UNIVERSITAT DE BARCELONA

Facultat de Matemàtiques
i Informàtica

Simultaneous Degrees of Mathematics and Computer Science

End-of-Degree Project

Enhancing Computer Vision Models Using Persistent Homology-Based Descriptors

Author: Soufiane Lyazidi Ahrillou Abraray

Supervisors: Dra. Petia Radeva
Dr. Bhalaji Nagarajan
Dr. Carles Casacuberta

Realitzat a: Departament de Matemàtiques
i Informàtica

Barcelona, 15 de gener de 2026

To my nearest and dearest.

Contents

Acknowledgements

I would like to express my sincere gratitude to my supervisors, Dra. Petia Radeva, Dr. Bhalaji Nagarajan and Dr. Carles Casacuberta for their invaluable mentorship.

I am deeply grateful to Dra. Petia Radeva for introducing me to the field of Deep Learning and providing the foundation to explore and delve into this domain. I owe a special debt of gratitude to Dr. Nagarajan, whose constant availability, prompt feedback, and technical guidance provided continuous support that was instrumental to the development of this thesis.

I also want to thank to Dr. Carles Casacuberta for his guidance in every aspect of the project, especially for helping me navigate the theoretical challenges of persistent homology. It has been an honor to walk this academic path alongside him.

Finally, I would like to thank my family and friends for their endless encouragement and patience throughout my double degree studies. Their support has been my foundation, giving me the strength to persevere.

Abstract

Deep Neural Networks (DNNs) often fail to capture the underlying hierarchical topological structures present in images, a limitation particularly critical in domains such as medical imaging, where fine anatomical details are essential. Topological Data Analysis (TDA), and specifically its prominent technique, persistent homology, serves as a robust tool to detect and represent these structures by summarizing them into Persistence Diagrams.

This final degree project explores the theoretical framework and foundations required to understand persistent homology and its applications to image analysis. It also explores the applications of persistent homology in machine learning, specifically applied to multiclass image classification tasks using Deep Neural Networks (DNNs).

An effective way to integrate topological information into DNNs is to encapsulate it into a feature vector. While many existing methods enhance architectures like Transformers or CNNs by generating topological descriptors based on static information, this work focuses on learnable strategies. Leveraging the recent architecture, PHG-Net (Peng et al. [46]) as a foundation, we employ a framework where homological descriptors are not fixed manually but are adaptively selected and optimized by the network during training. In this way, we achieve greater generalization and flexibility by enabling the model to autonomously learn the optimal persistent features, rather than relying on fixed, manually selected descriptors.

Consequently, this work aims to validate and extend this methodology by establishing a comprehensive experimental benchmark on ISIC 2018, a complex medical imaging dataset and CIFAR-100, which includes more general objects. Furthermore, we conduct a qualitative analysis to evaluate the robustness and decision-making confidence of topologically enhanced models. This includes examining the specific regions where the model focuses on using Grad-CAMs, and analyzing the geometric organization of learned features through dimensionality reduction techniques such as PCA, t-SNE, and UMAP.

2020 Mathematics Subject Classification. 68T07, 68T45, 55N31, 62R40, 92C55

Resum

Les Xarxes Neuronals Profundes (DNNs) sovint no aconsegueixen capturar les estructures topològiques jeràrquiques subjacents presents a les imatges, una limitació especialment crítica en imatges mèdiques, on els detalls anatòmics fins són essencials. L'Anàlisi Topològica de Dades (TDA), i específicament la seva tècnica més destacada, l'homologia persistent, serveix com una eina robusta per detectar i representar aquestes estructures resumint-les en Diagrames de Persistència.

Aquest Treball de Final de Grau explora el marc teòric i els fonaments necessaris per entendre l'homologia persistent i les seves aplicacions a l'anàlisi d'imatges. També s'exploren les aplicacions de l'homologia persistent en l'aprenentatge automàtic, aplicat específicament a tasques de classificació d'imatges utilitzant Xarxes Neuronals Profundes.

Una forma efectiva per integrar la informació topologica a les DNNs és encapsular-la en vectors de característiques. Mentre que molts mètodes existents milloren arquitectures com els Transformadors o les CNN, generant descriptors topològics basats en regles estàtiques, aquest treball es centra en estratègies aprenibles. Utilitzant l'arquitectura PHG-Net (Peng et al. [46]) com a base, emprem un marc on els descriptors homològics no es fixen manualment, sinó que se seleccionen de manera adaptativa i són optimitzats per la xarxa durant l'entrenament. D'aquesta manera, aconseguim una major generalització i flexibilitat permetent que el model aprengui de manera autònoma les característiques persistents òptimes, en lloc de dependre de descriptors fixos seleccionats manualment.

Consegüentment, aquest treball té com a objectiu validar i estendre aquesta metodologia establint un punt de referència experimental exhaustiu amb l'ISIC 2018, un conjunt de dades complex d'imatges mèdiques, i el CIFAR-100, que inclou objectes més generals. A més, realitzem una anàlisi qualitativa per avaluar la robustesa i la confiança en la presa de decisions dels models millorats topològicament. Això inclou examinar les regions específiques on el model focalitza la seva atenció utilitzant Grad-CAMs, i analitzar l'organització geomètrica de les característiques apreses mitjançant tècniques de reducció de dimensionalitat com PCA, t-SNE i UMAP.

Chapter 1

Introduction

1.1 Context and Motivation

The application of advanced mathematical tools combined with modern data science aims to capture features often omitted by standard Machine Learning approaches. Datasets from various fields, such as medical imaging (e.g., prostate cancer analysis, mammographies, ocular diseases), contain rich and complex structures embedded within their geometry. Standard models often struggle to fully capture these structures. Consequently, developing tools capable of detecting, encapsulating, and representing these structures as feature vectors would allow us to feed Machine and Deep Learning models with deeper structural information, thereby enhancing their performance.

Topological Data Analysis (TDA) addresses this challenge by utilizing concepts from algebraic topology to identify and model the intrinsic shape of data. Persistent Homology [12, 35], the most prominent TDA technique, traces its roots to the 1990s when Frosini [28] introduced a distance between equivalence classes of manifolds. For 1-dimensional curves, this metric is equivalent to 0-th persistent homology. Over a decade later, in 2002, Edelsbrunner et al. [21] formally introduced the concept of persistent homology (defined as topological persistence), providing the first visualization methods and a computational algorithm. This was followed by the foundational algebraic results of Zomorodian and Carlsson [59], who reformulated persistent homology as a study of persistence modules over polynomial rings.

The **main objectives of this work** are the following:

- **Theoretical Foundations:** To establish a rigorous theoretical framework for Persistent Homology applied to image analysis. This encompasses providing a proof of an extended version of the Structure Theorem as the fundamental algebraic tool for formalizing topological features, demonstrating the practical application of these results by modeling images as finite cubical complexes.

- **Methodological Implementation:** To investigate state-of-the-art applications of persistent homology, specifically building upon the PHG-Net architecture proposed by Peng et al. (2024) [46]. Using this architecture as a foundation, this work aims to optimize the integration of learnable topological features into Deep Learning models and rigorously evaluate their impact across diverse image domains.

- **Qualitative Analysis & Interpretability:** To conduct a comprehensive qualitative evaluation of the model's performance and decision-making process. This involves determining how topological structures influence the model's focus using Grad-CAM techniques, and visualizing the geometric organization of the learned feature descriptors using dimensionality reduction techniques such as PCA, t-SNE, and UMAP.

To compute these topological features on modern data, such as the images used in this case study, we require efficient software libraries. One of the most widely used tools is the GUDHI library (Geometry Understanding in Higher Dimensions) [40], which provides state-of-the-art algorithms and data structures for TDA, specifically for persistent homology.

1.2 Contributions

In this work, we go deeper into the PHG-Net model, by exploring it into details, optimizing it and extending it. The primary contributions of this thesis are as follows:

- **Exhaustive Optimization Studies:** We conducted extensive studies to optimize the preprocessing steps and hyperparameter configurations of the persistence diagrams. This resulted in a comprehensive benchmark that provides valuable insights into which factors drive performance, opening new avenues for generalizing these hypotheses to broader applications of TDA.
- **Performance Improvements over the Reference Model:** We identified specific configurations, validated across multiple seeds, that outperform the results reported by Peng et al. [46] by more than 1%.
- **Robustness Validation:** We validated the robustness of the topological enhancement by extending the scope of the experiments. We verified the positive impact of TDA not only on medical imaging (ISIC2018 [14]) but also on a more general and class-diverse dataset, such as CIFAR-100 [37]. We also validated using various DNN models as a backbone (ResNet-18, ResNet-50, ResNet-152, and Swin Transformer V2-B).
- **Qualitative Topological Analysis:** Beyond quantitative metric improvements, we analyzed the effect of persistent homology on the model's decision-making process. This analysis demonstrated that topological descriptors help the model cluster distinct classes more effectively and guide the model's attention toward relevant morphological structures in skin lesions.

1.3 Organization

The remainder of this work is organized as follows:

- **Chapter 2:** Introduces the necessary theoretical background regarding geometric complexes and persistent homology. It also details the construction of cubical complexes from image data.
- **Chapter 3:** Provides a brief and essential background on machine learning, focusing on the concepts and tools used in this work. It covers supervised learning, different types of Deep Neural Networks (DNNs), and aspects of model optimization.
- **Chapter 4:** Offers a detailed explanation of the PHG-Net architecture. This chapter also introduces the datasets used and the experimental framework established for this study.
- **Chapter 5:** Presents our most relevant results, including a quantitative metrics analysis and a qualitative analysis of the ISIC2018 and CIFAR-100 datasets and the backbones ResNet-18, ResNet-50, ResNet-152, and Swin Transformer V2-B.
- **Chapter 6:** Concludes the work by summarizing the global conclusions drawn from the experiments and outlining potential futures lines of work.

Chapter 2

Homological Persistence

2.1 Preliminaries

This section lays the algebraic and topological groundwork required for a rigorous formulation of persistent homology. Homological persistence refers to the evolution of topological features, such as connected components or cycles, along the stages of a filtration of topological spaces. In practice, one uses finite simplicial complexes or finite cubical complexes instead of topological spaces, due to their combinatorial nature.

Homology of simplicial complexes (or cubical complexes) is defined by means of formal sums of simplices (or cubes) with coefficients in a ring R. The resulting *homology groups*, which are defined in the next section, acquire a natural structure of R-modules. We first recall the definition of an R-module.

Definition 2.1. *Let R be a commutative ring with a unit element 1_R. An R-*module *is a set M equipped with an addition operation $+$ such that $(M, +)$ is an abelian group, and a scalar multiplication $\cdot : R \times M \longrightarrow M$ satisfying the following properties for every $r, s \in R$ and $m, n \in M$:*

- *Distributivity over module addition: $r \cdot (m + n) = r \cdot m + r \cdot n$.*
- *Distributivity over ring addition: $(r + s) \cdot m = r \cdot m + s \cdot m$.*
- *Compatibility with ring multiplication: $(r \cdot s) \cdot m = r \cdot (s \cdot m)$.*
- *Identity action: $1_R \cdot m = m$.*

If R is a field, then an R-module is an R-vector space. In this work, we use coefficients in a field; the default choice is the field $\mathbb{F}_2 = \mathbb{Z}/2\mathbb{Z}$.

2.2 Geometric Complexes

We define complexes in a way that allows us to indistinctly use simplices or cubes as building blocks, depending on the context. We define a *geometric complex* K as a finite set of cells (simplices or cubes) equipped with a partial order relation by inclusion. If $\sigma \in K$ and $\tau \subseteq \sigma$, we say that τ is a *face* of σ. We call τ a *proper face* of σ if the inclusion is strict. A geometric complex must be closed under the face relation, that is, if $\sigma \in K$ and τ is a face of σ then τ must also belong to K.

We primarily distinguish between two instances of geometric complexes: *simplicial complexes* and *cubical complexes*. We study simplicial complexes as they form the foundational structure for homology, and we utilize cubical complexes for their practical application in image analysis, particularly when representing grid data.

2.2.1 Simplicial Complexes

An *affine simplex* is the convex hull of a collection $\{v_0, \ldots, v_n\}$ of affinely independent points in a Euclidean space $\mathbb{R}^d$. We denote it as follows:

$$\Delta(v_0, \ldots, v_n) = \{\textstyle\sum_{i=0}^{n} t_i v_i \mid t_i \geq 0,\ \sum_{i=0}^{n} t_i = 1\}\,.$$

The *standard n-simplex* is the affine simplex $\Delta^n = \Delta(e_1, \ldots, e_{n+1})$ in $\mathbb{R}^{n+1}$, where e_i is the i-th unit coordinate point (Figure 2.1).

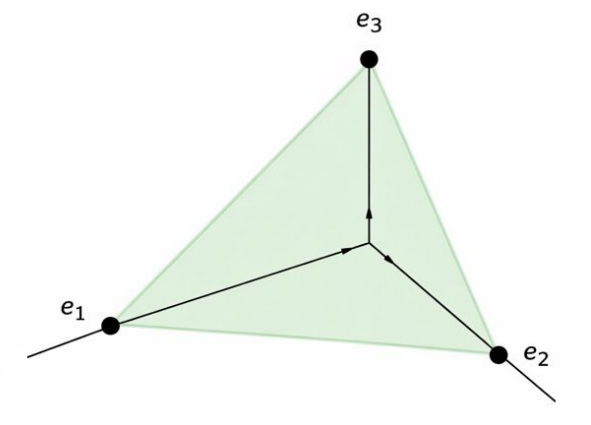


Figure 2.1: Standard 2-simplex $\Delta^2 \subset \mathbb{R}^3$. Vertices are located at unit distance along each axis.

A *simplicial complex* is a set K of affine simplices closed under the face relation. The *dimension* of a simplex $\Delta(v_0, \ldots, v_n)$ is defined to be n. Thus the 0-dimensional simplices are points, which are called *vertices*, and the 1-dimensional simplices are line segments, which are called *edges*. The *dimension* of a simplicial complex is the maximum dimension of its simplices. Simplicial complexes can be viewed as a high-dimensional generalization of (simple, undirected) graphs, which are simplicial complexes of dimension 1. A *subcomplex* of a simplicial complex K is a subset $W \subseteq K$ which is itself a simplicial complex.

The *geometric realization* or *underlying space* of a simplicial complex K is defined as the union $|K| = \cup_{\sigma \in K} \sigma$ of its simplices, viewed as topological subspaces of a common Euclidean space $\mathbb{R}^d$, where d is called *embedding dimension* and it is greater of equal than the dimension of K as a simplicial complex. The geometric realization $|K|$ is usually treated up to homeomorphism, regardless of the embedding dimension.

For many purposes, simplicial complexes are often treated as purely combinatorial objects, by disregarding their topological structure. Such objects are called *abstract simplicial complexes*, consisting of a set V, whose elements are called *vertices*, and a collection K of finite subsets of V, called *simplices*, where K is closed under the inclusion relation. Then one considers a set $\{v_0, \ldots, v_n\} \subseteq V$ instead of a geometric simplex $\Delta(v_0, \ldots, v_n)$. For simplicity, we will use this notation from now on.

A simplicial complex is said to be *ordered* if its vertex set V is equipped with a total order. In what follows, we will assume that simplicial complexes are ordered, and that $v_0 < \cdots < v_n$ in any simplex $\{v_0, \ldots, v_n\}$.

Homology of Simplicial Complexes

Let K be a simplicial complex and $n \geq 0$. We define an *n-chain* as a formal linear combination of n-simplices with coefficients in a ring R (typically $\mathbb{Z}/2\mathbb{Z}$). We denote an n-chain by $c = \sum a_i \sigma_i$, where σ_i are simplices of dimension n and $a_i \in R$.

We denote by $C_n(K, R)$ the set of all n-chains formed from the n-simplices of K. Two n-chains are added element-wise: $\sum a_i\sigma_i + \sum b_i\sigma_i = \sum(a_i + b_i)\sigma_i$. Thus, the set $C_n(K, R)$ forms an R-module under addition.

The *boundary* of an n-simplex $\sigma = \{v_0, \ldots, v_n\}$ is defined as the alternating sum of its $(n-1)$-dimensional faces:

$$\partial_n(\{v_0, \ldots, v_n\}) = \sum_{j=0}^{n}(-1)^j\{v_0, \ldots, \widehat{v_j}, \ldots, v_n\},$$

where $\{v_0, \ldots, \widehat{v_j}, \ldots, v_n\}$ denotes the $(n-1)$-simplex obtained by removing the vertex v_j.

This definition extends by linearity to chains: for an n-chain $c = \sum a_i\sigma_i$, the boundary of c is defined to be $\partial_n c = \sum a_i\partial_n\sigma_i$. Hence, the boundary defines a homomorphism

$$\partial_n : C_n(K, R) \longrightarrow C_{n-1}(K, R),$$

as it preserves the R-module structure. The 0-th boundary is defined to be the zero map $\partial_0 \colon C_0(K, R) \to 0$.

Boundary homomorphisms satisfy the fundamental property that $\partial_n \circ \partial_{n+1} = 0$ for all $n \geq 0$ (see Hatcher, 2002, Lemma 2.1 for proof). Consequently, the sequence $\{C_n(K, R)\}_{n\geq 0}$, together with the boundary homomorphisms $\{\partial_n\}_{n\geq 0}$, forms a *chain complex* of R-modules, as defined next.

Definition 2.2. *A* chain complex *of R-modules is a collection $C_* = \{C_n\}_{n\geq 0}$ of R-modules together with R-module homomorphisms $\partial_n : C_n \longrightarrow C_{n-1}$ such that $\partial_n \circ \partial_{n+1} = 0$ for all $n \geq 0$.*

The equality $\partial_n \circ \partial_{n+1} = 0$ implies that $\operatorname{Im}\partial_{n+1} \subseteq \operatorname{Ker}\partial_n$ for $n \geq 1$, which enables us to define the n-th *homology group* of a chain complex C_* as

$$H_n(C_*) = \operatorname{Ker}\partial_n / \operatorname{Im}\partial_{n+1},$$

where $\operatorname{Ker}\partial_0 = C_0$. The elements in $\operatorname{Ker}\partial_n$ are called *n-cycles* and those of $\operatorname{Im}\partial_{n+1}$ are called *n-boundaries*.

The *homology groups* of a simplicial complex K with coefficients in a ring R are then defined as the homology groups of the chain complex $C_*(K, R) = \{C_n(K, R)\}_{n\geq 0}$, which are in fact R-modules, since $\operatorname{Ker}\partial_n$ and $\operatorname{Im}\partial_{n+1}$ are R-modules.

2.2.2 Cubical Complexes

While the geometric representation of simplices (vertices, edges, triangles, tetrahedra) is intuitive for general topology, our primary case of study, digital images, relies on a different structure. Image data is typically represented as cubical grids: two-dimensional grids for grayscale images and three-dimensional grids for volumetric data. Cubical complexes provide the appropriate theoretical framework to apply homology theory to these rigid grid structures without requiring triangulation.

Definition 2.3. *An* elementary interval *is a closed interval in $\mathbb{R}$ of the form $I = [k, k+1]$ (called non-degenerate) or $I = [k, k]$ (called degenerate), where $k \in \mathbb{Z}$.*

Definition 2.4. *An* elementary cube *Q is defined as the finite product of d elementary intervals:*

$$Q = \prod_{i=1}^{d} I_i = I_1 \times I_2 \times \cdots \times I_d.$$

The *dimension* of an elementary cube is defined as the number of non-degenerate intervals in its product decomposition.

- A 0-cube (vertex) consists entirely of degenerate intervals.
- A 1-cube (edge) contains exactly one non-degenerate interval.
- A 2-cube (square) contains exactly two non-degenerate intervals.

We say that a cube P is a *face* of a cube Q if $P \subseteq Q$. For example, the edge $[1,2] \times [4,4]$ is a face of the square $[1,2] \times [4,5]$. Note that two cubes are comparable by inclusion only if they are embedded in the same ambient Euclidean space (i.e., defined by the same number of intervals).

Definition 2.5. *A* cubical complex *K is a finite set of elementary cubes that is closed under the operation of taking faces. Thus, if $Q \in K$ and P is a face of Q, then P must also belong to K.*

Homology of Cubical Complexes

Similarly as in the simplicial case, we define an n-chain of a cubical complex K as a formal linear combination of n-cubes with coefficients in a ring R. While this construction extends to arbitrary rings, we will restrict our attention to the field $\mathbb{Z}/2\mathbb{Z}$. The *n-th chain group*, denoted $C_n(K,R)$ or $C_n(K)$ for simplicity, is defined as the free R-module generated by the n-cubes of K.

The *boundary* of an elementary interval I is defined as follows:

- If $I = [k, k+1]$ is non-degenerate: $\partial[k, k+1] = [k+1, k+1] - [k, k]$.
- If $I = [k, k]$ is degenerate: $\partial[k, k] = 0$.

For a general elementary cube $Q = \prod_{i=1}^{d} I_i$, the boundary of Q is defined as:

$$\partial Q = \sum_{j=1}^{d} (-1)^{j-1} (I_1 \times \cdots \times \partial I_j \times \cdots \times I_d),$$

where the terms $I_1 \times \cdots \times \partial I_j \times \cdots \times I_d$ are formed by expanding the boundary of I_j. That is, if $I_j = [k, k+1]$ then

$$I_1 \times \cdots \times \partial I_j \times \cdots \times I_d = I_1 \times \cdots \times [k+1, k+1] \times \cdots \times I_d - I_1 \times \cdots \times [k,k] \times \cdots \times I_d$$

and if $I_j = [k,k]$ then $I_1 \times \cdots \times \partial I_j \times \cdots \times I_d = 0$.

The boundary operator extends linearly to the entire chain group, defining a homomorphism $\partial_n : C_n(K) \longrightarrow C_{n-1}(K)$. The fundamental property $\partial_n \circ \partial_{n+1} = 0$ holds, allowing us to define the cubical homology groups as

$$H_n(K) = \operatorname{Ker} \partial_n / \operatorname{Im} \partial_{n+1},$$

where $\operatorname{Ker} \partial_0 = C_0(K)$.

The *geometric realization* or *underlying space* $|K|$ of a cubical complex K is defined, similarly to the simplicial case, as the union of all the cubes in K within a common ambient Euclidean space.

2.2.3 Maps Between Complexes

Let K and W be simplicial or cubical complexes, and let ∂_n^K and ∂_n^W denote their respective n-dimensional boundary operators. A *map of complexes* (or simplicial/cubical map) $f : K \longrightarrow W$ is a function mapping the cells (simplices or cubes) of K to cells of W by preserving the structural relations, that is, it satisfies the property

$$\partial_n^W \circ f = f \circ \partial_n^K.$$

A map of complexes induces a *chain map* $f_\# : C_n(K) \longrightarrow C_n(W)$, defined by extending f linearly to chains:

$$f_\#\left(\sum \lambda_i \sigma_i\right) = \sum \lambda_i f(\sigma_i),$$

where σ_i are cells of K (cubes or simplices). This induced map still commutes with the boundary operators:

$$\partial_n^W \circ f_\# = f_\# \circ \partial_n^K.$$

Remark 2.6. From this section onwards, we adopt a standard abuse of notation by denoting the chain map simply as $f_\#$, omitting the dimension n. The correct dimension is implied by the context. For instance, in the composition $f_\# \circ \partial_n^K$, the operator $f_\#$ is implicitly understood to be the $(n-1)$-dimensional component acting on the output of the boundary map.

We define the *induced homomorphism* $f_* : H_n(K) \longrightarrow H_n(W)$ on homology as

$$f_*([c]) = [f_\#(c)],$$

where $[c]$ is the homology class of a cycle $c \in C_n(K)$.

To ensure that f_* is well defined, we must verify two conditions:

- Preservation of cycles: The image of a cycle must be a cycle: $f_\#(c) \in \operatorname{Ker} \partial_n^W$.
- Independence of representative: The definition must not depend on the representative chosen from each class.

Proof. First, let c be a cycle in K, so $\partial_n^K(c) = 0$. By commutativity,

$$\partial_n^W(f_\#(c)) = f_\#(\partial_n^K(c)) = f_\#(0) = 0.$$

Hence, $f_\#(c)$ is a cycle in W.

Second, suppose c and c' are two cycles that represent the same class, meaning their difference is a boundary: $c - c' = \partial_{n+1}^K(\alpha)$ for some $\alpha \in C_{n+1}(K)$. Applying $f_\#$, we obtain that

$$f_\#(c) - f_\#(c') = f_\#(\partial_{n+1}^K(\alpha)) = \partial_{n+1}^W(f_\#(\alpha)).$$

Since $f_\#(\alpha) \in C_{n+1}(W)$, the difference $f_\#(c) - f_\#(c')$ lies in $\operatorname{Im} \partial_{n+1}^W$. Thus, $[f_\#(c)] = [f_\#(c')]$. □

It is a standard result (see Hatcher, 2002, Theorem 2.27) that the homology of a simplicial complex is isomorphic to the singular homology of its geometric realization. This result extends naturally to cubical complexes.

2.2.4 Relating Simplicial and Cubical Complexes

To establish a topological relationship between cubical complexes and simplicial complexes, we employ the Freudenthal triangulation. This method subdivides cubes into simplices, allowing us to associate a simplicial complex $T(K)$ to any cubical complex K so that their geometric realizations are homeomorphic, i.e., $|T(K)| \cong |K|$.

