on knots. It is important to note that cannonically the knots are classified in category representing the minimal number of crossing that can be obtained by a representation of a knot. You can find the beginning of the knot table sorted by increasing number of crossings in Figure 15.

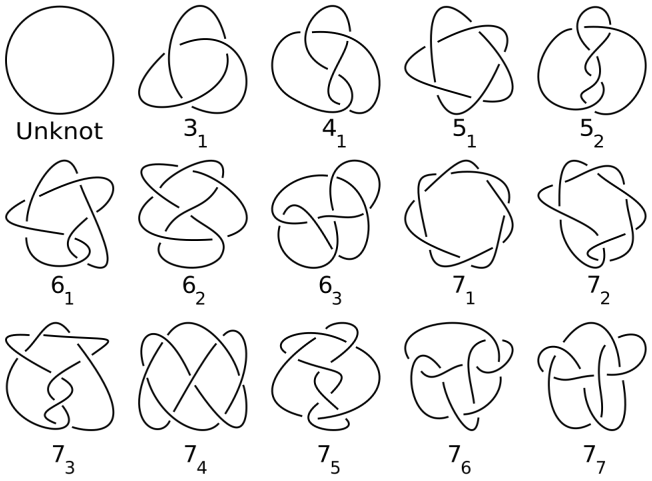


Figure 15: Knot table up to 7 crossings knots

2.3.2 Knot invariants

One of the main question of knot theory is to find a way to identify any two equivalent knots. As of today, such method is still to be found. Two knots that might seem very different can be equivalent as shown in Figure 16. To partially solve the problem, researcher have found knot invariants, that is to say mathematical object computable from any knot representation and that will always be equal for two representation of the same knot. We will be in this section introducing some of these and explain the one we will be using for our research.

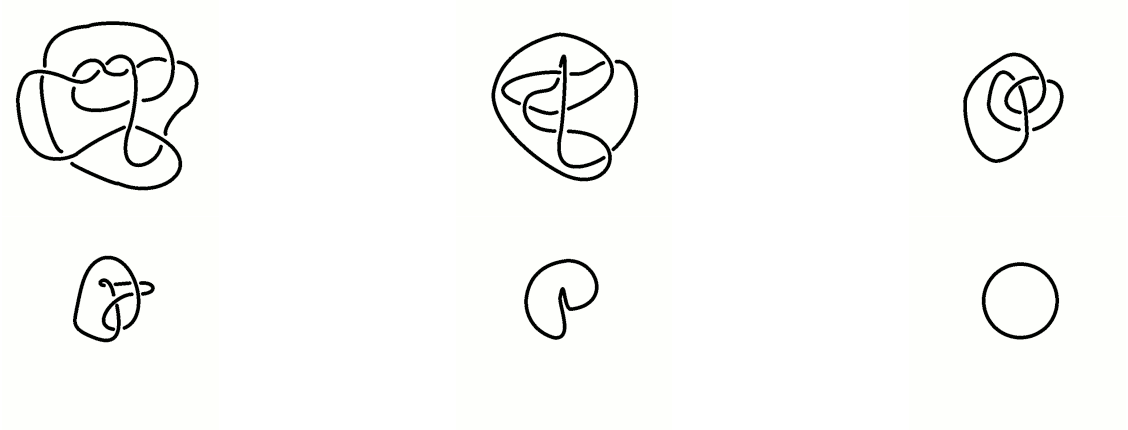


Figure 16: Knot unfolding

Tricolorability (or 3-colorability) Tricolorability is a criterion to color the components of a knot following two rules:

- All components must be colored

- At each crossing all three components are either all of the same color either all of different colors ($\Longleftrightarrow$ one can find only one or the three colors at each crossing).

For instance the trefoil is 3-colorable as shown is Figure 17 unlike the figure-eight knot (Figure 18)

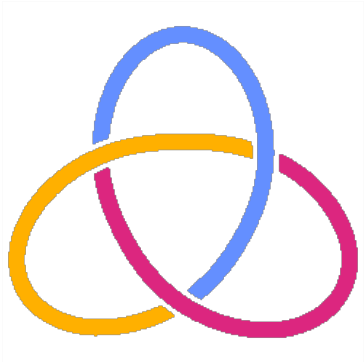


Figure 17: Colored trefoil

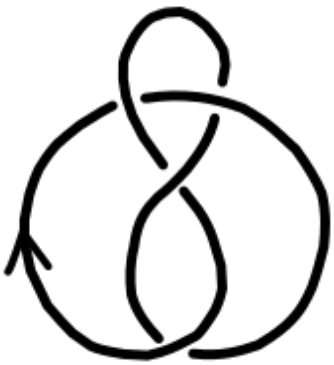


Figure 18: Not-colored figure-eight knot

Proposition 3 *Tricolorabitily is a knot invariant*

Proof:
To prove that 3-colorability is a knot invariant it is sufficient to show that the tricolorability of a knot is preserved with the Reidemeister moves. This task is fairly simple though it might involve to test a lot of cases.

- Type I: Twist. This move is straight forward since starting with only one component, the only starting point is of having a single color, the solution of keeping the same color is obvious.

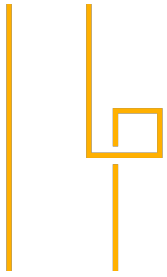


Figure 19: Twist tricolorability

- Type II: Poke. For this move there are two possible configuration as starter since both components can have the same color or a different one. In both case it is possible to find a coloration that fits.

Figure 20: Poke tricolorability

- Type III: Slide. This move could have up to 8 possibilities but the possibilities where exactly one of the crossings is of different colors is not possible since it goes against the rules of tricolorability. We end up with 5 cases that all admit a solution.

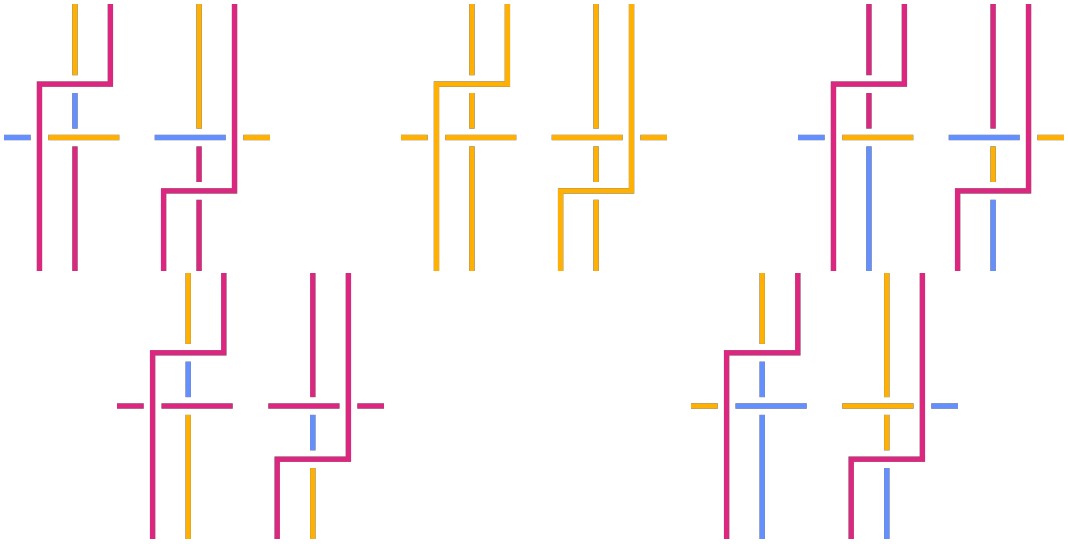


Figure 21: Slide tricolorability

Note that a solution is admissible only if the components that link with the rest of the knot can preserve their color so that the move do not interfere with the rest of the knot.

