TopoU-Net: A U-Net Architecture for Topological Domains

Anonymous authors
Paper under double-blind review

Abstract

Many modern datasets mix points, edges, regions, groups, objects, events, hyperedges, and relations. Yet neural architectures often force such data into grids, graphs, or sequences, obscuring higher-order structure and making encoder-decoder designs domain-specific. We view U-Net not as a grid-specific architecture, but as a hierarchical encoder-decoder principle: representation spaces, transport maps between levels, and skip connections between matched levels. Combinatorial complexes naturally supply these ingredients through cells, incidences, and ranks. We introduce TopoU-Net, a rank-path U-Net for topological domains. Given a path from an input rank to a bottleneck rank and back, the encoder lifts cochains upward along incidence maps, the decoder transports them downward, and skip connections merge features at matched ranks. Rank replaces spatial scale: choosing paths through nodes, edges, faces, hyperedges, or global cells becomes the central architectural decision. A key quantity is the bottleneck support ratio, the number of cells at the bottleneck relative to the number of cells at the input rank. This ratio is fixed by the complex and chosen path rather than by arbitrary pooling, and it clarifies when skip connections are optional, useful, or structurally important. Across node classification, graph classification, hypergraph node classification, point cloud classification, and image reconstruction, TopoU-Net provides a reusable encoder-decoder template for higher-order structured data. Among the evaluated baselines, it achieves the strongest mean accuracy on three of six node-classification datasets against topological baselines and four of five hypergraph datasets, with the largest gains on heterophilic graphs. Ablations show that removing skip connections is most damaging under severe bottleneck compression.

1 Introduction

Deep learning architectures are most effective when their inductive bias matches the structure of the data. CNNs assume signals live on grids, GNNs assume pairwise relational structure, and Transformers assume collections of tokens with learned interactions. Many modern datasets, however, are not naturally described by a single one of these abstractions. They contain features attached to points, relations attached to edges, regions attached to faces, groups attached to hyperedges, and sometimes global or higher-order objects. Forcing such data into only a grid, graph, or sequence can discard the very structure the model should exploit.

This raises a basic architectural question: how should information move between the different supports on which data lives? In images, U-Net architectures Ronneberger et al. (2015) answer this question through a spatial hierarchy: an encoder compresses local features into coarser context, a decoder returns this context to the output resolution, and skip connections restore fine-scale information lost through compression. The same principle, however, is not inherently grid-specific. What it requires is a hierarchy of representation spaces, transport maps between them, and skip connections between matched levels.

In non-Euclidean domains, the central question is therefore what should play the role of scale. Graph U-Nets Gao & Ji (2019) answer this through learned pooling and unpooling over nodes, but they remain tied to pairwise node–edge structure. Many scientific and relational domains contain higher-order objects—faces in meshes, hyperedges, triangles and cliques in networks, or patches

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

in images—where hierarchy is better understood as traversal across topological ranks rather than merely reduction in node count. This is the view taken in this paper: combinatorial complexes supply the representation spaces through cells, the transport maps through incidences, and the hierarchy through rank.

This paper introduces *TopoU-Net*, a U-Net architecture on combinatorial complexes Hajij et al. (2023). A combinatorial complex $\mathcal{X}$ organizes the domain into ranked cells (nodes, edges, triangles, hyperedges, or higher-order cells). TopoU-Net selects an encoder rank path $\mathcal{S} = (s_0 < \cdots < s_L)$ and transports features upward along incidence maps; the decoder reverses the path, while skip connections merge encoder and decoder features at matched ranks. Thus scale is replaced by rank traversal, turning the architecture into a U-Net over the cell structure of the domain rather than a Euclidean grid.

The rank path is the central design choice: edges capture pairwise relations, triangles or hyperedges incorporate higher-order interactions, and a global cell yields strong compression. This makes compression explicit via the *bottleneck support ratio* $\rho_{\mathrm{bot}} = n_{s_L}/n_{s_0}$, fixed by the complex and chosen path. When $\rho_{\mathrm{bot}} \ll 1$, reconstruction from the bottleneck is structurally constrained; skip connections provide a principled way to bypass severe compression, offering a concrete design principle for topological encoder–decoder networks.

TopoU-Net gives a unified encoder–decoder view across domains. Images use pixels, adjacencies, and patches; graphs use nodes, edges, triangles, and global components; hypergraphs use node–hyperedge incidence; and meshes use vertices, edges, and faces. In all cases, the architecture is unchanged: only the complex construction and rank path vary. We evaluate TopoU-Net on node classification, graph classification, hypergraph node classification, point cloud classification, and image reconstruction.

Across the evaluated baselines, TopoU-Net achieves the strongest mean accuracy on three of six node-classification datasets against topological baselines and four of five hypergraph datasets, with the largest gains on heterophilic graphs where edge-local aggregation is unreliable. Reconstruction experiments show that incidence-based rank transport can be parameter-efficient on grids, while skip-connection ablations support the predicted link between bottleneck support ratio and skip importance.

Contributions. We make four contributions. First, we formulate U-Net-style encoder–decoders on combinatorial complexes using incidence-based transport across ranks. Second, we identify the bottleneck support ratio as a structural measure of rank-path compression, providing a structural measure of rank-path compression and motivating when skip connections or wider bottleneck representations may become important. Third, we provide a canonical TopoU-Net with same-rank refinement and matched-rank skip merges. Fourth, we evaluate it across graph, hypergraph, point cloud, and image domains, with ablations isolating the roles of rank paths and skip connections.

2 Related Work

U-Net architectures. U-Net was introduced as an encoder-decoder architecture with skip connections for biomedical image segmentation Ronneberger et al. (2015). Its main architectural bias is to combine coarse contextual features with fine-scale features preserved through matched skip connections. Many variants refine this principle. Attention U-Net gates skip features using task-dependent attention signals Oktay et al. (2018), U-Net++ introduces nested skip pathways to improve feature fusion Zhou et al. (2018), ResUNet adds residual blocks for deeper encoder-decoder models Zhang et al. (2018), and UNet3+ uses full-scale skip aggregation across resolutions Huang et al. (2020). These architectures are defined on Euclidean grids, where locality and scale are provided by convolution, pooling, and upsampling. TopoU-Net keeps the encoder-decoder and skip-connection principle, but replaces Euclidean scale by rank structure in a combinatorial complex.

Graph neural networks and graph hierarchy. Graph neural networks learn representations by aggregating information over pairwise graph neighborhoods. Representative architectures include GCN Kipf & Welling (2017), GraphSAGE Hamilton et al. (2017), GAT Veličković et al. (2018), and GIN Xu et al. (2019). Extensions such as MixHop Abu-El-Haija et al. (2019) and H2GCN Zhu et al. (2020) incorporate multi-hop or heterophily-aware aggregation. Hierarchical graph models introduce coarser graph representations through learned assignment or node selection, including

DiffPool Ying et al. (2018), Graph U-Net Gao & Ji (2019), top-k pooling Cangea et al. (2018), and MinCutPool Bianchi et al. (2020). These methods define hierarchy inside a pairwise graph domain. By contrast, TopoU-Net defines hierarchy by transporting features across ranked cells, such as nodes, edges, triangles, hyperedges, or higher-order cells.

Topological deep learning. Topological deep learning (TDL) extends neural architectures from graphs to richer domains such as simplicial complexes, cell complexes, hypergraphs, and combinatorial complexes Hajij et al. (2023; 2024); Zia et al. (2024). Hodge-theoretic methods generalize graph Laplacians to signals on higher-order cells Lim (2020); Schaub et al. (2021); Roddenberry & Segarra (2019); Barbarossa & Sardellitti (2020), while message-passing and attention architectures aggregate over upper, lower, and boundary neighborhoods Hajij et al. (2020); Bodnar et al. (2022a); Giusti et al. (2022; 2023); Battiloro et al. (2023). These models provide higher-order analogues of graph neural networks, but they are usually not organized as U-shaped encoder-decoder architectures with matched cross-rank skip connections.

Topological pooling and our distinction. Pooling and coarsening on higher-order domains are less standardized than graph pooling. Existing simplicial pooling methods adapt graph-pooling ideas to simplicial complexes Cinque et al. (2023), and skip connections have been studied for higher-order networks Hajij et al. (2022). TopoU-Net takes a different approach. Instead of learning a new coarsened complex at each layer, TopoU-Net uses the existing rank structure of a combinatorial complex as the hierarchy. Incidence maps transport features across ranks, within-rank operators refine features at fixed support, and skip connections merge encoder and decoder states at matched ranks. This uses the compression induced by the rank path and links architectural choices to the cell counts of the underlying domain.

3 TopoU-Net: Architecture and Compression

TopoU-Net is an encoder–decoder architecture on combinatorial complexes (CCs). Ranked cells supply the hierarchy, incidence matrices provide transport between ranks, and skip connections merge encoder and decoder features at matched ranks. We first fix notation for cells, cochains, and incidence-based transport; the architecture is defined in Section 3.3.

