# 本科生毕业设计（论文）

题　　目：三维点云机器学习中拓扑特征嵌入方法的研究与分析


姓　　名：秦雨禾

学　　号：12211057

院　　系：数学系

专　　业：数学与应用数学

指导教师：朱一飞


2026 年 4 月 28 日

CLC____________ Number_________

UDC___________ Available for reference ☐ Yes ☐ No

# Undergraduate Thesis

**Thesis Title:** **Topological Feature Embedding and Representation for Machine Learning on 3D Point Clouds**

**Student Name:** **Yuhe Qin**

**Student ID:** **12211057**

**Department:** **Department of Mathematics**

**Program:** **Mathematics and applied Mathematics**

**Thesis Advisor:** **Yifei Zhu**

Date: April 28, 2026

# 诚信承诺

1.本人郑重承诺所呈交的毕业设计（论文），是在导师的指导下，独立进行研究工作所取得的成果，所有数据、图片资料均真实可靠。

2.除文中已经注明引用的内容外，本论文不包含任何其他人或集体已经发表或撰写过的作品或成果。对本论文的研究作出重要贡献的个人和集体，均已在文中以明确的方式标明。

3.本人承诺在毕业论文（设计）选题和研究内容过程中没有抄袭他人研究成果和伪造相关数据等行为。

4.在毕业论文（设计）中对侵犯任何方面知识产权的行为，由本人承担相应的法律责任。

作者签名：[redacted]

2026 年 4 月 28 日

# COMMITMENT OF HONESTY

1. I solemnly promise that the paper presented comes from my independent research work under my supervisor's supervision. All statistics and images are real and reliable.

2. Except for the annotated reference, the paper contents no other published work or achievement by person or group. All people making important contributions to the study of the paper have been indicated clearly in the paper.

3. I promise that I did not plagiarize other people's research achievement or forge related data in the process of designing topic and research content.

4. If there is violation of any intellectual property right, I will take legal responsibility myself.

Signature: [redacted]

Date: 2026.4.28

# Topological Feature Embedding and Representation for Machine Learning on 3D Point Clouds

秦雨禾

（数学系　指导教师：朱一飞）

[ABSTRACT] ：Point clouds are widely used for 3D data, but their unordered structure and lack of explicit connectivity make it difficult for neural networks to capture stable global structure. This thesis studies how persistent-homology-based topological features can complement geometric point cloud learning. The work first summarizes a topological feature extraction pipeline from complex construction and filtration to persistence diagrams and vectorization. It then organizes topological representations along two dimensions: spatial scope, including global and local representations, and adaptivity, including fixed and learnable representations. It also compares three integration strategies: feature-level augmentation, representation-level integration, and loss-level structural constraints. A PointNet-based ablation study on ModelNet40 is conducted using persistence-image descriptors from $H_0$ and $H_1$. The results show that topology-only features are not sufficient for competitive classification, while selected topological signals, especially $H_1$-based descriptors, provide small but useful complementary information. This suggests that topology should be used as an auxiliary structural prior rather than a replacement for geometric learning.

[Key words]: Point Clouds; Topological Data Analysis; Persistent Homology; Topology-aware Learning; Feature Integration

[摘要]：点云是三维数据的重要表示形式，但其无序性和缺乏显式连接关系的特点，使神经网络难以稳定捕捉全局结构信息。本文研究基于持久同调的拓扑特征如何为点云几何学习提供补充信息。本文首先总结拓扑特征提取流程，包括复形构造、滤过、持久图计算和向量化。随后，本文从两个维度组织拓扑表示：空间范围，即全局表示与局部表示；表示适应性，即固定表示与可学习表示。同时，本文比较了三类拓扑融合策略：特征级增强、表示级融合和损失级结构约束。在实验部分，本文基于 PointNet 在 ModelNet40 上进行消融实验，并使用来自 $H_0$ 和 $H_1$ 的持久图像作为拓扑描述符。实验结果表明，仅使用拓扑特征不足以获得有竞争力的分类性能；但经过选择的拓扑信号，尤其是 $H_1$ 描述符，能够提供有限但有用的补充结构信息。因此，拓扑更适合作为辅助结构先验，而不是替代几何学习。

[关键词]：点云；拓扑数据分析；持久同调；拓扑感知学习；特征融合

# Contents

**1. Introduction** 1

**2. Structural Limitations of Point Clouds and the Role of Topology** 1

2.1 Structural Limitations of Point Clouds . . . . . . . . . . . . . . . . . . . 2

2.2 How Topology Can Help . . . . . . . . . . . . . . . . . . . . . . . . . . 3

**3. Topological Foundations for Point Cloud Analysis** 3

3.1 Point Clouds and Simplicial Complexes . . . . . . . . . . . . . . . . . . 4

3.2 Vietoris–Rips Filtration . . . . . . . . . . . . . . . . . . . . . . . . . . 4

3.3 Persistent Homology . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

3.4 Vectorization of Topological Descriptors . . . . . . . . . . . . . . . . . . 7

3.5 Computational Pipeline . . . . . . . . . . . . . . . . . . . . . . . . . . 8

**4. Topological Feature Representation Mechanisms** 10

4.1 Spatial Scope: Global and Local Topological Representations . . . . . 11

4.1.1 Global Topological Representations . . . . . . . . . . . . . . . . . . 11

4.1.2 Local and Point-wise Topological Representations . . . . . . . . . . 12

4.2 Representation Adaptivity: Fixed and Learnable Topological Representations . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 13

**5. Integration with Point Cloud Learning** 15

5.1 Feature-level Augmentation . . . . . . . . . . . . . . . . . . . . . . . . 15

5.2 Representation-level Integration . . . . . . . . . . . . . . . . . . . . . . 16

5.3 Loss-level Structural Constraints . . . . . . . . . . . . . . . . . . . . . 17

**6. Comparative Insights** 19

6.1 Representative Methods under the Proposed Classification . . . . . . 19

6.2 Comparison of Topological Representations . . . . . . . . . . . . . . . 21

6.3 Comparison of Integration Strategies . . . . . . . . . . . . . . . . . . . . . . 21
6.4 Why Topology Alone is Insufficient . . . . . . . . . . . . . . . . . . . . . 22
6.5 Practical Implications . . . . . . . . . . . . . . . . . . . . . . . . . . . . 23
**7. Experiments** 23
7.1 Experimental Setup . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 23
7.2 Implementation Details . . . . . . . . . . . . . . . . . . . . . . . . . . . 24
7.3 Main Results and Quantitative Analysis . . . . . . . . . . . . . . . . . . . 25
7.4 Ablation Study . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 25
7.5 Qualitative Analysis . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 26
**8. Conclusion** 26
8.1 Main Findings and Contributions . . . . . . . . . . . . . . . . . . . . . . 26
8.2 Limitations and Future Work . . . . . . . . . . . . . . . . . . . . . . . . 26
**References** 28
**Acknowledgements** 30

# 1. Introduction

Point clouds have become a fundamental representation for 3D data in applications such as shape analysis, robotics, and computer graphics[1]. Unlike images or volumetric grids, a point cloud is an unordered set of points without explicit connectivity, which makes point cloud learning structurally different from learning on regular grids[2].