□

Tricolorability is a useful invariant because it can allow for example to prove that the trefoil (3_1) and the figure-eight knot (4_1) or different knots, however it does not allow to discriminate a lot of knots since for example with tricolorability only, there is no way to know wether the unknot is different from the figure-eight knot. In Figure 22 we see which knots are tricolorable. To be able to discriminate more knots we will introduce another tool, the knot determinant.

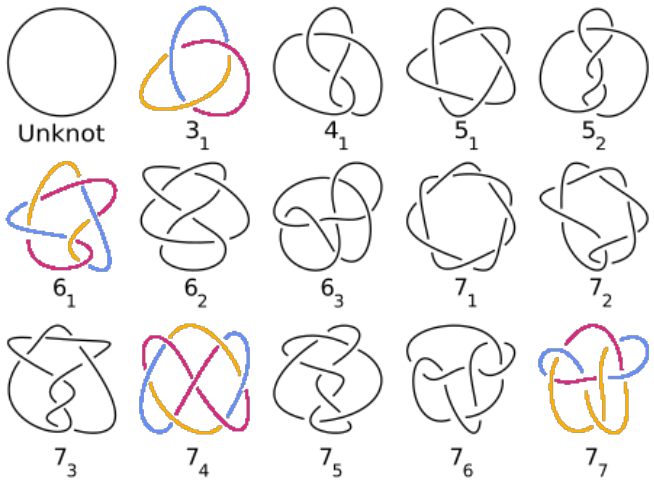


Figure 22: Tricolorability table

Knot determinant To be able to compute knot determinant, one must first number components and crossings of a knot as shown in Figure 23.

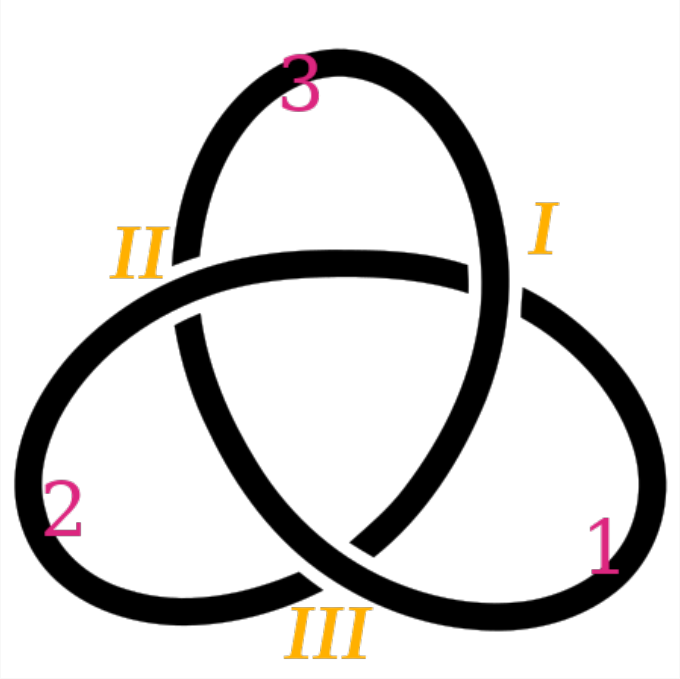


Figure 23: Trefoil with numbering to compute determinant

One can then create a matrix A with coefficient $a_{i,j}$ defined as follows:

$$a_{i,j} = \begin{cases} -1 & \text{if component } i \text{ is one of the lower components involved in crossing } j \\ 2 & \text{if component } i \text{ is the upper component involved in crossing } j \\ 0 & \text{otherwise} \end{cases}$$

For the trefoil in Figure 23 we have:

$$A = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} I & II & III \end{array} \\ \begin{bmatrix} -1 & -1 & 2 \\ -1 & 2 & -1 \\ 2 & -1 & -1 \end{bmatrix} \end{array}$$

The knot determinant is then the determinant of the matrix A with one line and column removed. In the case of the previously obtained matrix, 3.

In the same fashion as the proof done for tricolorability one can show that the determinant is a knot invariant by showing its invariability under the Reidemeister moves.

The results of the computation of knot determinants for the knots up to 7 crossings can be seen in Table 1.

knot	$det(K)$
3_1	3
4_1	5
5_1	5
5_2	7
6_1	9
6_2	11
6_3	13
7_1	7
7_2	11
7_3	13
7_4	15
7_5	17
7_6	19
7_7	21

Table 1: Knot table & determinant

We can see that even though more values are possible it can not allow to discriminate all knots.

Note that the result obtained from the computation of the knot determinant gives the p-colorability of a knot. Extending the concept of 3-coloration by keeping the same rules (1 or 3 colors at each crossing), we notice that a p-coloration is possible iff p is a divisor of the knot determinant.

Alexander polynomial The last knot invariant that we will introduce in this paper is the Alexander polynomial. Its definition is close to the one of the determinant. To compute the Alexander polynomial, one will first construct a matrix A with coefficients $a_{i,j}$ defined as such:

$$a_{i,j} = \begin{cases} t & \text{if component } i \text{ enters (exits) under left (right) handed crossing } j \\ -1 & \text{if component } i \text{ enters (exits) under right (left) handed crossing } j \\ 1-t & \text{if component } i \text{ is overcrossing at crossing } j \\ 0 & \text{otherwise} \end{cases}$$

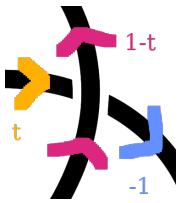


Figure 24: Left crossing

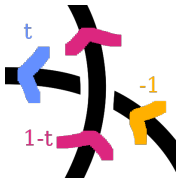


Figure 25: Right crossing

Then in the same manner, one can compute the determinant after removing one line and column of the matrix. For the trefoil we obtain:

$$A = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} I & II & III \end{array} \\ \begin{bmatrix} -1 & t & 1-t \\ t & 1-t & -1 \\ 1-t & -1 & t \end{bmatrix} \end{array}$$

After normalization, we obtain the polynomial: $t-1+t^{-1}$. The polynomial obtained by this method can differ from a factor $\pm t^n, n \in \mathbb{N}$. To solve that issue we normalize it so that it looks symetric (Laurent polynomial) and denote if $\Delta_k(t)$. We can with that method write a table of the Alexander polynomial as seen in Table 2 for the knots up to 7 crossings.

knot	$\Delta_k(t)$
3_1	$t - 1 + t^{-1}$
4_1	$-t + 3 - t^{-1}$
5_1	$t^2 - t + t - t^{-1} + t^{-2}$
5_2	$2t - 3 + 2t^{-1}$
6_1	$-2t + 5 - 2t^{-1}$
6_2	$-t^2 + 3t - 3 + 3t^{-1} - t^{-2}$
6_3	$t^2 - 3t + 5 - 3t^{-1} + t^{-2}$
7_1	$t^3 - t^2 + t - 1 + t^{-1} - t^{-2} + t^{-3}$
7_2	$3t - 5 + 3t^{-1}$
7_3	$2t^2 - 3t + 3 - 3t^{-1} + 2t^{-2}$
7_4	$4t - 7 + 4t^{-1}$
7_5	$2t^2 - 4t + 5 - 4t^{-1} + 2t^{-2}$
7_6	$-t^2 + 5t - 7 + 5t^{-1} - t^{-2}$
7_7	$t^2 - 5t + 9 - 5t^{-1} + t^{-2}$

Table 2: Knot table and Alexander polynomial

With the Alexander polynomial one can discrimainate any two knots with up to seven crossings. Though the Alexander polynomial can not discriminate any two knots as explained in the introduction. An interesting property of the Alexander polynomial is that $\Delta_k(-1) = det(K)$ (the knot determinant).

