Architectures of Topological Deep Learning: A Survey on Topological Neural Networks

Mathilde Papillon*[1] Sophia Sanborn[2] Mustafa Hajij[3] Nina Miolane[2]

[1]Department of Physics
[2]Department of Electrical and Computer Engineering
University of California, Santa Barbara

[3] Department of Data Science
University of San Francisco

*Corresponding Author: papillon@ucsb.edu

Abstract

The natural world is full of complex systems characterized by intricate relations between their components: from social interactions between individuals in a social network to electrostatic interactions between atoms in a protein. Topological Deep Learning (TDL) provides a comprehensive framework to process and extract knowledge from data associated with these systems, such as predicting the social community to which an individual belongs or predicting whether a protein can be a reasonable target for drug development. TDL has demonstrated theoretical and practical advantages that hold the promise of breaking ground in the applied sciences and beyond. However, the rapid growth of the TDL literature has also led to a lack of unification in notation and language across Topological Neural Network (TNN) architectures. This presents a real obstacle for building upon existing works and for deploying TNNs to new real-world problems. To address this issue, we provide an accessible introduction to TDL, and compare the recently published TNNs using a unified mathematical and graphical notation. Through an intuitive and critical review of the emerging field of TDL, we extract valuable insights into current challenges and exciting opportunities for future development.

Keywords: deep learning, topology, message passing, graph, hypergraph, simplicial complex, cellular complex, combinatorial complex

1 Introduction

Many natural systems as diverse as social networks (Knoke and Yang, 2019) and proteins (Jha et al., 2022) are characterized by *relational structure*. This is the structure of interactions between components in the system, such as social interactions between individuals or electrostatic interactions between atoms. In Geometric Deep Learning (Bronstein et al., 2021), Graph Neural Networks (GNNs) (Zhou et al., 2020) have demonstrated remarkable achievements in processing relational data using graphs—mathematical objects commonly used to encode *pairwise relations*.

However, the pairwise structure of graphs is limiting. Social interactions can involve more than two individuals, and electrostatic interactions more than two atoms. Topological Deep Learning (TDL) (Hajij et al., 2023; Bodnar, 2022) leverages more general abstractions to process data with higher-order relational structure. The theoretical guarantees (Bodnar et al., 2021a,b; Huang and Yang, 2021) of its models, Topological Neural Networks (TNNs), lead to state-of-the-art performance on many machine learning tasks (Dong et al., 2020; Hajij et al., 2022a; Barbarossa and Sardellitti, 2020; Chen et al., 2022)—and reveal high potential for the applied sciences and beyond.

However, the abstraction and fragmentation of mathematical notation across the TDL literature significantly limits the field's accessibility, while complicating model comparison and obscuring opportunities for innovation. To address this, we present an intuitive and systematic comparison of published TNN architectures. We contribute:

- **A pedagogical resource** accessible to newcomers interested in applying TNNs to real-world problems.
- **A comprehensive and critical review** of TNNs, their implementations and practical applications, with equations rewritten in our notation available at github.com/awesome-tnns.
- **A summary of open research questions**, challenges, and opportunities for innovation.

By establishing a common and accessible language in the field, we hope to provide newcomers and experienced practitioners alike with a solid foundation for cutting-edge research in TDL.

Other literature reviews at the intersection of topology and machine learning have focused on data representation (Torres et al., 2021) and physics-inspired models (Battiston et al., 2021). TDL is part of a broader field defined as topological machine learning, first surveyed in Hensel et al. (2021). This field encompasses additional methods such as topological data analysis for machine learning, where topological features computed from techniques such as persistent homology enhance the capabilities of neural networks.

2 Topological Neural Networks

Topological Neural Networks (TNNs) are deep learning architectures that extract knowledge from data associated with topologically rich systems such as protein structures, city traffic maps, or citation networks. A TNN, like a GNN, is comprised of stacked layers that transform data into a series of features (Figure 1). Each layer leverages the fundamental concepts of *data and computational domains*, *neighborhoods*, and *message passing*—presented in this section.