Since our work focuses on 2D images, we restrict our detailed proof to the case where the highest-dimensional cells are squares (2-cubes).

Consider a standard 2-cube Q, corresponding to $[0,1] \times [0,1]$. We label its vertices as follows: $v_0 = (0,0)$, $v_1 = (0,1)$, $v_2 = (1,1)$, $v_3 = (1,0)$.

The*Freudenthal triangulation* subdivides this square into two 2-simplices (triangles) by cutting along the diagonal connecting v_0 and v_3:

- The "lower" simplex σ_L spanned by vertices $\{v_0, v_2, v_3\}$.
- The "upper" simplex σ_U spanned by vertices $\{v_0, v_1, v_3\}$.

Formally, given a cubical complex K, we define its associated simplicial complex $T(K)$ as follows:

Definition 2.7. *For a 2-dimensional cubical complex K, the simplicial complex $T(K)$ is constructed from the elements of K satisfying the following conditions:*

1. *The vertex set of $T(K)$ is identical to the set of 0-cubes (vertices) of K.*
2. *Every 1-cube (edge) in K corresponds to a unique 1-simplex (edge) in $T(K)$ connecting the same endpoints.*
3. *Every 2-cube (square) in K is subdivided into two 2-simplices (triangles) as described above adding the diagonal as a new edge.*

Since it is straightforward to prove that the geometric realizations are homeomorphic, $|K| \cong |T(K)|$, it follows that their homology groups are isomorphic, i.e., $H_n(|K|) \cong H_n(|T(K)|)$. Furthermore, since the homology of a complex is isomorphic to the singular homology of its geometric realization, we obtain the result.

However, this result can also be proven algebraically without relying on geometric realizations, working directly with chain complexes. The proof can be found in Appendix A.1.

2.3 Filtrations

Definition 2.8. *Let P be a partially ordered set and K a geometric complex (simplicial or cubical). A collection of subcomplexes $\{K_t\}_{t\in P}$ is called a P-*filtration *of K if it exhausts the complex, $K = \bigcup_{t\in P} K_t$, and satisfies the inclusion property:*

$$K_s \subseteq K_t \quad \textit{whenever} \quad s \leq t.$$

A *P^*-filtration* is defined analogously but with the order reversed, that is, $K_i \subseteq K_j$ whenever $i \geq j$. In this work, we focus on the case where $P = \mathbb{R}$.

2.3.1 Filtering Functions and Sublevel Sets

Definition 2.9. *A P-*filtering function *on a complex K is a map $f : K \longrightarrow P$ satisfying the monotonicity property:*

$$f(\sigma) \leq f(\tau) \quad \textit{if} \quad \sigma \subseteq \tau$$

for any cells $\sigma, \tau \in K$.

This function induces a P-filtration of K via *sublevel sets*, defined as:

$$K_t = \{\sigma \in K \mid f(\sigma) \leq t\}.$$

The monotonicity of f ensures that if a cell τ belongs to K_t (i.e., $f(\tau) \leq t$), then any face $\sigma \subseteq \tau$ also satisfies $f(\sigma) \leq f(\tau) \leq t$, meaning $\sigma \in K_t$. Thus, each K_t is a subcomplex of K. Furthermore, the inclusion condition ($K_s \subseteq K_t$ for $s \leq t$) is an immediate consequence of the definition of sublevel sets.

2.3.2 Superlevel Sets

Analogously, we define superlevel sets.

Definition 2.10. *A P^*-*filtering function *on a complex K is a function $f : K \longrightarrow P$ satisfying the reversed monotonicity property:*

$$f(\tau) \leq f(\sigma) \quad \textit{if} \quad \sigma \subseteq \tau.$$

Such a function yields subspaces called *superlevel sets*:

$$K^t = \{\sigma \in K \mid f(\sigma) \geq t\}.$$

Similarly to the previous case, K^t is a subcomplex of K. These subcomplexes form a P^*-filtration of K (as the parameter t decreases, the subcomplex grows). Furthermore, it is straightforward to verify that if f induces a sublevel filtration, then the opposite function $-f$ induces a superlevel filtration.

2.4 Persistent Homology

Our main interest lies in the evolution of topological features along the complexes in a filtration, which is described by the sequence of homology groups induced by the filtration.

Let P be a partially ordered set (typically a finite set of indices $\{0, 1, \ldots, n\}$ for digital images) and $\{K_t\}_{t \in P}$ a P-filtration. For every $i, j \in P$ with $i \leq j$, there exists an inclusion map $K_i \hookrightarrow K_j$. This inclusion induces a homomorphism on the homology groups for every dimension $p \geq 0$:

$$f_p^{i,j} : H_p(K_i) \longrightarrow H_p(K_j).$$

Consequently, the filtration induces a sequence of homology groups connected by homomorphisms:

$$H_p(K_0) \longrightarrow H_p(K_1) \longrightarrow \cdots \longrightarrow H_p(K_n).$$

2.4.1 Persistent Homology Groups

We define the *p-th persistent homology groups* $H_p^{i,j}$ as the image of the homomorphism induced by the inclusion:

$$H_p^{i,j} = \operatorname{Im} f_p^{i,j}.$$

This group contains the homology classes born at or before time i that are still alive at time j.

Alternatively, this group can be defined explicitly in terms of cycles and boundaries as

$$H_p^{i,j} \cong \frac{Z_p(K_i)}{B_p(K_j) \cap Z_p(K_i)},$$

where $Z_p(K_i) = \operatorname{Ker} \partial_p^i$ is the cycle group of K_i and $B_p(K_j) = \operatorname{Im} \partial_{p+1}^j$ is the boundary of K_j.

2.4.2 Birth and Death of Classes

We say that a homology class $\alpha \in H_p(K_b)$ is *born* at time b if it does not belong to the image of the previous group. Formally:

$$\alpha \notin \operatorname{Im} f_p^{b-1,b}.$$

This implies that α cannot be traced back to any previous step; that is, $\alpha \notin \operatorname{Im} f_p^{j,b}$ for any $j < b$. This holds because the homomorphisms compose as $f_p^{j,b} = f_p^{b-1,b} \circ f_p^{j,b-1}$, and thus, if α were in the image of a map starting at $j < b - 1$, it would necessarily belong to the image of the map coming from $b - 1$.

Intuitively, a class can die in two ways:

- Trivial death: A cycle becomes a boundary (e.g., a loop is filled by a disk), meaning it becomes equivalent to the zero class in the homology group. This can be formalized as finding the smallest index $d > b$ such that

$$f_p^{b,d}(\alpha) = 0.$$

- Merging: Two separate components (or cycles) connect. When this happens, they become part of the same homology class. By convention (the Elder Rule), the younger class dies (merges into the older one), and the older class persists.

Formalizing the merging condition requires care. We say that a class α, born at b, dies at time d if

$$f_p^{b,d}(\alpha) \in H_p^{b-1,d} \quad \text{and} \quad f_p^{b,d-1}(\alpha) \notin H_p^{b-1,d-1}.$$

The condition $f_p^{b,d}(\alpha) \in H_p^{b-1,d}$ means that at time d, the class α has become equivalent to some cycle that was born before b (i.e., it has merged with an older class). The second condition ensures that d is the first time index for which this merging occurs.

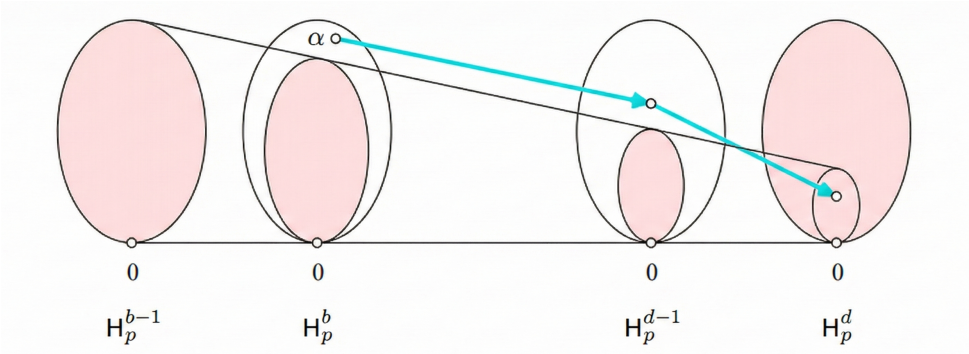


Figure 2.2: The class α is born at K_b and dies entering K_d. It merges into the image of H_p^{b-1} (Edelsbrunner et al. [20]).

2.5 Structure Theorem

Definition 2.11. *A* persistence module $\mathcal{M}$ *over a ring* R *is defined as a family of* R*-modules* $\{M_i\}_{i\geq 0}$ *together with a sequence of homomorphisms* $\varphi_i : M_i \longrightarrow M_{i+1}$ *for* $i \geq 0$.

A persistence module is said to be of *finite type* if each M_i is finitely generated as an R-module and there exists an integer $m \geq 0$ such that the homomorphisms φ_i are isomorphisms for all $i \geq m$.

We define a *graded ring* as a ring R that decomposes as a direct sum of additive subgroups $R = \bigoplus_{i\in\mathbb{Z}} R_i$ such that $R_n \cdot R_m \subseteq R_{n+m}$. For example, the standard grading of the polynomial ring $R[t]$ is the decomposition $R[t] = \bigoplus_{i\geq 0} R_i$, where R_i consists of all monomials of degree i (elements of the form $r \cdot t^i$ where $r \in R$).

A *graded module* M over a graded ring R is an R-module that decomposes into a direct sum of R-modules $M = \bigoplus_{i\in\mathbb{Z}} M_i$ such that the ring action respects the grading: $R_n \cdot M_m \subseteq M_{n+m}$.

Elements belonging to a summand R_i of a graded ring R (or to a summand M_i of a graded module M) are called *homogeneous* elements of degree i. For example, in the ring $\mathbb{F}[t]$ with the standard grading, the polynomial t^3 is homogeneous, whereas $t^3 + t^5$ is not, as it mixes degrees 3 and 5.

A graded ring R (or module M) is said to be *non-negatively graded* if $R_i = 0$ (or $M_i = 0$) for all $i < 0$. For example, $R[t]$ equipped with the standard grading is non-negatively graded.

2.5.1 Correspondence with $R[t]$-modules

In the context of complexes, let $\{K_i\}_{i\geq 0}$ be a filtration of a finite simplicial or cubical complex K. It follows directly from the definition that the homology groups $H_p(K_i, R)$ are R-modules (or vector spaces, if R is a field). Furthermore, the inclusion maps $K_i \hookrightarrow K_{i+1}$ induce homomorphisms $f_p^{i,i+1} : H_p(K_i, R) \to H_p(K_{i+1}, R)$.

This structure allows us to define a persistence module, denoted by $\mathcal{M}$, where the modules are given by $M_i = H_p(K_i, R)$ and the connecting homomorphisms are $\varphi_i = f_p^{i,i+1}$.

Since the complex K is finite, each subcomplex K_i is also finite. Consequently, every homology group $H_p(K_i, R)$ is finitely generated. Furthermore, because K is finite, the filtration must eventually stabilize, that is, there exists an integer $m \geq 0$ such that $K_i = K$ for all $i \geq m$. As a result, for all $i \geq m$, the induced homomorphisms $f_p^{i,i+1}$ are isomorphisms. These two conditions ensure that this persistence module is of finite type.

Starting from the persistence module $\mathcal{M} = \{M_i, \varphi_i\}$, we define a graded module $\alpha(\mathcal{M})$ over the graded ring $R[t]$ (equipped with the standard grading) as the direct sum

$$\alpha(\mathcal{M}) = \bigoplus_{i=0}^{\infty} M_i.$$

We define the action of the ring $R[t]$ on this module by extending the action of R linearly and defining the action of the variable t as the shift map induced by the linear maps φ_i. Specifically, for an element $m = (m_0, m_1, \dots) \in \alpha(\mathcal{M})$,

$$t \cdot (m_0, m_1, \dots) := (0, \varphi_0(m_0), \varphi_1(m_1), \dots).$$

Under this action, $\alpha(\mathcal{M})$ is a graded $R[t]$-module.

This construction establishes a fundamental equivalence of categories between the category of persistence modules of finite type over R and the category of finitely generated non-negatively graded modules over $R[t]$ (Zomorodian and Carlsson [59]).

This correspondence is powerful because it allows us to analyze the entire dynamic sequence of homology groups (the persistence module) as a single algebraic object (the associated $R[t]$-module).

2.5.2 The Structure Theorem

This correspondence is particularly useful due to the algebraic properties of the ring $R[t]$ (specifically when R is a field, making $R[t]$ a Principal Ideal Domain). The standard Structure Theorem for finitely generated modules over a PID states the following:

Theorem 2.12. *If D is a PID, then every finitely generated D-module is isomorphic to a direct sum of cyclic D-modules. That is, it decomposes uniquely into the form*

$$D^{\beta} \oplus \bigoplus_{i=1}^{m} D/d_i D,$$

for $d_i \in D$, $\beta \in \mathbb{Z}$, such that $d_i \mid d_{i+1}$.

A more complete statement, adapted for graded modules and provable by reasoning analogously to the ungraded case, is the following.

Theorem 2.13. *Every graded module M over a graded PID D decomposes uniquely into the form*

$$\bigoplus_{i=1}^{n} \Sigma^{\alpha_i} D \oplus \bigoplus_{j=1}^{m} \Sigma^{\gamma_j} \big(D/d_j D\big),$$

where $d_j \in D$ are homogeneous elements satisfying $d_j \mid d_{j+1}$, and $\alpha_i, \gamma_j \in \mathbb{Z}$. Here Σ^{α} denotes an upward shift in grading by α.

Proof. Let $\mathbb{F}$ be a field. We restrict our attention to the case where $D = \mathbb{F}[t]$, as this is the instance required for the application to persistent homology by Zomorodian and Carlsson.

Let M be a non-negatively graded module over the graded ring $R = \mathbb{F}[t]$. We assume that M is finitely generated. We can choose a set of homogeneous generators $\{g_1, \ldots, g_n\}$ with degrees $d_1, \ldots, d_n$, possibly repeated.

Since M is the direct sum of its homogeneous parts $M = \bigoplus M_i$, if we start with non-homogeneous generators, we can simply replace each generator with its homogeneous components. Therefore, the collection of all such homogeneous components still generates M.

We construct a degree-preserving epimorphism from a free graded module to M:

$$\varphi : \bigoplus_{i=1}^{n} \Sigma^{d_i} \mathbb{F}[t] \longrightarrow M,$$

defined by mapping the generator of the i-th summand to g_i as

$$\varphi(0, \ldots, t^{d_i}, \ldots, 0) = g_i.$$

By the First Isomorphism Theorem, we have that

$$M \cong \left(\bigoplus_{i=1}^{n} \Sigma^{d_i} \mathbb{F}[t] \right) \Big/ \operatorname{Ker} \varphi.$$

The domain $\bigoplus_{i=1}^{n} \Sigma^{d_i} \mathbb{F}[t]$ is a free $\mathbb{F}[t]$-module of rank n with a basis given by the generator t^{d_i} of every homogeneous component. Since $\mathbb{F}[t]$ is a Principal Ideal Domain (PID), the submodule $\operatorname{Ker} \varphi$ is also free, with rank $m \leq n$ ([18, Theorem 12.4]).

To determine the structure of the quotient, we apply the Smith Normal Form (SNF) algorithm (diagonalization) to the inclusion map $\iota : \operatorname{Ker} \varphi \hookrightarrow \bigoplus_{i=1}^{n} \Sigma^{d_i} \mathbb{F}[t]$, as in [59].

We represent the generators of $\operatorname{Ker} \varphi$ in terms of the basis of the free module $\bigoplus_{i=1}^{n} \Sigma^{d_i} \mathbb{F}[t]$ using an $n \times m$ matrix. Since we are working in the graded category, we must restrict the standard row and column operations to those that preserve the graded structure. Consequently, operations are restricted to rows and columns associated with generators of compatible degrees (or must use polynomial shifts to align the degrees), ensuring that the resulting matrix entries remain homogeneous.

This process yields a diagonal matrix with entries $r_1, \ldots, r_m, 0, \ldots, 0$. Due to the graded constraint, these entries must be homogeneous elements of $\mathbb{F}[t]$. In this ring, the only homogeneous elements are monomials of the form $c \cdot t^k$. Since the Smith Normal Form algorithm allows for multiplication by units (non-zero scalars from $\mathbb{F}$), we can normalize these entries to be monic. Thus, without loss of generality, we assume the non-zero diagonal entries are of the form t^{e_i} with $e_i \geq 1$. Furthermore, the algorithm guarantees the divisibility condition $t^{e_1} \mid t^{e_2} \mid \cdots \mid t^{e_m}$.

Consequently, the module M decomposes as:

$$M \cong \frac{\bigoplus_{i=1}^{n} \Sigma^{d_i} \mathbb{F}[t]}{\bigoplus_{j=1}^{m} t^{e_j + d_j} \mathbb{F}[t]}.$$

Therefore, we obtain:

$$M \cong \left(\bigoplus_{i=1}^{m} \frac{\Sigma^{d_i} \mathbb{F}[t]}{\Sigma^{d_i} t^{e_i} \mathbb{F}[t]} \right) \oplus \left(\bigoplus_{j=m+1}^{n} \Sigma^{d_j} \mathbb{F}[t] \right).$$

Recognizing that the initial terms represent a shift in degree for both the generating module and the relation, we can rewrite the expression to explicitly factor out the shift. Using the notation established in the Structure Theorem, we arrive at the final decomposition:

$$M \cong \left(\bigoplus_{i=1}^{m} \Sigma^{d_i} (\mathbb{F}[t] / t^{e_i} \mathbb{F}[t]) \right) \oplus \left(\bigoplus_{j=m+1}^{n} \Sigma^{d_j} \mathbb{F}[t] \right). \qquad \square$$

Remark 2.14. The condition that the elements $d_j \in D$ are homogeneous is crucial. This restriction ensures that the quotient modules D/d_jD preserve the graded structure.

Remark 2.15. The operator Σ^α defines a shift in grading. Let M be a graded module with decomposition $M = \bigoplus_{n\in\mathbb{Z}} M_n$. Then $\Sigma^\alpha M$ is a new graded module whose components are defined as

$$(\Sigma^\alpha M)_n = M_{n-\alpha}.$$

Intuitively, this means that the component originally at degree k (M_k) is moved to degree $k+\alpha$ in the shifted module.

Remark 2.16. The theorem relies on the fact that both the ring D and the quotients D/d_jD are D-modules. It is clear that D is a module over itself. For D/d_jD, the module action is well-defined by $r \cdot [x] = [r \cdot x]$, inheriting the structure from D.

Zomorodian and Carlsson exploit this graded version of the Structure Theorem to establish both the existence and uniqueness of the barcode decomposition associated with the persistent homology of a filtration.

Assuming that the coefficients come from a field $\mathbb{F}$, the polynomial ring $\mathbb{F}[t]$ becomes a Principal Ideal Domain (PID). Consequently, the Structure Theorem allows us to decompose the persistence module $\alpha(\mathcal{M})$ into a direct sum of cyclic graded $\mathbb{F}[t]$-modules, as follows.

Theorem 2.17 (Structure Theorem for Persistence Modules)**.** *Let $\mathcal{M}$ be a persistence module of finite type over a field $\mathbb{F}$. Then, there exists a decomposition of the associated graded $\mathbb{F}[t]$-module:*

$$\alpha(\mathcal{M}) \cong \left(\bigoplus_{i=1}^{n} \Sigma^{\alpha_i}\mathbb{F}[t]\right) \oplus \left(\bigoplus_{j=1}^{m} \Sigma^{\gamma_j}\mathbb{F}[t]/(t^{k_j})\right), \tag{2.1}$$

where Σ^k denotes a shift in grading.

This algebraic decomposition of the graded module $\alpha(\mathcal{M})$ translates directly back to the topological structure of the persistence module $\mathcal{M}$ due to the equivalence of categories established by the correspondence α.

Geometric Interpretation

This algebraic decomposition offers a straightforward geometric interpretation:

- Infinite Persistence: The free part of the decomposition, $\Sigma^{\alpha_i}\mathbb{F}[t]$, maps to the interval $[\alpha_i, +\infty)$. This corresponds to a topological feature that is born at step α_i and never dies (persisting indefinitely).
- Finite Persistence: The torsion part of the decomposition, $\Sigma^{\gamma_j}\mathbb{F}[t]/(t^{k_j})$, maps to the interval $[\gamma_j, \gamma_j + k_j)$. This corresponds to a topological feature that is born at step γ_j and dies at step $\gamma_j + k_j$ (having persisted for k_j steps).

Remark 2.18. In general, persistence features where $k_i = 0$ are not analyzed in detail. Such features corresponds to a torsion module of the form $\Sigma^{b_i}\mathbb{F}[t]/(t^0)$. Since $t^0 = 1$, this makes the quotient module trivial: $\mathbb{F}[t]/(1) \cong \{0\}$.

Compatible Bases

To understand the significance of this decomposition, consider the filtration $K_0 \subseteq K_1 \subseteq \dots$. At each step i, we compute the homology group $H_k(K_i)$, which is a vector space since we are operating under a field $\mathbb{F}$. If we were to simply compute a basis for $H_k(K_1)$ and a separate basis for $H_k(K_2)$, there would be no guarantee of a relationship between them. It would be impossible to determine if a cycle in K_1 "survived" to K_2 or if it died and a new, unrelated cycle appeared. We need compatible bases to track the evolution of cycle classes throughout the filtration.

The Structure Theorem guarantees precisely this. It states that, for every k, the module $\bigoplus_{i\geq 0} H_k(K_i)$ can be decomposed into cyclic submodules generated by a finite set of generators $\{g_1, \dots, g_N\}$.

Each generator g_i corresponds to one of the summands in the decomposition. Since the summand is shifted by Σ^b, the generator g_i is a homogeneous element of degree b, meaning it is a homology class born at step b: $g_i \in H_k(K_b)$.

The action of t in the module corresponds to the induced map φ in the filtration. Therefore, $t \cdot g = \varphi_b(g) \in H_k(K_{b+1})$.

Consequently, if g_i represents a cycle at step b, then $t \cdot g_i$ represents the same cycle (mapped forward) at step $b+1$.

Thus, the theorem allows us to choose a basis for every homology group $H_k(K_j)$ simultaneously, such that the basis vectors map directly to each other (or to zero) across the filtration steps. A generator g_i born at α_i contributes a basis vector to every group $H_k(K_j)$ for j in the interval $[\alpha_i, \infty)$ (for the free part) or $[\alpha_i, \alpha_i + k_i)$ (for the torsion part). Specifically, at step j, this contribution takes the form $t^{j-\alpha_i} \cdot g_i$. when viewed from the isomorphism side of the module decomposition

Remark 2.19. It is legitimate to question the uniqueness of the decomposition, since going back to the proof of the theorem, we see that we choose a set of generators for the module. Consequently, one might ask if choosing another set of generators would yield a different decomposition. Nevertheless, one can see through the proof that the set of barcodes is univocally determined regardless of the chosen generator set. Therefore, the specific basis selection as described in the previous section is not relevant for the retrieved set of barcodes.

Barcodes

Persistence intervals form what is commonly referred to as a *barcode*. As popularized by the foundational paper of Zomorodian and Carlsson (2005) [59], this visual representation directly encodes the algebraic structure derived earlier. Each bar in the diagram corresponds to a specific generator (or summand) from the Structure Theorem decomposition: the bar begins at the generator's birth index and extends until its death index (or to infinity if the feature persists).

2.6 Barcode Calculation Algorithms

While the Structure Theorem guarantees the existence of a finite set of unique generators tracking cycles through the filtration, for practical applications we require a constructive method to explicitly compute these generators and the resulting barcodes.

2.6.1 Computational Framework

These computations are performed by Topological Data Analysis software packages, such as GUDHI. While modern libraries often implement optimized variants (such as dual algo-

rithms or cohomology computations), the standard approach relies on matrix reduction algorithms over a field. As described in the foundational paper by Edelsbrunner et al. (2002), the core algorithm is essentially a variation of Gaussian elimination performed over $\mathbb{Z}/2\mathbb{Z}$.

2.6.2 Matrix Reduction Algorithm

Matrix Representation

First, we construct a matrix representation of the boundary operator ∂_k with entries in $\mathbb{Z}/2\mathbb{Z}$.

- The columns represent the k-simplices of the complex, ordered by their filtration index.
- The rows represent the $(k-1)$-simplices, also ordered by their filtration index.

We define the boundary matrix M_k such that the entry $M_k(i,j) = 1$ if the $(k-1)$-simplex at row i is a face of the k-simplex at column j, and 0 otherwise.

Property: Upper-Triangularity

It is a fundamental property that this matrix M_k is strictly upper-triangular. This is a direct consequence of the filtration condition: a simplex can only be included in the complex after all of its faces have been included. Therefore, if the indices i and j respect the filtration order, any face σ_i of a simplex σ_j must satisfy $\text{index}(\sigma_i) < \text{index}(\sigma_j)$, implying $i < j$ for all non-zero entries.