3 Previous research on quantification of neural network expressiveness

In this section we will explain how the topic of quantifying neural network expressiveness has been tackled by researchers and introduce the different research articles on which our work relies.

The concept of measuring neural network expressiveness induces to use some concept as a measure of difficulty of expression. In this section we will see how some previously introduced concepts can be used as measures: betti numbers and circle embeddings.

3.1 Using topological data analysis

In this section we will focus on the use of betti as a measure for neural netwok expresiveness.

In [13], Bianchini and Scarselli have offered proofs on the number of neurons required in different architecture of neural network to properly express a dataset betti numbers. They have worked on determining upper bounds and lower bounds on the number of neurons required in network of a certain depth to be able to express the homology represented by its sum of betti numbers with different activation functions. Their result are summed up in Table 3 that can also be found in [13].

Let X be a topological space. We denote $\mathcal{B}(X) = \sum_i \mathcal{B}_i(X)$ where $\mathcal{B}_i(X)$ is the number of i-dimensional homology in X.

Inputs	Layers	Activation function	Bound
Upper Bounds			
n	3	threshold	$O(h^n)$
n	3	arctan	$O((n+h)^{n+2})$
n	3	polynomial of degree r	$\frac{1}{2}(2+r)(1+r)^{n-1}$
1	3	arctan	h
n	any	arctan	$2^{h(2h-1)}O((nl+n)^{n+2h})$
n	any	tanh	$2^{(h(h-1))/2}O((nl+n)^{n+h})$
n	any	polynomial of degree r	$\frac{1}{2}(2+r^l)(1+r^l)^{n-1}$
Lower bounds			
n	3	any sigmoid	$(\frac{h-1}{n})^n$
n	any	any sigmoid	2^{l-1}
n	any	polynomial of degree $r \geq 2$	2^{l-1}

Table 3: Upper bounds and lowers bounds on the growth of $\mathcal{B}(X)$ for networks with h neurons, n input and l hidden layers

To prove these bounds they use the function sum representation of neural networks as introduced in **Section 1** as well as classical function analysis inequalities.

In [14], Guss and Salakhutdinov relying on [13] proposed a statistical approach to the problem of giving a value to the way each Betti numbers would affect the topological expressivity of the neural network since [13] only relies on the sum $\mathcal{B}(X)$.

To study the difficulty of learning homologically complex data, they considered fully-connected ReLU neural networks of depth $d \in [1...6]$ and width $h_l = \beta_0(X)$ when $1 \leq l \leq d$ and $h_l \in [1...500]$ when $l = 0$. It sums up with a first layer of variable width (up to 500) and the other layers width being fixed at the number of connected components (0-homology) of X. These networks would then be trained using standard cross-entropy loss with Adam optimizer, a fixed learning rate of 0.01 and an increasing batch size schedule.

They used datasets composed of two classes of 5000 points one of randomly distributed points the other one of mixtures of the uniform distribution on $\mathcal{S}^1$ and $\mathcal{B}^2$. They limited their study to $n = 2$. They studied the output for datasets continuously ranging from $H(X) = \{\mathbb{Z}^1, 0\}$ to $H(X) = \{\mathbb{Z}^{30}, \mathbb{Z}^{30}\}$. They trained 100 of the previously described networks on each dataset and compiled the results in Figure 26.

In Figure 26 we can see two distincts parts. The upper part refers to the characterization of $H_0(X)$. Each of the four plots is representing the probability of expressing $H_0(X)$ depending on $\mathcal{B}_0(X)$ and the number of hidden units on the fist layer of the network (the other layers having a number of hidden units fixed to $B_0(X)$). On these plots the number of hidden units ranges from 0 to 14 and $\mathcal{B}_0(X)$ from 3 to 15.

The second plot presents the same results for the proper characterization of $\mathcal{B}_1(X)$ with this time $\mathcal{B}_1(X) \in \{1, 2\}$ and hidden units ranging from 0 to 14.

We can see some clear tendency (that would be expected) that expression of $H_n(X)$ is improving with the reduction of $\mathcal{B}_n(X)$, and the increase of l and d. The interest of such result lies in the fact of giving a possible approximation of the evolution of the probability depending on each of these parameters.

Unfortunately the authors did not provide the full results and code of their research and did not answer my requests to obtain them.