3.1 Combinatorial complexes and cochain spaces

We use CC as the underlying topological domain. They allow signals to be placed on vertices, edges, faces, hyperedges, or other higher-order cells.

Definition 3.1. *A* combinatorial complex (CC) *is a triple* $(S, \mathcal{X}, \mathrm{k})$*, where* S *is a finite nonempty vertex set,* $\mathcal{X} \subseteq 2^S \setminus \{\emptyset\}$ *is a finite collection of nonempty subsets containing every singleton* $\{v\}$ *for* $v \in S$*, and* $\mathrm{k} : \mathcal{X} \to \mathbb{Z}_{\geq 0}$ *is a rank function satisfying* $\mathrm{k}(\{v\}) = 0$ *for all* $v \in S$*, and* $x \subseteq y \implies \mathrm{k}(x) \leq \mathrm{k}(y)$.

Elements of $\mathcal{X}$ are called *cells*. A cell $x \in \mathcal{X}$ with $\mathrm{k}(x) = r$ is called an r-cell. We write $\mathcal{X}^r := \{x \in \mathcal{X} : \mathrm{k}(x) = r\}$ and $n_r := |\mathcal{X}^r|$, and define the active ranks by $\mathcal{R}(\mathcal{X}) := \{r \in \mathbb{Z}_{\geq 0} : \mathcal{X}^r \neq \emptyset\}$. The dimension of the complex $\mathcal{X}$ is $\dim(\mathcal{X}) := \max\{\mathrm{k}(x)\}_{x \in \mathcal{X}}$. Unlike simplicial complexes, a CC does not need to contain all subsets of a cell, and the active ranks do not need to be consecutive. This is useful for learning because the model can use only the ranks that are present or useful for a given task.

For an active rank r, a feature signal on the r-cells is represented as an r-cochain. We identify the corresponding cochain space with $C^r(\mathcal{X}; \mathbb{R}^{d_r}) := \mathbb{R}^{n_r \times d_r}$, where rows are indexed by cells in $\mathcal{X}^r$, and d_r is the feature dimension at rank r. Thus $H^{(r)} \in C^r(\mathcal{X}; \mathbb{R}^{d_r})$ assigns one feature vector to each r-cell.

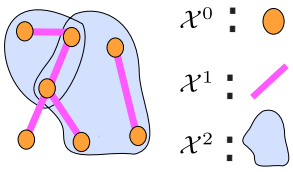

Figure 1: A combinatorial complex with cells at ranks 0, 1, and 2.

3.2 Incidence operators and rank transport

Fix an ordering of the cells at each active rank. For two active ranks $r < r'$, the incidence matrix $B_{r,r'} \in \{0,1\}^{n_r \times n_{r'}}$ is defined by $[B_{r,r'}]_{ij} = 1$ if and only if $x_i^r \subset x_j^{r'}$, where $x_i^r \in \mathcal{X}^r$ and $x_j^{r'} \in \mathcal{X}^{r'}$. Multiplying by $B_{r,r'}^\top$ moves features

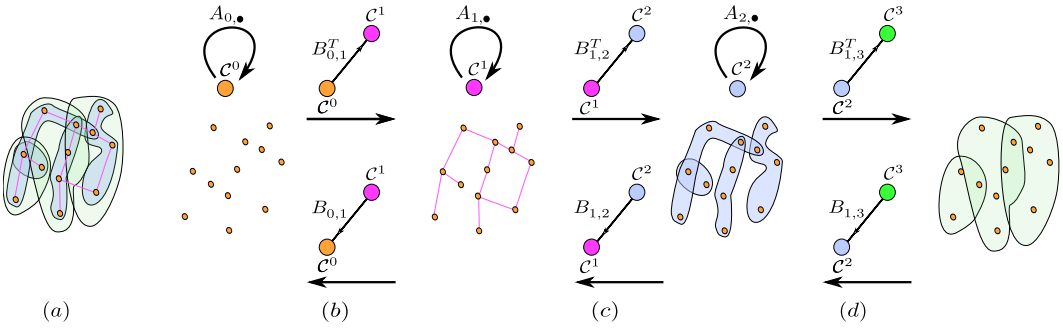

Figure 2: Rank-induced hierarchy. Incidence maps transport features from lower to higher ranks in the encoder and from higher to lower ranks in the decoder. Within-rank refinements act on cochains at a fixed rank, while skip connections merge encoder and decoder features at matched ranks.

upward from rank r to rank r', while multiplying by $B_{r,r'}$ moves features downward from rank r' to rank r: $B_{r,r'}^\top H^{(r)} \in \mathbb{R}^{n_{r'} \times d_r}$ and $B_{r,r'} H^{(r')} \in \mathbb{R}^{n_r \times d_{r'}}$. Because $B_{r,r'}$ is defined directly from inclusion, transport can be defined between any two active ranks $r < r'$.

Incidence also induces within-rank adjacency. For example, $A_{r|r'} := B_{r,r'} B_{r,r'}^\top \in \mathbb{R}^{n_r \times n_r}$ connects two r-cells when they are incident to a common r'-cell. In TopoU-Net, we distinguish these two roles; incidence matrices transport features across ranks, while adjacency-type operators such as $A_{r|r'}$ may be used to refine features within a fixed rank.

Definition 3.2 (Rank transport)**.** *Let $r, r' \in \mathcal{R}(\mathcal{X})$ with $r < r'$. An* upward rank transport *is a map $T^{\uparrow}_{r \to r'} : C^r(\mathcal{X}; \mathbb{R}^{d_r}) \to C^{r'}(\mathcal{X}; \mathbb{R}^{d_{r'}})$ constructed from the incidence relation between r-cells and r'-cells. The corresponding* downward rank transport *is a map $T^{\downarrow}_{r' \to r} : C^{r'}(\mathcal{X}; \mathbb{R}^{d_{r'}}) \to C^r(\mathcal{X}; \mathbb{R}^{d_r})$ constructed by reversing the incidence direction.*

The simplest transport pair is incidence convolution: $T^{\uparrow}_{r \to r'}(H) = \phi(B_{r,r'}^\top H W^{\uparrow}_{r,r'})$, with $W^{\uparrow}_{r,r'} \in \mathbb{R}^{d_r \times d_{r'}}$, and $T^{\downarrow}_{r' \to r}(G) = \phi(B_{r,r'} G W^{\downarrow}_{r',r})$, with $W^{\downarrow}_{r',r} \in \mathbb{R}^{d_{r'} \times d_r}$. Here ϕ is a pointwise nonlinearity. In implementations, $B_{r,r'}$ may be replaced by a fixed degree-normalized incidence matrix. This changes the weighting of incident cells, but not the type of transport map.

Throughout the paper, an *encoder rank path* is an increasing sequence of active ranks $\mathcal{S} = (s_0 < s_1 < \ldots < s_L) \subseteq \mathcal{R}(\mathcal{X})$. The bottleneck rank is s_L. The corresponding U-shaped traversal is $\mathcal{P} = (s_0, s_1, \ldots, s_L, s_{L-1}, \ldots, s_0)$. $\mathcal{S}$ determines the encoder compression structure, while $\mathcal{P}$ describes the full encoder–decoder path.

3.3 TopoU-Net definition

Let $\mathcal{S} = (s_0 < s_1 < \ldots < s_L)$ be an encoder rank path in $\mathcal{R}(\mathcal{X})$. For readability, write d_i for the feature dimension used at rank s_i. For each $i = 0, \ldots, L-1$, let $T^{\uparrow}_{s_i \to s_{i+1}} : C^{s_i}(\mathcal{X}; \mathbb{R}^{d_i}) \to C^{s_{i+1}}(\mathcal{X}; \mathbb{R}^{d_{i+1}})$ be an upward transport and let $T^{\downarrow}_{s_{i+1} \to s_i} : C^{s_{i+1}}(\mathcal{X}; \mathbb{R}^{d_{i+1}}) \to C^{s_i}(\mathcal{X}; \mathbb{R}^{d_i})$ be the corresponding downward transport. Let $\Phi_{s_{i+1}}$ and Ψ_{s_i} be within-rank refinement maps at ranks s_{i+1} and s_i, respectively. Let Ω_{s_L} be a bottleneck map at rank s_L, and let $M_{s_i} : C^{s_i}(\mathcal{X}; \mathbb{R}^{d_i}) \times C^{s_i}(\mathcal{X}; \mathbb{R}^{d_i}) \to C^{s_i}(\mathcal{X}; \mathbb{R}^{d_i})$ be a same-rank merge map.