The Reduction Algorithm

We define $\text{low}(j)$ as the row index of the lowest non-zero value in column j. We say that a matrix is reduced if the map $\text{low}(\cdot)$ is injective for non-zero columns, meaning no two columns share the same pivot row. The algorithm proceeds by reducing the matrix M_k, processing the columns from left to right.

Algorithm 1 Matrix Reduction Algorithm

```
for j ← 0 to m − 1 do
    while exists i < j such that (M_k)_i ≠ 0 and low(i) = low(j) do
        Add column i to column j (mod 2);
    end
end
```

Algebraically, every column operation corresponds to a change of basis in the chain group $C_k(K_j, \mathbb{Z}/2\mathbb{Z})$. Since adding two columns is equivalent to multiplying by an elementary matrix (which is invertible), this preserves the homology structure. Furthermore, the process of adding only columns $k < j$ to column j preserves the filtration order.

Interpretation: Positive and Negative Simplices

The reduced matrix allows us to classify simplices based on the topological events they trigger. A simplex is called *positive* if it creates a new homology class, and *negative* if it destroys one (by merging classes or filling a cycle). Once the matrix is reduced, we iterate through the columns to classify them (Edelsbrunner et al., 2002 [19]):

- If a column j contains only zeros, the simplex is a positive k-simplex. It creates a cycle that has not yet died, retrieving a persistence pair (j, ∞).

- If a column j has non-zero entries, it represents a negative simplex. This generates a persistence pair (i, j), where $i = \text{low}(j)$. Here, i represents the youngest positive $(k-1)$-simplex in the boundary of the simplex at column j .

Output: Barcodes

The collection of pairs (i, j) derived from the negative columns, along with the unpaired positive indices (j, ∞) are visualized as intervals $[i, j)$ and $[j, \infty)$, commonly referred to as barcodes. See Figure 2.3 (left) for a visual representation.

Relation of the Algorithm with the Structure Theorem

We find a correspondence between the free components of the Structure Theorem decomposition and the columns with all entries 0. Analogously, the columns with non-zero entries correspond to the torsion components of the decomposition.

2.6.3 Algorithm Complexity

As stated in the papers by Zomorodian and Carlsson (2004)[59] and Edelsbrunner et al. (2002), the worst-case time complexity of the algorithm is $O(m^3)$, where m is the total number of simplices (bounding the number of $(k-1)$-simplices and k-simplices).

This is due to the fact that reducing a single column j may require up to $O(m^2)$ operations. In the worst-case scenario, for a given column j, we may encounter a pivot collision with every preceding column $k < j$. Since there are j potential predecessors, and each column addition involves processing up to m entries (the maximum length of a column), the computational cost to reduce column j is proportional to $j \times m$, which is bounded by $O(m^2)$. Repeating this process for all m columns results in a total time complexity of $O(m^3)$. However, the boundary matrix sparsity reduces the complexity notoriously.

2.7 Persistence Diagrams

For a given barcode $B = \{[b_i, d_i)\}_{i \in I} \cup \{[b_j, \infty)\}_{j \in J}$, where the intervals may appear with multiplicity, its *persistence diagram* is defined as a multiset of points in the extended plane $\overline{\mathbb{R}}^2$ (where $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, +\infty\}$), given by

$$D(B) = \{(b_i, d_i)\}_{i \in I} \cup \{(b_j, \infty)\}_{j \in J}.$$

Since the birth of a topological feature must precede or coincide with its death, the first coordinate is always less than or equal to the second ($b \leq d$). Consequently, all points lie in the upper-left half-plane, on or above the diagonal $\Delta = \{(x, x) \mid x \in \overline{\mathbb{R}}\}$. The vertical distance of a point from the diagonal corresponds to the lifetime (or persistence) of the associated feature, given by $d_i - b_i$. Points closer to the diagonal typically represent noise.

Remark 2.20. In computational implementations and visualizations, points with an infinite death coordinate ($d = \infty$) are typically substituted by a constant value slightly larger than the maximum finite death value of the dataset. Furthermore, since the persistence diagram is a multiset, multiple features may map to the exact same coordinates, this multiplicity is usually visualized by representing the point with increased size.

Figure 2.3 illustrates the correspondence between the barcode representation (intervals) and the persistence diagram (points).

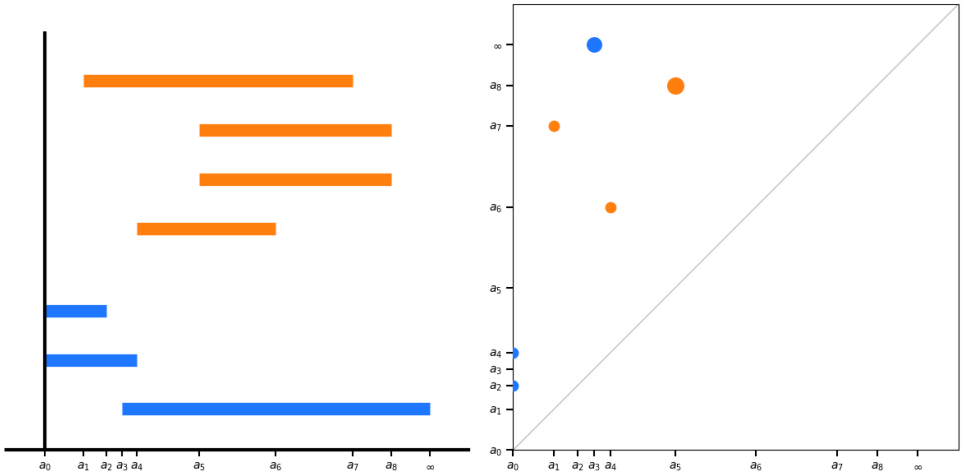


Figure 2.3: At the left we have an example of a persistence barcode and its corresponding persistence diagram at the right. We see how the point (a_5, a_8) is thicker since is represents two bars. The different colors indicate the different homological dimensions.

2.8 Numerical Descriptors

To integrate persistent homology into machine learning pipelines, we must transform the raw barcode (a multiset of intervals) into a fixed-size vector or scalar representation. Two of the most widely used scalar descriptors are *Total Persistence* and *Persistence Entropy*. Later in this work, we will explore more advanced methods for generating descriptors using deep learning architectures, such as PointNet.

Total Persistence

For a given barcode $B = \{[b_i, d_i)\}_{i \in I} \cup \{[b_j, \infty)\}_{j \in J}$, where we only consider the finite intervals ($d_i < \infty$), the Total Persistence is defined as the sum of the lifetimes of all features:

$$TP(B) = \sum_{i \in I} (d_i - b_i).$$

In the context of a filtered complex, this value is typically computed separately for the barcode of each homology dimension (e.g., H_0, H_1).

Persistence Entropy

Persistence Entropy is derived from Shannon entropy and measures the diversity of persistence interval lifetimes, effectively quantifying the "uncertainty" of the persistence diagram. First, we define a probability distribution $P = \{p_i\}_{i \in I}$ based on the relative contribution of each interval to the total persistence,

$$p_i = \frac{d_i - b_i}{TP(B)}.$$

The Persistence Entropy is then defined as the Shannon entropy of the distribution P:

$$PE(B) = -\sum_{i \in I} p_i \log(p_i).$$

As established by Atienza et al. (2020), Persistence Entropy exhibits stability under small perturbations of the input data. Specifically, under certain conditions, persistence entropy is continuous with respect to the Wasserstein distance.

2.9 From Images to Cubical Complexes

We choose to work with cubical complexes rather than simplicial complexes, as they naturally align with the intrinsic grid structure of digital images. A cubical complex is typically derived from an image using one of two primary methods: the *V-construction* or the *T-construction* (Bleile et al., 2022[8]).

Formally, we define a grayscale d-dimensional image $\mathcal{I}$ of size $(n_1, \ldots, n_d)$ as a map $\mathcal{I} : I \longrightarrow \mathbb{R}$, defined on a discrete domain I:

$$I = \{(l_1, \ldots, l_d) \in \mathbb{N}^d \mid 1 \leq l_i \leq n_i \text{ for all } i\}.$$

An element $(l_1, \ldots, l_d) \in I$ is called a *voxel* (or a *pixel* in the specific case where $d = 2$). The value $\mathcal{I}(l_1, \ldots, l_d)$ represents the voxel intensity (or pixel intensity when $d = 2$).

2.9.1 *V*-construction (Vertex Construction)

In this construction, the image voxels are defined as the vertices (0-cubes) of the cubical complex.

Formally, given a d-dimensional image of size $(n_1, \ldots, n_d)$, we define the complex $V(I)$ built from the grid $[1, n_1] \times \cdots \times [1, n_d]$, composed of elementary cubes of every dimension. Specifically, a voxel at coordinates $\mathbf{x} = (x_1, \ldots, x_d)$ is associated with the vertex $v = (x_1, \ldots, x_d)$ in the complex. The resulting complex is composed of $(n_1 - 1) \cdot \ldots \cdot (n_d - 1)$ elementary d-cubes connecting these vertices.

We define an associated filtration function $V(\mathcal{I}) : V(I) \longrightarrow \mathbb{R}$. The intensity values of the pixels are assigned to the vertices. For any higher-dimensional cube Q (where $1 \leq \dim(Q) \leq d$), its value is determined by the maximum intensity of its constituent vertices:

$$V(\mathcal{I})(Q) = \max\{V(\mathcal{I})(v) \mid v \text{ is a vertex of } Q\}.$$

This definition ensures monotonicity: if $P, Q \in V(I)$ are cubes such that $P \subseteq Q$, then the vertices of P are a subset of the vertices of Q, implying that $V(\mathcal{I})(P) \leq V(\mathcal{I})(Q)$. Consequently, this function induces a valid sublevel filtration.

To obtain a superlevel filtration, one simply replaces the maximum with the minimum in the definition above.

Remark 2.21. This construction relies on direct connectivity. In topological terms, this corresponds to the Manhattan distance: two voxels are connected if their distance is exactly 1. This retrieves $2d$ neighbors (4-connectivity in 2D, 6-connectivity in 3D).

2.9.2 *T*-construction (Top-cell Construction)

In this construction, the image voxels are defined as the highest-dimensional cells of the cubical complex (e.g., squares in 2D images, cubes in 3D images). Formally, we define $T(I)$ as the cubical complex constructed from the grid $[0, n_1] \times \cdots \times [0, n_d]$, composed of elementary cubes of every dimension. We associate the voxel at coordinates $\mathbf{x} = (x_1, \ldots, x_d)$ with the elementary d-cube defined by the interval product:

$$Q_{\mathbf{x}} = [x_1 - 1, x_1] \times \cdots \times [x_d - 1, x_d].$$

We define the associated filtration function $T(\mathcal{I}) : T(I) \longrightarrow \mathbb{R}$. The intensity values of the pixels are assigned to the d-cubes. For any lower-dimensional face P (where $0 \leq \dim(P) < d$), its value is given by the minimum intensity of the d-cubes that contain it:

$$T(\mathcal{I})(P) = \min\{T(\mathcal{I})(Q) \mid Q \text{ is a } d\text{-cube in } T(I) \text{ and } P \subseteq Q\}.$$

This definition ensures monotonicity: if $P \subseteq Q$, then any top-cell that contains Q must necessarily contain P. Consequently, the set of top-cells containing Q is a subset of the set of top-cells containing P. This implies from the definition $T(\mathcal{I})(P) \leq T(\mathcal{I})(Q)$. Thus, this function induces a valid sublevel filtration. To obtain a superlevel filtration, one replaces the minimum with the maximum.

This construction effectively models indirect connectivity. Since vertices and edges are shared by multiple top-cells, two voxels are considered connected if they share any common face. Retrieving $3^d - 1$ neighbors (8-connectivity in 2D, 26-connectivity in 3D).

2.9.3 Comparison of V-constructions and T-constructions

Figure 2.4 illustrates the cubical complexes obtained from the grid on the left via the V-construction and the T-construction, along with their corresponding filtration functions.

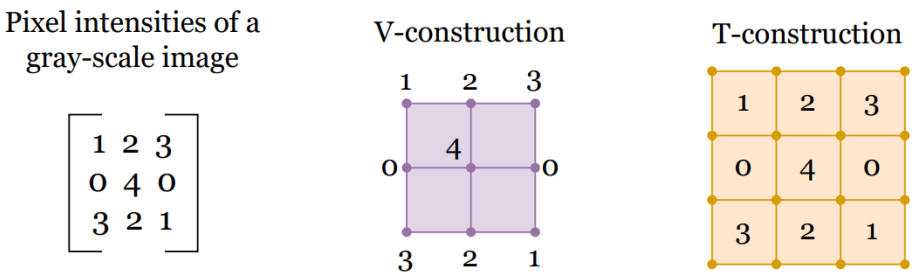


Figure 2.4: Visual comparison of V-construction and T-construction on a sample grid. Image extracted from Ferrà [26].

As Figure 2.5 demonstrates, these two constructions are not topologically equivalent and may yield different barcodes for the same image. Nevertheless, there exists a certain duality between the two constructions, specifically between the sublevel sets of the V-construction and the superlevel sets of the T-construction, which will be discussed in the following section.

2.10 Duality Between Constructions

As seen previously, the V-construction and T-construction do not yield identical sublevel filtrations. Nevertheless, they satisfy an almost-duality relationship, where the vertices in the V-construction correspond to the top-dimensional cells in the T-construction.

Formally, let $\mathcal{I}$ be an image. We define a padded image $\mathcal{I}^p$ by adding an external border of voxels with a value greater than all intensity values in the original image $\mathcal{I}$ (Figure 2.6). The theorem relating the T-construction to the V-construction states that the persistence diagram of the former coincides with the persistence diagram of the V-construction applied to the negated image $-\mathcal{I}^p$ (corresponding to the superlevel sets of $\mathcal{I}^p$).

We denote the total barcode of a filtered complex K by $B(K)$ (the union of every complex dimension barcode). For every bar $[b, d)_k$, the subscript k indicates the homological dimension. The following theorem formally relates the barcodes of the two constructions.

Theorem 2.22. *Let $\mathcal{I} : I \longrightarrow \mathbb{R}$ be a grayscale image of dimension m. The barcodes of the T-construction and V-construction satisfy the following relations:*

$$B(T(\mathcal{I})) = (\{[-d, -b)_{m-k-1} \mid [b, d)_k \in B(V(-\mathcal{I}^p))\} \setminus \{[\min \mathcal{I}, N)_0\}) \cup \{[\min \mathcal{I}, \infty)_0\}$$

$$B(V(\mathcal{I})) = (\{[-d, -b)_{m-k-1} \mid [b, d)_k \in B(T(-\mathcal{I}^p))\} \setminus \{[\min \mathcal{I}, N)_0\}) \cup \{[\min \mathcal{I}, \infty)_0\}$$

where N is the padding value (greater than $\max \mathcal{I}$).

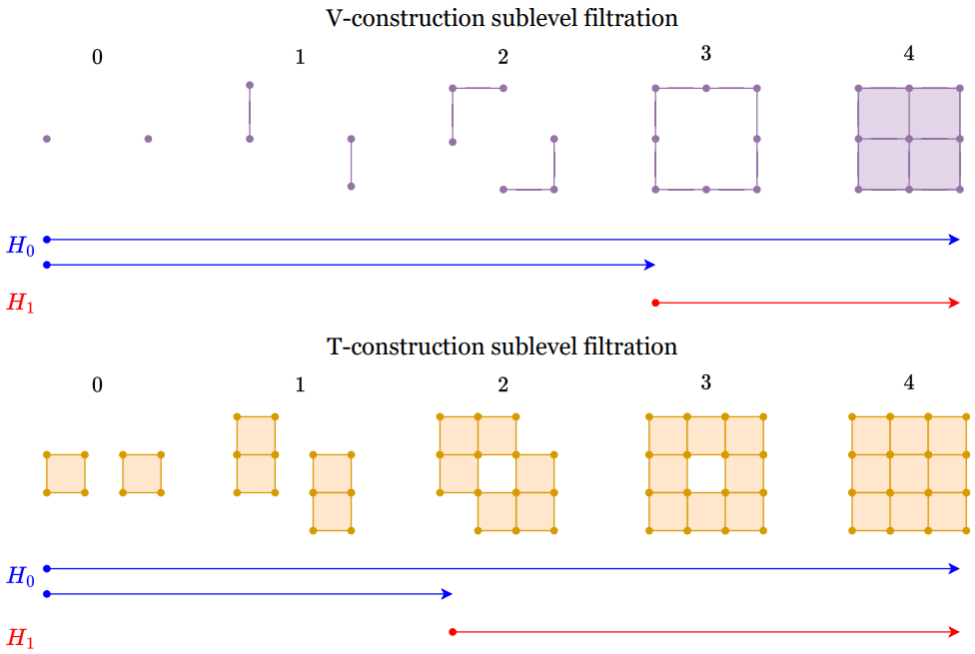


Figure 2.5: Comparison of barcodes generated by the V-construction and T-construction from Figure 2.4. Image extracted from Ferrà [26].

$$\begin{bmatrix} N & N & N & N & N \\ N & 5 & 1 & 6 & N \\ N & 2 & 9 & 4 & N \\ N & 8 & 3 & 7 & N \\ N & N & N & N & N \end{bmatrix}$$

Figure 2.6: Example of a padded image, where $N > 9$ (extracted from Bleile et al. (2022) [8]).

The proof of this theorem can be found in Bleile et al. (2022), Section 6, Theorems 6.1 and 6.2 [8]. The almost-duality can be seen in Figures 2.7 and 2.8. We observe that the birth and death values (b and d) in one construction correspond to $-d$ and $-b$ in the other. This construction is consistent with the topology of images, where there clearly exists exactly one component with infinite lifetime. At the final filtration level, all components merge into a single connected component, which, by the elder rule, corresponds to the one born first. We also observe how the homological features of dimension k in one construction map to features of dimension $m - k - 1$ in the other.

2.11 Library: GUDHI

The library used in this work to compute persistence diagrams is GUDHI (Geometric Understanding in Higher Dimensions). It is an open-source library implemented in C++ with a Python interface, providing robust implementations of state-of-the-art algorithms for Topological Data Analysis (TDA).

All images are processed as two-dimensional structures, even when the original images

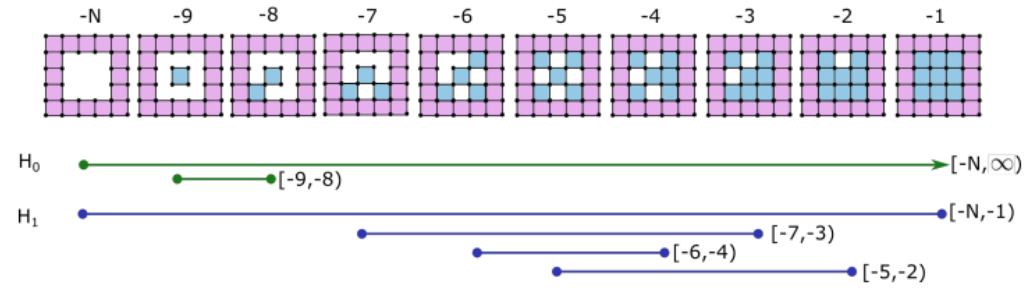


Figure 2.7: Example of a padded image V-construction barcode computation (extracted from Bleile et al. (2022) [8]).

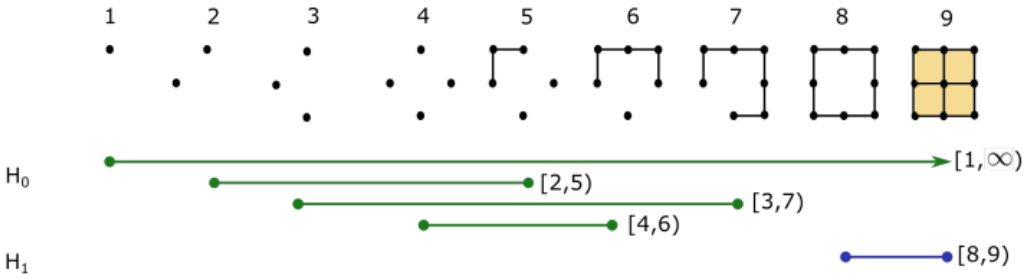


Figure 2.8: Example of a padded image T-construction barcode computation (extracted from Bleile et al. (2022) [8]).

possess a third dimension (such as RGB color channels). This approach relies on the observation that the third dimension typically contains color information, which does not fundamentally alter the underlying topological structure of the image domain.

For example, consider an image containing a loop that is partially red and partially blue. The underlying feature of interest is the loop itself, the variation in color does not create two distinct connected components, but rather represents the same topological feature with varying attributes.

Although GUDHI is designed to handle various types of simplicial complexes, we utilize GUDHI's implementation of cubical complexes, as this structure is naturally suited for the grid-based nature of digital images.

Chapter 3

Machine Learning Background

3.1 Machine Learning

Formally, a machine or computer program is said to learn if its performance measure P on a specific task T improves as a result of experience E. In such a case, we say that the program has learned from experience E [43].

Experience (E) refers to the dataset or interactions from which the program learns. In the context of this work, E corresponds to a dataset of images. Task (T) denotes the specific problem the program is designed to solve, such as classifying the images of the dataset. The Performance measure (P) is inherently tied to the task T and serves as a quantitative metric to evaluate the performance of the model. In classification problems, P is typically defined by accuracy, defined as the proportion of instances for which the model correctly predicts the corresponding label.

A well-trained model possesses the ability to generalize from its experience. Generalization is defined as the capacity to achieve high performance on previously unseen data by leveraging the patterns extracted during the learning process.

3.2 Supervised Learning: Classification Tasks

Our work focuses exclusively on supervised learning, therefore, although other learning paradigms exist (such as unsupervised and semi-supervised learning), we will limit our discussion to the scope of this end-of-degree project.

3.2.1 The Context of Supervised Learning

Learning scenarios are distinguished by the nature of the "experience" provided to the model. In supervised learning, the program is provided with a dataset consisting of feature vectors, each paired with an associated label (or labels). The "supervision" stems from the fact that we explicitly show the model the desired output for each input during the training phase.

Classically, a supervised learning model is defined as a function $f : \mathcal{X} \longrightarrow \mathcal{Y}$. Here, $\mathcal{X}$ represents the input space, typically modeled as d-dimensional vectors in $\mathbb{R}^d$ (or higher-dimensional tensors, as in the case of Convolutional Neural Networks). The target space $\mathcal{Y}$ is the discrete, finite set of all possible values associated with an input from $\mathcal{X}$.

3.2.2 Classification Tasks

One of the most fundamental supervised learning tasks is classification. Since this work is devoted exclusively to classification, we will not cover ranking or regression. In this setting,

the algorithm must map every input to a category (label). We distinguish three types of classification based on the cardinality and structure of the target set $\mathcal{Y}$:

- **Binary Classification:** The cardinality of $\mathcal{Y}$ is 2. The label set is typically defined as $\mathcal{Y} = \{0, 1\}$. Examples include binary image recognition (e.g., "cat" vs. "not cat") or spam filtering ("spam" vs. "not spam").
- **Multiclass Classification:** The cardinality of $\mathcal{Y}$ is greater than 2, but every input is associated with exactly one label. The label set is defined as $\mathcal{Y} = \{0, 1, \ldots, C-1\}$. An example is a dataset of animal images where each image contains exactly one distinct species.
- **Multilabel Classification:** When a single input may be associated with multiple labels simultaneously. Here, the target space is often defined as $\mathcal{Y} = \{0, 1\}^C$, where the i-th coordinate is 1 if the i-th class is present, and 0 otherwise. This is common in object detection, where an image may contain multiple distinct objects.

3.2.3 Formal Framework: Data and Optimization

The "experience" of a trained model is encapsulated in the training set:

$$\mathcal{T}_{train} = \{(x_i, y_i)\}_{i=1}^{N} \subseteq \mathcal{X} \times \mathcal{Y}.$$

To evaluate the model's performance, we define an analogous test set:

$$\mathcal{T}_{test} = \{(x_j, y_j)\}_{j=1}^{M} \subseteq \mathcal{X} \times \mathcal{Y}.$$

The test set represents data the model has not seen before, used to measure its ability to generalize. The full domain is $\mathcal{D} = \mathcal{T}_{train} \cup \mathcal{T}_{test}$, and it is a standard assumption that both sets are sampled from the same underlying probability distribution. However, due to diverse factors, they may in practice exhibit distinct distributions.

From a probabilistic perspective, supervised learning can be viewed as estimating the conditional probability $p(y \mid x)$ for an input $x \in \mathcal{X}$. For single-label classification tasks, function f is typically defined as selecting the target with the highest estimated probability:

$$f(x) = \underset{y \in \mathcal{Y}}{\operatorname{argmax}}\, p_{\text{estimated}}(y \mid x).$$

In the multilabel case, the task is typically performed as a binary classification for every label simultaneously. The model outputs a probability for each class independently, and we select all labels y_i such that the estimated probability exceeds a specific threshold (typically 0.5).

Consequently, we can treat the definition interchangeably, interpreting the target space $\mathcal{Y}$ either as the set of discrete class labels or as the vector space of probability distributions over those categories (e.g., via one-hot encoding).

We define our model space as a family of functions parameterized by θ:

$$\mathcal{F} = \{f_\theta : \mathcal{X} \longrightarrow \mathcal{Y}\}_{\theta \in \Theta},$$

where $\Theta \subseteq \mathbb{R}^n$ is the parameter space. In this context, f_θ outputs the predicted probability distribution over the categories.