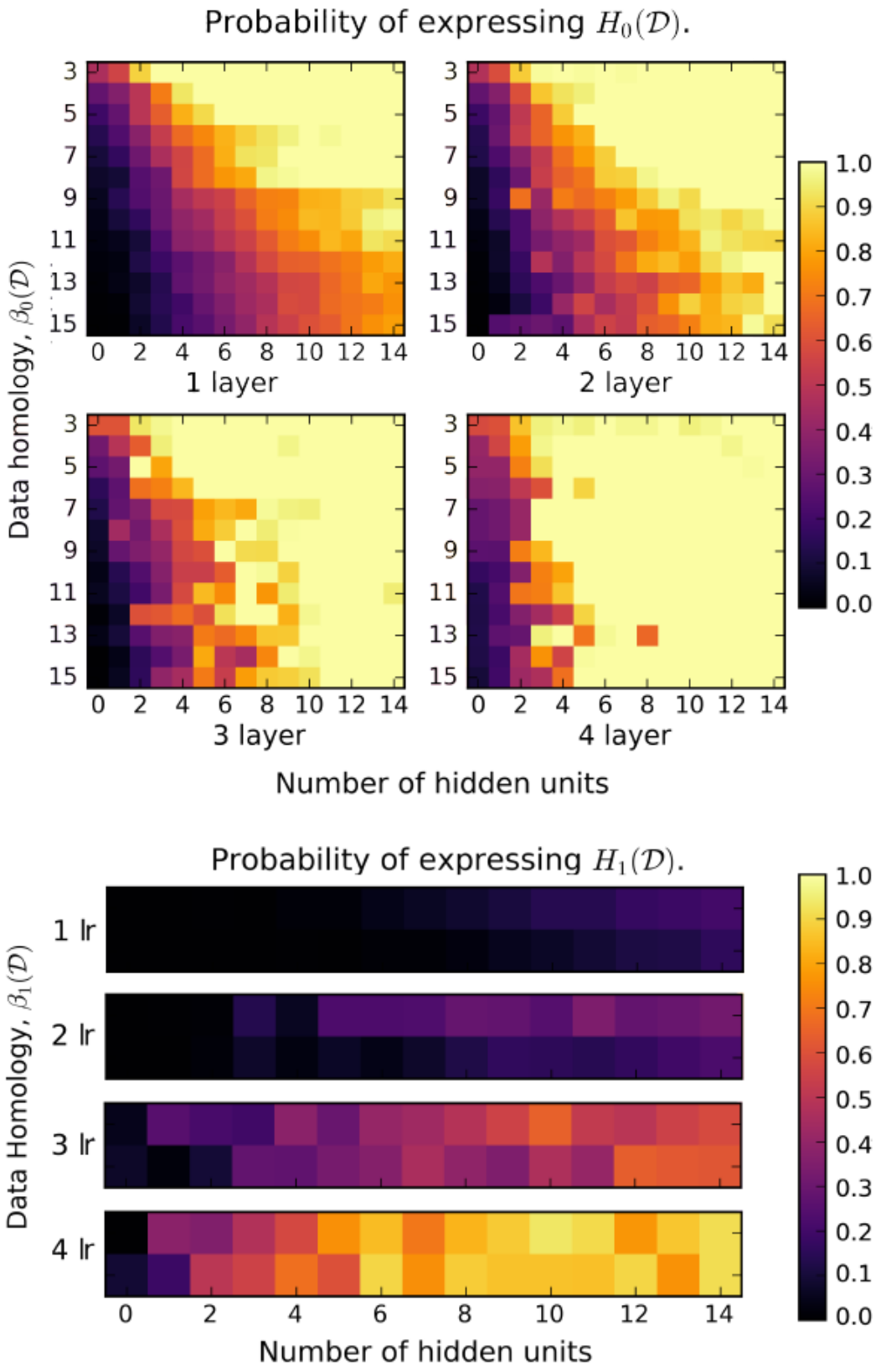


Figure 26: Results on the characterization of homologies with different neural networks architecture

3.2 Using trajectories

The idea of studying trajectories to evaluate neural network expressivity comes from [15] itself relying on [16] and [17].

In [15], Raghu introduces various variables of interest for neural network expressivity relying on the notion of transition of activation patterns. Activation patters denotes all the way threshold-based activation functions such as ReLU and hard-tanh can activate or not at the same time. For a ReLU function we would consider it active if in the non-constant part and non-active if in the constant part. Activation patterns is a concept at network scale to summarize the existence of inputs and activation function parameters leading to different activation and non-activation of each neuron of the network.

To study transition Raghu introduces the unifying concept of trajectories. Trajectories are defined as follows:

Definition 5 *Given two points $x_0, x_1 \in \mathbb{R}^n$, $x(t)$ between x_0 and x_1 is a trajectory if there exists $f : [0,1] \to \mathbb{R}^n$ with $f(0) = 0_{\mathbb{R}^n}$ and $f(1) = 1_{\mathbb{R}^n}$ such that $x(t) = (x_0 - x_1)f(t) + x_0$.*

This can include simple examples such as a line with $f(t) = t$ or a circular arc with $f(t) = sin(\frac{\pi t}{2})$ but in general $x(t)$ might be more complicated. This definition is obviously linked to the concept of path in topology and therefore includes loops.

For piecewise linear activation functions such as ReLU and hard tanh, the function computed by the network is also piecewise linear since the linear sum of composition of piecewise linear functions is itself piecewise linear.

Definition 6 *We say a neuron with piecewise linear activation function transitions between input x and $x + \delta$ if its activation function switches between x and $x + \delta$*

With all the previously introduced concepts we can now give a proper definition to the activation pattern:

Definition 7 *The activation pattern $AP(F_A(x, W))$ of network with weights W, architecture A and input x is a string of form $\{0,1\}^{number\ of\ neurons}$ for ReLU networks and $\{-1,0,1\}^{number\ of\ neurons}$ for hard tanh networks coding the linear region of the activation function of every neuron.*

We can aswell define $T(F_A(x(t), W))$ the number of transition undergone by the network when x sweeps along $x(t)$.

In [17], Montufar et al. provide a bound on the number of transition by constructing a set of weights W_0 resulting in an exponential increase of the number of transition with the depth of the network i.e.

$$\forall W,\ T(F_{A_1}([0,1],W)) < T(F_{A_1}([0,1],W_0))$$

with A_1 a fully connected network with one hidden layer.

We denote $U(n,k,m)$ the upper bound on the number of activation patterns for a fully connected network with n hidden layers of width k and inputs in $\mathbb{R}^m$. Raghu shows that for a ReLU network we have $U(n,k,m) = O(k^{nm})$ and for hard tanh $U(n,k,m) = O((2k)^{nm})$.

Going back to the definition of trajectory, we define the trajectory length:

Definition 8 *For a trajectory $x(t)$, we define $l(x(t))$ its length to be the standard arc length:*

$$l(x(t)) = \int_t \|\frac{dx(t)}{dt}\|$$

Raghu then show that length grows exponentially with length in most cases. Formally:

Theorem 2 *Let $F_A(x,W)$ be a ReLU or hard tanh random neural network and $x(t)$ a one-dimensional trajectory with $x(t+\delta)$ having a non trivial perpendicular component to $x(t)$ for all t,δ. Then with $z^{(d)}(x(t)) = z^{(d)}(t)$ the image of the trajectory in layer d of the network we have:*

$$\mathbb{E}[l(z^{(d)}(t))] \geq O(\frac{\sigma_w\sqrt{k}}{\sqrt{k+1}})^d l(x(t)) \text{ for ReLU networks}$$
$$\mathbb{E}[l(z^{(d)}(t))] \geq O(\frac{\sigma_w\sqrt{k}}{\sqrt{\sigma_w^2+\sigma_b^2+k\sqrt{\sigma_w^2+\sigma_b^2}}})^d l(x(t)) \text{ for hard tanh}$$

with σ_w the standard deviation of weights, σ_b the standard deviation of bias and k the width of the network.

Therefore, the length of the trajectory grows exponentially with depth and hence expressivity. A figure of that growth can be seen in Figure 27.

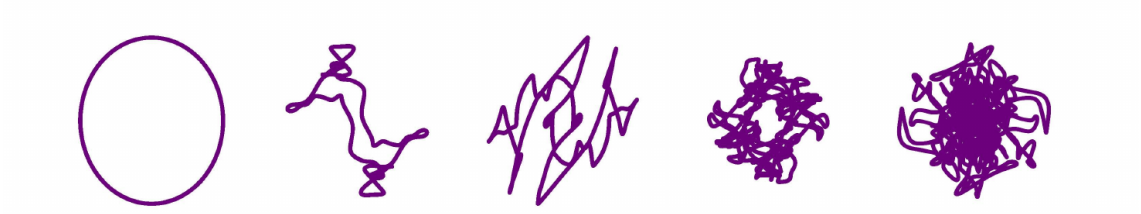


Figure 27: Projected circular trajectory at increasing layers in a width 100 neural network

4 Experiments

4.1 Studying trajectories from a knot theory perspective

In this section we will detail the work on trajectories inspired by the research detailed in **Section 3.2** where we study the trajectories previously introduced from a knot theory point of view.