Definition 3.3 (TopoU-Net)**.** *Given an input cochain $H^{(s_0)} \in C^{s_0}(\mathcal{X}; \mathbb{R}^{d_0})$, a TopoU-Net first sets $E_{s_0} = H^{(s_0)}$ and computes encoder states*

$$E_{s_{i+1}} = \Phi_{s_{i+1}} \left(T^{\uparrow}_{s_i \to s_{i+1}}(E_{s_i}) \right), \; i = 0, \ldots, L-1. \tag{1}$$

The decoder is initialized by $D_{s_L} = \Omega_{s_L}(E_{s_L})$ and then computes

$$\widetilde{D}_{s_i} = \Psi_{s_i} \left(T^{\downarrow}_{s_{i+1} \to s_i}(D_{s_{i+1}}) \right), \; D_{s_i} = M_{s_i}(E_{s_i}, \widetilde{D}_{s_i}), \; i = L-1, \ldots, 0. \tag{2}$$

The output at the input rank is D_{s_0}, optionally followed by a task-specific prediction head.

The definition separates three operations. Transport changes the rank and is implemented from incidence structure. Refinement acts within a fixed rank. Merge maps combine encoder and decoder features only after both are indexed by the same cells. This is the topological analogue of the alignment condition in classical U-Nets.

Proposition 3.4 (Structural compatibility)**.** *For every $i = 0, \dots, L$, the encoder state E_{s_i} lies in $C^{s_i}(\mathcal{X}; \mathbb{R}^{d_i})$. For every $i = 0, \dots, L-1$, the decoder state $\widetilde{D}_{s_i}$ and the merged state D_{s_i} also lie in $C^{s_i}(\mathcal{X}; \mathbb{R}^{d_i})$. Hence every skip merge in Definition 3.3 is type-compatible by construction.*

We next record the equivariance property used throughout the paper. Let P_r denotes a permutation matrix reordering the cells of rank r. Under a simultaneous reindexing of cells, cochains transform as $H^{(r)} \mapsto P_r H^{(r)}$, and incidence matrices transform as $B_{r,r'} \mapsto P_r B_{r,r'} P_{r'}^{\top}$. A transport family is reindexing-equivariant if, under this transformation, $T^{\uparrow}_{r \to r'}(P_r H) = P_{r'} T^{\uparrow}_{r \to r'}(H)$ and $T^{\downarrow}_{r' \to r}(P_{r'} G) = P_r T^{\downarrow}_{r' \to r}(G)$. The incidence-convolution transports in Section 3.2 satisfy this condition because they use shared feature maps and incidence matrices only through multiplication by $B_{r,r'}$ or $B_{r,r'}^{\top}$.

Proposition 3.5 (Equivariance)**.** *Assume that all transports in Definition 3.3 are reindexing-equivariant, and that the refinement maps Φ_{s_i}, Ψ_{s_i}, the bottleneck map Ω_{s_L}, and the merge maps M_{s_i} are equivariant to permutations of cells at their respective ranks. Then the TopoU-Net map $H^{(s_0)} \mapsto D_{s_0}$ is equivariant to joint reindexing of the cells along $\mathcal{S}$; if the input is replaced by $P_{s_0} H^{(s_0)}$ and all incidence matrices are reindexed consistently, then the output is replaced by $P_{s_0} D_{s_0}$.*

3.4 Bottleneck support ratios and skip connections

The encoder rank path $\mathcal{S} = (s_0 < s_1 < \ldots < s_L)$ determines the support size at each level of the architecture. Unlike classical U-Nets, where pooling ratios are design choices, these support sizes are fixed by the number of cells at the selected ranks. For each step in the encoder, define the support ratio $\rho_i := n_{s_{i+1}}/n_{s_i}$, and define the bottleneck support ratio $\rho_{\mathrm{bot}} := n_{s_L}/n_{s_0}$. A step with $\rho_i < 1$ compresses the number of cells, while a step with $\rho_i > 1$ expands it.

Proposition 3.6 (Bottleneck support and capacity)**.** *Let $\mathcal{X}$ be a CC and let $\mathcal{S} = (s_0 < s_1 < \ldots < s_L)$ be an encoder rank path. The support ratios $\rho_i = n_{s_{i+1}}/n_{s_i}$ and $\rho_{\mathrm{bot}} = n_{s_L}/n_{s_0}$ are determined by the cell counts of $\mathcal{X}$ and the chosen rank path. Moreover, consider a no-skip, bottleneck-only linear autoencoder that maps arbitrary inputs in $C^{s_0}(\mathcal{X}; \mathbb{R}^{d_0})$ through a bottleneck $C^{s_L}(\mathcal{X}; \mathbb{R}^{d_L})$ and reconstructs them at rank s_0. A necessary condition for exact reconstruction of arbitrary rank-s_0 cochains is $d_L \geq d_0/\rho_{\mathrm{bot}}$.*

The role of the capacity condition in Proposition 3.6 is to identify when a bottleneck is structurally severe. If $\rho_{\mathrm{bot}} \ll 1$, a no-skip model must either use a much larger bottleneck feature dimension or discard information that may be needed at the output rank. Skip connections provide an alternative by bypassing the compressed ranks and merging encoder features back into the decoder at matched supports.

Examples. For the Texas graph, the triangle bottleneck path $0 < 1 < 2$ has $n_0 = 183$, $n_1 = 325$, and $n_2 = 52$, giving $\rho_0 = 325/183 \approx 1.78$, $\rho_1 = 52/325 \approx 0.16$, and $\rho_{\mathrm{bot}} = 52/183 \approx 0.28$. If the path is extended to a global rank-3 cell, $0 < 1 < 2 < 3$, then $n_3 = 1$ and $\rho_{\mathrm{bot}} = 1/183 \approx 0.006$. Thus the global bottleneck is much more compressive than the triangle bottleneck. By contrast, a 28×28 image grid represented by pixels, grid adjacencies, and 2×2 patches has $n_0 = 784$, $n_1 = 1512$, and $n_2 = 729$, so the path $0 < 1 < 2$ has $\rho_{\mathrm{bot}} = 729/784 \approx 0.93$. In this case the bottleneck retains nearly the same number of cells as the input rank, so reconstruction without skip connections is structurally less constrained.

3.5 Canonical instantiation

The general definition allows different choices of transport, refinement, and merge maps. The canonical TopoU-Net used in our experiments takes cross-rank transport to be incidence convolution along the selected encoder rank path $\mathcal{S} = (s_0 < \ldots < s_L)$. For each consecutive pair of ranks in the path,

(s_i, s_{i+1}), let $\bar{B}_{s_i,s_{i+1}}$ denote either the raw incidence matrix $B_{s_i,s_{i+1}}$ or a fixed degree-normalized incidence matrix with the same support. The upward and downward transports are

$$T^{\uparrow}_{s_i \to s_{i+1}}(H) = \phi\left(\bar{B}^{\top}_{s_i,s_{i+1}} H W^{\uparrow}_i\right),\ T^{\downarrow}_{s_{i+1} \to s_i}(G) = \phi\left(\bar{B}_{s_i,s_{i+1}} G W^{\downarrow}_i\right),$$

where $W^{\uparrow}_i \in \mathbb{R}^{d_i \times d_{i+1}}$ and $W^{\downarrow}_i \in \mathbb{R}^{d_{i+1} \times d_i}$. If $s_{i+1} - s_i > 1$, the same formula uses the direct incidence matrix $B_{s_i,s_{i+1}}$. Within-rank refinements Φ_{s_i} and Ψ_{s_i} are applied after transport and do not change the rank or support. They may be implemented as pointwise MLPs, or as same-rank message-passing blocks using incidence-induced adjacencies. For active ranks $r < q$, the operator $A_{r|q} = B_{r,q}B^{\top}_{r,q}$ connects r-cells that are incident to a common q-cell. For $q < r$, the operator $A_{r|q} = B^{\top}_{q,r}B_{q,r}$ gives the analogous adjacency on r-cells through shared lower-rank cells. These refinements update features within a fixed cochain space. The hierarchy of the architecture is determined by the cross-rank incidence transports.

For example, if $\mathcal{S} = (0 < 2 < 3)$, the encoder transports features as $E_0 \xrightarrow{\bar{B}^{\top}_{0,2}} E_2 \xrightarrow{\bar{B}^{\top}_{2,3}} E_3$. The decoder starts from $D_3 = \Omega_3(E_3)$ and reverses the path as

$$D_3 \xrightarrow{\bar{B}_{2,3}} \widetilde{D}_2 \xrightarrow{M_2(E_2,\cdot)} D_2 \xrightarrow{\bar{B}_{0,2}} \widetilde{D}_0 \xrightarrow{M_0(E_0,\cdot)} D_0.$$

The skip merges occur only at matched ranks, here ranks 2 and 0, where the encoder and decoder states are indexed by the same cells.

The merge operator must act within a common cochain space. In the default additive merge, the decoder state at rank s_i is $D_{s_i} = \phi((E_{s_i} + \widetilde{D}_{s_i})W^m_i)$, where $W^m_i \in \mathbb{R}^{d_i \times d_i}$ is shared across cells of rank s_i. Other equivariant within-rank merges, such as concatenation or gated fusion, can be used without changing the definition. In no-skip ablations, the merge is removed and the decoder uses $D_{s_i} = \widetilde{D}_{s_i}$. This isolates the effect of bypassing compressed bottleneck ranks.