Deep learning methods such as PointNet, PointNet++, and DGCNN have made substantial progress by addressing permutation invariance and improving local feature aggregation[2-4]. However, because these models primarily learn from coordinates and local geometric relations, global structural properties such as connectivity, loops, and multi-scale organization may not be fully captured.

Topological data analysis (TDA) provides a complementary perspective for this problem. Persistent homology, in particular, describes structural patterns that appear and disappear across scales, producing information such as connected components and loops that can complement local geometric features[5]. This makes topology a natural source of structural information for point cloud learning.

This paper investigates how topological information can be represented and incorporated into point cloud learning. Rather than focusing on individual methods separately, it organizes topology-aware point cloud learning from two perspectives: how topological information is represented, and where it is integrated into the learning pipeline.

# 2. Structural Limitations of Point Clouds and the Role of Topology

Although point clouds are simple, flexible, and easy to acquire, their structural information is largely implicit[1]. A point cloud records sampled coordinates, but it does not directly provide edges, surfaces, or a stable neighborhood structure. As a result, relations such as adjacency, connectivity, and global organization must be constructed or inferred during learning.

This section examines the structural limitations that motivate the use of topology in point cloud learning. It first discusses why neighborhood construction, sampling irregularity, and limited global structure awareness make point cloud learning difficult. It then explains how topological methods, especially persistent homology, can provide complementary multi-scale structural information.

## 2.1 Structural Limitations of Point Clouds

A point cloud is fundamentally an unordered set of points sampled from a 3D object or scene. Although each point carries spatial coordinates, the representation itself provides only discrete geometric samples rather than an explicit structural description. Therefore, many important shape relations must be inferred from the point distribution instead of being directly given in the input[2].

- **Unstable connectivity modeling.** Unlike meshes or graphs, point clouds do not contain predefined edges or adjacency relations between points. As a result, neural networks must construct connectivity indirectly, often through $k$-nearest neighbors or radius-based search[3-4]. However, these neighborhoods are heuristic and can be sensitive to noise, point density, and sampling variation. A small perturbation in the input may change the neighborhood structure, which in turn affects the extracted features.
- **Irregular and incomplete sampling.** Point clouds are often acquired from real-world sensors such as LiDAR scanners and depth cameras, which usually produce non-uniform point distributions[1]. Some regions may be densely sampled, while others may be sparse, noisy, or missing due to occlusion and sensor limitations. This irregularity introduces ambiguity in local geometric interpretation. For example, a sparse region may correspond either to a genuinely flat surface or to missing observations, and a local neighborhood may fail to represent the true underlying surface structure.
- **Limited global structure awareness.** Many point cloud networks rely heavily on local feature aggregation[3-4]. While this strategy is effective for extracting fine-scale geometric patterns, it may not be sufficient for representing global shape organization. Two objects may have similar local surface geometry but different global structures. For instance, a shape with a loop and a shape without a loop may share similar local patches, but they differ in their topological organization. Purely local geometric features may not clearly capture this difference.

Therefore, the main challenge is not only that point clouds are sparse or unordered, but that their structural information is implicit. Connectivity, holes, loops, and global organization must be inferred from coordinates rather than read directly from the representation. This creates a gap between the raw point cloud input and the structural information needed for robust shape understanding[1].

## 2.2 How Topology Can Help

Topological data analysis provides a way to extract structural information from point clouds without requiring predefined connectivity[5-6]. Instead of assuming that the input already contains edges or faces, topology constructs these relationships from metric information. For example, Vietoris–Rips filtrations build simplicial complexes from pairwise distances at different scales[7]. As the scale parameter changes, connected components, loops, and higher-dimensional structures can appear and disappear. This process allows global structure to be recovered from the point set itself.

Persistent homology is especially useful because it captures topological features across multiple scales[5-6]. Short-lived features are often treated as noise, while long-lived features are interpreted as more stable structural patterns. This multi-scale property is well suited to point clouds, where the correct neighborhood scale is often unclear.

Topology can also provide robustness against irregular sampling and small geometric perturbations. Since persistent features are defined by their stability across scales, they are less dependent on individual points or local sampling density. This makes topological descriptors useful as complementary information when geometric features are unstable or incomplete[8].

Another advantage is that topology captures global shape properties that may be difficult to express through local geometric descriptors alone. Features such as connected components and $H_1$ loops describe the organization of the entire shape rather than only local surface patches. In this sense, topology provides a compact structural summary of the point cloud[7].

Topological features usually discard detailed metric information such as exact coordinates, curvature, and fine surface texture. Their value lies in complementing neural point cloud models with structural information that is invariant, multi-scale, and globally informative[5].

Overall, point clouds are flexible but structurally incomplete representations. Their unordered nature, lack of explicit connectivity, and irregular sampling make it difficult to recover stable structural information. Topological methods help address this limitation by extracting global structures from distance-based relations, providing information that complements local geometric features.

# 3. Topological Foundations for Point Cloud Analysis

This section introduces the basic topological concepts used in this paper. The goal is to define the main objects needed to extract topological features from point clouds and use them in learning models. Standard concepts in topological data analysis are adopted following classical references[5,9].

## 3.1 Point Clouds and Simplicial Complexes

A point cloud is treated as a finite set of sampled points. To study its topology, topological data analysis represents relations among these points through simplices and simplicial complexes[9]. In this setting, simplices serve as the basic building blocks, while simplicial complexes provide the combinatorial structure on which topological features can be computed.

**Definition 3.1** (Simplex) *Given $k+1$ distinct points $\{v_i\}_{i=0}^{k}$ in $\mathbb{R}^n$, the set $\{v_i\}_{i=0}^{k}$ is called affinely independent if the vectors*

$$\{v_i - v_0\}_{i=1}^{k}$$

*are linearly independent. The k-simplex spanned by the affinely independent points $\{v_i\}_{i=0}^{k}$, denoted by $[v_0, \ldots, v_k]$, is defined as*

$$[v_0, \ldots, v_k] = \left\{ \sum_{i=0}^{k} t_i v_i \;\middle|\; t_i \geq 0,\ \sum_{i=0}^{k} t_i = 1 \right\}.$$

**Definition 3.2** (Simplicial Complex) *A simplicial complex K is a collection of simplices satisfying the following two conditions:*

1. *every face of a simplex in K is also contained in K;*
2. *the intersection of any two simplices in K is either empty or a common face of both simplices.*

With these definitions, a point cloud can be converted into a simplicial complex and analyzed through its topological features, such as connected components, loops, and higher-dimensional voids. This conversion is the starting point for applying topological data analysis to point cloud learning.

## 3.2 Vietoris–Rips Filtration

**Definition 3.3** (Vietoris–Rips Complex) *Let $X$ be a point cloud and let $\varepsilon > 0$ be a scale parameter. The Vietoris–Rips complex $VR_\varepsilon(X)$ is the simplicial complex in which a simplex*

$$[x_{i_0}, x_{i_1}, \ldots, x_{i_k}]$$

*is included if all pairwise distances between its vertices satisfy*

$$\|x_{i_p} - x_{i_q}\| \leq \varepsilon \quad \text{for all } 0 \leq p, q \leq k.$$

The Vietoris–Rips construction is widely used in topological data analysis due to its reliance on pairwise distances[7].