4.1.1 Methodology

For the purpose of studying trajectories as knot we limit the definition of trajectories to the case where the beginning point is the end point ($x_0 = x_1$ in **Definition 5**). In that case the trajectory becomes an embedding of the circle. We also limit our study to embeddings of S^1 in $\mathbb{R}^n, n \geq 3$ to ensure the existence of the knot.

Knots can exist in dimension higher than 3 by simply living in a 3-dimensional hyperplane. We create trajectories corresponding to the first knots in increasing number of crossings, namely: the unknot, the trefoil 3_1 and the figure-eight knot 4_1.

Computationnaly trajectories are arrays of points uniformly distributed along the studied embedding. They are obtained from a B-spline of hand-picked points for each different trajectory.

Definition 9 *A spline of order n is a piecewise polynomial function of degree $n-1$. The values where the polynomial pieces meet are called knots and denoting them $t_0, t_1, ..., t_n$. When the knots are distincts the $n-2$ derivatives of the polynomials are continuous across each knot. When r knots are coincident, only the first $n-r-1$ derivatives are continuous.*

For some $t_0, t_1, ..., t_n$ there is up to a scaling factor a unique spline $B_{i,n}(x)$ such that

$$B_{i,n} = \begin{cases} 0 & \text{if } x < t_i \text{ or } x \geq t_{i+n} \\ \text{nonzero} & \text{otherwise} \end{cases}$$

Adding the constraint $\sum_i B_{i,n}(x) = 1$ for all x between the first and last knot, then we obtain unicity and $B_{i,n}(x)$ is called B-spline. Figuratively B-spline allow to create a smooth curve passing through a set of different points (called knots). When some points of a B-spline repeats the smoothness is lost at those points. That last property will be important since embeddings of the circle admit the same first and last point and its B-spline representation the same first and last knot. Figure 28 shows the B-spline for the trefoil and the figure-eight knot.

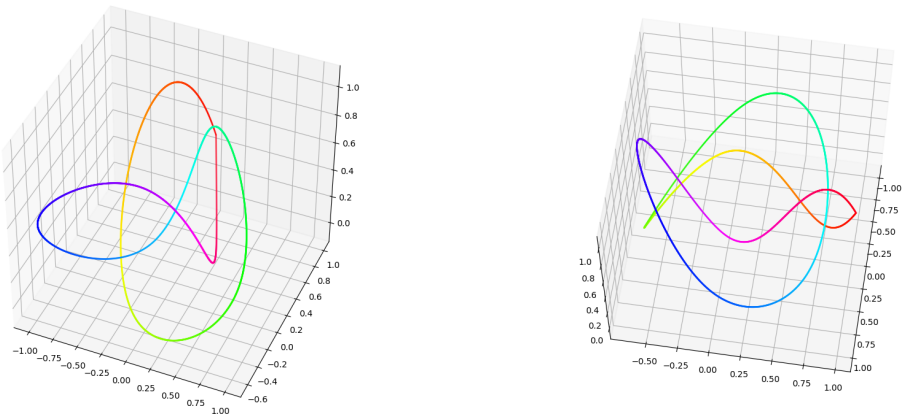


Figure 28: B-spline for 3_1 and 4_1 in $\mathbb{R}^3$

The points used to generate the B-spline of 3_1 and 4_1 in $\mathbb{R}^3$ are represented in Figure 29

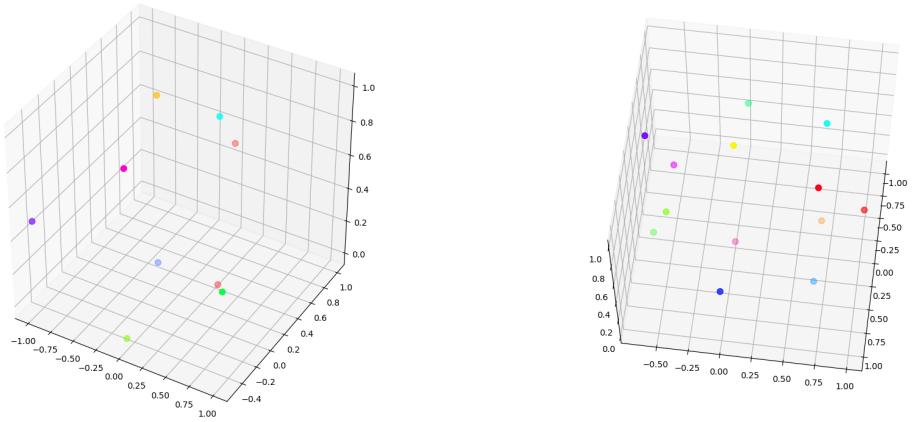


Figure 29: Points for the computation of the B-spline of 3_1 and 4_1 in $\mathbb{R}^3$

The aim is then to feed the trajectories to ReLU neural networks and to study the evolution of the different trajectories through the layers of the network. We proceed by feeding the trajectories to neural networks of depth ranging from one to eight hidden layers of width eight. The input dimension of the networks are either 3 or 4. At each layer we study the trajectory by taking the PCA in dimension 3 of the trajectory points after passing through each of the network layer. We then identify the obtained trajectory to a knot.

4.1.2 Algorithms

In this section we introduce the different algorithm implemented for the research described in this section. All the algorithm have been implemented in python and can be found in the code available on Gitlab `https://gitlab.com/Spiilgriim/nnexpy` and the file and line to look for the python file including the implementation can be found in the algorithm title.

We write N the number of points in the trajectory and T the array containing the points of the trajectory, $\forall i \in [1 \dots N]$, T_i denotes the point at index i in T.

Algorithm 1 Trajectory to Knot (`nnexpy/trajectories.py l.24`)

crossings $\leftarrow []$
components $\leftarrow []$
for $i = 1$ to $N - 1$ **do**
  for $j = i + 1$ to N **do**
    if $Overlap((T_i, T_{i+1}), (T_j, T_{j+1}))$ **then**
      add i in *crossings*
      if (T_j, T_{j+1}) is above **then**
        add $(components_{-1,2}, i)$ in *components*
      end if
    end if
  end for
end for
add $(components_{-1,2} + 1, components_{1,1})$ to *components*

This algorithm is a simplified version of the one implemented since we implemented the computation of the orientation of the knot (right-hand knot or left-hand knot) but we will not be using this feature in the rest of the section. An implementation for `Overlap` and the other computational geometry functions used to compute the orientation of the knot and determine which piece is over the other can be found in `nnexpy/trajectories_utils.py l.139` $\rightarrow$ `l.175`

This algorithm is valid because it relies on the fact that topologically any representation (in our case projection hyperplane) of a knot is the same knot and the different knot determinant we could use would have the same value whatever the representation used. In our case we then chose to project the knot on the two dimensional hyperplane with base $\{(1, 0, 0), (0, 1, 0)\}$ in dimension 3 and $\{(1, 0, 0, 0), (0, 1, 0, 0)\}$ in dimension 4.

We denote Cr of length L_{Cr} the crossings array and Co of length L_{Co} the components array obtained from Algorithm 1. $0_{n,m}$ represents the matrix of size $n \times m$ with only null coefficient. For a matrix M of size $n \times m$, $M_{\widehat{i,j}}$ represents the comatrix of M with line i and row j removed ($1 \leq i \leq n$ and $1 \leq j \leq m$).