4 Experimental study

We evaluate TopoU-Net on node classification, graph classification, hypergraph node classification, 3D point-cloud classification, and image segmentation. Image reconstruction results are reported in Appendix B.2.

Rank structure construction. Each domain is represented as a combinatorial complex with domain-specific rank assignment. Graphs use TopoNetX Hajij et al. (2024) to construct nodes (rank 0), edges (rank 1), and triangles as maximal 3-cliques (rank 2). For heterophilic graph datasets, we additionally use a single global rank-3 cell incident to all rank-2 triangle cells. This is the construction used in the compression tables, where $n_3 = 1$. Hypergraphs use native node–hyperedge incidence as the $0 \to 1$ transport operator $H \in \{0,1\}^{|V| \times |\mathcal{E}|}$, with rank 2 cells formed from triples of hyperedges satisfying a fixed pairwise-overlap criterion. For point-cloud classification on ModelNet10/40 Wu et al. (2015), each shape is sampled into 512 points. Sampled points are treated as rank 0 cells, symmetric kNN adjacencies as rank 1 cells, and local kNN clique triangles as inferred rank 2 cells.

Rank path selection. Rank paths are selected based on dataset structure rather than using a single architecture across all datasets. For homophilic graphs (Computers, Photo), the path $0 \to 1 \to 2 \to 1 \to 0$ is used, allowing information to propagate from nodes through edges and triangles before being decoded back to the node rank. For heterophilic graphs (Cornell, Wisconsin, Texas, Actor, Chameleon, Squirrel), the path $0 \to 1 \to 2 \to 3 \to 2 \to 1 \to 0$ additionally introduces a global aggregation channel through the rank-3 cell. Graph classification, hypergraph node classification, and point-cloud classification use the path $0 \to 1 \to 2 \to 1 \to 0$. The relevant structural quantity is the bottleneck support ratio ρ_{bot} from Proposition 3.6, analyzed in Section 4.6.1.

4.1 Node classification

Table 1 reports accuracy across eight benchmarks compared against GCN Kipf & Welling (2017), GraphSAGE Hamilton et al. (2017), GAT Veličković et al. (2018), GIN Xu et al. (2019),

H2GCN Zhu et al. (2020), MixHop Abu-El-Haija et al. (2019), DiffPool Ying et al. (2018), and Graph U-Net Gao & Ji (2019). TopoU-Net achieves the highest mean accuracy on four of the eight datasets and three of the six datasets for strong topological baselines.

Table 1: Test accuracy (%) across node classification benchmarks.

Model	Computers	Photo	Actor	Chameleon	Squirrel	Cornell	Wisconsin	Texas
GCN	66.4 ± 4.0	77.2 ± 4.9	33.1 ± 0.9	41.1 ± 2.6	28.1 ± 0.9	47.8 ± 9.3	62.1 ± 7.3	62.9 ± 4.3
GraphSAGE	$\mathbf{80.4 \pm 0.8}$	$\mathbf{89.1 \pm 1.5}$	33.7 ± 0.8	47.3 ± 2.8	31.7 ± 1.7	70.5 ± 5.1	82.7 ± 3.1	81.0 ± 5.8
GAT	74.4 ± 2.3	85.7 ± 2.5	29.3 ± 1.0	44.8 ± 2.8	26.1 ± 0.5	46.4 ± 8.5	61.1 ± 6.3	59.4 ± 4.5
GIN	73.6 ± 2.9	81.8 ± 3.7	23.7 ± 1.6	48.2 ± 2.4	31.5 ± 1.9	52.9 ± 7.1	62.5 ± 3.3	64.3 ± 7.9
H2GCN	46.0 ± 2.2	75.2 ± 2.0	30.0 ± 1.9	37.7 ± 3.2	24.4 ± 1.6	72.4 ± 6.9	81.7 ± 3.2	79.1 ± 6.2
MixHop	67.6 ± 2.4	78.1 ± 4.0	29.8 ± 2.0	36.4 ± 1.7	23.2 ± 1.9	71.3 ± 2.4	83.1 ± 3.0	75.1 ± 5.9
DiffPool	57.7 ± 12	82.4 ± 4.8	25.0 ± 2.2	$\mathbf{49.3 \pm 2.6}$	$\mathbf{33.7 \pm 1.4}$	48.6 ± 6.9	58.6 ± 5.0	59.1 ± 5.3
Graph U-Net	71.2 ± 1.0	54.4 ± 6.6	35.9 ± 1.0	41.3 ± 1.4	33.5 ± 1.6	46.7 ± 6.0	60.3 ± 6.1	60.0 ± 6.9
TopoU-Net	73.6 ± 1.9	83.2 ± 1.1	$\mathbf{36.8 \pm 1.2}$	46.8 ± 1.3	33.2 ± 4.0	$\mathbf{73.5 \pm 5.8}$	$\mathbf{85.6 \pm 4.2}$	$\mathbf{81.6 \pm 5.6}$

These results are consistent with the mechanism described in Section 3.4. On heterophilic graphs, immediate edge neighborhoods can be less reliable for label prediction, so aggregation through triangles and the global rank-3 cell can provide useful additional context.

On the homophilic co-purchase graphs, Computers and Photo, GraphSAGE remains the strongest evaluated baseline. This is consistent with strong local homophily, where edge-neighborhood aggregation is already effective and higher-order rank paths are less important.

Table 2: Accuracy (%) over 10 splits.

Dataset	SheafD	SCCNN	CWN	TopoU-Net
Actor	35.41 ± 1.09	35.36 ± 1.14	35.58 ± 0.72	$\mathbf{36.89 \pm 1.28}$
Squirrel	34.35 ± 1.90	$\mathbf{36.12 \pm 2.18}$	35.77 ± 1.86	33.25 ± 4.04
Chameleon	$\mathbf{51.64 \pm 2.03}$	51.14 ± 2.97	49.74 ± 2.72	46.86 ± 2.31
Cornell	$\mathbf{75.41 \pm 4.59}$	72.16 ± 4.36	71.62 ± 4.87	73.51 ± 5.81
Wisconsin	82.75 ± 7.48	83.33 ± 4.32	84.12 ± 3.87	$\mathbf{85.69 \pm 4.24}$
Texas	80.27 ± 4.84	77.84 ± 4.65	77.30 ± 6.64	$\mathbf{81.62 \pm 5.67}$

We further compare TopoU-Net against SheafD (Bodnar et al., 2022b), SCCNN (Yang & Isufi, 2023), and CWN (Bodnar et al., 2021) in Table 2. TopoU-Net achieves the highest mean accuracy on Actor, Wisconsin, and Texas.

4.2 Graph classification

Table 3 reports graph classification accuracy on MUTAG, PROTEINS, and IMDB-BINARY using the rank path $0 \to 1 \to 2 \to 1 \to 0$ with global mean pooling. TopoU-Net achieves the highest mean accuracy on MUTAG and IMDB-BINARY and remains competitive on PROTEINS.

Table 3: Test@BestVal accuracy (%) on graph classification.

Model	MUTAG	PROTEINS	IMDB
GCN	84.5 ± 7.6	74.5 ± 5.5	48.8 ± 4.1
GraphSAGE	82.9 ± 8.1	74.2 ± 5.2	49.8 ± 4.8
GAT	71.8 ± 3.7	$\mathbf{74.7 \pm 4.0}$	48.3 ± 3.1
GIN	91.4 ± 5.1	73.4 ± 3.9	69.2 ± 3.7
Graph U-Net	82.9 ± 9.5	74.7 ± 3.9	50.5 ± 3.6
TopoU-Net	$\mathbf{91.5 \pm 7.9}$	73.4 ± 4.2	$\mathbf{71.1 \pm 3.3}$

These results suggest that rank-2 structure is useful for graph-level prediction in domains with higher-order substructure, such as molecular graphs and interaction networks.

4.3 Hypergraph node classification

Given a hypergraph $\mathcal{H} = (V, \mathcal{E})$ with incidence matrix $H \in \{0,1\}^{|V| \times |\mathcal{E}|}$, we use H as the transport operator between ranks 0 and 1. Rank-2 cells are constructed from triples of hyperedges satisfying the fixed overlap criterion, giving incidence matrix $B_{1,2}$ and the path $0 \to 1 \to 2 \to 1 \to 0$ with skip connections at ranks 0 and 1. Table 4 reports accuracy on co-citation and co-authorship hypergraph benchmarks Yadati et al. (2019), implemented using the DHG library Feng et al. (2019). TopoU-Net attains the highest mean accuracy on four of five datasets compared with MLP, HGNN Feng et al. (2019); Gao et al. (2023), HyperSAGE Arya et al. (2020), and HNHN Dong et al. (2020).

The MLP is competitive on several co-citation datasets, suggesting that node features are already informative. TopoU-Net improves over the MLP on all reported hypergraph datasets, indicating that the rank path can add useful higher-order signal. The largest gains occur on co-authorship datasets, where hyperedges directly encode group interactions.