**Definition 3.4** (Filtration) *A filtration is a nested sequence of simplicial complexes*

$$K_{\varepsilon_1} \subseteq K_{\varepsilon_2} \subseteq \cdots \subseteq K_{\varepsilon_m},$$

*where*

$$\varepsilon_1 \leq \varepsilon_2 \leq \cdots \leq \varepsilon_m.$$

In this paper, the filtration is generated by increasing the Vietoris–Rips scale parameter:

$$VR_{\varepsilon_1}(X) \subseteq VR_{\varepsilon_2}(X) \subseteq \cdots \subseteq VR_{\varepsilon_m}(X).$$

As $\varepsilon$ increases, more edges and higher-dimensional simplices are added. This allows connectivity and loop structures to be observed across multiple scales.

## 3.3 Persistent Homology

Persistent homology provides a multi-scale description of topological features and is a central tool in topological data analysis[6-7]. It extends ordinary homology by studying how homological features appear, persist, and disappear along a filtration.

**Definition 3.5** (Homology Group and Betti Number) *Let $C_k(K)$ be the group of k-chains of a simplicial complex K, and let $\partial_k : C_k(K) \to C_{k-1}(K)$ be the boundary operator. The k-th homology group of K is defined as*

$$H_k(K) = \ker \partial_k / \operatorname{im} \partial_{k+1}.$$

*The k-th Betti number is*

$$\beta_k = \operatorname{rank}(H_k(K)),$$

*which measures the number of independent k-dimensional topological features.*

**Definition 3.6** (Persistent Homology) *Let*

$$K_0 \subseteq K_1 \subseteq \cdots \subseteq K_m$$

*be a filtration of simplicial complexes. For each homological dimension $k \geq 0$ and indices*

*$0 \le i \le j \le m$, the inclusion map*

$$K_i \hookrightarrow K_j$$

*induces a homomorphism*

$$f_k^{i,j} : H_k(K_i) \to H_k(K_j).$$

*The k-th persistent homology group from $K_i$ to $K_j$ is defined as*

$$H_k^{i,j} = \mathrm{im}(f_k^{i,j}).$$

*Its rank,*

$$\beta_k^{i,j} = \mathrm{rank}(H_k^{i,j}),$$

*is called the k-th persistent Betti number.*

Intuitively, persistent homology records which homological features survive from one scale to another. In a Vietoris–Rips filtration, as the scale parameter $\varepsilon$ increases, connected components may merge, loops may appear and disappear, and higher-dimensional voids may form and later be filled. Therefore, persistent homology captures not only the existence of topological features, but also their stability across scales.

**Definition 3.7** (Birth, Death, and Persistence) *For a topological feature in a filtration, its birth value b is the scale at which it first appears, and its death value d is the scale at which it disappears. The persistence of the feature is defined as*

$$p = d - b.$$

A feature with longer persistence is usually interpreted as a more stable structural pattern, while a feature with very short persistence is often regarded as noise or local sampling variation.

**Definition 3.8** (Persistence Diagram) *The k-dimensional persistence diagram is a multiset of birth–death pairs:*

$$D_k = \{(b_i, d_i)\}_{i=1}^{M_k}.$$

*Each point $(b_i, d_i)$ represents one k-dimensional topological feature, where $b_i$ and $d_i$ denote its birth and death values, respectively.*

In this work, $D_0$ and $D_1$ are mainly used, corresponding to connected components and one-dimensional loops. These diagrams provide compact multi-scale summaries of the global structure of a point cloud.

## 3.4 Vectorization of Topological Descriptors

Persistence diagrams are unordered multisets of points in the birth–death plane and cannot be directly used as inputs to standard neural networks. Therefore, they must be mapped to fixed-length vector representations.

**Definition 3.9** (Persistence Image) *Let*

$$D = \{(b_i, d_i)\}_{i=1}^{M}$$

*be a persistence diagram. Each point* $(b_i, d_i)$ *is first transformed into the birth–persistence coordinate*

$$u_i = (b_i, p_i) = (b_i, d_i - b_i).$$

*Given a nonnegative weight function* $w : \mathbb{R}^2 \to \mathbb{R}$ *and a smoothing kernel* $\phi_{u_i}$ *centered at* $u_i$*, the persistence surface associated with* $D$ *is defined as*

$$\rho_D(x, y) = \sum_{i=1}^{M} w(u_i) \phi_{u_i}(x, y).$$

*A persistence image is obtained by discretizing this surface over a fixed grid. For each grid cell* $P_j$*, the corresponding pixel value is*

$$I_j = \int_{P_j} \rho_D(x, y)\, dx\, dy.$$

*Thus, the persistence image is the finite-dimensional vector*

$$\mathrm{PI}(D) = (I_1, I_2, \ldots, I_N) \in \mathbb{R}^N.$$

This construction converts an unordered persistence diagram into a fixed-length vector representation that can be used by standard machine learning models. More generally, for a selected set of homology dimensions $S \subseteq \{0, 1, 2\}$, the vectorized topological descriptor can be written as

$$z_{\text{topo}} = V_{\eta_v}\left(\{D_k\}_{k \in S}\right),$$

where $D_k$ denotes the persistence diagram in homology dimension $k$, and $V_{\eta_v}$ denotes a general vectorization operator. Different choices of $V_{\eta_v}$ lead to different topological representations, such as persistence images, persistence landscapes, Betti curves, or learnable persistence embeddings.

In the experimental part of this work, persistence images are used as the concrete vectorization method. When $S = \{0,1\}$, this gives

$$z_{\text{topo}} = [\psi_{\text{PI}}(D_0), \psi_{\text{PI}}(D_1)],$$

where $D_0$ and $D_1$ are the persistence diagrams for connected components and one-dimensional loops, respectively, and $\psi_{\text{PI}}$ denotes the persistence image mapping. This mapping transforms birth–death pairs into birth–persistence coordinates, smooths them into a persistence surface, and discretizes the surface over a fixed grid.

Other vectorization methods can also be used. For example, persistence landscapes map persistence diagrams into functional summaries in a vector space, while Betti curves summarize the number of active topological features across filtration scales. These methods differ in how persistence information is transformed, weighted, and aggregated.

## 3.5 Computational Pipeline

The topological feature extraction pipeline can be written in a general form as

$$X \longrightarrow C_{\eta_c}(X) \longrightarrow F_{\eta_f}(C_{\eta_c}(X)) \longrightarrow \{D_k(X)\}_{k\in S} \longrightarrow z_{\text{topo}}.$$

Here, $X$ denotes the input point cloud, and $C_{\eta_c}$ is the complex construction operator. The term $F_{\eta_f}(C_{\eta_c}(X))$ denotes the filtration induced by the chosen complex construction and filtration rule. Persistent homology is then computed on this filtration to obtain persistence diagrams

$$\{D_k(X)\}_{k\in S},$$

where $S \subseteq \{0,1,2\}$ is the selected set of homology dimensions. Finally, these diagrams are mapped into a machine-learning-compatible representation by a vectorization operator:

$$z_{\text{topo}} = V_{\eta_v}(\{D_k(X)\}_{k\in S}).$$

In the experimental setting of this paper, the complex construction is chosen as the Vietoris–Rips complex, and the filtration is generated by increasing the scale parameter. Thus, the concrete pipeline becomes

$$X \longrightarrow VR_\varepsilon(X) \longrightarrow \{VR_{\varepsilon_j}(X)\}_{j=1}^m \longrightarrow \{D_k(X)\}_{k\in S} \longrightarrow z_{\text{topo}}.$$