Algorithm 2 Knot determinant (`trajectories.py l.234`)

if $L_{Cr} = 0$ **or** $L_{Co} = 1$ **then**
  return 1
end if
$M \leftarrow 0_{L_{Cr}, L_{Co}}$
for $i = 1$ to L_{Cr} **do**
  $c1 \leftarrow$ upper component of Cr_i
  $c2, c3 \leftarrow$ lower component of Cr_i
  $M_{i,c1} \leftarrow 2$
  $M_{i,c2} \leftarrow -1$
  $M_{i,c3} \leftarrow -1$
end for
$M \leftarrow M_{\widehat{L_{Cr}}, \widehat{L_{Co}}}$
return $\det(M)$

We then use the determinant to discriminate knots. As explained in **Section 2.3.2**, it is not sufficient to discriminate all knots but it turns out to be enough for our purposes.

4.1.3 Results

The result of the experiment described in this section are summed up in Figure 30.

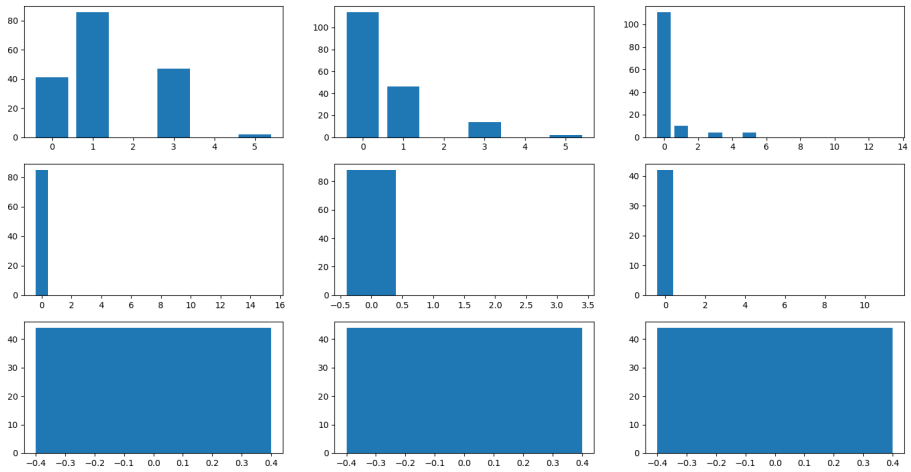


Figure 30: Result of the experiment on knots with trefoil input

Figure 30 represents the evolution of the knot determinant through the layers of networks. Each of the subgraph represents a layer from 1 on the top left to 9 on the bottom right. On each of the subgraph the bar represents the absolute number of occurences of a determinant for instance at layer 1 we obtained about 40 knots

with determinant 0, 85 with determinant 1, 45 with determinant 3 and 5 with determinant 5.

Note that using our alorithm unknots have determinant 0 if they are entangled unknots (more than 1 component) and 1 if they are trivial unknots (1 component and 0 crossing).

4.1.4 Considerations

We can see that as layer index increases, the knots become entangled unknots. An interesting following to this research would be to see if it is possible to create a neural network or set of weights that result in a different knot that an entangled unknot.

4.2 Extending the study of expressiveness with topological data analysis

In this section, we detail the work on using homology for neural entwork expressiveness inspired by the research detailed in **Section 3.1** where we try to extend the experiment introduced by Guss in [14] to understand quantitatively the influence of a neural network architecture on it capacity to characterizing homologies in different dimension.

4.2.1 Methodology

Definition 10 *Let (X, D) be a metric space with $D : X^2 \to \mathbb{R}^+$. Let $c \in X$ and $r \in \mathbb{R}$.*

$\mathcal{B}(c, r)$ is the topological ball of radius $r > 0$ centered on x:

$$\mathcal{B}(c, r) = \{x \in X \mid D(x, c) \leq r\}$$

$\mathcal{S}(c, r)$ is the topological sphere of radius $r > 0$ centered on x:

$$\mathcal{S}(c, r) = \{x \in X \mid D(x, c) = r\}$$

$\mathfrak{S}(c, [r_1, r_2])$ is the thick sphere of border ranging from r_1 to $r_2 \neq 0$ ($0 \leq r1 \leq r_2$) centered on x:

$$\mathfrak{S}(c, [r_1, r_2]) = \{x \in X \mid r1 \leq D(x, c) \leq r_2\}$$

Note that a ball is a particular case of thick sphere where $r_1 = 0$ and spheres are thick spheres with $r_1 = r_2$. Thick sphere is unlike spheres and balls not a rigorous term of topology since it is always equivalent to a sphere or a ball but it is the concept we will use to informatically modelize our balls and spheres.

Let $n \in \mathbb{N}^*$, we will consider $X = \mathbb{R}^n$ containing N_B balls $\mathcal{B}_i, 1 \leq i \leq nB$ and N_S spheres or thick sphere $\mathcal{S}_j, 1 \leq j \leq N_S$ (cf Definition 10) of dimension at most n. We informatically modelize this set in the following manner:

We used datasets of 50000 points split in two non-intersecting classes labelled "0" and "1". The points of class "1" are uniformly spread across various topological spheres and balls. In our case uniformly will mean that the points are uniformly distributed across the volume of the topological feature and that the number of points in each topological feature is proportional to its importance in the sum of volumes of topological features in our set of interest.

The points of class "0" are distributed uniformly in the space $X \backslash \{\bigcup_{i=1}^{N_B} \mathcal{B}_i \bigcup \bigcup_{j=1}^{N_S} \mathcal{S}_i\}$.

For the result that we will present in **Section 4.2.3**. We trained network on twelve different sets of dimension 2, 3 and 4 with one to three topological features. On each of these sets we trained 20 sets of fully-connected networks of width 8 and depth 1,2,4,6,8,10,12,14,16 with random initialization for 2000 epochs. To train the networks we used the sparse categorical entropy as loss function, a batch size of 10 and the Adam optimizer with 80% of the data used as training data and 20% as validation data. We used Keras [18] with a tensorflow backend in python3.

4.2.2 Algorithms

In this section we introduce the different algorithm implemented for the research described in this section. All the algorithm have been implemented in python and can be found in the code available on Gitlab `https://gitlab.com/Spiilgriim/nnexpy` and the file and line to look for the python file including the implementation can be found in the algorithm title.

First we will detail the algorithm to generate our datasets consisting of balls and spheres. Denote $n \geq 2$ the dimension of the space,

$$B = \{(\mathcal{B}(c_i, r_i), d_i, o_i)\}_{1 \leq i \leq N_B}$$

the set of balls in X as well as their dimension d_i and orientation o_i. Finally

$$S = \{(\mathfrak{S}(c_j, [r_{1_j}, r_{2_j}]), d_j, o_i)\}_{1 \leq j \leq N_S}$$

the set of thick sphere and P the number of points in each class. The orientation of balls and spheres is an array of that indicates the index of the elements of the cannonical basis constituing the hyperplane of $\mathbb{R}^n$ in which the feature does not live. For instance in $\mathbb{R}^3$ writing the cannonical basis $\{(1,0,0),(0,1,0),(0,0,1)\}$, $(\mathcal{B}(0,1),2,[1])$ is a 2-dimensional ball centered at the point $(0,0,0)$ of radius 1 and living in the plane spanned by $\{(0,1,0),(0,0,1)\}$. In the python implementation some of the parameters might be decided randomly or with the help of other parameters.