Table 4: Hypergraph node classification accuracy (%).

Model	CocitationCora	CocitationCiteseer	CocitationPubmed	CoauthorshipCora	CoauthorshipDBLP
MLP	46.15 ± 1.93	47.65 ± 0.94	36.13 ± 8.59	46.15 ± 1.93	78.55 ± 0.23
HGNN	42.79 ± 1.01	35.83 ± 0.70	22.32 ± 2.14	59.36 ± 1.20	87.65 ± 0.11
HyperSAGE	44.58 ± 0.72	44.44 ± 0.91	$\mathbf{39.83 \pm 0.31}$	47.10 ± 0.47	82.64 ± 0.34
HNHN	47.94 ± 1.79	45.72 ± 0.64	36.81 ± 1.38	55.97 ± 1.82	86.12 ± 0.46
TopoU-Net	$\mathbf{51.63 \pm 1.80}$	$\mathbf{48.73 \pm 1.28}$	39.67 ± 0.37	$\mathbf{63.57 \pm 1.62}$	$\mathbf{87.81 \pm 0.16}$

4.4 Point-cloud classification

We evaluate TopoU-Net on ModelNet10/40 Wu et al. (2015) in a point-cloud setting. Using the path $0 \to 1 \to 2 \to 1 \to 0$, TopoU-Net achieves the highest mean accuracy on ModelNet40 and remains competitive on ModelNet10. PointNet operates directly on the point set, GraphSAGE uses a kNN graph, and TopoU-Net lifts the point cloud to a rank-$0/1/2$ complex with sampled points as 0-cells, symmetric kNN edges as 1-cells, and local kNN clique triangles as inferred 2-cells.

Table 5: Point-cloud classification accuracy.

Model	ModelNet10	ModelNet40
PointNet	0.8893 ± 0.0008	0.8258 ± 0.0006
GraphSAGE	$\mathbf{0.8959 \pm 0.0008}$	0.8179 ± 0.0066
TopoU-Net	0.8855 ± 0.0062	$\mathbf{0.8318 \pm 0.0074}$

4.5 Image segmentation

For image segmentation, we evaluate a learned-transport variant inspired by TopoU-Net. In this variant, the rank-transport matrices are learned soft incidence maps rather than fixed incidence matrices from an explicit combinatorial complex. Given pixel features, the model transports information from pixels to latent topological slots and back. We report two variants: TopoU-Net Soft, a small learned-transport model without U-Net skips, and TopoU-Net-S2, a compact skip-connected multi-kernel version. Results are shown on Oxford-IIIT Pet Parkhi et al. (2012) and Pascal VOC 2012 Everingham et al.. See Table 6.

The learned-transport variants give competitive segmentation accuracy in a smaller parameter regime. TopoU-Net Soft uses 0.42M parameters and remains close to the classical baselines on Oxford-IIIT Pet. TopoU-Net-S2 improves over the soft version while using roughly half the parameters of U-Net, and obtains the highest Pascal VOC accuracy among the reported models.

Table 6: Image segmentation accuracy.

Model	Params	Pet	VOC
U-Net	1.93M	0.8508 ± 0.0030	0.7440 ± 0.0002
ResUNet	2.88M	0.8505 ± 0.0005	0.7421 ± 0.0002
Attention U-Net	1.95M	$\mathbf{0.8545 \pm 0.0025}$	0.7449 ± 0.0021
DeepLab-lite	2.06M	$\mathbf{0.8545 \pm 0.0018}$	0.7421 ± 0.0041
TopoU-Net Soft	0.42M	0.8440 ± 0.0023	0.7360 ± 0.0018
TopoU-Net-S2	0.97M	0.8449 ± 0.0028	$\mathbf{0.7453 \pm 0.0019}$

4.6 Ablation studies

We study how bottleneck support, rank-path complexity, and skip connections affect performance.

4.6.1 Compression profiles

Table 7 reports support profiles along the encoder rank paths. We use $\rho_i = n_{s_{i+1}}/n_{s_i}$ for consecutive encoder steps and $\rho_{\mathrm{bot}} = n_{s_L}/n_{s_0}$ for the bottleneck support ratio. Ratios below one indicate support compression; ratios above one indicate support expansion.

The heterophilic paths ending in a global rank-3 cell have $\rho_{\mathrm{bot}} < 0.01$, so the bottleneck support is extremely compressed. The grid path has $\rho_{\mathrm{bot}} \approx 0.93$, so the bottleneck retains nearly the same number of cells as the input rank. For the homophilic datasets, the first transport from rank 0 to rank 1 expands the support, while the subsequent transport to rank 2 produces moderate overall bottleneck compression, with $\rho_{\mathrm{bot}} \approx 0.65$ for Computers and $\rho_{\mathrm{bot}} \approx 0.59$ for Photo.

Table 7: Support profiles along rank paths. The Path column reports the full U-shaped encoder–decoder traversal. The step ratios ρ_i and bottleneck ratio ρ_{bot} are computed along the encoder branch from the input rank to the bottleneck rank.

Dataset	n_0	n_1	n_2	n_3	ρ_0	ρ_1	ρ_2	ρ_{bot}	Path
Heterophilic									
Texas	183	325	52	1	1.78	0.16	0.02	0.006	$0\to1\to2\to3\to2\to1\to0$
Wisconsin	251	515	89	1	2.05	0.17	0.01	0.004	$0\to1\to2\to3\to2\to1\to0$
Cornell	183	298	47	1	1.63	0.16	0.02	0.005	$0\to1\to2\to3\to2\to1\to0$
Chameleon	2277	36101	1507	1	15.9	0.04	0.001	0.0004	$0\to1\to2\to3\to2\to1\to0$
Homophilic									
Computers	13752	245861	8943	—	17.9	0.04	—	0.65	$0\to1\to2\to1\to0$
Photo	7650	119081	4521	—	15.6	0.04	—	0.59	$0\to1\to2\to1\to0$
Grid									
28×28	784	1512	729	—	1.93	0.48	—	0.93	$0\to1\to2\to1\to0$

4.6.2 Rank path complexity

Table 8 varies the rank path on heterophilic datasets. Richer paths improve performance, with gains up to 12.5 points on Texas over the minimal $0\to1\to0$ path. Since higher ranks also increase depth and parameter count, these results demonstrate the practical value of rank-path selection and support the view that rank-path selection is a central part of the TopoU-Net design

Table 8: Node accuracy (%) versus rank-path complexity on heterophilic datasets.

Path	Cells	ρ_{bot}	Params	Actor	Chameleon	Squirrel	Cornell	Wisconsin	Texas
$0\to1\to0$	Nodes, edges	1.0	280K	29.7±2.6	32.9±4.0	25.6±1.3	53.2±8.6	71.3±4.4	64.5±6.9
$0\to1\to2\to1\to0$	+Triangles	0.28	301K	34.9±5.7	45.5±2.1	33.1±2.2	56.2±7.8	73.3±4.3	69.1±7.0
$0\to1\to2\to3\to2\to1\to0$	+Global	0.006	322K	**36.8±1.2**	**46.8±2.3**	**33.1±4.0**	**73.5±5.8**	**85.6±4.2**	**81.6±5.6**

4.6.3 Skip connections

Table 9 tests the prediction of Proposition 3.6: when the bottleneck support is severely compressed, removing skip connections should be especially damaging. This trend is visible on Texas, Wisconsin, and Cornell, where $\rho_{\mathrm{bot}} < 0.01$ and removing skips reduces accuracy by 12–14 percentage points. By contrast, the homophilic paths $0\to1\to2\to1\to0$ exhibit substantially milder bottleneck compression, with $\rho_{\mathrm{bot}} = 0.65$ for Computers and $\rho_{\mathrm{bot}} = 0.59$ for Photo, and show smaller degradation when skip connections are removed. For the grid reconstruction row, the metric is MSE rather than accuracy, so the reported effect is a relative error increase.

Table 9: Effect of removing skip connections, grouped by rank path.

Dataset	ρ_{bot}	Skip	No Skip	Δ
Path: $0\to1\to2\to3\to2\to1\to0$				
Texas	.006	81.6±5.6	67.3±6.2	−14.3
Wisconsin	.004	85.6±4.2	71.1±5.8	−14.5
Cornell	.005	73.5±5.8	61.4±7.1	−12.1
Path: $0\to1\to2\to1\to0$				
Computers	.65	73.6±1.9	68.2±2.4	−5.4
Photo	.59	83.2±1.1	79.1±1.8	−4.1
Path: $0\to1\to2\to1\to0$				
Grid	.93	9.5×10^{-4}	1.2×10^{-3}	26%

5 Conclusion

TopoU-Net shows that the U-Net principle extends beyond spatial resolution: on combinatorial complexes, rank provides the hierarchy, incidence provides transport, and matched ranks provide natural skip connections. This view yields a single encoder–decoder construction across graphs, hypergraphs, meshes, and grids, while the bottleneck support ratio makes the amount of structural compression explicit. Empirically, richer rank paths can improve prediction, while skip connections matter most when the chosen path induces a severe bottleneck. The main open problem is therefore not the encoder–decoder mechanism itself, but how to construct and select useful higher-order ranks efficiently and automatically.