The sequence

$$\{VR_{\varepsilon_j}(X)\}_{j=1}^m$$

denotes the Vietoris–Rips filtration obtained by increasing the scale parameter from $\varepsilon_1$ to $\varepsilon_m$. In the main experiments, $S = \{0,1\}$, so the diagrams $D_0(X)$ and $D_1(X)$ are used. Here, $D_0(X)$ records persistent connected-component information, while $D_1(X)$ records persistent one-dimensional loop structures. These diagrams are then vectorized as persistence images and concatenated to form the fixed-length topological descriptor

$$z_{\text{topo}} = [\psi_{\text{PI}}(D_0(X)), \psi_{\text{PI}}(D_1(X))],$$

where $\psi_{\text{PI}}$ denotes the persistence image transform applied to a single persistence diagram. Persistence images are finite-dimensional vector representations of persistence diagrams and are designed to be compatible with standard vector-based machine learning models[10].

**Computational cost and subsampling.** A practical limitation of the Vietoris–Rips filtration is its high computational cost. For a point cloud with $n$ points, the full Vietoris–Rips complex may contain exponentially many simplices in the number of vertices; therefore, practical implementations often restrict the maximum dimension or the filtration range[11]. Computing persistent homology on full point clouds is therefore expensive, especially for datasets such as ModelNet40, where each shape may contain 1024 sampled points[12].

To make the computation feasible, persistent homology is computed on a smaller subset

$$X' \subset X, \qquad |X'| \ll |X|.$$

In this work, 96 sampled points are used as a reduced representation of each point cloud. This improves efficiency, but it may remove small-scale topological features. Thus, the resulting descriptors should be viewed as approximate structural summaries rather than complete topological descriptions.

## 4. Topological Feature Representation Mechanisms

This section studies the *representation form* of topological information. Given a point cloud

$$X = \{x_i\}_{i=1}^{n} \subset \mathbb{R}^d,$$

topological information can be extracted through a general persistent-homology pipeline. For a single homology dimension $k$, the topological representation can be written as

$$\mathscr{T}_\eta^{(k)}(X) = V_{\eta_v}\left(D_k(X)\right), \qquad D_k(X) = PH_k\left(F_{\eta_f}\left(C_{\eta_c}(X)\right)\right).$$

Equivalently,

$$\mathscr{T}_\eta^{(k)}(X) = V_{\eta_v}\left(PH_k\left(F_{\eta_f}\left(C_{\eta_c}(X)\right)\right)\right).$$

Here, $C_{\eta_c}$ denotes a complex construction operator, such as a Vietoris–Rips, alpha, witness, or cubical complex. The operator $F_{\eta_f}$ denotes the filtration rule, such as distance-based, function-based, or learned filtration. The term $PH_k$ computes persistent homology in dimension $k$ and outputs the persistence diagram $D_k(X)$. The operator $V_{\eta_v}$ maps the resulting persistence information into a machine-learning-compatible representation, such as a persistence image, persistence landscape, Betti curve, or learnable persistence embedding.

The parameter collection

$$\eta = (\eta_c, \eta_f, \eta_v)$$

summarizes the design choices in the topological representation pipeline. In this work, topological representations are discussed along two dimensions. The first dimension is the spatial scope of the representation, namely whether topology is computed from the whole point cloud or from local neighborhoods. This distinction gives rise to global and local topological representations. The second dimension is the adaptivity of the representation, namely whether the topological operator is predefined before training or includes learnable components.

For a set of selected homology dimensions $S \subseteq \{0,1,2\}$, we write

$$\mathscr{T}_\eta^{(S)}(X) = V_{\eta_v}\left(\{D_k(X)\}_{k\in S}\right),$$

where $D_k(X)$ denotes the persistence diagram in homology dimension $k$. When the vectorization is applied separately to each homology dimension, this notation represents the concatenation of the resulting vectors:

$$\mathscr{T}_{\eta}^{(S)}(X) = [V_{\eta_v}(D_k(X))]_{k \in S}\,.$$

## 4.1 Spatial Scope: Global and Local Topological Representations

The spatial scope of a topological representation specifies the domain on which the topological operator $\mathscr{T}_{\eta}^{(k)}$ or $\mathscr{T}_{\eta}^{(S)}$ is applied. At the global level, topology is computed from the entire point cloud. At the local level, topology is computed from neighborhoods or patches around individual points.

### 4.1.1 Global Topological Representations

A global topological representation computes persistent homology from the whole point cloud. For a point cloud $X$, one constructs a filtration and obtains persistence diagrams

$$\mathscr{D}(X) = \{D_k(X)\}_{k \in S},$$

where $S \subseteq \{0,1,2\}$ denotes the selected homology dimensions. A fixed-length global descriptor is then obtained by

$$z_{\text{topo}}^{\text{global}} = V_{\eta_v}(\mathscr{D}(X)) = V_{\eta_v}(\{D_k(X)\}_{k \in S})\,.$$

Equivalently, using the general topological operator, this can be written as

$$z_{\text{topo}}^{\text{global}} = \mathscr{T}_{\eta}^{(S)}(X).$$

In the specific setting of this paper, the selected homology dimensions are $S = \{0,1\}$, and persistence images are used for vectorization. Let $\psi_{\text{PI}}$ denote the persistence image transform applied to a single persistence diagram. This transform maps a persistence diagram into a fixed-dimensional vector by smoothing persistence points into a persistence surface and discretizing the surface over a grid. Thus,

$$z_{\text{topo}}^{\text{global}} = [\psi_{\text{PI}}(D_0(X)), \psi_{\text{PI}}(D_1(X))]\,.$$

Here, $D_0(X)$ and $D_1(X)$ encode the persistent connected-component and loop structures of $X$, respectively, and the brackets denote concatenation of the two resulting persistence-image vectors.

The descriptor $z_{\text{topo}}^{\text{global}} \in \mathbb{R}^m$ summarizes the object-level topology of the whole point cloud. It captures multi-scale structural information such as connected components and loops. However, because it is a global summary, it does not preserve the spatial location of topological features. For example, it may indicate the existence of a loop, but it does not directly identify where the loop occurs in the point cloud.

Global topological descriptors are therefore suitable for object-level tasks such as shape classification and retrieval. Representative examples include methods that use persistence images as global descriptors for shape classification, such as TopoRec[13].

4.1.2 Local and Point-wise Topological Representations

Global descriptors lose spatial localization. To obtain spatially resolved topological information, persistent homology can be computed on local neighborhoods or patches.