In the algorithm z will refer to a random number between 0 and 1.

Algorithm 3 Generation of topological features
(`nnexpy/homology_data_generation.py l.191`)

points $\leftarrow$ []
$X \leftarrow S$
for $\mathcal{B}_i \in B$ **do**
  $X \leftarrow X :: (\mathfrak{S}(c_i, [0, r_i]), d_i, o_i)$
end for
sum $\leftarrow 0$
for $i = 1$ to $N_B + N_S$ **do**
  sum $\leftarrow$ sum $+ {r_{i_1}}^{d_i} - {r_{i_2}}^{d_i}$
end for
for $i = 1$ to $N_B + N_S$ **do**
  for $j = 1$ to $\lfloor \frac{{r_{i_1}}^{d_i} - {r_{i_2}}^{d_i}}{sum} \rfloor$ **do**
    temp $\leftarrow 0_{\mathbb{R}^n}$
    radius $\leftarrow (r_{i_1} * z_1 + r_{i_2} * (1 - z_1))$
    for $e = 1$ to n **do**
      if $e \notin o_i$ **then**
        $\text{temp}_i \leftarrow (2 * z_2 - 1)$
      end if
    end for
    points $\leftarrow$ points:: $\frac{\text{radius}}{\sqrt{\sum_{i=1}^{n} \text{temp}_i}} \times$ temp
  end for
end for

We have also designed an algorithm to compute betti numbers for spheres and balls that is a more efficient than persistent homology memory-wise due to the limitation to simple topological features. We write $T \in \mathbb{R}^+$ the threshold chosen to compute betti numbers, it corresponds to the radius up to which imaginary balls around each point grow in the persistent homology algorithm. We denote G a graph, $C(G)$ the list of connected components of G, $M(X)$ the matrix corresponding to the concatenation of the points of the set X and we extend the definition of $D : X \times A \to \mathbb{R}$ with A a subset of X as $D(x, A) = \min_{y \in A} D(x, y)$.

Algorithm 4 Custom persistent homology (`nnexpy/homology_data_generation.py l.414`)

betti $\leftarrow \underbrace{[0...0]}_{n \text{ times}}$
G_node $\leftarrow$ points
G_edge $\leftarrow$ []
for $i, j \in$ points **do**
  if $d(i, j) \leq T$ **then**
    G_edge $\leftarrow G$_edge :: (i, j)
  end if
end for
for $\mathcal{C} \in C(G)$ **do**
  compute barycenter B of $\mathcal{C}$
  if $D(\mathcal{C}, B) \leq T$ **then**
    $\text{betti}_0 \leftarrow \text{betti}_0 + 1$
  else
    $\text{betti}_{rank(\mathcal{C})-1} \leftarrow \text{betti}_{rank(\mathcal{C})-1} + 1$
  end if
end for

4.2.3 Results

In Table 4 and Table 5, one can see the results obtained for the experiments described in this section.

depth \ betti	[3,0]	[1,1]	[1,0]	[0,0,1]	[0,1,0]	[1,1,0]	[1,0,1]	[0,0,0,1]	[0,0,1,0]	[1,0,1,0]	[1,0,0,1]
1	13	12	11	15	13	4	2	19	20	13	9
2	18	19	2	19	15	14	12	17	19	18	6
4	18	19	5	17	12	12	14	17	20	19	13
6	19	17	1	19	15	14	13	18	20	20	14
8	17	15	2	18	15	14	19	17	20	20	15
10	15	13	1	19	13	14	18	18	19	18	10
12	15	10	2	16	9	13	17	16	20	17	13
14	16	9	1	14	7	12	15	16	17	14	10
16	10	6	1	7	6	12	10	13	12	13	8

Table 4: The number of occurences of the same betti numbers between the original data persistence diagram and the networks data persistence diagram

depth \ betti	[3,0]	[1,1]	[1,0]	[0,0,1]	[0,1,0]	[1,1,0]	[1,0,1]	[0,0,0,1]	[0,0,1,0]	[1,0,1,0]	[1,0,0,1]
1	0.077	0.088	0.033	0.035	0.038	0.137	0.137	0.025	0.031	0.198	0.181
2	0.033	0.058	0.016	0.021	0.035	0.046	0.079	0.025	0.027	0.110	0.125
4	0.023	0.019	0.011	0.019	0.043	0.050	0.034	0.025	0.029	0.051	0.054
6	0.013	0.012	0.006	0.020	0.040	0.035	0.024	0.024	0.024	0.035	0.059
8	0.025	0.011	0.010	0.017	0.042	0.041	0.019	0.028	0.028	0.050	0.041
10	0.028	0.016	0.006	0.019	0.039	0.036	0.024	0.028	0.026	0.056	0.047
12	0.031	0.009	0.007	0.018	0.043	0.041	0.023	0.026	0.032	0.051	0.071
14	0.023	0.010	0.007	0.019	0.041	0.042	0.021	0.023	0.027	0.040	0.067
16	0.025	0.020	0.012	0.020	0.044	0.041	0.024	0.023	0.034	0.055	0.046

Table 5: The average bottleneck distance between the original data persistence diagram and the networks data persistence diagram

4.2.4 Considerations

In table 4 we can see a tendency for the number of proper characterizations to first go up since the network gets deeper and more expressive as expected and then down. The drop in the number of proper characterizations is due to the difficulty to train big networks since they would require more epochs than used but we could not for time reasons allocate more time to the network training considering how long the process is. Another problem we faced in the case where the topological feature only live in a strict subspace of X is that we might see the appearance of training artifacts in the region where there should be no points. This causes the computation of betti numbers to return a result different from the original one because it changes the dimension in which the feature lives in.

Finally another problem with this experiment is the time required to train enough networks and train them properly. To reproduce Guss' experiment described in **Section 3.1** with only 2000 epoch of training on an NVIDIA RTX 2080 and 16 GB of RAM we would need approximately 50 days of uninterrupted computation. The results obtained with the various training are yet insufficent to draw any significant conclusion but with the availability of the code and its documentation should allow anyone with enough computation power to run statistically significant experiments and draw the numeric conclusion.

References

[1] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals and Systems*, vol. 2, pp. 303–314, Dec. 1989.

[2] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, pp. 251–257, Jan. 1991.

[3] R. Arora, A. Basu, P. Mianjy, and A. Mukherjee, "Understanding Deep Neural Networks with Rectified Linear Units," *arXiv:1611.01491 [cond-mat, stat]*, Feb. 2018. arXiv: 1611.01491.

[4] S. Wang and X. Sun, "Generalization of hinging hyperplanes," *IEEE Transactions on Information Theory*, vol. 51, pp. 4425–4431, Dec. 2005.

[5] B. Hanin and M. Sellke, "Approximating Continuous Functions by ReLU Nets of Minimal Width," *arXiv:1710.11278 [cs, math, stat]*, Mar. 2018. arXiv: 1710.11278.

[6] A. Hatcher, *Algebraic topology*. Cambridge ; New York: Cambridge University Press, 2002.

[7] P. Albin, "1. History of Algebraic Topology; Homotopy Equivalence - Pierre Albin," 2018.