In the context of a classification task, the goal of the learning algorithm is to select the optimal parameters $\theta^* \in \Theta$ that maximize the accuracy of predictions on the training set. This is formalized as finding the optimal parameters θ^* that minimizes an error function, known as the loss function, over the training set. Specifically, we seek to find the parameters that minimize:

$$\theta^* = \underset{\theta \in \Theta}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(f_\theta(x_i), y_i).$$

Here, $\mathcal{L} : \mathbb{R}^{|\mathcal{Y}|} \times \mathcal{Y} \longrightarrow \mathbb{R}_{\geq 0}$ is a loss function that evaluates the model by penalizing incorrect predictions and rewarding correct ones (ideally 0 for a perfect prediction). Note that strictly speaking, θ^* may not exist if the infimum of the loss is not attainable.

3.3 Deep Neural Networks

Deep Neural Networks (DNNs) provide a powerful mechanism by enabling models to learn complex hierarchical structures and patterns. Formally, a DNN is defined as the composition of a sequence of functions, or layers:

$$\mathcal{F} = \{f_{\theta_L} \circ \cdots \circ f_{\theta_1} : \mathcal{X} \longrightarrow \mathcal{Y}\}_{\theta=(\theta_1,\ldots,\theta_L)\in\Theta}.$$

Each function f_{θ_i} represents the i-th layer of the network. Typically, a layer consists of two distinct operations:

1. Affine Transformation: A linear mapping defined by $x_i = W_i x_{i-1} + b_i$, where the matrix W_i (weights) and the vector b_i (bias) constitute the learnable parameters $\theta_i = \{W_i, b_i\}$.

2. Non-Linear Activation: A function σ applied to the output of the linear step.

Thus, the propagation through layer i is defined as:

$$x_i = f_{\theta_i}(x_{i-1}) = \sigma_i(W_i x_{i-1} + b_i).$$

The most common choices for the activation function σ are:

- **Sigmoid:** $\sigma(z) = \frac{1}{1+e^{-z}}$ applied element-wise to the tensor. This maps values to the range $(0, 1)$, historically used for probabilities.

- **ReLU (Rectified Linear Unit):** $\sigma(z) = \max(0, z)$ applied element-wise to the tensor. This is the standard choice for modern deep networks as it significantly improves gradient propagation.

Convolutional Neural Networks (CNNs) and Transformers are examples of Deep Neural Networks.

3.3.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) emphasize the spatial structure of the data, which is particularly relevant for images, the focus of this end-of-degree project. Convolution is defined as an operation applied to a two-dimensional matrix M using a smaller matrix $K \in \mathbb{R}^{m\times n}$, called a kernel. This produces an output matrix defined as:

$$S_{i,j} = \sum_m \sum_n M_{m,n} K_{i-m,j-n}.$$

For 3-dimensional matrices and 2-dimensional kernels, the operation is simply applied across the third axis.

Convolutional Component of the Layer

The primary component is the convolutional layer, which consists of applying the convolution operation previously defined to an image represented as a matrix. This process involves sliding the kernel spatially over the input image, with a fixed number of rows (or columns), d, separating consecutive positions. This interval, determining the jump from one kernel position to the next, is called the stride. Figure 3.1 summarizes this process:

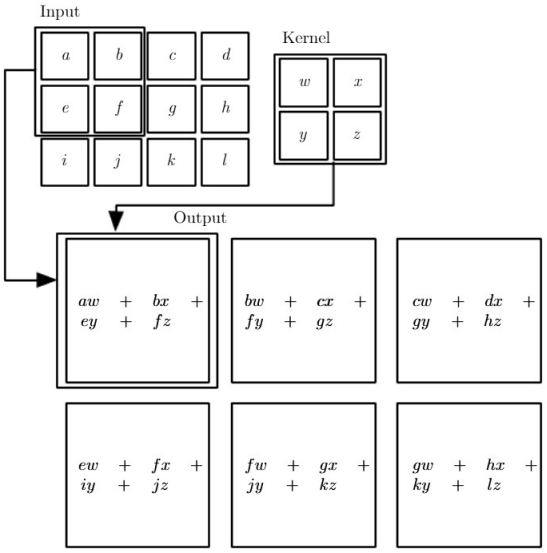


Figure 3.1: The figure shows an example of convolving with stride 1. (Image from Goodfellow et al. [29])

This spatial processing allows the model to learn to detect specific patterns corresponding to each convolution kernel. For example, in a "face vs. not face" classification task, deeper kernels might specialize in detecting high-level features like eyes or mouths. Formally, convolutional layers can be defined using matrix multiplication, similar to the general definition of DNN layers, but with the constraint that appropriate parameters are shared across the spatial dimensions.

Pooling Component of the Layer

Following the activation function, the final and optional stage is the pooling layer. This layer serves to reduce the spatial dimensions of the input while preserving its dominant features.

Formally, we define a pooling function P that operates on a local neighborhood. The resulting matrix S at position (i, j), derived from the input matrix M using a window of size $m \times n$, is given by:

$$S_{i,j} = P(M_{i,\ldots,i+m;\ j,\ldots,j+n}).$$

The most widely used variants are Max Pooling, where P selects the maximum value within the neighborhood, and Average Pooling, where P computes the arithmetic mean of the values in the neighborhood.

Full Connected Layer

When the layer parameter matrix W_i connects every input unit to every output unit (i.e., it is a dense matrix), the layer is referred to as a Fully Connected (FC) or Dense layer.

3.3.2 Transformers

While CNNs excel at leveraging the spatial structure of data, tasks involving inherently sequential data, such as natural language, were historically dominated by Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks, which provide a memory mechanism. Transformers, as introduced by Vaswani et al. [56], dispense with recurrence and convolutions entirely, relying solely on a mechanism called attention to establish global dependencies between input and output.

The core component of the Transformer is the attention function, which maps a query and a set of key-value pairs to an output. The specific mechanism employed is called Scaled Dot-Product Attention.

The input consists of queries (q) and keys (k) of dimension d_k, and values (v) of dimension d_v, typically packed together into matrices Q, K and V respectively. The attention function is defined as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V.$$

To enable the model to attend to information from different representation subspaces at different positions, the Transformer utilizes Multi-Head Attention. This is defined as:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O$$

where each head is computed as $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$, and W^O is the output projection matrix [56].

Here, the different representation subspaces are induced by the specific projection matrices (W_i^Q, W_i^K, W_i^V) associated with each head. These multiple heads allow the model to attend to various aspects of the input simultaneously.

Relating Transformers to the typical DNN structure $\sigma(y = Wx + b)$, the role of the activation function, which determines where the model focuses, is fulfilled by the Attention function. The matrix operations are performed on the three inputs, with all biases set to 0, as these are projection matrices typically mapping data to lower-dimensional spaces.

As previously noted, the Multi-Head Attention mechanism is a combination of two layers of projection matrices applied to the three inputs, followed by an activation function consisting of the Attention mechanism. The queries, keys, and values can originate from the same input or different inputs depending on the encoder-decoder architecture. However, since this work does not delve deeply into Transformers, we will not detail this further.

Although Transformers originally emerged as tools for natural language processing, their high degree of parallelization and ability to model global context have made them highly effective for computer vision tasks, leading to adaptations such as the Swin Transformer V2 [39], which adapts transformers for vision tasks by computing attention locally within image windows, utilizing a shifted window mechanism to establish global context. Furthermore, it introduces residual-post-normalization and scaled cosine attention to enhance stability in high-resolution tasks. These innovations facilitate a higher degree of efficiency while effectively granting the model higher capacity (more details about our implementation can be found in Subsection 4.1.2).

3.4 Capacity

A central challenge in machine learning is ensuring that models perform as well as on unseen data (test data) as they do on their training experience (train data). This ability is governed by the capacity of the model, which corresponds to its complexity or the amount of information it can retain. Formally, this is defined by the richness of the hypothesis space $\mathcal{F}$.

We distinguish between two primary failure modes related to capacity:

- **Overfitting:** This occurs when the capacity of the model is too high relative to the amount of available training data. Instead of learning generalizable patterns, the model effectively "memorizes" the training set, even if noisy. This phenomenon is characterized by a low training loss but a significantly higher test loss over the training period.
- **Underfitting:** This occurs when the capacity of the model is too low to capture the underlying structure and complexity of the dataset. This results in the model being unable to sufficiently minimize the loss, leading to poor performance on both the training and test sets.

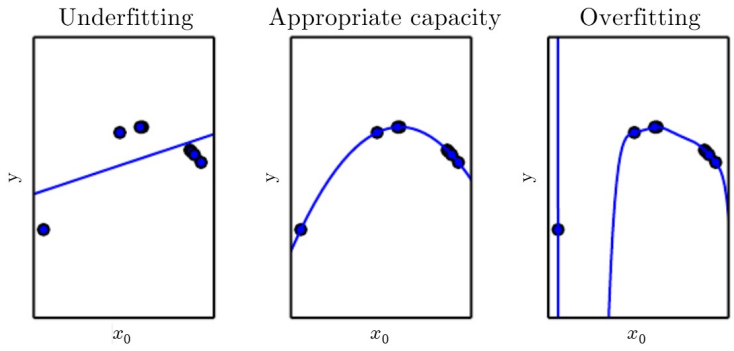


Figure 3.2: Three models fits the training set composed by the represented inputs in the x axis and their label on the vertical axis. Model on Left is underfitting since it cannot capture the pattern of the data making the training loss too big. On the other hand, the right one even having a zero loss is overfitting since it does not learn the underlying patterns understanding that the central one is the model that generated the data (Image from Goodfellow et al. [29]).

3.5 Regularization

We refer to regularization as any modification made to a learning program or its training process that is intended to reduce the generalization error (test loss) without necessarily reducing the training loss. These strategies are the primary defense against overfitting in high-capacity models. The most common techniques include:

- **Weight Decay (L^2 Regularization):** This technique involves adding a penalty term to the loss function proportional to the squared magnitude of the model parameters. This forces the weights to remain small, preventing the model from relying excessively on any single feature or learning effectively noise. The modified objective function becomes:

$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}(f_\theta(x_i), y_i) + \frac{\lambda}{2}\|\theta\|_2^2,$$

  where λ is a hyperparameter that controls the regularization strength.
- **Early Stopping:** Consists of monitoring the loss (or accuracy) on unseen data, typically a validation set, and halting the learning process when performance begins to degrade.

3.6 How the algorithm finds the optimal parameters?

As previously established, a machine learning task can be reduced to an optimization problem, necessitating algorithms capable of minimizing the loss function. Unlike simpler

models (e.g., linear regression), for DNNs no closed-form solution exists, thus, we must rely on iterative methods to approximate the minimum.

The most fundamental of these is the Gradient Descent algorithm. Formally, it is defined iteratively by updating the parameter vector θ in the opposite direction of the gradient:

$$\theta_{t+1} = \theta_t - \mu \nabla_\theta \mathcal{L}(\theta_t), \tag{3.1}$$

where the hyperparameter μ is called the learning rate. This hyperparameter controls the step size of each iteration, effectively determining how fast the model converges to the optimal parameters. Or, intuitively, how fast the model "learns".

To compute these gradients efficiently, a strategy known as backpropagation is employed. Rather than computing gradients individually, backpropagation systematically applies the chain rule of calculus starting from the output layer and moving backward through the network. It employs the activation values stored during the forward pass to efficiently calculate the partial derivatives of the loss function with respect to every parameter.

3.6.1 Loss Functions

The standard metric for evaluating classification performance is the Cross-Entropy loss, which measures the divergence between the estimated probability distribution and the true distribution. In the context of this work, we employ two variations of this metric:

- **Cross-Entropy Loss (Standard):** This is the standard loss for multiclass classification tasks. Let the set of classes be defined as $\mathcal{Y} = \{0, \dots, C-1\}$. We represent the ground truth as a one-hot encoded vector y, where the entry corresponding to the true class is 1 and all others are 0. The model is expected to predict a probability distribution over the labels, denoted as $\hat{y} \in [0,1]^C$ (typically the output of a Softmax function). The loss function is defined as:
$$\mathcal{L}_{CE}(\hat{y}, y) = -\sum_{i=0}^{C-1} y_i \log(\hat{y}_i).$$

- **Weighted Cross-Entropy Loss:** In scenarios where the dataset is imbalanced, there is a risk that the model will bias its predictions toward the classes with the highest frequency. To mitigate this, we modify the loss function by introducing a weight w_i for each class (typically inversely proportional to the class frequency), ensuring that minority classes contribute more significantly to the error signal:
$$\mathcal{L}_{WCE}(\hat{y}, y) = -\sum_{i=0}^{C-1} w_i y_i \log(\hat{y}_i).$$

3.6.2 Stochastic Gradient Descent (SGD)

Computing the exact gradient over the entire training dataset is often computationally prohibitive. To address this bottleneck, Stochastic Gradient Descent (SGD) approximates the gradient by computing the loss function only on a reduced subset of the training inputs. We refer to this subset as a batch (or mini-batch).

The optimization algorithm partitions the training dataset into subsets of a fixed batch size and applies the update rule (Equation 3.1) based on the loss calculated for that specific batch. An epoch is defined as one complete pass through the entire training set, occurring when all batches have been processed.

3.6.3 Adaptive Moment Estimation (ADAM)

SGD has inherent limitations, for instance, in scenarios where the loss function forms a "ravine", steep in one dimension and flat in another, SGD tends to oscillate across the steep slopes while making very slow progress along the flat valley floor.

Adaptive Moment Estimation (ADAM) addresses these issues by combining the advantages of other optimization strategies such as RMSProp and Momentum.

We define the moment updates as:

$$m_{t+1} = \beta_1 m_t + (1 - \beta_1)\nabla_\theta \mathcal{L}(\theta_t)$$
$$v_{t+1} = \beta_2 v_t + (1 - \beta_2)\left(\nabla_\theta \mathcal{L}(\theta_t)\right)^2$$

To counteract the initialization bias, bias-corrected moment updates are computed as:

$$\hat{m}_{t+1} = \frac{m_{t+1}}{1 - \beta_1^{t+1}}, \quad \hat{v}_{t+1} = \frac{v_{t+1}}{1 - \beta_2^{t+1}}.$$

Here, β_1 and β_2 are hyperparameters typically set to $\beta_1 = 0.9$ and $\beta_2 = 0.999$. Finally, the parameter update rule is defined as:

$$\theta_{t+1} = \theta_t - \frac{\mu}{\sqrt{\hat{v}_{t+1}} + \epsilon}\hat{m}_{t+1},$$

where μ is the learning rate and ϵ is a small constant (e.g., 10^{-8}) added to prevent division by zero. Like SGD, ADAM is typically implemented using the mini-batch strategy.

Chapter 4

Methodology and Architecture

While CNNs and Transformers excel at capturing local spatial features and intensity patterns, they often overlook inherent global topological structures, such as connected components and loops. Building upon PHG-Net, the architecture proposed by Peng et al. [46], this work integrates image persistence diagrams to enhance the representational capacity of standard computer vision models, specifically ResNets and Swin Transformers.

4.1 The PHG-Net Architecture & Modifications

The architecture utilized, PHG-Net, is organized into two distinct streams: a vision stream and a topological stream. In the context of this work, the vision component employs backbone models such as ResNet-18, ResNet-50, ResNet-152, or the Swin Transformer V2-B. At specific intermediate stages, the extracted visual features are fused with topological feature vectors derived from the persistence diagrams of the images using a Pointnet-like encoder. The overall architecture is illustrated in Figure 4.1.

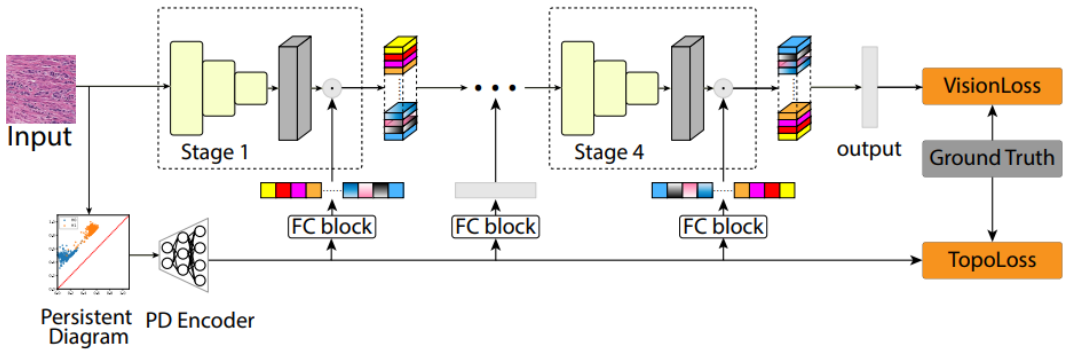


Figure 4.1: Architecture overview from [46].

Although many models share a similar architecture, the primary contribution and distinguishing aspect of this approach lies in the specific methodology used to transform persistence diagrams into feature vectors suitable for integration into machine learning pipeline. In related literature, persistence diagram descriptors are typically generated either by first encoding the persistence diagram and passing it through a dedicated neural network (e.g., PersLay [11]), or by employing handcrafted vectorization techniques such as Persistence Landscapes [9], Total Persistence (see Section 2.8), or Persistence Entropy (see Section 2.8).

4.1.1 Persistence Diagram Encoder

A pivotal component of the PHG-Net architecture is the use of a learnable encoder to generate the homology feature descriptor. Crucially, a persistence diagram should not be processed as an image, as its semantic value lies in the discrete set of (birth, death) coordinates rather than pixel intensities. Treating it as an image would introduce ambiguity, as varying visual rasterizations could correspond to the exact same underlying topological data.

Consequently, the Persistence Diagram Encoder must be designed to process an unordered set of points and remain invariant to permutation, that is, the output must not depend on the order in which the points are input. To address this requirement, PHG-Net utilizes a PointNet-like architecture, which effectively ensures permutation invariance over the persistence diagram point-set.

Mathematically, this encoder is defined as the composition of a shared local mapping and a global aggregation function:

$$f(p_1, \ldots, p_n) = g(m(p_1), \ldots, m(p_n)),$$

where

- $\{p_1, \ldots, p_n\}$ is the set of preprocessed persistence diagram points.
- The function m is implemented as a shared Multi-Layer Perceptron (MLP), which maps each point individually to a higher-dimensional feature space.
- To ensure permutation invariance, the aggregation function g utilizes a symmetric pooling operation. In this architecture, max pooling is employed to extract the most salient features across the set.

Before entering the network, the raw persistence points are augmented with additional coordinates to encode topological information explicitly. Typically, this involves adding an indicator for the homology dimension:

- A scalar label (e.g., 0 for connected components H_0, 1 for loops H_1).
- Alternatively, a one-hot encoded vector (e.g., $[1, 0]$ for H_0 and $[0, 1]$ for H_1).

We exclude H_2 (voids) features in this work, a decision that will be justified in subsequent sections along with the discussion of the final additional coordinate. The complete structure of the encoder is illustrated in Figure 4.2.

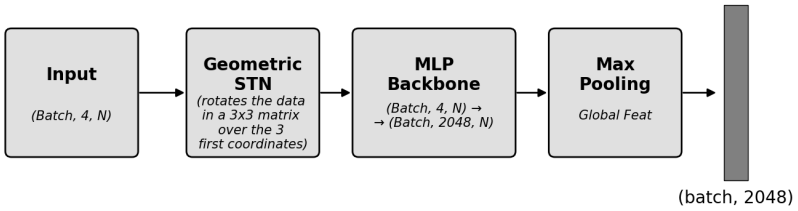


Figure 4.2: Structure of the Persistence Diagram Encoder adapted from [46]. It consists of a shared MLP applied to each point followed by a symmetric max-pooling operation.[1]

As illustrated in the previous figure, a Spatial Transformer Network (STN) is introduced preceding the MLP and pooling structure to canonically align the persistence points.[2]The

[1]Note: The Spatial Transformer Network (STN) is present in the official code, though not explicitly detailed in the original paper.

specific role of this module and an analysis of its actual benefits (or lack thereof) will be discussed in a subsequent chapter.

4.1.2 Vision Model

The vision component of the architecture employs standard backbone model, specifically ResNet-18, ResNet-50, ResNet-152, and Swin Transformer V2-B. These models are organized into 4 distinct stages, at the conclusion of each stage, the extracted topological vectors are fused with the visual feature maps.

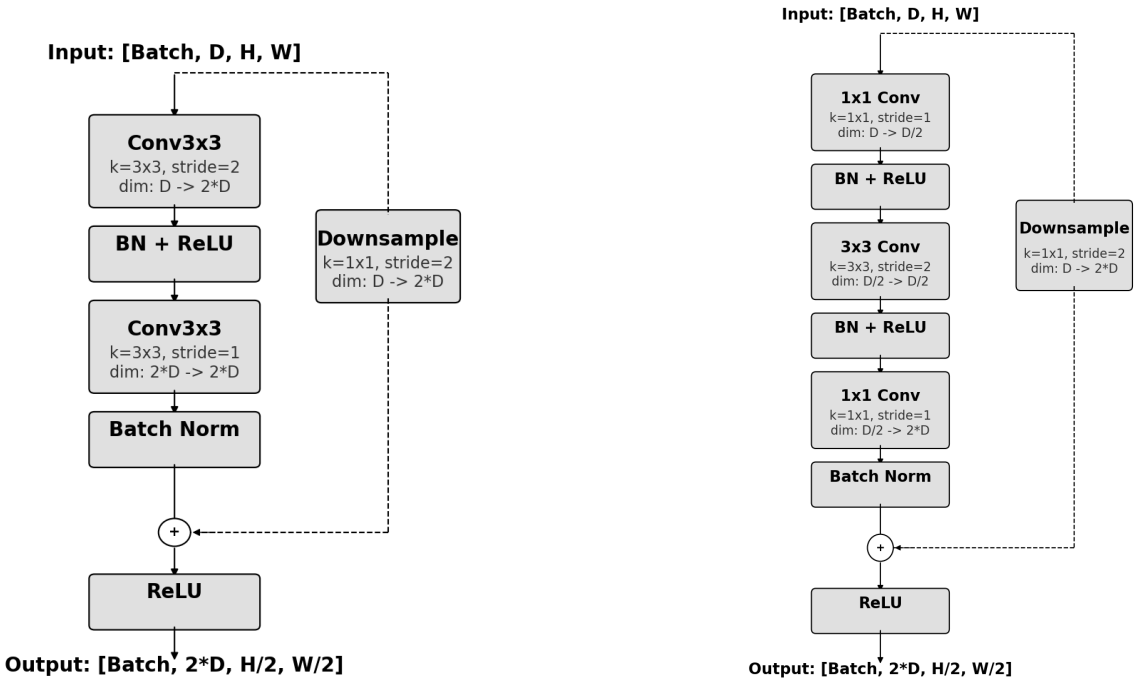
ResNet Backbones

For the ResNet architectures, the network begins with an initial stem module designed to reduce the spatial resolution of the input image. This module consists of:

1. A Convolutional layer with a 7×7 kernel and a stride of 2.
2. Batch Normalization followed by a ReLU activation function.
3. A Max-pooling layer with a 3×3 kernel and a stride of 2.

This stem is followed by four stages of residual blocks. ResNet-18 employs the standard BasicBlock [32], while ResNet-50 and ResNet-152 utilize the Bottleneck block structure [32]. The output channel dimensions for the BasicBlock stages are 64, 128, 256, and 512, respectively. Conversely, the Bottleneck stages produce outputs of size 256, 512, 1024, and 2048.

The core innovation of ResNet is the introduction of residual learning, which forces the network to learn the residual mapping rather than the direct mapping. This is achieved by adding the input of a block directly to its output: $y = \mathcal{F}(x) + x$.



(a) BasicBlock Structure

(b) Bottleneck Structure

Figure 4.3: Comparison of ResNet building blocks. (a) The standard BasicBlock used in ResNet-18. (b) The Bottleneck block used in deeper networks (ResNet-50/152) [32].

[2]While the usage of this STN was not explicitly documented in the original paper by Peng et al. [46], we discovered its implementation within their source code.

Swin Transformer V2-B

The Swin Transformer V2-B [39] follows a hierarchical structure similar to CNNs. Each stage consists of a sequence of Transformer blocks, where the core mechanism is Multi-Head Self-Attention (MSA) applied within local non-overlapping windows. The number of attention heads increases across the four stages, set to 4, 8, 16, and 32, respectively.

Each block processes the image using windows of size 8×8. The feature dimension is divided proportionally among the heads (e.g., if the total embedding dimension is 128 and there are 4 heads, each head processes a dimension of 32).

To enable cross-window connections, allowing information to propagate beyond the local 8×8 boundaries, the partitioning scheme alternates between two configurations:

Window-MSA (W-MSA): Used in even blocks. The image is partitioned starting from the top-left pixel $(0, 0)$ in a regular grid.

Shifted Window-MSA (SW-MSA): Used in odd blocks. The windowing grid is shifted by $([\frac{M}{2}], [\frac{M}{2}])$, where M is the window size pixels ($M = 8$, resulting in a shift of 4 rows and 4 columns). This displacement is illustrated in Figure 4.4.