For each point $x_i \in X$, let

$$\mathcal{N}(x_i) \subset X$$

denote a local neighborhood. This neighborhood can be constructed by $q$-nearest neighbors,

$$\mathcal{N}_q(x_i) = \mathrm{kNN}_q(x_i; X),$$

or by radius search,

$$\mathcal{N}_r(x_i) = \{x_j \in X : \|x_j - x_i\| \leq r\}.$$

A local topological descriptor is then defined as

$$h_i^{\text{topo}} = \mathcal{T}_\eta^{(S)}(\mathcal{N}(x_i)) = V_{\eta_v}\left(\{D_k(\mathcal{N}(x_i))\}_{k \in S}\right).$$

The collection

$$H^{\text{topo}} = \{h_i^{\text{topo}}\}_{i=1}^n$$

forms a point-wise or patch-wise topological representation of the input point cloud. This representation can be combined with local geometric features

$$h_i^{\text{geo}} = f_\theta^{\text{local}}(x_i, \mathcal{N}(x_i))$$

through a fusion operator

$$\tilde{h}_i = \Gamma_\phi\left(h_i^{\text{geo}}, h_i^{\text{topo}}\right).$$

Thus, local topological representations may be used in two forms. For point-wise prediction tasks, the model can directly use

$$H^{\text{topo}} = \{h_i^{\text{topo}}\}_{i=1}^n.$$

For object-level prediction tasks, the point-wise descriptors can be aggregated as

$$z_{\text{topo}}^{\text{local}} = \text{Agg}\left(H^{\text{topo}}\right),$$

where Agg is a permutation-invariant aggregation operator.

This formulation makes local topology analogous to other point-wise geometric features, such as normals, curvature, or local density. The main difference is that $h_i^{\text{topo}}$ describes the persistent structure of a neighborhood rather than only its metric geometry.

There is an inherent trade-off in this representation. A larger neighborhood $\mathscr{N}(x_i)$ provides broader structural context but increases computational cost. A smaller neighborhood is more efficient but may fail to capture larger-scale loops or connectivity patterns. Thus, local topological representations are especially relevant to segmentation and part-level tasks, where spatially localized structure matters. Topology-aware segmentation methods such as TopoSeg[14] are representative examples of this direction.

## 4.2 Representation Adaptivity: Fixed and Learnable Topological Representations

The second dimension concerns the adaptivity of the topological representation. Both global and local representations can be either fixed or learnable. A fixed representation uses a predefined topological pipeline, while a learnable representation allows some components of the pipeline to adapt to the downstream learning task.

For a fixed topological representation, the design parameters

$$\eta = (\eta_c, \eta_f, \eta_v)$$

are manually chosen and remain unchanged during model training. For example, a typical fixed global representation may use a Vietoris–Rips complex, a distance-based filtration, and a persistence image:

$$z_{\text{topo}}^{\text{fixed}} = \mathscr{T}_{\eta}^{(S)}(X), \qquad \eta \text{ fixed.}$$

Similarly, a fixed local representation is given by

$$h_i^{\text{topo,fixed}} = \mathscr{T}_{\eta}^{(S)}(\mathscr{N}(x_i)), \qquad \eta \text{ fixed.}$$

In contrast, a learnable topological representation introduces trainable components into the topological pipeline. This can be written as

$$\mathscr{T}_{\eta_\theta}^{(k)}(X) = V_{\eta_v,\theta_v}\left(PH_k\left(F_{\eta_f,\theta_f}\left(C_{\eta_c,\theta_c}(X)\right)\right)\right),$$

where some components of the complex construction, filtration, vectorization, or embedding are parameterized by learnable parameters $\theta$. A common case is a learnable filtration:

$$F_{\eta_f} = F_{\theta_f},$$

which gives

$$z_{\text{topo}}^{\text{learn}} = V_{\eta_v}\left(PH_k\left(F_{\theta_f}\left(C_{\eta_c}(X)\right)\right)\right).$$

This formulation can be applied globally or locally. A learnable global topological representation has the form

$$z_{\text{topo}}^{\text{global,learn}} = \mathscr{T}_{\eta_\theta}^{(S)}(X),$$

whereas a learnable local representation has the form

$$h_i^{\text{topo,learn}} = \mathscr{T}_{\eta_\theta}^{(S)}(\mathscr{N}(x_i)).$$

The learnable topological representation can then be integrated with geometric features through a neural mapping:

$$z_{\text{latent}} = \Phi_\theta\left(f_{\theta_g}(X), z_{\text{topo}}^{\text{learn}}\right),$$

where $f_{\theta_g}$ is the geometric feature extractor and $\Phi_\theta$ models the interaction between geometric and topological information.

Compared with fixed representations, learnable topological representations have two main advantages. First, the topological signal can become task-adaptive. Second, topology and geometry can interact during feature learning rather than only after feature extraction.

However, this also introduces additional difficulty, because persistent homology involves discrete changes in filtrations and is not naturally compatible with gradient-based optimization.

Representative examples include methods that embed topology into latent representations or network architectures, such as TopologyNet[15], RipsNet[16], and TopoDiT-3D[17].

# 5. Integration with Point Cloud Learning

Section 4 describes how topological information is represented. This section studies how such information is integrated into point cloud learning models. The distinction is important: representation concerns the form of topological information, while integration concerns the position and mechanism by which topology affects the learning pipeline.

Let

$$z_{\text{geo}} = f_\theta(X)$$

denote the geometric feature extracted from the point cloud by a neural backbone, and let

$$z_{\text{topo}} = \mathscr{T}_\eta^{(S)}(X)$$

denote a topological representation. Different integration strategies can be understood by where $z_{\text{topo}}$, or the corresponding topological signal, is introduced into the model.

## 5.1 Feature-level Augmentation

Feature-level augmentation is the most direct integration strategy. The geometric and topological branches are computed separately:

$$z_{\text{geo}} = f_\theta(X),$$

$$z_{\text{topo}} = \mathscr{T}_\eta^{(S)}(X).$$

They are then combined by a feature fusion operator

$$z = \Gamma_\phi\left(z_{\text{geo}}, z_{\text{topo}}\right),$$

and the final prediction is obtained by

$$\hat{y} = h_\omega(z).$$

The simplest instance of $\Gamma_\phi$ is concatenation:

$$\Gamma_\phi\left(z_{\text{geo}}, z_{\text{topo}}\right) = [z_{\text{geo}}, z_{\text{topo}}],$$

which gives

$$\hat{y} = h_\omega\left([z_{\text{geo}}, z_{\text{topo}}]\right).$$

More generally, the fusion operator may include learnable projections:

$$z = [W_g z_{\text{geo}}, W_t z_{\text{topo}}],$$

or a gated fusion mechanism:

$$z = z_{\text{geo}} + g_\phi(z_{\text{geo}}, z_{\text{topo}}) \odot W_t z_{\text{topo}},$$

where $g_\phi(\cdot) \in [0,1]^m$ controls the contribution of topological features and $\odot$ denotes element-wise multiplication.

Feature-level augmentation treats topology as an external descriptor. It is simple, stable, and easy to apply to existing architectures. However, its coupling with the geometric backbone is weak, because the topological descriptor is usually computed before training and remains fixed. Moreover, persistent homology is a many-to-one representation: geometrically different shapes may share similar or identical topological descriptors. Therefore, topology alone is usually insufficient for fine-grained classification and is more appropriate as auxiliary structural information.

A representative example is TopoRec[13]. The experiments in this paper also follow this strategy by concatenating persistence-image features with PointNet global features.