[8] P. Simard, D. Steinkraus, and J. Platt, "Best practices for convolutional neural networks applied to visual document analysis," in *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.*, vol. 1, (Edinburgh, UK), pp. 958–963, IEEE Comput. Soc, 2003.

[9] L. Wasserman, "Topological Data Analysis," *arXiv:1609.08227 [stat]*, Sept. 2016. arXiv: 1609.08227.

[10] A. Markov, "Insolubility of the Problem of Homeomorphy," 2001.

[11] Anthony Bosman, "Knot Theory 1: Coloring," Jan. 2019.

[12] K. Reidemeister, "Elementare Begründung der Knotentheorie," *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, vol. 5, pp. 24–32, Dec. 1927.

[13] M. Bianchini and F. Scarselli, "On the Complexity of Neural Network Classifiers: A Comparison Between Shallow and Deep Architectures," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, pp. 1553–1565, Aug. 2014. Conference Name: IEEE Transactions on Neural Networks and Learning Systems.

[14] W. H. Guss and R. Salakhutdinov, "On Characterizing the Capacity of Neural Networks using Algebraic Topology," *arXiv:1802.04443 [cs, math, stat]*, Feb. 2018. arXiv: 1802.04443.

[15] M. Raghu, B. Poole, J. Kleinberg, S. Ganguli, and J. Sohl-Dickstein, "On the Expressive Power of Deep Neural Networks," *arXiv:1606.05336 [cs, stat]*, June 2017. arXiv: 1606.05336.

[16] B. Poole, S. Lahiri, M. Raghu, J. Sohl-Dickstein, and S. Ganguli, "Exponential expressivity in deep neural networks through transient chaos," *arXiv:1606.05340 [cond-mat, stat]*, June 2016. arXiv: 1606.05340.

[17] R. Pascanu, G. Montufar, and Y. Bengio, "On the number of response regions of deep feed forward networks with piece-wise linear activations," *arXiv:1312.6098 [cs]*, Feb. 2014. arXiv: 1312.6098.

[18] F. Chollet *et al.*, "Keras." `https://keras.io`, 2015.

Acknowledgements

Thanks to my family and friends for their constant support through my research

Merci à ma famille et mes amis pour leur soutient sans faille pendant mes recherches

私の研究を通して絶え間ない支援をしてくれた家族や友人に心から感謝します

5 Appendix

5.1 Figures credit

Figure 31: By Josef Steppan - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=64810040

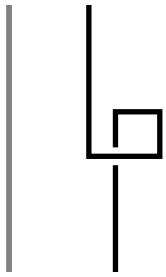


Figure 32: Parcly Taxel- Own work, FAL, https://commons.wikimedia.org/w/index.php?curid=68523335

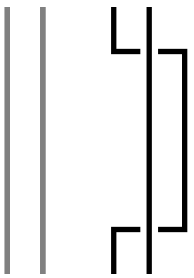


Figure 33: By Parcly Taxel- Own work, FAL, https://commons.wikimedia.org/w/index.php?curid=68523334

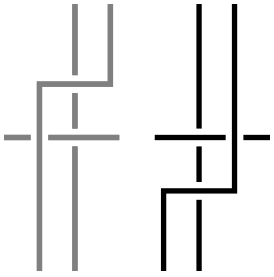


Figure 34: By Parcly Taxel- Own work, FAL, https://commons.wikimedia.org/w/index.php?curid=68523336

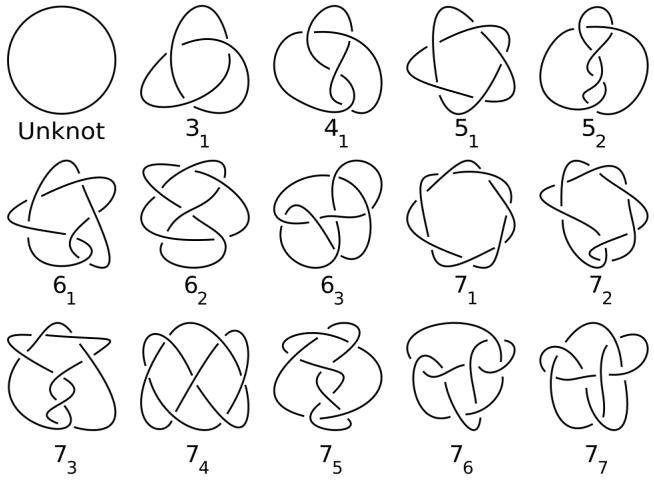


Figure 35: By Jkasd - Own work, Public Domain,
https://commons.wikimedia.org/w/index.php?curid=3937733

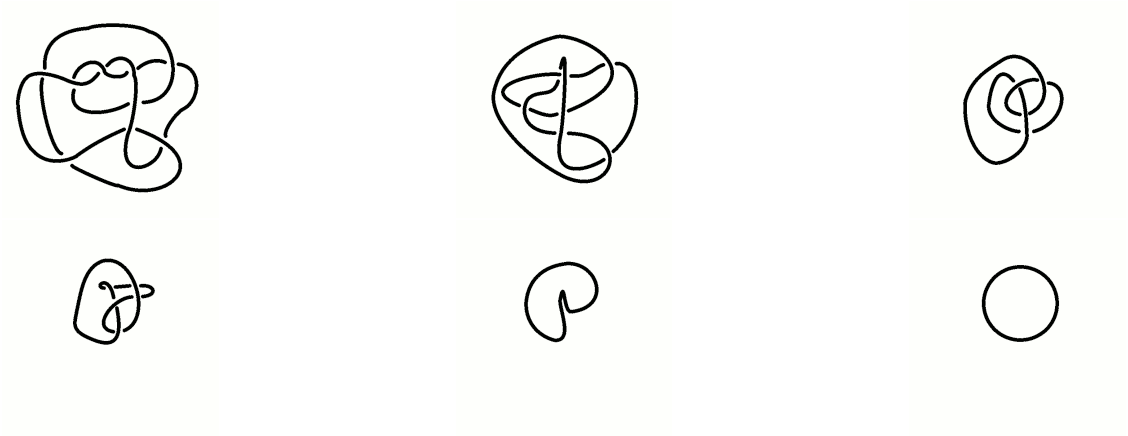


Figure 36: By Kuchtact - Own work, CC BY-SA 4.0,
https://commons.wikimedia.org/w/index.php?curid=77971069

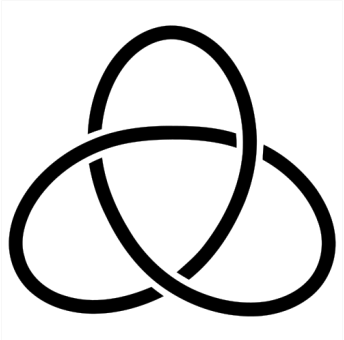


Figure 37: By Marnanel - Image:TrefoilKnot-01.png, Public Domain,
https://commons.wikimedia.org/w/ index.php?curid=3246080

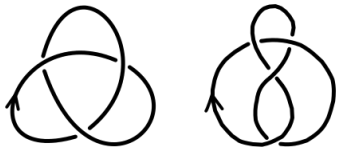


Figure 38: By Talifero - File:Knots eight shamrock.svg, Public Domain, https://commons.m.wikimedia.org/wiki/File:Knots eight shamrock.svg