AI USE STATEMENT

Generative AI tools were used to assist with language editing, clarity, and presentation of the manuscript. All technical content, results, analyses, and conclusions were reviewed and verified by the authors. We take full responsibility for the final content of this work, including any text or artifacts produced with the aid of generative AI.

REFERENCES

Sami Abu-El-Haija, Bryan Perozzi, Amol Kapoor, Nazanin Alipourfard, Kristina Lerman, Hrayr Harutyunyan, Greg Ver Steeg, and Aram Galstyan. MixHop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 21–29, 2019.

Devanshu Arya, Deepak K. Gupta, Stevan Rudinac, and Marcel Worring. Hypersage: Generalizing inductive representation learning on hypergraphs. *arXiv preprint arXiv:2010.04558*, 2020.

Sergio Barbarossa and Stefania Sardellitti. Topological signal processing over simplicial complexes. *IEEE Transactions on Signal Processing*, 68:2992–3007, 2020. doi: 10.1109/TSP.2020.2981920.

Claudio Battiloro, Lucia Testa, Lorenzo Giusti, Stefania Sardellitti, Paolo Di Lorenzo, and Sergio Barbarossa. Generalized simplicial attention neural networks. *arXiv preprint arXiv:2309.02138*, 2023.

Filippo Maria Bianchi, Daniele Grattarola, and Cesare Alippi. Spectral clustering with graph neural networks for graph pooling. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 874–883, 2020.

Cristian Bodnar, Fabrizio Frasca, Nina Otter, Yuguang Wang, Pietro Liò, Guido F. Montufar, and Michael Bronstein. Weisfeiler and lehman go cellular: Cw networks. In *Advances in Neural Information Processing Systems*, volume 34, pp. 2625–2640, 2021.

Cristian Bodnar, Francesco Di Giovanni, Benjamin Paul Chamberlain, Pietro Liò, and Michael M. Bronstein. Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in GNNs. In *Advances in Neural Information Processing Systems*, volume 35, pp. 18527–18541, 2022a.

Cristian Bodnar, Francesco Di Giovanni, Benjamin Paul Chamberlain, Pietro Liò, and Michael M. Bronstein. Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in GNNs. In *Advances in Neural Information Processing Systems*, volume 35, 2022b.

Cătălina Cangea, Petar Veličković, Nikola Jovanović, Thomas Kipf, and Pietro Liò. Towards sparse hierarchical graph classifiers. In *NeurIPS Workshop on Relational Representation Learning*, 2018. arXiv:1811.01287.

Domenico Mattia Cinque, Claudio Battiloro, and Paolo Di Lorenzo. Pooling strategies for simplicial convolutional networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, 2023. doi: 10.1109/ICASSP49357.2023.10096866.

Yihe Dong, Will Sawin, and Yoshua Bengio. Hnhn: Hypergraph networks with hyperedge neurons. In *ICML 2020 Workshop on Graph Representation Learning and Beyond (GRL+)*, 2020.

Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88 (2):303–338.

Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 3558–3565, 2019. doi: 10.1609/aaai.v33i01.33013558.

Hongyang Gao and Shuiwang Ji. Graph U-Nets. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2083–2092, 2019.

Yue Gao, Yifan Feng, Shuyi Ji, and Rongrong Ji. Hgnn+: General hypergraph neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3181–3199, 2023. doi: 10.1109/TPAMI.2022.3182052.

Lorenzo Giusti, Claudio Battiloro, Paolo Di Lorenzo, Stefania Sardellitti, and Sergio Barbarossa. Simplicial attention neural networks. *arXiv preprint arXiv:2203.07485*, 2022.

Lorenzo Giusti, Claudio Battiloro, Lucia Testa, Paolo Di Lorenzo, Stefania Sardellitti, and Sergio Barbarossa. Cell attention networks. In *International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2023. doi: 10.1109/IJCNN54540.2023.10191962.

Mustafa Hajij, Kyle Istvan, and Ghada Zamzmi. Cell complex neural networks. In *NeurIPS Workshop on Topological Data Analysis and Beyond*, 2020. arXiv:2010.00743 [cs.LG].

Mustafa Hajij, Karthikeyan Natesan Ramamurthy, Aldo Guzmán-Sáenz, and Ghada Zamzmi. High skip networks: A higher order generalization of skip connections. In *ICLR 2022 Workshop on Geometrical and Topological Representation Learning*, 2022.

Mustafa Hajij, Ghada Zamzmi, Theodore Papamarkou, Nina Miolane, Aldo Guzmán-Sáenz, Karthikeyan Natesan Ramamurthy, Tolga Birdal, Tamal K. Dey, Soham Mukherjee, Shreyas N. Samaga, Neal Livesay, Robin Walters, Paul Rosen, and Michael T. Schaub. Topological deep learning: Going beyond graph data. *arXiv preprint arXiv:2206.00606*, 2023.

Mustafa Hajij, Mathilde Papillon, Florian Frantzen, Jens Agerberg, Ibrahem AlJabea, Ruben Ballester, Claudio Battiloro, Guillermo Bernárdez, Tolga Birdal, Aiden Brent, Peter Chin, Sergio Escalera, Simone Fiorellino, Odin Hoff Gardaa, Gurusankar Gopalakrishnan, Devendra Govil, Josef Hoppe, Maneel Reddy Karri, Jude Khouja, Manuel Lecha, Neal Livesay, Jan Meißner, Soham Mukherjee, Alexander Nikitin, Theodore Papamarkou, Jaro Pílepok, Karthikeyan Natesan Ramamurthy, Paul Rosen, Aldo Guzmán-Sáenz, Alessandro Salatiello, Shreyas N. Samaga, Simone Scardapane, Michael T. Schaub, Luca Scofano, Indro Spinelli, Lev Telyatnikov, Quang Truong, Robin Walters, Maosheng Yang, Olga Zaghen, Ghada Zamzmi, Ali Zia, and Nina Miolane. Topox: A suite of python packages for machine learning on topological domains. *arXiv preprint arXiv:2402.02441*, 2024.

William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, volume 30, pp. 1024–1034, 2017.

Huimin Huang, Lanfen Lin, Ruofeng Tong, Hongjie Hu, Qiaowei Zhang, Yutaro Iwamoto, Xianhua Han, Yen-Wei Chen, and Jian Wu. Unet 3+: A full-scale connected unet for medical image segmentation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1055–1059, 2020. doi: 10.1109/ICASSP40776.2020.9053405.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017. arXiv:1609.02907.

Lek-Heng Lim. Hodge laplacians on graphs. *SIAM Review*, 62(3):685–715, 2020. doi: 10.1137/18M1202688.

Ozan Oktay, Jo Schlemper, Loic Le Folgoc, Matthew Lee, Mattias Heinrich, Kazunari Misawa, Kensaku Mori, Steven McDonagh, Nils Y. Hammerla, Bernhard Kainz, Ben Glocker, and Daniel Rueckert. Attention u-net: Learning where to look for the pancreas. In *Medical Imaging with Deep Learning (MIDL)*, 2018. arXiv:1804.03999.

Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3498–3505, 2012.

T. Mitchell Roddenberry and Santiago Segarra. Hodgenet: Graph neural networks for edge data. In *Asilomar Conference on Signals, Systems, and Computers*, pp. 220–224, 2019.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *Lecture Notes in Computer Science*, pp. 234–241. Springer, 2015. doi: 10.1007/978-3-319-24574-4_28.

Michael T. Schaub, Yu Zhu, Jean-Baptiste Seby, T. Mitchell Roddenberry, and Santiago Segarra. Signal processing on higher-order networks: Livin' on the edge... and beyond. *Signal Processing*, 187:108149, 2021. doi: 10.1016/j.sigpro.2021.108149.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations (ICLR)*, 2018. arXiv:1710.10903.

Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1912–1920, 2015.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations (ICLR)*, 2019. arXiv:1810.00826.

Naganand Yadati, Madhav Nimishakavi, Prateek Yadav, Vihari Nitin, Anand Louis, and Partha Talukdar. Hypergcn: A new method for training graph convolutional networks on hypergraphs. In Hanna Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché Buc, Emily Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32, pp. 1509–1520. Curran Associates, Inc., 2019.

Maosheng Yang and Elvin Isufi. Convolutional learning on simplicial complexes. *arXiv preprint arXiv:2301.11163*, 2023.

Rex Ying, Jiaxuan You, Christopher Morris, Xiang Ren, William L. Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. In *Advances in Neural Information Processing Systems*, volume 31, pp. 4800–4810, 2018.