## 5.2 Representation-level Integration

Representation-level integration introduces topology into intermediate feature learning rather than only at the final feature vector. Let $H^{(\ell)}$ denote the feature representation at layer $\ell$. A general topology-aware representation update can be written as

$$H^{(\ell+1)} = \Phi_\theta^{(\ell)}\left(H^{(\ell)}, \mathscr{T}_\eta^{(\ell)}(H^{(\ell)})\right), \qquad \ell = 0, \ldots, L-1.$$

The final prediction is then

$$\hat{y} = h_\omega(H^{(L)}).$$

Here, $\mathscr{T}_\eta^{(\ell)}(H^{(\ell)})$ denotes a topological signal computed from the input, an intermediate feature graph, a latent representation, or a learned filtration at layer $\ell$. This strategy allows topology to affect how representations are formed inside the network.

A simpler formulation is

$$z_{\text{latent}} = \Phi_\theta\left(f_{\theta_g}(X), \mathscr{T}_\eta^{(S)}(X)\right),$$

where topology and geometry interact before the final prediction head:

$$\hat{y} = h_\omega(z_{\text{latent}}).$$

Compared with feature-level augmentation, representation-level integration provides stronger coupling between topology and geometry. The model can learn not only whether to use topological information, but also how topology should influence intermediate features. This is especially relevant when topology is used to guide attention, neighborhood aggregation, latent embeddings, or learned filtrations.

However, this strategy is technically more difficult. Persistent homology depends on discrete changes in topological features along a filtration, while neural networks are usually trained by gradient-based optimization. Differentiable or surrogate formulations are therefore needed. These approximations can increase computational cost and may introduce optimization instability.

Representative examples include RipsNet[16] and related differentiable persistent homology methods.

## 5.3 Loss-level Structural Constraints

Loss-level integration uses topology as a supervisory or regularizing signal rather than as an input feature. Let $\hat{Y}$ denote the model output and $Y$ denote the target structure. A general topology-aware objective is

$$\mathscr{L} = \mathscr{L}_{\text{task}}(\hat{Y}, Y) + \lambda \mathscr{L}_{\text{topo}}(\hat{Y}, Y),$$

where $\lambda > 0$ controls the strength of the topological constraint. Here, $\mathscr{L}_{\text{task}}$ denotes the standard task-specific loss used without topological constraints. For example, it can be cross-entropy loss for classification or segmentation, and Chamfer distance for point cloud reconstruction or completion.

A persistence-diagram-based topological loss can be written as

$$\mathscr{L}_{\text{topo}}(\hat{Y}, Y) = d_{\mathscr{D}}\left(D_k(\hat{Y}), D_k(Y)\right),$$

where $d_{\mathscr{D}}$ is a distance between persistence diagrams, such as a bottleneck or Wasserstein-type distance. Thus,

$$\mathscr{L} = \mathscr{L}_{\text{task}}(\hat{Y}, Y) + \lambda d_{\mathscr{D}}\left(D_k(\hat{Y}), D_k(Y)\right).$$

In tasks where only coarse topological properties are required, the topological loss may instead be defined through Betti numbers:

$$\mathscr{L}_{\text{topo}} = \sum_{k=0}^{K_{\max}} \alpha_k \left|\beta_k(\hat{Y}) - \beta_k(Y)\right|,$$

where $\beta_k$ denotes the $k$-th Betti number, $K_{\max}$ is the maximum homology dimension considered, and $\alpha_k$ controls the importance of each homology dimension.

For example, if a predicted segmentation mask is expected to be connected, one may penalize deviations from one connected component:

$$\mathscr{L}_{\text{conn}} = \left|\beta_0(\hat{Y}) - 1\right|.$$

If the task requires preserving loop structures, one may impose

$$\mathscr{L}_{\text{loop}} = \left|\beta_1(\hat{Y}) - \beta_1(Y)\right|.$$

Loss-level integration does not require topology to be used as an explicit input feature. Instead, topology constrains the structure of the output. This makes it particularly suitable for segmentation, reconstruction, completion, and generation tasks, where the structural correctness of the output is important. Its main limitation is optimization sensitivity: the effect of topology depends on the form of $\mathscr{L}_{\text{topo}}$, the value of $\lambda$, and the stability of gradients or surrogate gradients.

Topology-aware losses can therefore be understood as enforcing structural consistency at the output level, rather than enriching the input representation directly.

# 6. Comparative Insights

## 6.1 Representative Methods under the Proposed Classification

The proposed taxonomy can be used to organize existing topology-aware point cloud learning methods. Table 1 summarizes representative methods according to their spatial scope, representation adaptivity, integration level, and main role of topology. Some methods may span multiple categories; therefore, the table assigns each method according to its dominant use of topological information.

For loss-level methods such as TopoSeg, spatial scope and adaptivity are marked as "–" because topology is not used as an explicit input representation. Instead, it is imposed as a structural constraint on the model output.

Methods such as PHGCN are marked with multiple integration levels when topology affects both feature learning and the training objective.

**Table 1 Classification of Representative Topology-aware Point Cloud Learning Methods**

| Method | Task | Spatial Scope | Adaptivity | Integration Level | Main Topology Role |
|---|---|---|---|---|---|
| TopoRec[13] | Recognition / retrieval | Global | Fixed | Feature-level | Uses vectorized topological descriptors for point cloud recognition |
| TOPF[18] | Point-level learning | Local | Fixed | Feature-level | Extracts node-level topological features from point clouds |
| TopologyNet[15] | Classification / generation | Global | Learnable | Representation-level | Learns topological representations from point clouds and reduces PH computation cost |
| RipsNet[16] | PH estimation / classification support | Global | Learnable | Representation-level | Learns fast and robust approximations of vectorized persistence diagrams |
| TopoDiT-3D[17] | Point cloud generation | Global | Learnable | Representation-level | Integrates persistent-homology information into a diffusion transformer for topology-aware generation |
| TopoSeg[14] | Segmentation | – | – | Loss-level | Applies persistent-homology-based constraints to improve topological correctness of segmentation |
| PHGCN[19] | Segmentation | Local | Fixed | Representation-level / Loss-level | Uses persistence-diagram loss with graph convolution for fine-grained 3D segmentation |

## 6.2 Comparison of Topological Representations

The comparison of topological representations should be made along two separate dimensions. The first dimension is spatial scope, which distinguishes global representations from local representations. The second dimension is adaptivity, which distinguishes fixed representations from learnable representations. Tables 2 and 3 summarize the advantages and limitations of these two dimensions.

**Table 2 Comparison of Global and Local Topological Representations**

| Type | Main Form | Advantage | Limitation |
|---|---|---|---|
| Global | $z_{\text{topo}} = \mathscr{T}_\eta^{(S)}(X)$ | Provides a compact object-level structural summary. | Loses spatial localization and cannot indicate where topological features occur. |
| Local | $h_i^{\text{topo}} = \mathscr{T}_\eta^{(S)}(\mathscr{N}(x_i))$ | Preserves local or point-wise structural information. | Requires repeated local PH computation and is usually more expensive. |

**Table 3 Comparison of Fixed and Learnable Topological Representations**

| Type | Main Form | Advantage | Limitation |
|---|---|---|---|
| Fixed | $\mathscr{T}_\eta^{(S)}(\cdot)$, $\eta$ fixed | Simple, stable, and easy to precompute. | Cannot adapt the topological representation to the downstream task. |
| Learnable | $\mathscr{T}_{\eta_\theta}^{(S)}(\cdot)$ | Produces task-adaptive topological representations. | Requires differentiable or surrogate design and may introduce optimization instability. |

## 6.3 Comparison of Integration Strategies

The three integration strategies exhibit distinct advantages and limitations, which can be summarized in Table 4.