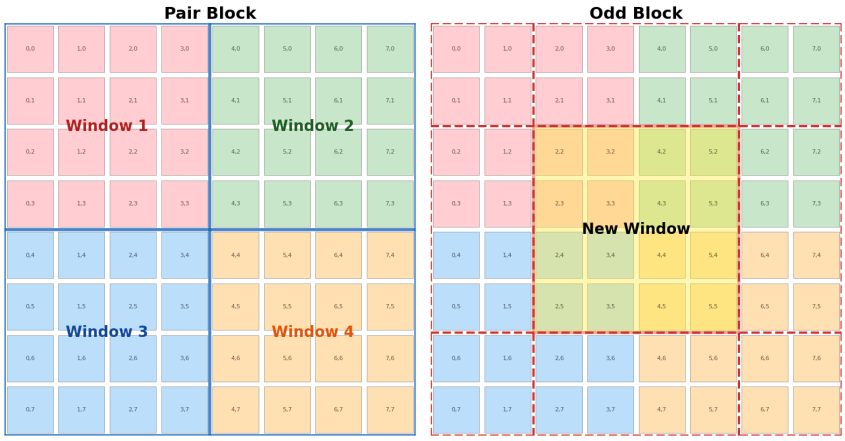


Figure 4.4: Illustration of the Shifted Window mechanism for window size of 4×4. While edges are processed via cyclic shifts to maintain window size, masking mechanisms ensure that non-adjacent pixels in the original image do not interact (i.e., do not 'talk' to each other). Pair Block: Pixels only talk to same colors. Odd Block: Different colors mix in the new windows

The Patch Embedding layer (Stage 1 input) serves a similar purpose to the ResNet stem. Given an input image of size $224 \times 224 \times 3$, it applies a convolution with a 4×4 kernel and a stride of 4. This reduces the resolution to 56×56 while projecting the input to 128 channels (output dimension: $56 \times 56 \times 128$). After the completion of all 4 stages and the fusion of topological features, the final output of dimension $(1024, 7, 7)$ is typically average-pooled to $(1024, 1, 1)$ for classification.

4.1.3 Persistent Homology Guidance

The guidance mechanism is applied at the end of each stage of the vision backbone, fusing the global homology descriptor, $t \in \mathbb{R}^N$, with the local vision feature map, v, which has channel dimension m. The dimensionality N depends on the architecture, with $N = 2048$ for ResNet backbones and $N = 1024$ for the Swin Transformer.

Since the topological vector has a fixed size of N while the vision channel dimension m varies by stage, t must first be projected to match the dimensions of v. This is achieved through

the FC Block (illustrated in Figure 4.1). The projection is defined as a two-layer perceptron:

$$h = \text{ReLU}(W_1 t + b_1)$$
$$t' = \sigma(W_2 h + b_2)$$

where:

- $W_1 \in \mathbb{R}^{\frac{m}{r} \times N}$ and $b_1 \in \mathbb{R}^{\frac{m}{r}}$ are the weights and biases of the reduction layer.
- $W_2 \in \mathbb{R}^{m \times \frac{m}{r}}$ and $b_2 \in \mathbb{R}^m$ are the weights and biases of the expansion layer.
- r is a reduction ratio parameter. In this work, we set $r = 8$, which balances model complexity and performance, consistent with the findings of Peng et al. [46]
- σ is the sigmoid activation function, ensuring the output values lie in the range $(0, 1)$.

It is important to note a discrepancy regarding this component: while the original paper describes a simple linear projection with biases initialized to zero, the official code implementation actually utilizes a single-layer linear structure. We will analyze the optimization of these architectural choices and their impact on performance in the next chapter.

Fusion Operation

Once the topological descriptor is transformed into the attention vector t', the final fusion is performed. While the paper states that this is done via a simple element-wise product ($v' = v \odot t'$), the code implements a residual gating mechanism:

$$v' = v + (v \odot t'),$$

where $\odot$ denotes the element-wise product.

We prioritized this second implementation for our experiments. Our reasoning is that if the topological vector is "noisy" or uninformative (i.e., t' values close to zero), the simple product method would suppress the vision features entirely, potentially discarding critical information. In contrast, the residual formulation ensures that the original vision features are preserved, allowing the topological signal to strictly enhance or modulate the features rather than destructively filtering them.

4.1.4 Classification Heads

Both classification heads are fully connected layers mapping the feature vector to the number of classes, followed by a softmax activation for classification tasks. During the experiments, we will discuss some modification explored.

The loss function is defined as a weighted sum of the vision model loss ($\mathcal{L}_V$) and the topological model loss ($\mathcal{L}_{Topo}$). Both components utilize Cross-Entropy loss, regulated by a balancing hyperparameter α. Let y denote the ground truth labels, while y_v and y_{topo} represent the output predictions of the vision head and topological head, respectively (see Figure 4.1):"

$$\mathcal{L} = \mathcal{L}_V(y_v, y) + \alpha \mathcal{L}_{topo}(y_{topo}, y)$$

4.2 Experimental Framework

The experimental framework focuses on two primary datasets to evaluate the PHG-Net architecture, along with our corresponding modifications. The ISIC2018 dataset [14], which was also utilized in the original paper, contains medical lesion images rich in topological structures. Additionally, we employed CIFAR-100 dataset [37] to evaluate the model's generalization capabilities across a dataset with large number of classes and with a balanced class distribution.

ISIC2018 (Skin Lesion Analysis Toward Melanoma Detection)

The primary dataset for this work is derived from Task 3 of the ISIC 2018 Challenge. The dataset consists of RGB images captured via dermoscopy [52], a non-invasive skin imaging technique. It consists of 10,015 training images and 193 validation images. We merged these sources into a single corpus of 10,208 images, classified into 7 diagnostic categories: Melanoma (MEL), Melanocytic nevus (NV), Basal cell carcinoma (BCC), Actinic keratosis / Bowen's disease (AKIEC), Benign keratosis (BKL), Dermatofibroma (DF), and Vascular lesion (VASC).

A critical characteristic of this dataset is its significant class imbalance, with common conditions such as Melanocytic nevus being overrepresented compared to rare classes. To ensure a representative evaluation, we employed a stratified split strategy, reserving 20% of the data for validation while ensuring that the original class distribution was maintained in both the training and validation sets.

CIFAR-100

To analyze the model's generalization performance over a wider array of categories, we selected the CIFAR-100 dataset. Created by Krizhevsky et al. [37] as a subset of the "80 Million Tiny Images" dataset, it consists of 60,000 RGB images of resolution 32×32. The dataset is perfectly balanced across 100 classes, which are organized into 20 superclasses (e.g., the 'food container' superclass encompasses bottles, bowls, cans, cups, and plates).

In contrast to the procedure used for ISIC2018, we adhered to the standard dataset partition provided by the authors, utilizing 50,000 images for training and 10,000 for testing.

Dataset Setup

All input images were resized to 224×224 to align with the standard input specifications of the vision backbones (ResNet and Swin Transformer).

While it is technically possible to modify the model's input layer to accept smaller dimensions (e.g., 32×32), we opted against this because our experiments rely heavily on pretrained weights. These weights encode kernels specialized in detecting features at a specific spatial scale (e.g., the curve of an eye or the texture of a lesion). Modifying the input resolution without resizing the image would alter the receptive field ratio, causing a significant scale mismatch that would render the pretrained weights ineffective for feature extraction.

4.3 Persistence Diagram Computation

Regarding the computation of persistence diagrams, while several configurable parameters were systematically evaluated (as detailed in the subsequent chapter), certain fundamental settings remained fixed throughout the experiments.

Most notably, the construction of the cubical complex from the input image requires a choice between the T-construction and the V-construction. Although a semi-duality relationship exists between these two methods, we consistently employed the T-construction for this work. We determined that this approach provides a more faithful representation of the underlying topology of the image, as it maps image pixels directly to the top-dimensional cells (elementary cubes or squares) of the complex, rather than ~~to~~ the vertices.

Other methodological aspects, such as the specific preprocessing of diagrams before the Persistence Diagram Encoder (PDE) or the normalization of images prior to filtration, were treated as experimental variables and are subject to detailed discussion in the next chapter.

All persistence diagrams were computed utilizing the GUDHI library, as noted in the previous chapter. Figure 4.5 illustrates a representative example of a Persistence Diagram extracted from the ISIC2018 dataset.

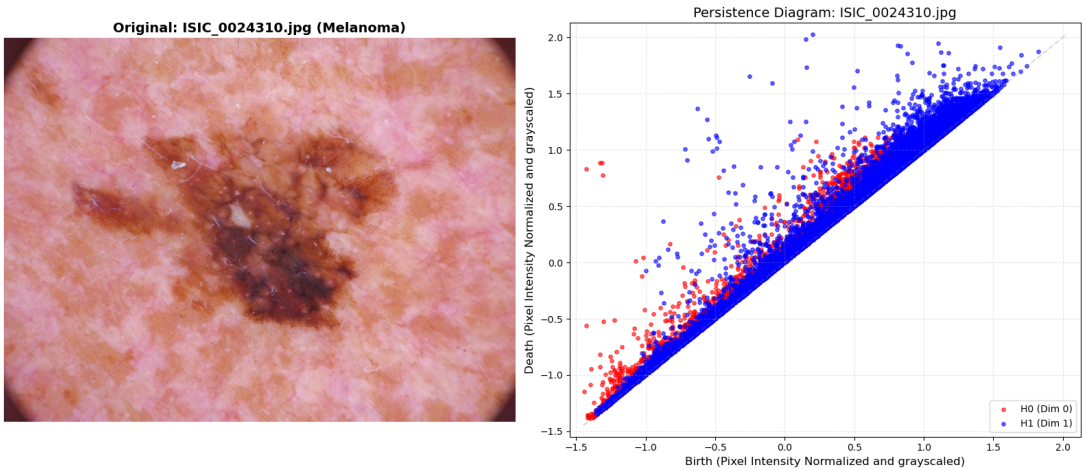


Figure 4.5: Illustration of the persistence diagram computation for image ISIC_0024310 (Melanoma) from the ISIC2018 dataset. The figure depicts the preprocessing pipeline required for the Persistence Diagram Encoder (PDE): the raw input is converted to a float representation, normalized to align with the model's expected distribution, and subsequently grayscaled for the topological filtration process. Infinite persistence point is not included.

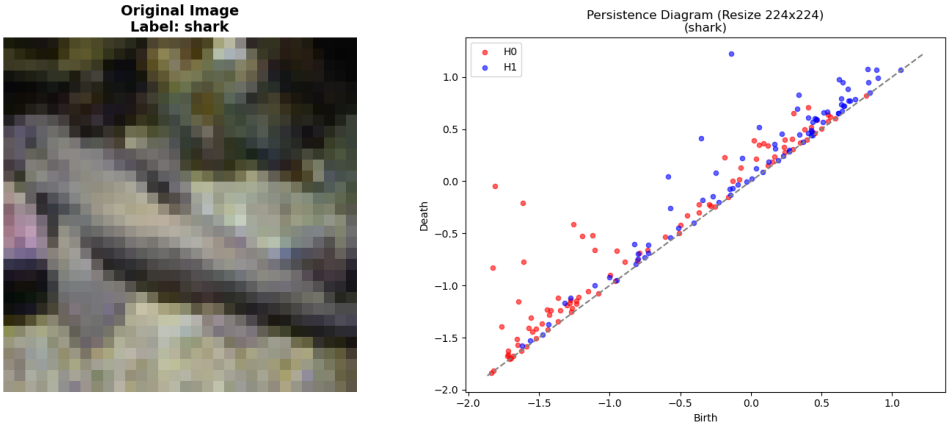


Figure 4.6: Illustration of the persistence diagram computation for a shark image from the CIFAR-100 dataset. The figure depicts the preprocessing pipeline required for the Persistence Diagram Encoder (PDE): the raw input is rescaled to 224×224 and converted to a float representation, grayscaled for the topological filtration process, and subsequently normalized to align with the model's expected distribution. Infinite persistence point is not included.

Chapter 5

Results and Discussions

The main goal of this end-of-degree project is to demonstrate the enhancement of vision models using homology-based descriptors, especially the ones generated by the persistence diagram encoder presented in the previous chapter.

In order to prove or refute this topological improvement, we ran an comprehensive set of experiments across the mentioned datasets using the three ResNets (ResNet-18, ResNet-50 adn ResNet-152) and the Swin Transformer V2-B architectures. These experiments covered a comprehensive exploration of hyperparameters and computation of the persistence diagram, as well as the vision component. All experiments were conducted on an `NVIDIA RTX 2080 Ti` GPU with `12 GB` of VRAM. The complete list of experiments performed during this project are summarized in Annex B. We present the most relevant ones in this chapter.

5.1 Quantitative Performance

5.1.1 Validation Strategies and Evaluation Metrics

Validation Strategies

To ensure the robustness and comparability of our results, we primarily utilized seed variation as a core technique. By executing experiments across multiple seeds for weight initialization and other stochastic elements of the code, we aim to guarantee the stability of our findings.

While a k-fold cross-validation on the ISIC dataset would have provided a more rigorous and direct comparison with PHG-Net results, such a process exceeded our available GPU resources. Consequently, seed variation serves as a necessary and effective alternative to validate the consistency of the model's performance.

Evaluation Metrics

We denote true positives as TP, true negatives as TN, false positives as FP, false negatives as FN. To study and compare our results, we relied on these four metrics:

- **Accuracy**: The most relevant one, defined as $\frac{\text{Total Correct Predictions}}{\text{Total Number of Samples}}$.
- **Sensitivity**: Defined as $\frac{TP}{TP+FN}$.
- **Specificity**: Defined as $\frac{TN}{TN+FP}$.
- **AUC (Area Under the ROC Curve)**: Defined as the area under the curve where the x-axis is $(1 - \text{Specificity})$ and the y-axis is Sensitivity, computed through various thresholds over the classification score.

Following Peng et al. [46], for multiclass classification tasks, the sensitivity, specificity, and AUC are computed using a one-versus-all approach. The corresponding metric is calculated for each class independently, treating the specific class as positive and the rest as negative, and then averaged to produce the final score.

5.1.2 Comparation with Baseline Models

The ISIC2018 Dataset

We conducted extensive experimentation with persistence diagram parameters, as the specific configurations used by Peng et al. [46] were not detailed in their published results.

The primary objective was to identify the optimal parameter combinations and demonstrate the resulting performance improvements. Due to the limited resources the majority of the experiments were run on 16 batch size. Table 5.1 summarizes the best-performing configurations for the various vision models evaluated. Additionally, it details a descriptor composed of four channels, incorporating both Total Persistence (TP) and Persistent Entropy (PE) for each homology group (H_0 and H_1). These channels are processed through a Multi-Layer Perceptron (MLP) to generate a 128-channel feature vector, this ensures the topological data is not ignored by the high-parameter backbone models.

It is important to acknowledge the Swin Transformer V2-B inherently captures global context through its self-attention mechanism, hence improving upon its baseline is a challenging task. Consequently, even marginal gains in this context represent a meaningful enhancement of the models.

Table 5.1: Performance comparison for ISIC2018 dataset. Due to the limited GPU resources we report results with mean and std only for ResNet-152. The best result is SwinT V2-B+PHG. The averaged results were obtained across the same two distinct seeds, with all experiments run for 250 epochs.

Architecture	**Acc**	**AUC**	**Sen**	**Spe**
SwinT V2-B	91.82	98.89	**86.63**	97.70
SwinT V2-B + PHG	**92.61**	**99.07**	86.52	**98.07**
ResNet-152 (Best)	90.35	98.53	82.63	97.42
ResNet-152 + TP and PE (Best)	91.04	98.75	84.84	97.54
ResNet-152 + PHG (Best)	**91.77**	**98.76**	**85.86**	**97.74**
ResNet-152	90.72 ± 0.53	98.61 ± 0.11	83.48 ± 1.20	97.50 ± 0.11
ResNet-152 + TP and PE	91.04 ± 0.00	98.69 ± 0.08	84.16 ± 0.96	97.51 ± 0.04
ResNet-152 + PHG	$\mathbf{91.57 \pm 0.27}$	$\mathbf{98.75 \pm 0.01}$	$\mathbf{84.58 \pm 1.80}$	$\mathbf{97.66 \pm 0.11}$

The results presented in the table demonstrate a clear performance improvement when incorporating topological descriptors via the Persistence Diagram encoder.

As previously noted, enhancing the ResNet-152 architecture proved challenging. Nonetheless, a higher than 1% gain in accuracy was achieved. Most notably, we observed a 3.23% increase in sensitivity at the best run, a critical metric in medical diagnostics, where minimizing false negatives is paramount.

Furthermore, our exhaustive experimentation with hyperparameters and persistence diagram preprocessing yielded an accuracy improvement of over 1% compared to the benchmarks reported by Peng et al. [46] (who obtained $88.83 \pm 0.45\%$ for the baseline ResNet-152 and $90.03 \pm 0.24\%$ for the ResNet-152 + PHG).

For the Swin Transformer (SwinT V2-B), the performance gap between the homology-enhanced model and the baseline was similarly pronounced, further substantiating the claim

of a robust improvement. Consistent with the ResNet results, this backbone configuration also outperformed the benchmarks set by Peng et al. [46] by nearly 1%.

Finally, the accuracy gains observed when utilizing Total Persistence (TP) and Persistent Entropy (PE) reaffirm that the underlying topological structures in medical images provide valuable discriminative information. While these fixed descriptors offer a smaller improvement than the learned PDE descriptors, they confirm that even simplified topological features enhance performance. To enable a fair comparison, the persistence diagrams used to compute the TP and PE, as well as those fed into the PDE, underwent identical preprocessing: 85 samples per homology group, with noise removed via a lower lifetime threshold.

CIFAR-100

For the CIFAR-100 dataset, we restricted the study to the ResNet-18 and ResNet-50 architectures. Compared to ISIC, CIFAR-100 provides significantly fewer training inputs per class: 500, whereas ISIC averages approximately 1,167. This scarcity presents a substantial challenge for higher-capacity models like ResNet-152, such architectures often struggle to learn generalizable features from limited data tending instead toward overfitting or memorization. Due to the smaller architectures the majority of the experiments were run on 64 batch size.

Table 5.2: Performance comparison for CIFAR-100 dataset. The averaged results were obtained by executing all experiments across the same three distinct seeds to ensure statistical consistency. The best result is ResNet-50+PHG. Furthermore, the best-performing results within each backbone category used identical configurations, ensuring a direct comparison. All experiments were run with 150 epochs.

Arch	Acc	AUC	Sen	Spe
ResNet-50 (Best)	86.28	99.62	86.28	**99.86**
ResNet-50 + PHG (Best)	**86.80**	**99.68**	**86.80**	**99.86**
ResNet-18 (Best)	83.53	**99.62**	83.53	**99.84**
ResNet-18 + PHG (Best)	**84.30**	**99.62**	**84.30**	**99.84**
ResNet-50	86.29 ± 0.07	99.65 ± 0.02	86.29 ± 0.07	$\mathbf{99.86 \pm 0.00}$
ResNet-50 + PHG	$\mathbf{86.36 \pm 0.45}$	$\mathbf{99.67 \pm 0.01}$	$\mathbf{86.36 \pm 0.45}$	99.86 ± 0.01
ResNet-18	83.65 ± 0.12	$\mathbf{99.63 \pm 0.01}$	83.65 ± 0.12	99.84 ± 0.01
ResNet-18 + PHG	$\mathbf{84.00 \pm 0.12}$	$\mathbf{99.63 \pm 0.01}$	$\mathbf{84.00 \pm 0.12}$	$\mathbf{99.84 \pm 0.00}$

As demonstrated in the results table 5.2, there is a subtle yet consistent and stable improvement when integrating TDA over baseline models. Notably, due to the perfect class balance, the sensitivity and accuracy metrics are identical.

The accuracy gain is more pronounced for ResNet-18 than for ResNet-50. As a shallower network, ResNet-18 extracts fewer complex hierarchical features. Consequently, the topological descriptors serve as a compensatory mechanism, providing structural information that the shallower model might otherwise miss. In contrast, the high-capacity ResNet-50 is inherently more capable of learning these features implicitly. Therefore, the PDs information becomes partially redundant, as it overlaps with information of the features already extracted by the deeper vision model. However, this sufficiency depends on the context: in highly complex domains like ISIC, containing fine-grained morphological structures such as veins, even deeper models (e.g., ResNet-152) are unable to fully resolve these features as we have seen previously.

While the accuracy improvement is more modest compared to the ISIC dataset or the other medical benchmarks reported by Peng et al. [46], this is likely due to the nature of the data. CIFAR-100 images, consisting of general objects, are not as inherently rich in distinct

topological structures as medical images (e.g., skin lesions or mammographies). Nevertheless, the improvement consistently yielded even in these cases underscores the robustness of topological features.

Regarding the generalization capabilities of the framework proposed by Peng et al. [46], our results confirm that the method generalizes effectively to datasets with a significantly higher number of classes, such as CIFAR-100.

5.2 Experimental Configuration

Throughout this section, we refer to the results in Appendix B by indicating the specific experiment index (#X) used in the comparison.

5.2.1 Data Augmentation Strategies

Significant overfitting was observed in both the baseline and the topology-enhanced models. To mitigate this, we explored several strategies, including fine-tuning the learning rate for the vision backbone and initializing the Persistence Diagram Encoder (PDE) without pretrained weights, however, these adjustments yielded no substantial accuracy improvements.

The most critical factor in stabilizing the model proved to be the optimization of data augmentation strategies. We found that the model was highly sensitive to the transformation pipeline: a lack of augmentation led to rapid overfitting, with accuracy failing to surpass 85%. Conversely, an overly aggressive pipeline, incorporating flipping, intense color jittering, random rotation, and random affine transformations, was equally detrimental, causing accuracy to drop to approximately 75% (#41).

Given the high number of parameters in models such as ResNet-152 architecture, data augmentation is essential to prevent the memorization of training samples. However, the specific nature of the transformations is critical in medical imaging:

- Color Jittering: The observed performance decline when using aggressive color jittering (#49 with 84.15%) is likely due to the loss of chromatic information. Skin lesion classification relies heavily on specific color intensities and gradients, excessive jittering destroys these discriminatory features.

- Affine Transformations: The further decrease in accuracy to 75.21% (#41) when including random affine transformations can be explained by the distortion of lesion morphology. In dermatology, the shape and symmetry of a lesion are key diagnostic indicators, and affine distortions likely introduced noise that confused the model.

Ultimately, the most effective augmentation strategy was found to be the combination of horizontal flipping and random rotation. This configuration provided sufficient variation to enhance the model's resilience to input data diversity while strictly preserving the color intensities and lesion morphology essential for diagnostic accuracy.

In contrast, for the CIFAR-100 dataset, we followed standard transformation protocols, specifically horizontal flipping and random rotation, which proved effective for general object classification.

Finally, while these studies were conducted prior to identifying the optimal topological parameters, the findings remain valid and translatable, as key experiments were successfully replicated using the final optimized configuration.

5.2.2 Persistence Diagram Preprocessing

Once the model reached a performance plateau with accuracies between 90% and 91%, our efforts shifted toward optimizing the hyperparameters and the specific processing components of the persistence diagram.

Given the high capacity of the architecture and the already elevated baseline accuracy, the margin for further learning was narrow. Consequently, substantial individual gains from single hyperparameter adjustments were not anticipated. While the vast majority of these optimization experiments were conducted on the ISIC dataset, the successful modifications were subsequently adapted and applied to the CIFAR-100 experiments, with minor variations to account for the different data characteristics.

Datatype and Image Normalization

Regarding the persistence diagram computation, we evaluated various input formats, including uint8 (0-255) and float32 (0-1) data types with and without normalization, as well as histogram matching.

The optimal configuration involved computing persistence diagrams using normalized float32 images. This choice made sense intuitively, as it ensures that the birth and death points are derived from the same normalized data that the vision backbone processes, maintaining consistency across both descriptors.

In comparison, using non-normalized integer images, as suggested by Peng et al. [46], yielded an accuracy of 90.14% (#53), which is consistent with their reported results. However, this was significantly outperformed by the normalized float32 (0-1) setup, which achieved an accuracy of 91.04% (#1) compared to 90.14% (#53). Furthermore, experiments involving histogram matching (#40, #39) resulted in a substantial performance decline of over 5%, dropping to 83.59% compared to the 91.38% achieved in similar setups like #23 (91.38%).

Persistence Diagram Representation

Initially, persistence diagrams were computed using images in uint8 format (values 0–255), without prior normalization. Following the methodology of Peng et al. [46], each point in the diagram was represented by four channels: the Birth coordinate, the Death coordinate, and a one-hot encoding for the homology group ($[1, 0]$ for H_0 and $[0, 1]$ for H_1). This preliminary setup achieved accuracies of up to 91% without the use of persistence thresholds or sample balancing (Appendix B, experiment #1).

It is reasonable to expect that topological features with a very short lifetime (low persistence) often act as noise rather than meaningful structural data. Consequently, we implemented a persistence threshold of 10, as suggested by Peng et al. [46], which successfully filtered out these insignificant features. In tandem, we updated the point representation to a more efficient four-coordinate vector: the Birth coordinate, the Death coordinate, one coordinate for the homology Dimension (0 for H_0, 1 for H_1), and a dummy flag a binary indicator to distinguish real points from the zero-padding required for fixed-size neural network inputs).