Zhengxin Zhang, Qingjie Liu, and Yunhong Wang. Road extraction by deep residual u-net. *IEEE Geoscience and Remote Sensing Letters*, 15(5):749–753, 2018. doi: 10.1109/LGRS.2018.2802944.

Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. Unet++: A nested u-net architecture for medical image segmentation. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, volume 11045 of *Lecture Notes in Computer Science*, pp. 3–11. Springer, 2018. doi: 10.1007/978-3-030-00889-5_1.

Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs. In *Advances in Neural Information Processing Systems*, volume 33, pp. 7790–7801, 2020.

Ali Zia, Abdelwahed Khamis, James Nichols, Usman Bashir Tayab, Zeeshan Hayder, Vivien Rolland, Eric Stone, and Lars Petersson. Topological deep learning: A review of an emerging paradigm. *Artificial Intelligence Review*, 57(4):77, 2024. doi: 10.1007/s10462-023-10692-9.

A Proofs and additional theoretical elaborations

A.1 Additional rank-transport parameterizations

This appendix lists several transport parameterizations compatible with the rank-transport definition in Section 3.2. The important constraint is that any attention or gating weights must be computed by shared functions from equivariant cell features, or be tied across incidence types. Arbitrary cell-indexed or incidence-indexed parameters are generally not permutation-equivariant.

For $y \in \mathcal{X}^{r'}$, define the incident lower-rank neighborhood $\mathcal{I}_r(y) := \{x \in \mathcal{X}^r : x \subset y\}$. The downward variants are obtained by reversing the incidence direction.

Parameterization	**Upward update for** $y \in \mathcal{X}^{r'}$	**Condition**
Incidence convolution	$\phi\left(\sum_{x \in \mathcal{I}_r(y)} H_x W\right)$	Shared W
Degree-normalized incidence	$\phi\left(\sum_{x \in \mathcal{I}_r(y)} \bar{B}_{xy} H_x W\right)$	Fixed incidence weights $\bar{B}_{xy}$
Attention-weighted incidence	$\phi\left(\sum_{x \in \mathcal{I}_r(y)} \alpha_{xy} H_x W\right)$	α_{xy} from shared equivariant scoring
Gated incidence	$\phi\left(\sum_{x \in \mathcal{I}_r(y)} g_{xy} H_x W\right)$	g_{xy} from shared equivariant gating

Table 10: Representative rank-transport parameterizations. Here H_x is the feature vector on the source cell x, W is shared across cells, and $\bar{B}$ denotes a fixed raw or normalized incidence matrix. Attention and gating preserve equivariance only when their weights are produced by shared functions of equivariant features, not by arbitrary cell-indexed parameters.

A.2 Proofs for Section 3

Proof of Proposition 3.4. The claim follows by induction over the encoder and decoder recursions. The input satisfies $E_{s_0} \in C^{s_0}(\mathcal{X}; \mathbb{R}^{d_0})$ by definition. If $E_{s_i} \in C^{s_i}(\mathcal{X}; \mathbb{R}^{d_i})$, then $T^{\uparrow}_{s_i \to s_{i+1}}(E_{s_i})$ lies in $C^{s_{i+1}}(\mathcal{X}; \mathbb{R}^{d_{i+1}})$, and the within-rank map $\Phi_{s_{i+1}}$ preserves this cochain space. Hence $E_{s_{i+1}} \in C^{s_{i+1}}(\mathcal{X}; \mathbb{R}^{d_{i+1}})$.

For the decoder, $D_{s_L} = \Omega_{s_L}(E_{s_L})$ lies in $C^{s_L}(\mathcal{X}; \mathbb{R}^{d_L})$. If $D_{s_{i+1}} \in C^{s_{i+1}}(\mathcal{X}; \mathbb{R}^{d_{i+1}})$, then $T^{\downarrow}_{s_{i+1} \to s_i}(D_{s_{i+1}})$ lies in $C^{s_i}(\mathcal{X}; \mathbb{R}^{d_i})$, and Ψ_{s_i} preserves that space. Thus $\widetilde{D}_{s_i} \in C^{s_i}(\mathcal{X}; \mathbb{R}^{d_i})$. Since E_{s_i} lies in the same cochain space, the merge map M_{s_i} is well-defined and returns $D_{s_i} \in C^{s_i}(\mathcal{X}; \mathbb{R}^{d_i})$. □

Proof of Proposition 3.5. Let primed variables denote the states obtained after reindexing cells and incidence matrices. We prove by induction that $E'_{s_i} = P_{s_i} E_{s_i}$ and $D'_{s_i} = P_{s_i} D_{s_i}$ for all ranks in the path. The encoder claim holds at s_0 by the definition of the reindexed input. If $E'_{s_i} = P_{s_i} E_{s_i}$, reindexing-equivariance of the transport gives $T'^{\uparrow}_{s_i \to s_{i+1}}(E'_{s_i}) = P_{s_{i+1}} T^{\uparrow}_{s_i \to s_{i+1}}(E_{s_i})$. Equivariance of $\Phi_{s_{i+1}}$ then gives $E'_{s_{i+1}} = P_{s_{i+1}} E_{s_{i+1}}$.

At the bottleneck, equivariance of Ω_{s_L} gives $D'_{s_L} = P_{s_L} D_{s_L}$. Assume $D'_{s_{i+1}} = P_{s_{i+1}} D_{s_{i+1}}$. Reindexing-equivariance of the downward transport and equivariance of Ψ_{s_i} imply $\widetilde{D}'_{s_i} = P_{s_i} \widetilde{D}_{s_i}$. Since the merge map is equivariant and $E'_{s_i} = P_{s_i} E_{s_i}$, we obtain $D'_{s_i} = P_{s_i} D_{s_i}$. Taking $i = 0$ proves the result. □

Proof of Proposition 3.6. The first claim follows directly from the definitions. Once the complex $\mathcal{X}$ and rank path $\mathcal{S}$ are fixed, each support size $n_{s_i} = |\mathcal{X}^{s_i}|$ is fixed, and hence so are $\rho_i = n_{s_{i+1}}/n_{s_i}$ and $\rho_{\mathrm{bot}} = n_{s_L}/n_{s_0}$.

For the capacity claim, a no-skip linear autoencoder whose output depends on the input only through the bottleneck factors through a vector space of dimension $n_{s_L} d_L$. Exact reconstruction of arbitrary inputs in $C^{s_0}(\mathcal{X}; \mathbb{R}^{d_0}) \cong \mathbb{R}^{n_{s_0} d_0}$ requires the linear encoder into the bottleneck to be injective. A linear injective map from $\mathbb{R}^{n_{s_0} d_0}$ to $\mathbb{R}^{n_{s_L} d_L}$ can exist only if $n_{s_L} d_L \geq n_{s_0} d_0$. Dividing by n_{s_L} gives $d_L \geq d_0/\rho_{\mathrm{bot}}$. □

B ADDITIONAL EXPERIMENTAL RESULTS

B.1 PARAMETER EFFICIENCY

Table 11 reports parameter counts for MNIST reconstruction. TopoU-Net uses roughly 8.7–$9.4\times$ fewer parameters than the convolutional encoder-decoder baselines in this comparison, while also achieving the lowest reported MSE.

Table 11: Parameters and MNIST reconstruction MSE.

Model	**Parameters**	**MSE**
UNet	1.05×10^5	1.63×10^{-3}
ResUNet	1.08×10^5	1.58×10^{-3}
AttentionUNet	1.07×10^5	1.90×10^{-3}
UNet++	1.14×10^5	1.53×10^{-3}
TopoU-Net	$\mathbf{1.21 \times 10^4}$	$\mathbf{9.5 \times 10^{-4}}$

B.2 IMAGE RECONSTRUCTION

Image reconstruction is a useful stress test because pixel grids are the domain where convolutional U-Nets are most natural. All inputs are resized to 28×28 grayscale. Table 12 reports test MSE.

TopoU-Net achieves the lowest MSE on four of five datasets and remains close to the best baseline on FashionMNIST. This is consistent with the support-ratio analysis: for the 28×28 grid construction in Table 7, the bottleneck support ratio is $\rho_{\text{bot}} = 0.93$, so the bottleneck retains nearly the same number of cells as the input rank.

Table 12: Test MSE on image reconstruction benchmarks.

Model	**CIFAR10**	**EMNIST**	**FashionMNIST**	**MNIST**	**SVHN**
AttentionUNet	2.90×10^{-3}	1.72×10^{-3}	5.96×10^{-3}	1.90×10^{-3}	8.04×10^{-4}
ResUNet	1.77×10^{-3}	1.42×10^{-3}	$\mathbf{3.95 \times 10^{-3}}$	1.58×10^{-3}	4.82×10^{-4}
UNet	2.46×10^{-3}	1.46×10^{-3}	5.34×10^{-3}	1.63×10^{-3}	7.43×10^{-4}
UNetPP	2.06×10^{-3}	1.53×10^{-3}	6.07×10^{-3}	1.53×10^{-3}	5.65×10^{-4}
TopoU-Net	$\mathbf{1.15 \times 10^{-3}}$	$\mathbf{1.01 \times 10^{-3}}$	4.27×10^{-3}	$\mathbf{9.46 \times 10^{-4}}$	$\mathbf{2.63 \times 10^{-4}}$

B.3 FULL COMPRESSION PROFILE ANALYSIS

Table 13 extends Table 7 to all node-classification datasets. The Path column reports the full encoder–decoder traversal, while the support ratios are computed along the encoder branch from the input rank to the bottleneck rank.