**Table 4 Comparison of Topology Integration Strategies**

| Criterion | Feature-level | Representation-level | Loss-level |
|---|---|---|---|
| Topology role | External descriptor | Intermediate feature signal | Structural constraint |
| Coupling strength | Weak | Moderate | Strong |
| Topology gradient | Not propagated | Partial / approximated | Via topology loss |
| Computational cost | Low | High | Medium–High |
| Training stability | High | Medium | Low–Medium |
| Main advantage | Simple and stable | Expressive and flexible | Structure-aware |
| Main limitation | Representation isolation | High complexity | Optimization sensitivity |
| Typical use case | Classification / retrieval | Feature learning | Segmentation / reconstruction |

Feature-level integration is limited by the fact that topological descriptors are computed independently of the learning process. As a result, the interaction between topology and geometry is fixed and cannot adapt to task-specific representations. This restriction reduces the ability of the model to fully exploit topological information during feature learning[13].

In representation-level integration, topology is incorporated into intermediate features, allowing it to influence the evolution of learned representations. This enables a more flexible interaction between geometric and topological information. However, since persistent homology is based on discrete operations, incorporating it into differentiable pipelines requires approximations, which can introduce instability during training[20-21].

Loss-level integration affects the model indirectly by constraining the output structure rather than modifying the feature space itself. The impact of topology therefore depends on how effectively the optimization process propagates these constraints back to the learned representations. This indirect influence can be sensitive to the choice of loss formulation and optimization dynamics[14].

## 6.4 Why Topology Alone is Insufficient

A central observation across existing work and empirical studies is that topology alone is insufficient for many point cloud learning tasks, particularly fine-grained classification[5,13]. This limitation can be understood from three complementary perspectives.

First, topological representations are inherently many-to-one. Persistent homology captures global structural properties while discarding detailed metric information. Therefore, multiple geometrically distinct shapes may share similar or even identical topological descriptors, leading to an information bottleneck[5-6].

Second, standard topological descriptors often lack spatial localization. Vectorized summaries such as persistence images describe the existence and persistence of topological features, but they do not directly encode where these features occur in the point cloud[10]. As a result, topology alone may fail to distinguish objects whose semantic differences depend on localized geometry.

Third, there is a trade-off between invariance and discriminability. The robustness of topological features comes from their relative insensitivity to small geometric perturbations[8], but this same invariance also removes information that may be necessary for distinguishing similar shapes.

### 6.5 Practical Implications

Based on the above analysis, several practical guidelines can be drawn.

Feature-level integration provides a simple and efficient baseline for incorporating topology. Representation-level integration enables stronger interaction between topology and geometry, but requires careful design for stability and efficiency. Loss-level integration is particularly suitable for tasks that require structural correctness[13-14,20].

In practice, the choice of integration strategy is task-dependent and should balance computational cost, stability, and representational power. Hybrid approaches that combine multiple integration levels may further improve robustness.

## 7. Experiments

To further examine the practical role of topological information in point cloud classification, this section presents a PointNet-based ablation study on ModelNet40[12]. The purpose of this experiment is to evaluate whether precomputed persistent-homology features can provide complementary information when combined with standard geometric features.

### 7.1 Experimental Setup

**Dataset.** Experiments are conducted on the ModelNet40 dataset[12], which contains 12,311 CAD models from 40 object categories. Following the standard split, 9,843 samples are used for training and 2,468 samples are used for testing. Each object is represented as a point cloud sampled to 1024 points.

**Backbone.** PointNet is used as the baseline model[2]. This choice provides a simple and interpretable setting for evaluating whether topological descriptors can improve a standard point cloud classification pipeline.

**Data Augmentation.** During training, standard point cloud augmentations are applied, including random point dropout, random scaling, and random translation, following the common PointNet training setting[2]. The same augmentation setting is used for all comparable experiments.

**Evaluation Metrics.** Two metrics are reported: Overall Accuracy (OA) and Mean Class Accuracy (mAcc). To reduce the influence of random initialization, the main results are averaged over three random seeds.

## 7.2 Implementation Details

**Training Configuration.** All models are implemented in PyTorch[22] and trained with the Adam optimizer[23]. The initial learning rate is set to 0.001, the batch size is set to 24, and all models are trained for 100 epochs. The same training configuration is used for all compared settings unless otherwise specified.

**Topology Representation.** Persistent homology is used to extract topological descriptors from point clouds[6]. To control computational cost, persistent homology is computed on a reduced point set, with 96 sampled points used in the main experiments. The resulting persistence diagrams are vectorized as persistence images[10]. Separate topological features are considered for different homology dimensions, including $H_0$ and $H_1$.

**Topology Integration.** This experiment focuses on feature-level integration. Specifically, the precomputed topological descriptor is concatenated with the global feature learned by PointNet before the final classification layers. The following variants are compared:

- **Baseline**: the original PointNet model using only geometric point cloud features;
- **Topo-Only**: a classifier using only vectorized topological descriptors;
- **Topo-Concat (All)**: PointNet features concatenated with both $H_0$ and $H_1$ topological descriptors;
- **Topo-Concat ($H_0$)**: PointNet features concatenated only with $H_0$-based descriptors;
- **Topo-Concat ($H_1$)**: PointNet features concatenated only with $H_1$-based descriptors.

## 7.3 Main Results and Quantitative Analysis

**Table 5 Classification Performance on ModelNet40**

| Method | OA | mAcc |
|---|---|---|
| Baseline (PointNet) | 0.8954 | 0.8569 |
| Topo-Only (PI) | 0.2241 | 0.1833 |
| Topo-Concat (All) | 0.8926 | 0.8533 |
| Topo-Concat ($H_0$) | 0.8970 | 0.8581 |
| Topo-Concat ($H_1$) | 0.8992 | 0.8597 |

The results show three main observations. First, the topology-only model performs much worse than the geometry-based baseline, indicating that topological descriptors alone are not sufficiently discriminative for fine-grained point cloud classification. Second, Topo-Concat (All) slightly underperforms the baseline, suggesting that simply adding all topological features may introduce redundant or weakly aligned information. Third, the $H_1$-based variant achieves the best performance, improving both OA and mAcc over the baseline. This suggests that loop-level structures provide more useful complementary information than coarse connectivity features in this setting.

## 7.4 Ablation Study

The ablation results provide a more detailed interpretation of the role of different topological signals. The gap between Baseline and Topo-Only reflects the many-to-one nature of persistent homology: while topology captures global structure, it discards metric and spatial details that are important for object recognition.

The comparison between $H_0$, $H_1$, and All further shows that topology should not be treated as a single uniform feature. $H_0$ mainly reflects connected components and provides only marginal additional information for complete object-level point clouds. In contrast, $H_1$ captures loop-like structures, such as handles or holes, which may better correspond to semantic differences among object categories. The weaker result of Topo-Concat (All) suggests that irrelevant or redundant topological components can dilute useful structural signals when fused naively.