With these refinements, specifically using 150 sampled points per homology group and the threshold of 10, the model reached an accuracy of 91.23% (#54).

However, shifting the computation to normalized float32 data (ranging tipically from -2.5 to 2.5) rendered the original threshold of 10 obsolete, as it erased all persistence points. To maintain the filtering logic, we rescaled the threshold relative to the new data range $\text{Threshold}_{new} = \frac{10 \times 5}{255} = \frac{10}{51} \approx 0.196$, which retrieved a good performance (even compared to a threshold of $\frac{5}{51}$).

By calibrating the threshold to the normalized input space, we achieved superior accuracies of 91.28% (#11) and 91.33% (#15).

For the CIFAR-100 case, we only needed to adjust fewer hyperparameters. We lowered the threshold to $\frac{5}{51}$. The requirement for a lower threshold indicates a key difference from ISIC: in CIFAR-100, short-lifetime components represent relevant features instead of noise, boosting accuracy from 83.16% (#86) to 85.10% (#90).

Persistence Diagram Points Normalization

An essential aspect of our study was the impact of scaling and normalization on the persistence diagram (PD) samples after preprocessing. We defined scaling as a Min-Max transformation to the $[0, 1]$ range, and standardization (or normalization) as the subtraction of the mean followed by division by the standard deviation.

While Peng et al. [46] mentioned scaling features to a $[0, 1]$ range, they did not specify the exact methodology. Given that normalizing inputs is standard practice to prevent outliers from skewing the classification, we tested these techniques on the PD coordinates.

Our findings indicate that scaling only the first two coordinates (Birth and Death) was beneficial. For instance, experiment #23 achieved 91.38%, improving upon the 90.94% of #19. This improvement may be due to the nature of neural network activation layers (such as ReLU), which suppress negative values, scaling the birth-death coordinates ensures the negative values information has not been lost.

Conversely, scaling the entire four-coordinate vector proved detrimental. In experiment #14, accuracy fell to 90.70% compared to 91.33% in #15 or 91.04% in #13. This suggests that scaling discrete labels, such as the homology group and the dummy flag, distorts the feature space. If the maximum coordinate value is 2.5, the discrete 0 and 1 values are shifted spatially, leading the model to interpret categorical homology dimensions as continuous "sizes", thereby losing the discrete distinction between connected components and loops.

Finally, global normalization did not yield improvements (#20 at 90.55% vs. #19 at 90.94%). This suggests that the absolute spatial structure and relative distances of the persistence diagram are inherently meaningful. Unlike raw image intensities, which fluctuate based on lighting conditions, the relative distances between points in a PD represent the persistence lifetimes of topological features. Standardizing these values can inadvertently remove the structural information that the model relies on for classification.

The Role of Alpha Loss

Consistent with the principle that topological features are intended to guide and enhance the vision backbone rather than serve as a primary classifier, the inclusion of a topological classification loss is designed to provide class-based guidance during training (see Equation 4.1.4).

To achieve this without overshadowing the vision descriptor, the weight of this loss (α) should be the minimum required to influence the model effectively. Experimental results showed that lower values, such as $\alpha = 0.01$, led to a decrease in accuracy from 91.38% (#36) to 90.84% (#23). Consequently, the value of $\alpha = 0.1$ suggested by Peng et al. [46] proved to be the optimal choice for maintaining this balance.

Sensitivity to Sample Size

Regarding the noise reduction parameters of the Persistence Diagram (PD), an aspect addressed by Peng et al. [46] is the filtering strategy: prioritizing the points with the greatest persistence values, of the maximum number of samples per homology group. We found that the optimal parameter was 85 samples, which outperformed values of 50, 75, and 150. This confirms that limiting the sample count acts as an effective noise filtration measure.

Furthermore, we observed that not differentiating features by homology group, specifically, selecting the highest global persistence points regardless of their group, does not yield positive results. For example, experiment #32 retrieved an accuracy of 90.45% compared to the 90.94% of #18, where both configurations selected 150 total samples (combined and separated by group, respectively).

For the CIFAR-100 case, we adjusted the number of samples according to the ResNet backbone: 85 samples for ResNet-18 and 150 samples for ResNet-50. This distinction indicates that smaller models benefit more from high-lifetime topological features because they have less internal capacity to learn complex features. In contrast, larger models require a higher number of topological features to induce a measurable improvement in accuracy.

It is also interesting to note the higher sample requirement for CIFAR-100 with ResNet-50 compared to the 85 samples used for ISIC, where 150 samples actually proved detrimental. This can be explained by the structural sparsity of skin lesion images in ISIC, where excessive features can result in noise. On the other hand, CIFAR-100 contains rich, natural images with multiple complex structures and detailed backgrounds.

Although Figures 4.5 and 4.6 suggest ISIC is structurally richer, the need for equal or greater samples in CIFAR-100 is consistent with the fact that fewer of its short-lifetime features constitute noise. Furthermore, this does not contradict the higher accuracy improvement observed in ISIC: while fewer topological features are required, they represent complex structures that are difficult for the baseline model to learn autonomously.

5.2.3 Spatial Transformer Network (STN)

A pivotal modification that yielded improvements in accuracy was the integration of a Spatial Transformer Network (STN) block into the Persistence Diagram Encoder. Inspired by the T-Net module introduced in the original PointNet architecture by Qi et al. [47], depicted in Figure 4.2.

In our code, the STN learns a 3×3 affine transformation. We expect this network to rotate, scale, and translate the data to highlight key variations between distinct classes. For example, it can learn to scale up the death coordinates, thereby assigning more importance to high lifetime values while compressing lower lifetime ones. This approach was applied to the persistence diagram encoding, where, of the four input channels, the third was designated for the homology dimension and the last for distinguishing valid points from dummy ones. Consequently, the affine transformation was applied only to the first three coordinates.

Regarding the ISIC dataset, although the STN yielded interesting high accuracy results (#23 with 91.38%), the highest accuracy for the ResNet-152 model was actually obtained without it. In that optimal configuration, the persistence diagram points were processed using the first two columns for birth and death, and the last two as a one-hot encoding ($[1, 0]$ for connected components and $[0, 1]$ for loops), with dummy points represented as $[0, 0]$, achieving the 91.77% accuracy (#79).

We hypothesize that one-hot encoding is the superior approach because it correctly treats homology dimensions as distinct categories. In contrast, a simple binary or scalar encoding (e.g., assigning 0 to H_0 and 1 to H_1) may inadvertently imply an ordinal relationship or a magnitude-based difference between the groups. Since the homology dimension is a categorical property, not a continuous variable where one group is "greater" than another, one-hot encoding prevents the network from interpreting the dimension index as a gradual feature.

The experiments conducted on the CIFAR-100 dataset consistently utilized a one-hot encoding to represent the two additional homology coordinates for the persistence diagram, and all configurations operated without a Spatial Transformer Network (STN).

The only exception was the best-performing isolated run for ResNet-18, which employed an STN and binary encoding for the homology group, achieving 84.30% (#113) compared

to 83.53% (#120). However, subsequent runs using different seeds failed to replicate this performance gain, suggesting that the initial result was potentially an outlier.

5.2.4 Classification Head Architecture

Through extensive experimentation to identify the optimal classification head, we determined that a single linear layer mapping the final feature descriptor to the output classes was the most effective architecture.

We evaluated alternative configurations, such as employing two fully connected layers (e.g., comparing #4 to #1 or #55 to #54) to mitigate potential information loss when transitioning from high-dimensional descriptors (2048 or 1024) to the final classes. However, results indicated that such information loss does not occur, instead, these multi-layer structures often resulted in a performance decrease of up to 1% (e.g., dropping from 91.23% in #54 to 89.23% in #55). Similarly, the addition of dropout layers for further regularization failed to improve performance, preventing the model from reaching higher accuracy thresholds (e.g., comparing #2, #3, and #4 to #1 or stabilizing at 91.04% for #1 and #2).

5.3 Qualitative Analysis

5.3.1 ISIC2018 Total Persistence and Persistence Entropy Analysis

We computed the mean TP and PE, stratified by homology group and class, to provide a global overview of the ISIC2018 dataset persistence diagrams.

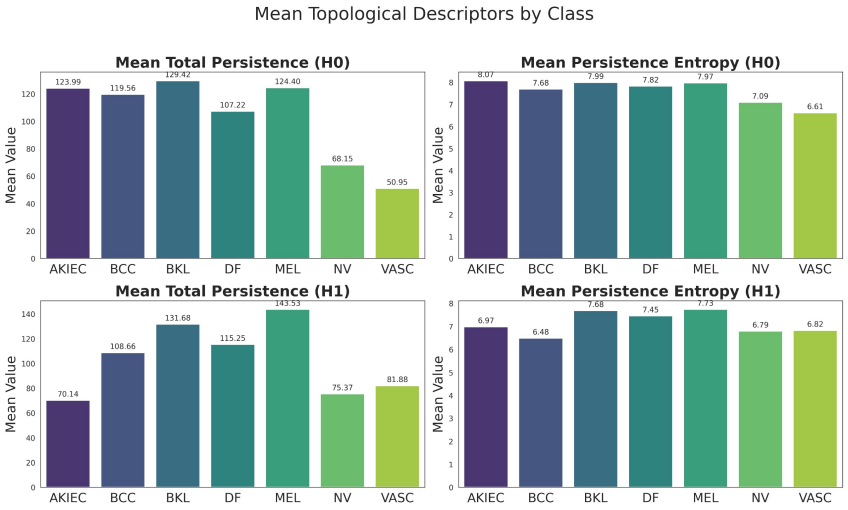


Figure 5.1: Visualization of the mean Total Persistence (TP) and Persistence Entropy (PE), stratified by homology group and class for ISIC2018 dataset. The mean values are calculated by averaging the respective persistence metrics across all samples belonging to each specific class. To ensure consistency with the model's input, the persistence diagrams used were subjected to the same preprocessing: 85 samples were selected per homology group, and noise was removed by applying a threshold to points with lower lifetime.

We observe that the Melanocytic nevus (NV) and Vascular lesion (VASC) classes exhibit the lowest Persistence Entropy (PE) values. This indicates that they possess the lowest variance in their Total Persistence distributions. Furthermore, these classes also demonstrate the lowest mean Total Persistence (TP) values.

This reduced variability suggests that these lesions are characterized by a more uniform and consistent topological structure. In contrast, classes such as Melanoma (MEL) display

the highest Total Persistence and Persistence Entropy values, reflecting a significantly higher degree of structural complexity and variability within these lesions.

An interesting observation is that the AKIEC (Actinic keratosis) class exhibits significantly higher persistence lifetimes for connected components (H_0) compared to loops (H_1). Biologically, this type of lesion is characterized by dry, scaly patches with uniform intensity. This morphological consistency aligns well with high persistence values for connected components, as the dense, uniform patches appear as stable features that persist through a wide range of filtration thresholds.

5.3.2 Feature Representation

To validate the quantitative improvements observed in the previous section, the objective of this section is to visualize the final vision feature vectors extracted immediately before the classification head. These vectors are the most relevant ones, as they already integrate the topological information and form the direct input for the final classification task.

It is important to note that the primary purpose of the homology descriptors is to "guide" the vision feature vector, effectively emphasizing specific features at each stage. Consequently, the inclusion of the topological descriptor in the loss function is intended to ensure that the vision descriptor is optimized for improved classification performance. This explains why we focus on representing only the final vision feature vectors in a lower-dimensional space.

For this visualization, we utilize the feature representations from our best-performing experiment: the Swin Transformer V2-B with a PointNet-based encoder including the STN (see Table 5.1). As the output consists of 1024-dimensional feature descriptors, which cannot be visualized directly, we employed three standard dimensionality reduction techniques to project the feature vectors onto a lower-dimensional space: PCA, t-SNE, and UMAP.

We briefly define each technique as follows:

- **Principal Component Analysis (PCA) [34]**: PCA performs a linear projection of the data into a lower-dimensional space defined by the directions of maximum variance. In our context, these directions correspond to the most significant variations among the feature descriptors. PCA generally fails to capture non-linear relationships within the data.
- **t-Distributed Stochastic Neighbor Embedding (t-SNE) [55]**: t-SNE is a non-linear technique that measures the similarity between pairs of points using a Student's t-distribution. It minimizes the Kullback-Leibler divergence between the conditional probability distributions of point pairs in the original high-dimensional space and the new lower-dimensional space. This technique excels at capturing local structures and can reveal global clusters across multiple scales.
- **Uniform Manifold Approximation and Projection (UMAP) [42]**: Grounded in manifold theory and topological data analysis, UMAP aims to preserve both local and global structures. It assumes the data lies on a Riemannian manifold and models it using a fuzzy topological structure, specifically, a simplicial set where each simplex is assigned an existence probability. The algorithm then seeks a lower-dimensional representation with a topological structure that most closely resembles the original.

Figure 5.2 presents a comparison of these three dimensionality reduction techniques applied to the validation set of ISIC2018.

Separation of Malignant vs. Benign Classes

Looking at t-SNE, we observe that classes such as BCC (orange), AKIEC (blue), and partly BKL (red) initially exhibit diffuse boundaries. However, when the model is enhanced with

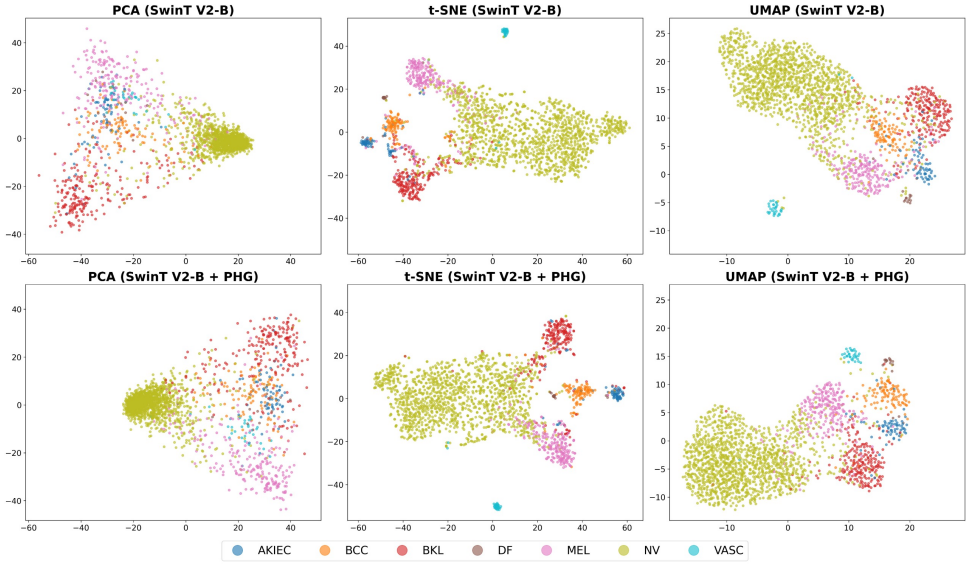


Figure 5.2: Visualization of the high-dimensional feature representations projected onto $\mathbb{R}^2$, comparing the baseline model against the version enhanced with the Persistence Diagram Encoder (PDE) using the SwinT V2-B backbone. The top row illustrates the clusters incorporating homology data, while the bottom row displays the baseline results without topological features. The columns correspond to the PCA, t-SNE, and UMAP dimensionality reduction techniques, respectively.

topological descriptors, these clusters become visibly more compact. This improved isolation mitigates the issue of class overlap, indicating that the 'guide' provided by the topological descriptors enables the model to establish clearer decision between these classes.

The most relevant improvement is the clearer isolation of BKL (Benign Keratosis) from the malignant classes: MEL (Melanoma), BCC (Basal cell carcinoma), and AKIEC (Actinic keratosis). Since the latter three are confirmed malignant categories [54], misclassifying them as benign is significantly more critical than confusing different malignant types. Therefore, this increased separation holds substantial clinical significance.

Reduction of Class Fragmentation

Additionally, some points of the MEL (pink) class at t-SNE initially displays significant overlap with other classes, an issue that the PHG-Net method clearly reduces. Furthermore, the baseline model splits the MEL class into two fragmented clusters, notably, this fragmentation does not occur when topology layers are included. This improvement suggests that the additional structural information makes the model less dependent on variable lesion attributes, such as intensity, which can otherwise lead to such intra-class fractures.

Global Structure and Variance

Similar trends are observed in the UMAP representation. We observe a marked improvement in the spatial differentiability of the BKL, MEL, AKIEC, and BCC classes compared to the baseline model without persistence diagram input.

In contrast, the PCA representation shows no distinct improvement. This is attributed to PCA's inherent limitation in capturing non-linear data variance. Nevertheless, it is worth noting that the MEL class, which appears entangled with other points in the top-left corner of the baseline plot, shifts toward a slightly more clustered configuration in the bottom-right corner when topology is applied.

Topology of Vascular Networks

An interesting aspect of the UMAP projections is the proximity of Vascular lesions (VASC) to the Melanoma (MEL) and Basal cell carcinoma (BCC) clusters. This spatial arrangement can be interpreted through the inherent vascular characteristics shared by these classes.

One of the primary differential criteria for BCC is the presence of arborizing vessels [41], while melanomas are frequently characterized by dotted and linear-irregular vessels [3]. Since vascular lesions are inherently defined by abnormalities in the vascular network, they share a fundamental structural similarity with the vascular features found in BCC and MEL.

We can interpret this clustering behavior as an evidence that the homology enhancement model highlights the morphology of the vessel network, while the baseline vision model may overlook these fine structural details in favor of other aspects as color or texture.

5.3.3 Confusion Matrix

Given the 100-class resolution of the CIFAR-100 dataset, a direct low-dimensional representation of its feature vectors would be difficult to interpret. Consequently, we utilized a confusion matrix that aggregates these results into their respective superclasses (20 superclasses each including 5 class of CIFAR-100 dataset).

A primary objective of this approach was to determine whether topological features significantly reduce misclassifications between different superclasses or if their primary contribution lies in refining intra-class classification. It is reasonable to hypothesize that the majority of failed predictions occur between classes within the same superclass (e.g., misclassifying a maple tree as an oak tree). Therefore, expecting that the structural details provided by the Persistence Diagram (PD) may assist the model in resolving these fine-grained distinctions.

The results are presented in Figure 5.3. Due to similarity between the confusion matrices for our best-performing ResNet-50 combination on CIFAR-100, we calculated the 'superclass accuracy', where a prediction is considered valid if it falls within the correct superclass, comparing the topology-enhanced model against the baseline. The results were 92.9% and 92.68%, respectively.

Interestingly, the margin of improvement at the superclass level ($\approx 0.2\%$) was lower than the improvement observed in the exact accuracy ($\approx 0.5\%$, see Table 5.2). This suggests that while topological features did not substantially alter the model's ability to distinguish between broad superclasses, potentially due to structural noise in varied backgrounds, the gain in exact accuracy is primarily driven by improved intra-class classification. This confirms that homology descriptors provide the fine-grained structural guidance necessary to distinguish between closely related classes. This is evidenced by the enhanced model achieving higher per-class accuracy on 58 classes, compared to only 38 classes for the baseline model (Appendix B.4).

5.3.4 Gradient-weighted Class Activation Mapping

Another interesting aspect to analyze, which allows us to determine the influence of the homology-based descriptor, is examining where the model 'looks at' when making a classification decision. By comparing the cases with and without TDA, we can assess whether the model incorporates the underlying topological structures. If it does, we would expect the TDA descriptors to guide the model to focus more accurately on the correct discriminative regions of the image.

For this purpose, we implemented a method based on the Gradient-weighted Class Activation Mapping (Grad-CAM [48]) approach. We extracted every 7×7 feature map from the 1024 channels available before the final pooling and normalization stage of the Swin Transformer V2-B model (which produces the 1024-dimensional feature vector used for the final class prediction).

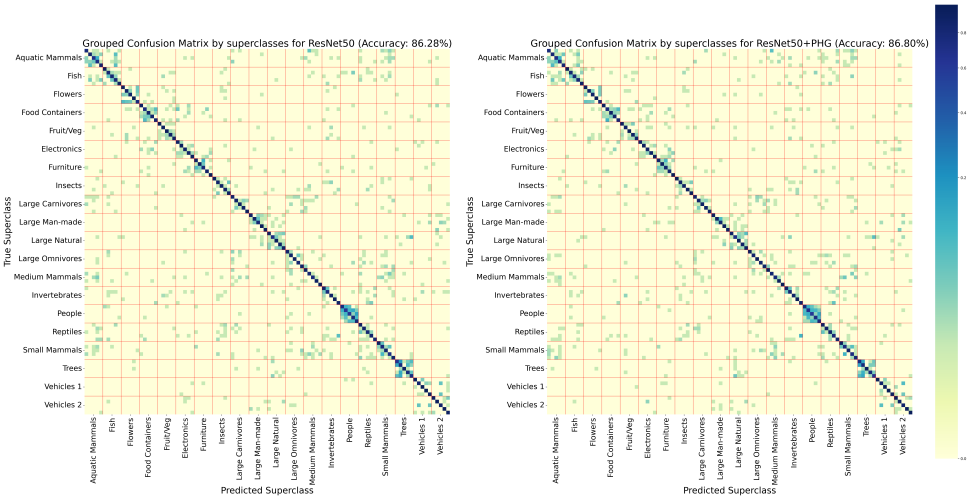


Figure 5.3: Visualization of the confusion matrix comparing the baseline model against the version enhanced with the Persistence Diagram Encoder (PDE) using the ResNet-50 backbone over the CIFAR-100 dataset with the best result (Table 5.2). The left matrix illustrates the baseline model, while the right matrix illustrates the enhanced one.

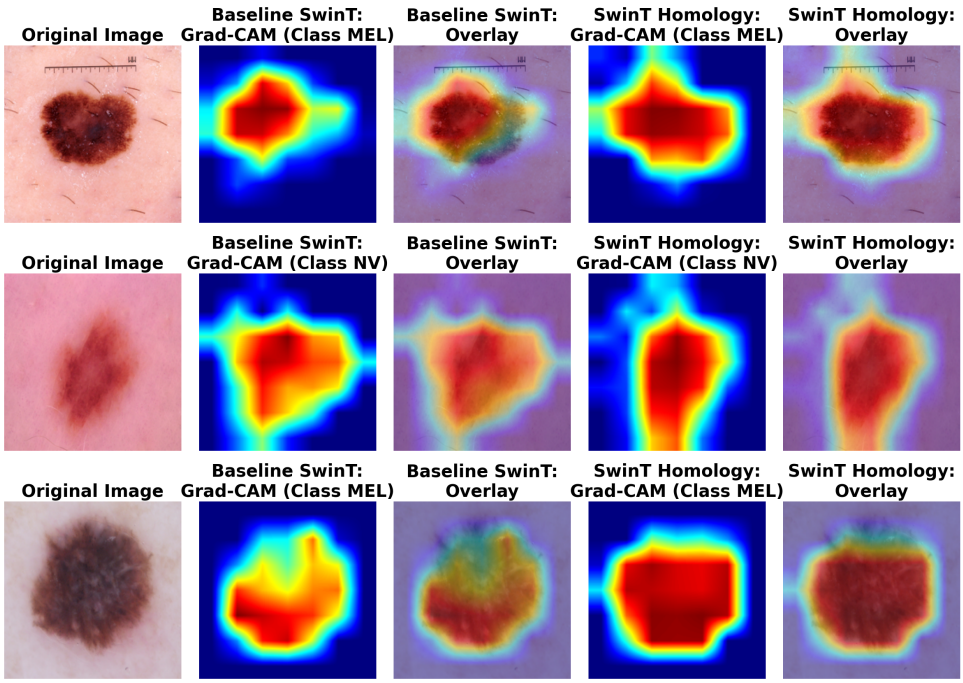


Figure 5.4: Visualization using the Grad-CAM technique applied to the Swin Transformer (Swin-T) V2-B model. The figure displays for three validation set different images the comparison of the attention maps generated by the baseline model (without topology) versus the model with topological enhancement best result (Table 5.1). Central row indicates Melanocytic nevi (NV) and the rest Melanoma (MEL)

For each feature in the final descriptor, we computed the gradient with respect to its corresponding 7×7 feature map. This gradient is pooled into a single scalar via mean computation, serving as a weight associated with that specific feature map. Subsequently, each feature map is multiplied by its associated scalar. To obtain the final activation map, all weighted feature maps are summed together, followed by a ReLU function. ReLU is applied because we aim to isolate and highlight only those regions that have a positive influence on the selected class.

Finally, to visualize the most influential areas on the original image, the resulting activation map is rescaled to the image size of 224×224.