Table 13: Complete compression profiles across all node classification datasets.

Dataset	n_0	n_1	n_2	n_3	ρ_0	ρ_1	ρ_2	ρ_{bot}	**Path**
Heterophilic									
Texas	183	325	52	1	1.78	0.16	0.02	0.006	$0\to1\to2\to3\to2\to1\to0$
Wisconsin	251	515	89	1	2.05	0.17	0.01	0.004	$0\to1\to2\to3\to2\to1\to0$
Cornell	183	298	47	1	1.63	0.16	0.02	0.005	$0\to1\to2\to3\to2\to1\to0$
Actor	7600	33544	1188	1	4.41	0.04	0.001	0.0001	$0\to1\to2\to3\to2\to1\to0$
Chameleon	2277	36101	1507	1	15.9	0.04	0.001	0.0004	$0\to1\to2\to3\to2\to1\to0$
Squirrel	5201	217073	2346	1	41.7	0.01	0.0004	0.0002	$0\to1\to2\to3\to2\to1\to0$
Homophilic									
Computers	13752	245861	8943	—	17.9	0.04	—	0.65	$0 \to 1 \to 2 \to 1 \to 0$
Photo	7650	119081	4521	—	15.6	0.04	—	0.59	$0 \to 1 \to 2 \to 1 \to 0$
WikiCS	11701	216123	9834	—	18.5	0.05	—	0.84	$0 \to 1 \to 2 \to 1 \to 0$

B.3.1 MATCHED PARAMETER/DEPTH ABLATION

We compare $0 \to 1 \to 0$, $0 \to 1 \to 2 \to 1 \to 0$, and $0 \to 1 \to 2 \to 3 \to 2 \to 1 \to 0$

All models use the same number of residual processing blocks; hidden widths are adjusted so parameter counts are comparable. Results are mean ($\pm$) standard deviation over 10 seeds.

The capacity explanation is directly contradicted on Roman-empire under compression. At support 0.50 and 0.25, the edge-only model has 137,707 parameters—slightly more than the rank-2 model with 137,004—yet falls to 71.92% and 68.56%, while every higher-order model remains above 82.6%. An 11–14 point gap in favor of the model with the smaller parameter budget cannot be attributed to width or depth.

On Minesweeper, all paths are comparable at full support, but the edge-only model loses 3.55 points under compression, whereas both higher-order paths remain near 84.1%. Amazon-ratings shows smaller but consistent numerical gains for rank 3. The robust conclusion is that higher-order transport preserves predictive information under severe bottleneck compression, and on Roman-empire also improves peak accuracy.

Table 14: Rank-path performance under different support fractions.

Dataset	Support	Rank 0-1-0	Rank 0-1-2-1-0	Rank-3 original	Rank-3 components
Roman-empire	1.00	82.45 ± 0.36	84.22 ± 0.61	$\mathbf{84.32 \pm 0.34}$	84.11 ± 0.53
Roman-empire	0.50	71.92 ± 0.66	83.26 ± 0.30	84.32 ± 0.34	$\mathbf{84.38 \pm 0.41}$
Roman-empire	0.25	68.56 ± 0.60	82.69 ± 0.65	$\mathbf{84.32 \pm 0.34}$	84.11 ± 0.48
Minesweeper	1.00	84.21 ± 0.62	84.12 ± 0.71	84.19 ± 0.57	$\mathbf{84.29 \pm 0.71}$
Minesweeper	0.50	81.40 ± 0.72	$\mathbf{84.19 \pm 0.79}$	84.02 ± 0.72	84.11 ± 0.75
Minesweeper	0.25	80.66 ± 0.82	$\mathbf{84.16 \pm 0.69}$	84.09 ± 0.66	84.05 ± 0.68
Amazon-ratings	1.00	38.66 ± 0.94	38.69 ± 1.30	38.95 ± 1.09	$\mathbf{39.12 \pm 0.87}$
Amazon-ratings	0.50	38.80 ± 0.59	38.89 ± 1.08	38.84 ± 0.84	$\mathbf{38.97 \pm 1.07}$
Amazon-ratings	0.25	38.77 ± 0.74	38.81 ± 1.41	$\mathbf{39.31 \pm 0.87}$	39.04 ± 0.97

B.4 SCALABILITY

We quantify the number of cells at each constructed rank, retained and total triangles, preprocessing time, training time, inference latency, and CPU/GPU memory. The measurements identify higher-order cell construction as the principal scalability cost. Preprocessing takes only 0.03–0.06 s on the WebKB datasets, 42.70 s on Actor, 160.92 s on Chameleon, and 1,054.54 s on Squirrel, which contains approximately 9.6 million candidate triangles.

To control this growth, we retain at most 30,000 triangles on Chameleon and Squirrel. This bounded cell budget makes the subsequent model practical: training ranges from 4.5 to 28.6 s, inference from 3.3 to 41.4 ms, and GPU memory from approximately 56 to 815 MB. The experiments therefore show a clear computational separation: higher-order construction is the dominant one-time cost, while training and inference remain efficient after the complex is built.

Table 15: Scalability statistics for TopoU-Net across datasets.

Dataset	N_0	N_1	$N_2^{\text{kept}}/N_2^{\text{total}}$	Preprocess (s)	Train (s)	Infer. (ms)	CPU (MB)	GPU (MB)
Actor	7,600	26,659	7,121/7,121	42.70	5.52	5.52	1,788.6	179.5
Chameleon	2,277	31,371	30,000/343,066	160.92	7.20	8.21	2,040.3	254.2
Squirrel	5,201	198,353	30,000/9,595,609	1,054.54	28.60	41.41	2,752.8	815.5
Cornell	183	277	59/59	0.03	4.5	3.36	2,077.8	56.0
Wisconsin	251	450	118/118	0.06	4.6	3.28	2,075.7	56.9
Texas	183	279	67/67	0.04	4.6	3.30	2,074.8	56.0

B.5 COMPUTATIONAL COMPLEXITY

Let n_r denote the number of retained cells of rank r, F the input feature dimension, H the hidden dimension, and K the number of classes. All rank transitions are implemented using sparse incidence

operators. here nnz is number if non zero entries and For the complexes considered here,

$$\mathrm{nnz}(S_{01}) = 2n_1, \qquad \mathrm{nnz}(S_{12}) = 3n_2,$$

and, for the connected-component rank-3 construction,

$$\mathrm{nnz}(S_{23}) = n_2.$$

Ignoring architecture-dependent constant factors, the forward-pass complexities of the three rank paths are

$$T_{010} = O\big(n_0FH + (n_0+n_1)H^2 + n_1H + n_0HK\big)\,,$$
$$T_{01210} = O\big(n_0FH + (n_0+n_1+n_2)H^2 + (n_1+n_2)H + n_0HK\big)\,,$$

and

$$T_{0123210} = O\big(n_0FH + (n_0+n_1+n_2+n_3)H^2 + (n_1+n_2)H + n_0HK\big)\,.$$

Hence, for fixed feature dimensions,

$$T_{010} = O(n_0+n_1), \qquad T_{01210} = O(n_0+n_1+n_2),$$

and

$$T_{0123210} = O(n_0+n_1+n_2+n_3).$$

Thus, after construction of the cell complex, TopoU-Net scales linearly with the total number of retained cells and does not require dense all-to-all interactions.

The corresponding memory complexity is linear in the stored cell features and sparse incidence operators,

$$O\left(\sum_r n_rH + \sum_r \mathrm{nnz}(S_{r,r+1})\right),$$

up to model-parameter storage.

Higher-order paths additionally incur preprocessing cost from cell construction. In particular, triangle enumeration requires

$$O\left(\sum_{(u,v)\in E} \min(d_u, d_v)\right)$$

using adjacency-set intersections, while connected-component construction requires $O(n_0 + n_1)$ time.

B.6 Hyperparameter settings

All TopoU-Net models use hidden dimension $d_h = 64$ for encoder/decoder blocks, learning rate $\eta = 0.001$ with weight decay 10^{-5}, and dropout rate 0.3. Graph classification uses batch size 32, node classification uses full-batch training. For heterophilic graphs, the rank-3 cell is a single global cell incident to all rank-2 triangle cells. Its feature is initialized by mean pooling over the triangle features. Baseline models (GCN, GAT, GIN, GraphSAGE, H2GCN, MixHop, DiffPool, Graph U-Net) use hyperparameters from original implementations, adapted to match our train/validation/test splits and normalization.