## 7.5 Qualitative Analysis

The quantitative results can be illustrated by objects such as mugs and cylinders. These categories may share similar local surface geometry, but a mug handle introduces a non-trivial $H_1$ loop structure that is absent in a simple cylinder. This example explains why $H_1$-based descriptors can provide useful complementary information to PointNet's geometric features.

Overall, the experiment supports the view that topology is most effective as a selected auxiliary structural signal, rather than as a standalone representation or an unfiltered collection of features.

# 8. Conclusion

## 8.1 Main Findings and Contributions

This paper investigates topological feature embedding and representation for machine learning on 3D point clouds. Starting from the structural limitations of point clouds, this work explains why topology can provide complementary information for point cloud learning, especially in terms of connectivity, global structure, and multi-scale shape patterns.

The main contribution of this work is threefold. First, it provides a structured taxonomy of topological representations for point cloud learning, organized along two dimensions: spatial scope, which distinguishes global and local representations, and representation adaptivity, which distinguishes fixed and learnable representations. Second, it analyzes how topological information can be integrated into learning pipelines through feature-level augmentation, representation-level integration, and loss-level structural constraints. Third, it presents a PointNet-based ablation study on ModelNet40 to examine the practical role of persistent-homology descriptors.

The experimental results show that topology-only features are insufficient for competitive classification, while selected topological signals, especially $H_1$-based descriptors, can provide useful complementary information. This result supports the view that topology should not replace geometric learning, but should be used as a structural prior or auxiliary signal.

## 8.2 Limitations and Future Work

There are still several issues that require further study. Topological descriptors may lose metric and spatial information, and their computation can be expensive for large point clouds. Future research may focus on adaptive filtrations, more efficient persistent homology computation, and hybrid models that combine geometric and topological representations more tightly.

Overall, topology-aware learning provides a meaningful direction for improving structural understanding in point cloud models. Its effectiveness, however, depends on careful feature selection, task relevance, and appropriate integration with geometric learning.

## References


[1] GUO Y, WANG H, HU Q, et al. Deep Learning for 3D Point Clouds: A Survey[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020.

[2] QI C R, SU H, MO K, et al. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017.

[3] QI C R, YI L, SU H, et al. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space[C]//Advances in Neural Information Processing Systems. 2017.

[4] WANG Y, SUN Y, LIU Z, et al. Dynamic Graph CNN for Learning on Point Clouds [J]. ACM Transactions on Graphics, 2019.

[5] CHAZAL F, MICHEL B. An Introduction to Topological Data Analysis: Fundamental and Practical Aspects for Data Scientists[J]. arXiv preprint arXiv:1710.04019, 2017.

[6] EDELSBRUNNER H, HARER J. Persistent Homology: A Survey[J]. Contemporary Mathematics, 2008.

[7] GHRIST R. Barcodes: The Persistent Topology of Data[J]. Bulletin of the American Mathematical Society, 2008.

[8] COHEN-STEINER D, EDELSBRUNNER H, HARER J. Stability of Persistence Diagrams[J]. Discrete and Computational Geometry, 2007.

[9] EDELSBRUNNER H, HARER J. Computational Topology: An Introduction[M]. American Mathematical Society, 2010.

[10] ADAMS H, EMERSON T, KIRBY M, et al. Persistence Images: A Stable Vector Representation of Persistent Homology[J]. Journal of Machine Learning Research, 2017, 18(8): 1-35.

[11] GUDHI Editorial Board. GUDHI: Rips Complex[Z]. https://gudhi.inria.fr/doc/latest/group__rips__complex.html. Accessed 2026-05-21.

[12] WU Z, SONG S, KHOSLA A, et al. 3D ShapeNets: A Deep Representation for Volumetric Shapes[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015.

[13] GHOSH A, KULBAKA I, DAHLIN I, et al. TopoRec: Point Cloud Recognition Using Topological Data Analysis[EB/OL]. 2025. https://arxiv.org/abs/2506.18725. arXiv:

2506.18725 [cs.RO].

[14] LIU W, GUO H, ZHANG W, et al. TopoSeg: Topology-Aware Segmentation for Point Clouds[C/OL]//Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence. International Joint Conferences on Artificial Intelligence Organization, 2022: 1201-1208. DOI: 10.24963/ijcai.2022/168.

[15] ZHOU C, DONG Z, LIN H. Learning persistent homology of 3D point clouds[J/OL]. Computers & Graphics, 2022, 102: 269-279. DOI: 10.1016/j.cag.2021.10.022.

[16] De SURREL T, HENSEL F, CARRIÈRE M, et al. RipsNet: A General Architecture for Fast and Robust Estimation of the Persistent Homology of Point Clouds[EB/OL]. 2022. https://arxiv.org/abs/2202.01725. arXiv: 2202.01725 [cs.CG].

[17] GUAN Z, YAN F, DU S, et al. TopoDiT-3D: Topology-Aware Diffusion Transformer with Bottleneck Structure for 3D Point Cloud Generation[EB/OL]. 2025. https://arxiv.org/abs/2505.09140. arXiv: 2505.09140 [cs.CV].

[18] GRANDE V P, SCHAUB M T. Point-Level Topological Representation Learning on Point Clouds[C]//Proceedings of Machine Learning Research: Proceedings of the 42nd International Conference on Machine Learning: vol. 267. PMLR, 2025.

[19] WONG C C, VONG C M. Persistent Homology Based Graph Convolution Network for Fine-Grained 3D Shape Segmentation[C/OL]//Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). 2021: 7098-7107. DOI: 10.1109/ICCV48922.2021.00701.

[20] HOFER C, KWITT R, NIETHAMMER M, et al. Deep Learning with Topological Signatures[C]//Advances in Neural Information Processing Systems. 2017.

[21] GABRIELSSON R B, NELSON B J, DWARAKNATH A, et al. A Topology Layer for Machine Learning[C]//Proceedings of the International Conference on Artificial Intelligence and Statistics. 2020.

[22] PASZKE A, GROSS S, MASSA F, et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library[EB/OL]. 2019. https://arxiv.org/abs/1912.01703. arXiv: 1912.01703 [cs.LG].

[23] KINGMA D P, BA J. Adam: A Method for Stochastic Optimization[EB/OL]. 2017. https://arxiv.org/abs/1412.6980. arXiv: 1412.6980 [cs.LG].

# Acknowledgements

I would like to express my sincere gratitude to my advisor, Prof. Yifei Zhu. In some of the most difficult moments of my undergraduate studies, Prof. Zhu offered me timely help and generous support, for which I am deeply grateful. As a rigorous and dedicated teacher, he has guided me not only in academic work but also in the way I approach learning and research. Looking back on my four years of university study, I truly appreciate his continuous guidance, mentorship, and encouragement.

I would also like to thank my life mentor, Prof. Qiman Shao, for his care, guidance, and support throughout my undergraduate years.

My heartfelt thanks also go to my family and friends, whose love and patience have accompanied me through many important stages of this journey. I am grateful to my three dear roommates for the warmth and companionship they have given me in daily life, and to my cat, Jianghu, for always being by my side.

Finally, I am especially grateful to the brothers and sisters in the Bay Area and around the world. Their constant care, encouragement, and support behind the scenes have given me the courage to go the extra mile and continue moving forward.