Analyzing the three examples presented in Figure 5.4, it is observable that when the models are enhanced with homology descriptors from the PDE, the region of highest attention (indicated in red) during the classification decision encapsulates the skin lesion much more accurately than the baseline model.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

Based on the results presented in the preceding chapters, we draw the following conclusions:

- **Validation and Optimization:** This work successfully validates and improves the PHG-Net architecture proposed by Peng et al. [46]. This improvement was achieved through the optimization of persistence diagram preprocessing and hyperparameter selection.
- **Efficiency on Smaller Models and Generalization:** From a quantitative analysis, we demonstrated that lower-capacity backbones exhibited higher accuracy improvements on the CIFAR-100 dataset. Furthermore, we conducted a set of experiments exploring many combinations, serving as a tool for potential generalizable conclusions in TDA.
- **Improved Clustering and Interpretability:** Our quantitative analysis confirms that structural information from topological descriptors enhances data clustering by focusing on the specific morphology of lesions. Feature projections reveal improved class segregation, particularly between malignant and benign lesions, and reduced intra-class fragmentation. Additionally, Grad-CAM visualizations confirm that the model focuses on more clinically relevant regions.
- **Automated Feature Selection:** Overall, these results demonstrate that enhancing models with persistent homology offers a solid improvement, serving as an effective bridge between algebraic topology and deep learning. Crucially, it offers a robust alternative to fixed, handcrafted persistence descriptors, removing the need of selecting persistence diagram summaries in the context of neural network training.

6.2 Future Work

The insights from this study suggest several avenues for future research:

- **Adaptive Learning:** Research could focus on enabling the network to learn a dynamic threshold rather than using a fixed one, allowing the model to autonomously distinguish topological noise from relevant structural features. Additionally, a curriculum learning approach, gradually introducing topological features, could prevent noise propagation while visual features are being established.
- **Integration with Early Ensemble Method:** Zhuang et al. [58] achieve high accuracy on ISIC2018 by ensembling SENet [33] and PNASNet-5-Large [38], along with specific image preprocessing techniques. Integrating the PHG-Net topological approach with such high-performance ensembles represents a promising direction for improvement.

- **Multi-label Scenarios:** Preliminary experiments on the ODIR-5K ocular disease dataset (see Appendix B.3) to assess persistence diagrams in multi-label classification yielded no significant improvement (97.39% / 88.5% without topology vs. 97.43% / 88.38% with topology using a ResNet-50 backbone). However, this remains an open line of investigation to determine if specific adaptations are required for multi-label tasks.

Appendix A

Homological Persistence Results

A.1 Relating Simplicial and Cubical Complexes: Algebraic Proof

We define a family of maps between the chain groups, $\Phi_n : C_n(K) \longrightarrow C_n(T(K))$. For $n = 0$ and $n = 1$, Φ_0 and Φ_1 map vertices and edges of the cube to their corresponding vertices and edges in the triangulation. Since the boundaries of the edges coincide for both complexes, these are clearly well defined chain maps.

For 2-chains, we define Φ_2 on the basis element Q as:

$$\Phi_2(Q) = \sigma_L + \sigma_U.$$

Remark A.1. From this point forward, we refer to any component of the family of homomorphisms $\{\Phi_n\}_{n\in\{0,1,2\}}$ simply as Φ. The specific dimensional component Φ_n is implied by the context (the dimension of the chain upon which it operates).

To prove that Φ is a chain map, we must verify that it commutes with the boundary operator, i.e., $\partial_2^{T(K)} \circ \Phi_2 = \Phi_1 \circ \partial_2^K$.

The boundary of the square Q consists of its four outer edges:

$$\partial_2^K(Q) = \{v_0, v_1\} + \{v_1, v_2\} + \{v_2, v_3\} + \{v_3, v_0\}.$$

Remark A.2. Throughout this work, we operate over coefficients in the field $\mathbb{Z}/2\mathbb{Z}$. Consequently, signs are ignored, as $+1 \equiv -1$ in this field. For context, the general definition of the boundary for an ordered square Q (over a ring like $\mathbb{Z}$) requires alternating signs to ensure cancellation:

$$\partial_2^K(Q) = \{v_0, v_1\} + \{v_1, v_2\} - \{v_2, v_3\} - \{v_3, v_0\}.$$

However, in our context, this simplifies to the sum of the four edges without signs.

Now, we compute the boundary of the image in the simplicial complex:

$$\partial_2^{T(K)}(\Phi_2(Q)) = \partial_2^{T(K)}(\sigma_L + \sigma_U) = \partial_2^{T(K)}(\sigma_L) + \partial_2^{T(K)}(\sigma_U).$$

Applying the standard boundary formula for simplices (sum of faces):

$$\partial\sigma_L = \partial\{v_0, v_2, v_3\} = \{v_2, v_3\} + \{v_0, v_3\} + \{v_0, v_2\}$$
$$\partial\sigma_U = \partial\{v_0, v_1, v_3\} = \{v_1, v_3\} + \{v_0, v_3\} + \{v_0, v_1\}.$$

Adding these two results (modulo 2):

$$\partial\sigma_L + \partial\sigma_U = \{v_2, v_3\} + \{v_0, v_2\} + \{v_1, v_3\} + \{v_0, v_1\} + 2 \cdot \{v_0, v_3\}.$$

In $\mathbb{Z}/2\mathbb{Z}$, the term $2 \cdot \{v_0, v_3\}$ (the shared diagonal) vanishes ($2 \equiv 0$). Thus

$$\partial_2^{T(K)}(\Phi_2(Q)) = \{v_0, v_1\} + \{v_1, v_3\} + \{v_3, v_2\} + \{v_2, v_0\},$$

which exactly matches $\Phi_1(\partial_2^K Q)$.

To prove the isomorphism of homology groups, we proceed by induction on the number of squares (2-cubes) in the complex K. Suppose K contains no squares. In this scenario, the complex consists exclusively of vertices and edges. Since the construction of $T(K)$ establishes a unique, bijective correspondence between the vertices and edges of the cubical complex and those of the simplicial complex, the chain complexes are identical. Consequently, the isomorphism holds trivially.

We establish two preliminary results to serve as the base for our induction:

- Connected components (H_0): Regardless of the number of squares, the map Φ_0 establishes a bijection between the vertex sets of K and $T(K)$. It acts as the identity map, since the vertices of $T(K)$ are defined to be identical to those of K. Since the triangulation process only introduces new edges (diagonals) within existing squares, where all vertices are already connected by the edges of the square (its 1-dimensional faces), these new edges remain within the same homology class in H_0. In other words, the triangulation does not alter the connected components of the complex. Consequently, since Φ_0 forms a bijection between 0-dimensional chains, it induces an isomorphism on 0-th homology groups: $H_0(K) \cong H_0(T(K))$.

- Consider the case where the cubical complex K consists of a single square Q. K has 4 vertices and 4 edges.

  For the cubical complex K: The kernel of the boundary map ∂_1^K is generated by the unique cycle formed by the perimeter (the sum of the 4 edges). It is straightforward to verify that any other linear combination of edges has a non-zero boundary. The image of the 2-boundary map ∂_2^K is generated by ∂Q, which is exactly this perimeter cycle. Thus, every cycle is a boundary, implying that $H_1(K) = 0$.

  For the second dimension, the single square Q has a non-zero boundary (the perimeter), so $\operatorname{Ker} \partial_2^K = 0$. Thus, $H_2(K) = 0$.

  For the simplicial complex $T(K)$: The associated simplicial complex $T(K)$ consists of two triangles, σ_L and σ_U, sharing a diagonal edge. The kernel of $\partial_1^{T(K)}$ is generated by the boundaries of these 2-dimensional simplices (the "lower" loop and the "upper" loop). The image of the 2-boundary map is generated precisely by these two boundaries. Since the kernel and image coincide, $H_1(T(K)) = 0$.

  For the second dimension, a general 2-chain can only be σ_L, σ_U, or their sum $\sigma_L + \sigma_U$. None of these combinations has a zero boundary. Hence, $\operatorname{Ker} \partial_2^{T(K)} = 0$, implying that $H_2(T(K)) = 0$.

  Consequently, the homology groups are isomorphic in all dimensions for the base case: $H_n(K) \cong H_n(T(K))$.

Now for the inductive step. We assume we have it proven for cubical complexes with k squares. Let K' be a cubical complex with $k+1$ squares. Let W be the subcomplex formed by the new square (and its faces), and let K be the subcomplex containing the remaining k squares. We define the intersection as $I = K \cap W$, which consists of the faces (edges and vertices) shared between the new complexes K and W. Also we define the union as $K' = K \cup W$.

Our aim is to prove that $H_n(K') \cong H_n(T(K'))$. By the induction hypothesis, we have that $H_n(K) \cong H_n(T(K))$.

We apply the Mayer-Vietoris Sequence (see Hatcher, 2002, p. 116–149) to the complex K' as the union of the subcomplexes K and W. This yields the following long exact sequence for cubical homology:

$$\cdots \to H_n(I) \to H_n(K) \oplus H_n(W) \to H_n(K') \to H_{n-1}(I) \to \dots$$

It is straightforward to verify that the triangulation preserves intersections and unions: $T(I) = T(K) \cap T(W)$ and $T(K') = T(K) \cup T(W)$. Thus, we have the corresponding Mayer-Vietoris sequence for the simplicial homology groups:

$$\cdots \to H_n(T(I)) \to H_n(T(K)) \oplus H_n(T(W)) \to H_n(T(K')) \to H_{n-1}(T(I)) \to \dots$$

The chain map Φ induces a homomorphism between these sequences, resulting in the following commutative diagram:

$$\begin{array}{ccccccccccc}
\dots & \longrightarrow & H_n(I) & \longrightarrow & H_n(K) \oplus H_n(W) & \longrightarrow & H_n(K') & \longrightarrow & H_{n-1}(I) & \longrightarrow & H_{n-1}(K) \oplus H_{n-1}(W) \dots \\
 & & \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow \\
\dots & \longrightarrow & H_n(T(I)) & \longrightarrow & H_n(T(K)) \oplus H_n(T(W)) & \longrightarrow & H_n(T(K')) & \longrightarrow & H_{n-1}(T(I)) & \longrightarrow & H_{n-1}(T(K)) \oplus H_{n-1}(T(W)) \dots
\end{array}$$

The vertical map $\Phi_* : H_n(I) \to H_n(T(I))$ is an isomorphism, since the intersection has no squares. The map $\Phi_* : H_n(K) \to H_n(T(K))$ is an isomorphism by the induction hypothesis. The map $\Phi_* : H_n(W) \to H_n(T(W))$ is an isomorphism (single square base case). Consequently, the map $H_n(K) \oplus H_n(W) \to H_n(T(K)) \oplus H_n(T(W))$ on the direct sum is also an isomorphism.

By the Five Lemma (Hatcher, 2002, p. 129), the middle map is an isomorphism, since the surrounding four maps at the commutative diagram are isomorphisms. Therefore, $\Phi_* : H_n(K') \to H_n(T(K'))$ is an isomorphism.

By induction, we conclude that for every cubical complex K and dimension $n \geq 0$,

$$H_n(K) \cong H_n(T(K)).$$

Appendix B

Experimental Results

B.1 ISIC2018 Dataset Experiments

Notation:

Arch: Architecture (R152: ResNet-152, R50: ResNet-50, R18: ResNet-18, Swin: Swin Transformer V2-B).
Topo: Topology Module (PN: PointNet-like PDE, STN: PointNet-like PDE with Spatial Transformer, None: Baseline without TDA).
Params: Hyperparameters (Samp: Samples/Homology Group, Scale: Scaling, Th: Threshold, WD: Weight Decay, Norm: Normalization). Stop x: experiment stopped at epoch X. RR: Random Rotate degrees=20, RA: Random Affine, CJ: Color Jitter, RVF: Random Vertical Flip p=0.3, RHF: Random Horizontal Flip p=0.3, RC: Random Crop, RC X: Random Crop with window size X, RRC: RandomResizedCrop, LR: Learning rate.
Img: Image Preprocessing (F: Processed as floats normalizing, I: Processed as integer without normalizing).
PD processing: 4OneHot: 4 Coordinates with [1, 0] and [0, 1] for homology group, 4Binary: 4 Coordinates with 0 or 1 for homology group and 4th if is dummy or not, 3Binary: 3 Coordinates with 0 or 1 for homology group, 5OneHot: 4 Coordinates with [1, 0] and [0, 1] for homology group and 5th if is dummy or not. Norm Centroid: substract centroid and divide over the max for pd points. Scale First2: Scale pd two first coordinates to range 0-1 with global coordinates min and max. Norm: Normalizes substracting mean and diving over the std.
FC Block connecting topology: FC2L: FC Block with two layers, FC1L: FC Block with one layer.
Classification head: 2CL: 2 layers at the classification head, Drop: DropOut.
Extra notation: Standard: without any extra configuration than the mentioned, X ep: set up with X epochs. X Th: we apply to the PD X lower lifetime Th. Baseline: Without PDE.
Default Settings: 250 Epochs, LR 10^{-4}, Alpha 0.1, Batch 16, seed 12345, fold 2, RandomHorizontalFlip, RandomVerticalFlip and RandomRotate, F (unless specified), and th 10/51

Experiments were usually stopped in cases where the training set was already overfitting, reaching nearly 100% accuracy for various epochs, while the test accuracy remained static or even dropped. This criterion enabled us to establish valid comparisons throughout the sections of this work. Regarding the ISIC dataset, although a 5-fold cross-validation scheme was implemented, the detailed results presented focus on Fold 2 due to GPU limitations and just run for various seeds. The remaining folds were utilized primarily for hyperparameter optimization and parameter selection.

Table B.1: Baseline Models (No Topology). Comparison of architectures without topological features. No PDE.

#	Configuration	Acc	AUC	Sen	Spe
35	R152, Batch 48	0.9104	0.9872	0.8395	0.9753
63	R152, Fold 3	0.9054	0.9792	0.7998	0.9735
73	R152, Stopped 230	0.9020	-	-	-
82	R152, Standard	0.9035	0.9853	0.8263	0.9742
83	R152, Seed 7213	0.9109	0.9869	0.8433	0.9758
75	R50, Standard	0.9006	0.9854	0.8163	0.9715
65	Swin, Standard	0.9182	0.9889	0.8663	0.9770

Table B.2: High-Sample PointNet Experiments. **Shared Configurations:** 4 channel PD points last two one-hot encoding for homology group, 150 Samples per homology group, Scaling to 0-1 all PD points, No Threshold to low lifetime. ResNet-152 + PointNet

#	Configuration	Acc	AUC	Sen	Spe
1	Standard Setup	0.9104	0.9867	0.8353	0.9759
2	300 ep, Drop 0.2, Stop 189	0.9104	0.9867	0.8353	0.9759
3	300 ep, Drop 0.1	0.9099	0.9871	0.8440	0.9746
4	100 ep, Drop 0.5, 2CL, WD 1e-4	0.8918	0.9801	0.7800	0.9698
6	Seed 42	0.9030	0.9854	0.8227	0.9733
7	Seed 99999	0.9079	0.9856	0.8275	0.9737
5	Stop 209	0.9021	0.9851	0.8278	0.9731

Table B.3: PointNet with Spatial Transformer (STN) and only PointNet (PN). And some alpha variations. **Shared Configurations:** 4 channels per PD point where the 3rd is a binary coordinate being 1 for loops and 0 for connected components and last one 0 for dummy points and 1 otherwise, 85 Samples per homology group, Scaling to 0-1 only 2 first coordinates of PD, lower lifetime threshold 10/51. R152+PointNet+STN

#	Configuration	Acc	AUC	Sen	Spe
13	PointNet without STN	0.9104	0.9851	0.8337	0.9751
24	Apply STN after expading to 64 channels	0.8967	0.9833	0.8040	0.9702
23	Standard PN & STN	0.9138	0.9889	0.8537	0.9768
34	Batch 48	0.9070	0.9872	0.8253	0.9757
36	Batch 16, Alpha 0.01	0.9084	0.9873	0.8259	0.9740
37	Batch 48, Alpha 0.01	0.9099	0.9863	0.8336	0.9757
38	Batch 64, Alpha 0.01, Stop 138	0.8462	0.9615	0.6710	0.9572
39	Batch 16, Histogram matching previous PD computation	0.8359	0.9576	0.6565	0.9544
40	Batch 16, Histogram matching previous PD computation, Fold 3	0.8550	0.9629	0.7233	0.9591
62	Standard, PN & STN	0.9104	0.9874	0.8452	0.9751
77	Normalizing PD first two points	0.9079	0.9878	0.8398	0.9756
28	Seed 42	0.9065	0.9866	0.8182	0.9744
29	Seed 99999	0.9011	0.9861	0.8284	0.9740
80	Fusion FC Blocks starting closed by initialization	0.9099	0.9869	0.8354	0.9751

Table B.4: Swin Transformer + PointNet with Spatial Transformer (STN) . **Shared Configurations:** 4 channels per PD point where the 3rd is a binary coordinate being 1 for loops and 0 for connected components and last one 0 for dummy points and 1 otherwise, 85 Samples per homology group, Scaling to 0-1 only 2 first coordinates of PD, lower lifetime threshold 10/51. SwinT+PointNet+STN

#	Configuration	Acc	AUC	Sen	Spe
64	1 layer FC Block	0.9261	0.9907	0.8652	0.9807
67	No Bias at FC Block, Stop 121	0.9182	0.9882	0.8399	0.9786
68	Standard	0.9167	0.9876	0.8285	0.9792
70	1 layer FC Block, Fold 4	0.9190	-	-	-
72	1 layer FC Block, Fold 0	0.9110	-	-	-
76	1000 ep, No pretrained weights, Fold 4, Stop 120	0.7070	0.8857	0.3135	0.9107

Table B.5: Hyperparameter Ablation (Optimization focus). **Shared Configurations:** PointNet only, 4 channels per PD point where the 3rd is a binary coordinate being 1 for loops and 0 for connected components and last one 0 for dummy points and 1 otherwise, 85 Samples per homology group, lower lifetime threshold 10, PD computed with non normalized images as integers. ResNet-152+PointNet

#	Configuration	Acc	AUC	Sen	Spe
41	Flip + Jitter + Affine + Rotation	0.7521	-	-	-
42	No Transformation, ResNet LR=1e-5 PointNet LR=1e-4	0.8605	-	-	-
43	1000 ep, No Pretrained weights, LR=1e-3, No Transformation, Stop 476	0.7656	-	-	-
44	100 ep, No Transformation, LR 1e-4	0.7214	-	-	-
45	500 ep, DropOut 0.3, Stop 75, No Transformation	0.8615	-	-	-
46	500 ep, No Transformation, Stop 41	0.8633	-	-	-
47	No Transformation, Stop 49	0.7538	-	-	-
48	Stop 128, No Transformation, WD 1e-4	0.8450	-	-	-
49	Jitter, Stop 56	0.8415	-	-	-
50	Rotate Crop, Stop 61	0.7115	-	-	-
51	500 ep, Flip + Rotate, Stop 170	0.9050	-	-	-
52	500 ep, Flip + Jitter + Affine + Rotation, 2-layer final classification head, Stop 57	0.8481	-	-	-

Table B.6: Miscellaneous Configurations and Ablations. **Shared Configurations:** Total persistence and persistence entropy decriptor: [TP_H0, TP_H1, PE_H0, PE_H1] ampliated to 128 channels through a MLP. ResNet-152

#	Configuration	Acc	AUC	Sen	Spe
33	Standard	0.9104	0.9863	0.8348	0.9748
84	seed 7213	0.9104	0.9875	0.8484	0.9754

Table B.7: Miscellaneous Configurations and Ablations. **Shared Configurations:** 150 samples variations. 4 channels per PD point where the 3rd is a binary coordinate being 1 for loops and 0 for connected components and last one 0 for dummy points and 1 otherwise. R152+PointNet

#	Configurations	Acc	AUC	Sen	Spe
25	PN+STN, Scale 0-1 First2, Th 10/51	0.9119	0.9864	0.8452	0.9755
27	PN+STN, Scale 0-1 First2, Th 10/51, Fusion FC Blocks starting closed by initialization	0.9089	0.9865	0.8389	0.9753
32	PN+STN, 150 Samp from both homology groups, Scale 0-1 First2, Th 10/51	0.9045	0.9877	0.8342	0.9737
54	Th 10, I	0.9123	-	-	-
55	Th 10, 100 ep, Drop 0.5, WD 1e-4, 2CL	0.8923	-	-	-
59	R50, Scale First2, Th 10, I, Stop 206	0.9023	-	-	-
60	Norm Centroid, 500 ep, Stop 320	0.9082	-	-	-
8	Scale 0-1 First2	0.9079	0.9859	0.8282	0.9746

Table B.8: Miscellaneous Configurations and Ablations. **Shared Configurations:** 150 samples variations. 4 channel PD points last two one-hot encoding for homology group. R152+PointNet

#	Configurations	Acc	AUC	Sen	Spe
57	Norm Centroid, No Th, Drop 0.1, 300 ep	0.9111	-	-	-
58	Norm Centroid, No Th, Drop 0.2, Stop 190	0.8994	-	-	-
56	Norm Centroid, No Th	0.9100	-	-	-
9	Scale First2	0.9094	0.9853	0.8202	0.9749

Table B.9: Miscellaneous Configurations and Ablations. **Shared Configuration:** 50 samples per homology group. R152+PointNet+STN

#	Configuration	Acc	AUC	Sen	Spe
22	Sin STN 3OneHot, Scale 0-1 First2, No Th	0.9055	0.9861	0.8273	0.9749
66	4Binary, Scale 0-1 First2, No Bias at FC Block, Stop 168	0.9030	0.9858	0.8296	0.9730
69	4Binary, Scale 0-1 First2, Th 10/51, Element prod Fusion	0.8981	0.9841	0.8083	0.9723
71	4Binary, No Scale 0-1, Th 10/51	0.9105	-	-	-
78	4onehot, Scale 0-1 First2, Th 10/51	0.9055	0.9871	0.8398	0.9743

Table B.10: Miscellaneous Configurations and Ablations. **Shared Configuration:** 85 samples per homology group. Scaling to 0-1 only 2 first coordinates of PD. Channel PD points last two one-hot encoding for homology group. R152+PN

#	Configuration	Acc	AUC	Sen	Spe
11	Scaling 0-1 per axis	0.9128	0.9875	0.8460	0.9761
16	PD points extra coordinate (1 for dummy 0 otherwise, that is 5OneHot)	0.9099	0.9860	0.8323	0.9756

Table B.10: Miscellaneous Configurations and Ablations. **Shared Configuration:** 85 samples per homology group. Scaling to 0-1 only 2 first coordinates of PD. Channel PD points last two one-hot encoding for homology group. R152+PN

#	Configuration	Acc	AUC	Sen	Spe
31	PointNet+STN, PD points extra coordinate (1 for dummy 0 otherwise), Stop 218	0.9055	0.9860	0.8333	0.9751
26	PointNet+STN	0.9055	0.9879	0.8404	0.9738
79	Standard	0.9177	0.9876	0.8586	0.9774
81	seed 7213	0.9138	0.9875	0.8331	0.9758
12	DropOut 0.3	0.9094	0.9859	0.8340	0.9751
53	No theshold and PD computed with 0-255 unnormalized images	0.9014	-	-	-
10	No theshold	0.9094	0.9859	0.8340	0.9751

Table B.11: Detailed Ablations: Scaling variations. **Shared Configurations:** 4 channels per PD point where the 3rd is a binary coordinate being 1 for loops and 0 for connected components and last one 0 for dummy points and 1 otherwise, 85 Samples per homology group, lower lifetime threshold 10/51. R152+PoinNet

#	Configuration	Acc	AUC	Sen	Spe
14	Scale all PD points	0.9070	0.9858	0.8375	0.9747
15	No Scale PD points	0.9133	0.9863	0.8400	0.9761
19	No Scale PD points	0.9094	0.9870	0.8402	0.9759
20	No Scale PD points, Normalize PD points	0.9055	0.9869	0.8221	0.9737
21	No Scale PD points, Normalize PD points, Alpha 1	0.9060	0.9878	0.8403	0.9741
17	Scale First2, 1 layer FC Block	0.9060	0.9865	0.8221	0.9739
30	PointNet+STN, Scale First2	0.9138	0.9889	0.8537	0.9768

Table B.12: Detailed Ablations: Normalization, Sampling, and Architecture variations.

#	Arch	Topo	Modifications	Acc	AUC	Sen	Spe
18	R152	PN	4Binary, 75 Samp, Scale First2, Th 10/51	0.9094	0.9870	0.8402	0.9759
31	R152	STN	5OneHot, 85 Samp, Scale First2, Th 10/51, Stop 218	0.9055	0.9860	0.8333	0.9751
74	R50	STN	4Binary, 85 Samp, Scale First2, Th 10/51	0.9006	0.9850	0.8139	0.9715

B.2 CIFAR-100 Dataset Experiments

Default Settings: 150 Epochs, LR 10^{-4}, Alpha 0.1, Batch 64, seed 12345, RandomHorizontalFlip and RandomCrop with 224, F (unless specified), 4binary for PD preprocessing and Scaling two first coordinated together to 0-1 of the PD points, 85 samples per homology group.

Table B.13: CIFAR-100: ResNet-50 Architectures with PDE including Spatial Transformer Networks (STN).

#	Configuration	Acc	AUC	Sen	Spe
85	Baseline, Batch 16, 150 ep	0.8304	0.9961	0.8304	0.9983
86	85 Samp, Scale First2, Th 10/51, Batch 16, PD with 32	0.8316	0.9959	0.8316	0.9983
87	85 Samp, Scale First2, Th 10/51, Batch 16, PD with 32, RHF, RVF, CJ, RA, RR	0.8311	0.9962	0.8311	0.9983
88	50 Samp, Scale First2, Th 10/255, Batch 16, PD with 32, Batch 16, RHF, RVF, RR	0.8362	0.9962	0.8362	0.9983
89	50 Samp, Scale First2, Th 10/255, RHF, RC with 32, PD with 32, batch 16, PD with 32	0.1526	0.8700	0.1526	0.9914
90	85 Samp, Scale First2, Th 10/255, RHF, RC 224, batch 16, PD with 32	0.8510	0.9964	0.8510	0.9985
91	50 Samp, Scale First2, Th 10/255, Batch 64, PD with 32	0.8580	0.9969	0.8580	0.9986
92	50 Samp, Scale First2, Th 10/255, SGD, Stop 73, PD with 32, batch 16	0.8204	0.9960	0.8204	0.9982
93	50 Samp, Scale First2, Th 10/51, PD with 224, batch 16	0.8524	0.9965	0.8524	0.9985
94	Baseline, Batch 64, Stop 73	0.8623	0.9969	0.8623	0.9986
95	85 Samp, Scale First2, Th 10/255, Batch 64	0.8634	0.9969	0.8634	0.9986
96	150 Samp, Scale First2, Th 10/255, Batch 64	0.8654	0.9970	0.8654	0.9986
97	85 Samp, Scale First2, Th 10/255, Batch 64, Drop 0.3	0.8634	0.9969	0.8634	0.9986
109	150 Samp, Th 10/255, WD 1e-4	0.8650	-	0.8650	-

Table B.14: CIFAR-100: ResNet-18 Architectures with PDE including Spatial Transformer Networks (STN).

#	Configuration	Acc	AUC	Sen	Spe
98	85 Samp, Scale First2, Th 10/255, Batch 64	0.8384	0.9964	0.8384	0.9984
99	85 Samp, Th 10/255, RHF, RR, CJ, RRC, Stop 84	0.4313	0.9364	0.4313	0.9943
100	85 Samp, Th 10/255, RHF, RR, CJ, RRC, Drop 0.5, Stop 80	0.4117	0.9350	0.4117	0.9941
101	85 Samp, Th 10/255, No Trans, Drop 0.5, Stop 80	0.8170	0.9959	0.8170	0.9982
102	85 Samp, Th 10/255, LR 1e-5	0.8194	0.9964	0.8194	0.9982
103	85 Samp, Th 10/255, WD 1e-4	0.8402	0.9963	0.8402	0.9984
104	85 Samp, Th 10/255, Topo Head with 2-Layers	0.8352	0.9962	0.8352	0.9983
105	85 Samp, Th 10/255, Alpha 0.0 until ep 10	0.8376	0.9963	0.8376	0.9984
106	Baseline	0.8349	0.9962	0.8349	0.9983
107	150 Samp, Th 10/255, WD 1e-4	0.8366	0.9963	0.8366	0.9983

Table B.14 – Continued from previous page

#	Configuration	Acc	AUC	Sen	Spe
108	150 Samp, Th 10/255, WD 1e-4	0.8380	-	0.8380	-
110	Baseline, Seed 42, WD 1e-4	0.8376	0.9962	0.8376	0.9984
111	150 Samp, Th 10/255, WD 1e-4, Seed 99999	0.8378	0.9949	0.8378	0.9984
112	Baseline, Seed 99999	0.8370	0.9962	0.8370	0.9984
113	150 Samp, Th 10/255, WD 1e-4, Seed 42	0.8430	0.9962	0.8430	0.9984

Table B.15: CIFAR-100: PDE Pointnet-like (PN) Integration and Stability Studies (Seed Variations) without STN.

#	Configuration	Acc	AUC	Sen	Spe
114	R50, 4onehot, 150 Samp, Th 10/255, WD 1e-4	0.8680	0.9968	0.8680	0.9987
115	R50, Baseline, WD 1e-4, seed 42	0.8637	0.9966	0.8638	0.9986
116	R18, 4onehot, 85 Samp, Th 10/255, WD 1e-4, seed 12345	0.8401	0.9963	0.8401	0.9984
117	R18, 4onehot, 85 Samp, Th 10/255, WD 1e-4, seed 2026	0.8387	0.9962	0.8387	0.9984
118	R18, Baseline, WD 1e-4, seed 2026	0.8367	0.9964	0.8367	0.9984
119	R18, 4onehot, 85 Samp, Th 10/255, WD 1e-4, seed 42	0.8411	0.9963	0.8411	0.9984
120	R18, Baseline, WD 1e-4, seed 12345	0.8353	0.9962	0.8353	0.9983
121	R50, 4onehot, 150 Samp, Th 10/255, WD 1e-4, seed 42	0.8638	0.9966	0.8638	0.9986
122	R50, Baseline, WD 1e-4, seed 12345	0.8628	0.9962	0.8628	0.9986
123	R50, 4onehot, 150 Samp, Th 10/255, WD 1e-4, seed 2026	0.8590	0.9966	0.8590	0.9986
124	R50, Baseline, WD 1e-4, seed 2026	0.8623	0.9967	0.8623	0.9986

B.3 ODIR-5K Dataset Experiments

Default Settings: 250 Epochs, LR 10^{-4}, Alpha 0.1, Batch 64, seed 12345, RandomHorizontalFlip, RandomVerticalFlip and RandomRotate, F (unless specified).

Table B.16: Performance comparison for ODIR-5K dataset.

#	Configuration	Ham Acc	exact Acc
125	R50, Baseline	0.9739	0.8850
126	R50, PN, 4onehot, 85 samples, th 10/51	0.9726	0.8780
127	R50, PN, 4onehot, 85 samples, th 10/255	0.9736	0.8835
128	R50, STN, 4binary, 85 samples, th 10/51	0.9743	0.8838
129	R50, STN, 4binary, 85 samples, th 10/255	0.9746	0.8803

Continued on next page

Table B.16 – Continued from previous page

#	Configuration	Ham Acc	exact Acc
130	R50, STN, 4binary, 150 samples, th 10/51	0.9723	0.8706

B.4 CIFAR-100 Class-wise Performance

In this section, we present the results intra-class accuracy for the best-performing topological model (#114), which achieved 86.80%, and its corresponding baseline (#122 with 86.28%) on the CIFAR-100 dataset using the ResNet-50 backbone.

Table B.17: CIFAR-100 Aquatic Mammals Superclass. ResNet-50: **0.8640**. PHG+Net: 0.8580.

Class	ResNet-50 Accuracy	PHG+Net Accuracy
beaver	**0.7800**	0.7600
dolphin	**0.8400**	0.8200
otter	0.6900	**0.7000**
seal	**0.7400**	0.7300
whale	**0.8500**	0.8100

Table B.18: CIFAR-100 Fish Superclass. ResNet-50: **0.9060**. PHG+Net: 0.8960.

Class	ResNet-50 Accuracy	PHG+Net Accuracy
aquarium_fish	**0.9600**	0.9400
flatfish	**0.8200**	**0.8200**
ray	0.8000	**0.8100**
shark	0.7900	**0.8000**
trout	**0.9100**	0.9000

Table B.19: CIFAR-100 Flowers Superclass. ResNet-50: 0.9680. PHG+Net: **0.9720**.

Class	ResNet-50 Accuracy	PHG+Net Accuracy
orchid	0.9100	**0.9400**
poppy	**0.8700**	**0.8700**
rose	0.8700	**0.8800**
sunflower	0.9800	**0.9900**
tulip	0.7700	**0.8200**

Table B.20: CIFAR-100 Food Containers Superclass. ResNet-50: 0.9180. PHG+Net: **0.9300**.

Class	ResNet-50 Accuracy	PHG+Net Accuracy
bottle	0.9000	**0.9300**
bowl	0.7400	**0.7500**

Continued on next page

Table B.20 – Continued from previous page

Class	ResNet-50 Accuracy	PHG+Net Accuracy
can	**0.8900**	0.8700
cup	0.8600	**0.8900**
plate	**0.8100**	0.8000

Table B.21: CIFAR-100 Fruit/Veg Superclass. ResNet-50: **0.9600**. PHG+Net: 0.9500.

Class	ResNet-50 Accuracy	PHG+Net Accuracy
apple	**0.9700**	0.9300
mushroom	**0.9200**	0.8800
orange	0.9600	**0.9800**
pear	0.9000	**0.9100**
sweet_pepper	**0.8800**	0.8600

Table B.22: CIFAR-100 Electronics Superclass. ResNet-50: **0.9380**. PHG+Net: 0.9280.

Class	ResNet-50 Accuracy	PHG+Net Accuracy
clock	**0.9000**	0.8900
keyboard	**0.9500**	0.9400
lamp	**0.8700**	**0.8700**
telephone	**0.8800**	**0.8800**
television	**0.9300**	0.9200

Table B.23: CIFAR-100 Furniture Superclass. ResNet-50: 0.9540. PHG+Net: **0.9580**.

Class	ResNet-50 Accuracy	PHG+Net Accuracy
bed	**0.8800**	0.8600
chair	0.9100	**0.9200**
couch	0.7900	**0.8500**
table	**0.8700**	0.8400
wardrobe	0.9700	**0.9800**

Table B.24: CIFAR-100 Insects Superclass. ResNet-50: 0.9340. PHG+Net:**0.9520**.

Class	ResNet-50 Accuracy	PHG+Net Accuracy
bee	0.9400	**0.9600**
beetle	0.8700	**0.8900**
butterfly	0.9000	**0.9200**
caterpillar	0.8700	**0.9100**
cockroach	0.9100	**0.9400**

Table B.25: CIFAR-100 Large Carnivores Superclass. ResNet-50: **0.9160**. PHG+Net: 0.9140.

Class	ResNet-50 Accuracy	PHG+Net Accuracy
bear	**0.8500**	**0.8500**
leopard	**0.8700**	0.8600
lion	**0.9000**	**0.9000**
tiger	**0.9000**	0.8900
wolf	**0.9200**	0.9100

Table B.26: CIFAR-100 Large Man-made Superclass. ResNet-50: **0.9440**. PHG+Net: 0.9400.

Class	ResNet-50 Accuracy	PHG+Net Accuracy
bridge	0.8700	**0.9300**
castle	**0.8900**	0.8800
house	0.8200	**0.8500**
road	**0.9700**	0.9600
skyscraper	**0.9400**	0.9100

Table B.27: CIFAR-100 Large Natural Superclass. ResNet-50: **0.9440**. PHG+Net: 0.9380.

Class	ResNet-50 Accuracy	PHG+Net Accuracy
cloud	**0.9000**	0.8800
forest	**0.7300**	0.7100
mountain	**0.9500**	**0.9500**
plain	**0.9000**	**0.9000**
sea	0.8900	**0.9100**

Table B.28: CIFAR-100 Large Omnivores Superclass. ResNet-50: 0.9080. PHG+Net: **0.9120**.

Class	ResNet-50 Accuracy	PHG+Net Accuracy
camel	0.8600	**0.9300**
cattle	**0.8700**	0.8600
chimpanzee	0.9200	**0.9400**
elephant	0.8500	**0.8600**
kangaroo	**0.8800**	**0.8800**

Table B.29: CIFAR-100 Medium Mammals Superclass. ResNet-50: 0.8800. PHG+Net:**0.9060**.

Class	ResNet-50 Accuracy	PHG+Net Accuracy
fox	0.9200	**0.9500**
porcupine	0.8200	**0.8400**
possum	0.7700	**0.7800**
raccoon	0.9100	**0.9300**
skunk	0.9500	**0.9800**

Table B.30: CIFAR-100 Invertebrates Superclass. ResNet-50: 0.9040. PHG+Net: **0.9180**.

Class	ResNet-50 Accuracy	PHG+Net Accuracy
crab	**0.8700**	0.8600
lobster	0.7700	**0.8200**
snail	0.8800	**0.8900**
spider	0.9300	**0.9600**
worm	**0.9000**	0.8700

Table B.31: CIFAR-100 People Superclass. ResNet-50: **0.9740**. PHG+Net: 0.9700.

Class	ResNet-50 Accuracy	PHG+Net Accuracy
baby	0.7500	**0.7800**
boy	0.6200	**0.6700**
girl	**0.6400**	0.6100
man	**0.7700**	0.7300
woman	**0.8200**	0.7700

Table B.32: CIFAR-100 Reptiles Superclass. ResNet-50: 0.8800. PHG+Net: **0.8900**.

Class	ResNet-50 Accuracy	PHG+Net Accuracy
crocodile	0.8900	**0.9000**
dinosaur	0.8600	**0.8700**
lizard	0.8200	**0.8600**
snake	**0.8500**	0.7900
turtle	0.8200	**0.8400**

Table B.33: CIFAR-100 Small Mammals Superclass. ResNet-50: 0.8840. PHG+Net: **0.8900**.

Class	ResNet-50 Accuracy	PHG+Net Accuracy
hamster	0.9100	**0.9400**
mouse	**0.7800**	0.7500
rabbit	0.8500	**0.8700**
shrew	0.7100	**0.7500**
squirrel	**0.8500**	0.8200

Table B.34: CIFAR-100 Trees Superclass. ResNet-50: **0.9680**. PHG+Net: 0.9580.

Class	ResNet-50 Accuracy	PHG+Net Accuracy
maple_tree	0.6600	**0.7600**
oak_tree	0.7600	**0.7700**
palm_tree	0.9200	**0.9300**
pine_tree	0.7600	**0.7700**
willow_tree	**0.7400**	0.6800

Table B.35: CIFAR-100 Vehicles 1 Superclass. ResNet-50: 0.9540. PHG+Net: **0.9560**.

Class	**ResNet-50 Accuracy**	**PHG+Net Accuracy**
bicycle	0.9400	**0.9700**
bus	0.8400	**0.8600**
motorcycle	**0.9800**	0.9700
pickup_truck	**0.9900**	0.9800
train	0.9000	**0.9200**

Table B.36: CIFAR-100 Vehicles 2 Superclass. ResNet-50: 0.9380. PHG+Net: **0.9440**.

Class	**ResNet-50 Accuracy**	**PHG+Net Accuracy**
lawn_mower	0.9200	**0.9500**
rocket	**0.9300**	0.9100
streetcar	0.8600	**0.8700**
tank	**0.9600**	0.9400
tractor	0.9300	**0.9700**

Bibliography

[1] H. Adams, *Applied Topology Software*, Department of Mathematics, Colorado State University. Available at https://www.math.colostate.edu/~adams/advising/appliedTopologySoftware/ (accessed Jan. 2026).

[2] H. Adams and B. Coskunuzer, *Geometric Approaches on Persistent Homology*, arXiv: 2103.06408 [math.AT], 2021. https://arxiv.org/abs/2103.06408

[3] G. Argenziano et al., *Vascular structures in skin tumors: a dermoscopy study*, Archives of Dermatology, **140** (12) (2004), 1485–1489. DOI: 10.1001/archderm.140.12.1485.

[4] M. F. Atiyah, *Vector bundles over an elliptic curve*, Proc. London Math. Soc., **7**, no. 3 (1957), 414–452. DOI: 10.1112/plms/s3-7.1.414

[5] N. Atienza, L. M. García-Ludeña, and M. Soriano-Trigueros, *Stability of Persistent Entropy and its applications to support vector machines*, Chaos, Solitons & Fractals, **140** (2020).

[6] R. Ballester, X. Arnal Clemente, C. Casacuberta, M. Madadi, C. A. Corneanu, and S. Escalera, *Predicting the generalization gap in neural networks using topological data analysis*, arXiv:2203.12330 [cs.LG], 2022. https://arxiv.org/abs/2203.12330

[7] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, New York, 2006.

[8] B. Bleile, A. Garin, T. Heiss, K. Maggs, and V. Robins, *The Persistent Homology of Dual Digital Image Constructions*, In: Research in Computational Topology 2, vol. 30 of Association for Women in Mathematics Series, Springer, Cham, 2022, pp. 1–26. https://doi.org/10.1007/978-3-030-95519-9_1

[9] P. Bubenik, *Statistical topological data analysis using persistence landscapes*, Journal of Machine Learning Research, **16**, no. 3 (2015), 77–102.

[10] R. Buchweitz, G. Greuel and F. O. Schreyer, *Cohen–Macaulay modules on hypersurface singularities, II*, Invent. Math., **88**, no. 1 (1987), 165–182. DOI: 10.1007/BF01388481

[11] M. Carrière, F. Chazal, Y. Ike, T. Lacombe, M. Royer and Y. Umeda, *PersLay: A neural network layer for persistence diagrams and new graph topological signatures*, International Conference on Artificial Intelligence and Statistics (PMLR), (2020), 2786–2796.

[12] F. Chazal and B. Michel, *An introduction to Topological Data Analysis: fundamental and practical aspects for data scientists*, arXiv:1710.04019 (2017). https://arxiv.org/abs/1710.04019

[13] C. Chen, *Topology and Invisibility*, Technical report, 2011.

[14] N. Codella, V. Rotemberg, P. Tschandl, M.E. Celebi, S. Dusza, D. Gutman, B. Helba, A. Kalloo, K. Liopyris, M. Marchetti, H. Kittler and A. Halpern, *Skin Lesion Analysis Toward Melanoma Detection 2018: A Challenge Hosted by the International Skin Imaging Collaboration (ISIC)*, arXiv preprint arXiv:1902.03368, (2019). https://arxiv.org/abs/1902.03368

[15] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer, *Stability of Persistence Diagrams*, Discrete & Computational Geometry, **37** (2007), 103–120. (Available as *Stability of Persistence Diagrams* on ResearchGate).

[16] DataScienceUB, *PointNet Implementation Explained Visually*, Medium, (2018). Available at `https://datascienceub.medium.com/pointnet-implementation-explained-visually-c7e300139698` (accessed Jan. 2026).

[17] M. Demazure, *A very simple proof of Bott's theorem*, Invent. Math., **33** (1976), 271–272. DOI: `10.1007/BF01425501`

[18] D. S. Dummit and R. M. Foote, *Abstract Algebra*, 3rd Edition, Wiley, 2004, Section 12.3.

[19] H. Edelsbrunner and J. Harer, *Persistent Homology—a Survey*, in *Surveys on Discrete and Computational Geometry*, Contemporary Mathematics, vol. 453, American Mathematical Society, 2008, 257–282.

[20] H. Edelsbrunner and J. Harer, *Computational Topology: An Introduction*, American Mathematical Society, Providence, RI, 2010.

[21] H. Edelsbrunner, D. Letscher, and A. Zomorodian, *Topological Persistence and Simplification*, Discrete & Computational Geometry, **28** (4) (2002), 511–533. DOI: `10.1007/s00454-002-2885-2`

[22] D. Eisenbud, *Commutative Algebra with a View Toward Algebraic Geometry*, Springer, New York, NY, 1995.

[23] D. Eisenbud and J. Herzog, *The classification of homogeneous Cohen–Macaulay rings of finite representation type*, Math. Ann., **280** (1988), 347–352. DOI: `10.1007/BF01457023`

[24] D. Eisenbud, F. O. Schreyer and J. Weyman, *Resultants and Chow forms via exterior syzygies*, J. Amer. Math. Soc., **16** (2003), 537–579. DOI: `10.1090/S0894-0347-03-00427-0`

[25] D. Faenzi and F. Malaspina, *The CM representation type of homogeneous spaces*, Preprint (2014).

[26] A. Ferrà Marcús, *Topological Data Analysis Across Domains: From Point Clouds to Graphs*, Doctoral Thesis, Universitat de Barcelona, 2025.

[27] A. Ferrà Marcús, R. Jankowski, M. V. Miñana, C. Casacuberta, and M. Á. Serrano, *Chordless cycle filtrations for dimensionality detection in complex networks via topological data analysis*, arXiv:2509.08350 [physics.soc-ph], 2025. `https://arxiv.org/abs/2509.08350`

[28] P. Frosini, *A distance for similarity classes of submanifolds of a Euclidean space*, Bulletin of the Australian Mathematical Society, **42** (3) (1990), 407–415. DOI: `10.1017/S000497270002766X`

[29] I. Goodfellow, Y. Bengio and A. Courville, *Deep Learning*, MIT Press, 2016.

[30] G. Guorgis, C. D. Anderson, J. Lyth and M. Falk, *Actinic Keratosis Diagnosis and Increased Risk of Developing Skin Cancer: A 10-year Cohort Study of 17,651 Patients in Sweden*, Acta Derm. Venereol., **100** (2020), adv00128. DOI: `10.2340/00015555-3486`.

[31] A. Hatcher, *Algebraic Topology*, Cambridge University Press, Cambridge, 2002.

[32] K. He, X. Zhang, S. Ren, and J. Sun, *Deep Residual Learning for Image Recognition*, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, 770–778.

[33] J. Hu, L. Shen, and G. Sun, *Squeeze-and-Excitation Networks*, arXiv preprint arXiv: 1709.01507, 2017.

[34] I. Jolliffe, *Principal Component Analysis*, Springer Berlin Heidelberg, 2002.

[35] A. J. Kemme and C. A. Agyingi, *Persistent Homology: A Pedagogical Introduction with Biological Applications*, arXiv:2505.06583 [math.AT], 2025.

[36] V. Kovalevsky, *Image Processing with Cellular Topology*, Springer, Singapore, 2021.

[37] A. Krizhevsky, *Learning Multiple Layers of Features from Tiny Images*, Technical Report, University of Toronto, (2009).

[38] C. Liu, B. Zoph, J. Shlens, W. Hua, L.-J. Li, L. Fei-Fei, A. Yuille, J. Huang, and K. Murphy, *Progressive Neural Architecture Search*, arXiv preprint arXiv:1712.00559, 2017.

[39] Z. Liu, H. Hu, Y. Lin, Z. Yao, Z. Xie, Y. Wei, J. Ning, Y. Cao, Z. Zhang, L. Dong, F. Wei and B. Guo, *Swin Transformer V2: Scaling Up Capacity and Resolution*, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022.

[40] C. Maria, J.-D. Boissonnat, M. Glisse, and M. Yvinec, *The Gudhi Library: Simplicial Complexes and Persistent Homology*, Research Report No. 8548, INRIA, 2014. Available at `https://inria.hal.science/hal-01005601v2/document`.

[41] A. Mateos-Mayo, A. Sánchez-Herrero, and J. A. Avilés-Izquierdo, *When We Can't See the Forest for the Trees (Cuando los árboles no dejan ver el bosque)*, Actas Dermo-Sifiliográficas, 2019.

[42] L. McInnes, J. Healy and J. Melville, *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction*, arXiv preprint arXiv:1802.03426 (2018).

[43] T. M. Mitchell, *Machine Learning*, McGraw-Hill, 1997.

[44] J. M. Møller, *From singular chains to Alexander duality*, Lecture Notes, Institut for Matematiske Fag, University of Copenhagen. Available at `https://web.math.ku.dk/~moller/f03/algtop/notes/homology.pdf`.

[45] J. R. Munkres, *Elements of Algebraic Topology*, Addison-Wesley, Menlo Park, CA, 1984.

[46] Y. Peng, H. Zhang, X. Li and J. Wu, *PHG-Net: Persistent Homology Guided Medical Image Classification*, Proc. IEEE/CVF Winter Conf. on Applications of Computer Vision (WACV), 2024.

[47] C. R. Qi, H. Su, K. Mo and L. J. Guibas, *PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation*, Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), (2017), 652–660.

[48] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh and D. Batra, *Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization*, arXiv preprint arXiv:1610.02391 (2016).

[49] S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms*, Cambridge University Press, New York, 2014.

[50] Stanford University, *CS230: Deep Learning*, Lecture Notes and Materials. Available at `https://cs230.stanford.edu/lecture/` (accessed Jan. 2026).

[51] E. Stevens, L. Antiga, and T. Viehmann, *Deep Learning with PyTorch*, Manning Publications, 2020.

[52] W. Stolz, U. Semmelmayer, and A. W. Kopf, *History of Skin Surface Microscopy and Dermoscopy*, in *An Atlas of Dermoscopy*, A. Marghoob, R. P. Braun, and A. W. Kopf (eds.), Taylor & Francis, New York, NY, 2004.

[53] M. Stone, *Cross-Validatory Choice and Assessment of Statistical Predictions*, J. R. Stat. Soc. Ser. B (Methodol.), **36** (1974), 111–147.

[54] P. Tschandl, C. Rosendahl and H. Kittler, *The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions*, Sci. Data, **5** (2018), 180161. DOI: 10.1038/sdata.2018.161.

[55] L. van der Maaten and G. Hinton, *Visualizing Data using t-SNE*, Journal of Machine Learning Research, **9** (2008), 2579–2605.

[56] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser and I. Polosukhin, *Attention Is All You Need*, Advances in Neural Information Processing Systems (NIPS), 2017.

[57] M. Wright, *Topological Data Analysis with Applications*, Cambridge University Press, Cambridge, 2021.

[58] J. Zhuang, W. Li, S. Manivannan, R. Wang, J. Zhang, J. Liu, J. Pan, G. Jiang, and Z. Yin, *Skin Lesion Analysis Towards Melanoma Detection Using Deep Neural Network Ensemble*, in *Proceedings of the ISIC 2018 Challenge: Skin Lesion Analysis Towards Melanoma Detection*, 2018.

[59] A. Zomorodian and G. Carlsson, *Computing Persistent Homology*, Proc. 20th Annual Symposium on Computational Geometry (SCG), ACM Press, 2004, 347–356. DOI: 10.1145/997817.997870