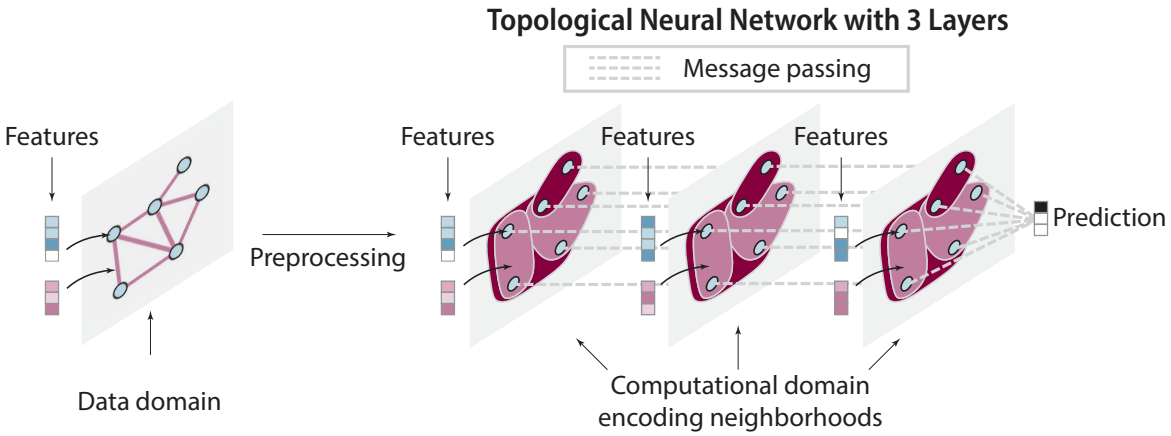

Figure 1: Topological Neural Network: Data associated with a complex system are features defined on a *data domain*, which is preprocessed into a *computational domain* that encodes interactions between the system's components with *neighborhoods*. The TNN's layers use *message passing* to successively update features and yield an output, e.g. a categorical label in classification or a quantitative value in regression. The output represents new knowledge extracted from the input data.

2.1 Domains

In Topological Deep Learning (TDL), data are features defined on discrete domains (Hajij et al., 2023; Bodnar, 2022). Traditional examples of discrete domains include sets and graphs (Figure 2, Left). A **set** is a collection of points called *nodes* without any additional structure. A **graph** is a set with *edges* that encode *pairwise relations* between nodes, representing either geometric proximity or more abstract relationships. For example, a graph may represent a protein, with nodes encoding its atoms and edges encoding the pairwise bonds between them. Alternatively, a graph may represent a social network, where nodes represent individuals and edges denote social relationships. The domains of TDL generalize the pairwise relations of graphs to *part-whole* and *set-types* relations that permit the representation of more complex relational structure (Figure 2, Right) (Hajij et al., 2023). Here, we describe the key attributes of each domain and highlight their suitability for different data types. We refer the reader to Torres et al. (2021) and Hajij et al. (2023) for more extensive discussions.

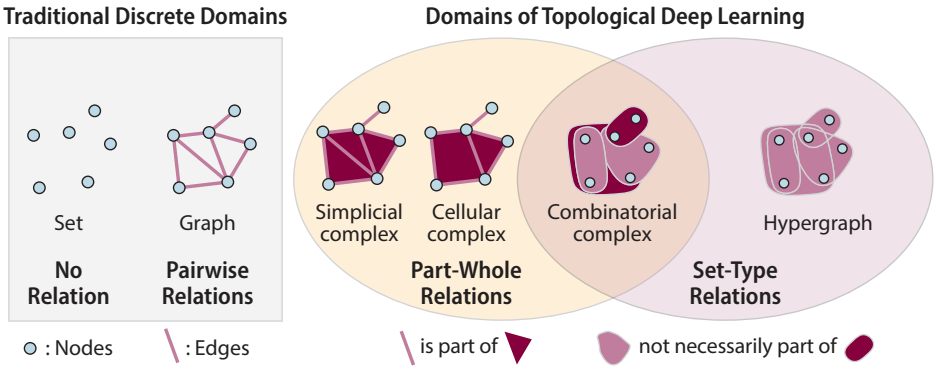

Figure 2: Domains: Nodes in blue, (hyper)edges in pink, and faces in dark red. Figure adapted from Hajij et al. (2022a).

Beyond Graphs: The Domains of Topological Deep Learning

Set + Pairwise Relations

Graph: A set of points (*nodes*) connected with *edges* that denote pairwise relationships.

Set + Part-Whole Relations

Simplicial Complex (SC): A generalization of a graph in which three edges can form a *triangular face*, four triangles can form a *tetrahedral volume*, and so on. Edges only connect pairs of nodes.

Cellular Complex (CC): A generalization of an SC in which *faces, volumes, etc* are not restricted to be triangles or tetrahedrons but may instead take any shape. Still, edges only connect pairs of nodes.

Set + Set-Type Relations

Hypergraph (HG): A generalization of a graph, in which higher-order edges called *hyperedges* can connect arbitrary *sets* of two or more nodes.

Set + Part-Whole and Set-Type Relations

Combinatorial Complex (CCC): A structure that combines features of HGs and CCs. Like an HG, edges may connect any number of nodes. Like a CC, cells can be combined to form higher-ranked structures.

Simplicial complexes (SCs) generalize graphs to incorporate hierarchical *part-whole relations* through the multi-scale construction of *cells*. Nodes are *rank 0* cells that can be combined to form edges (*rank 1* cells). Edges are, in turn, combined to form faces (*rank 2* cells), which are combined to form volumes (*rank 3* cells), and so on. As such, an SC's faces must be triangles, volumes must be tetrahedrons, and so forth. SCs are commonly used to encode discrete representations of 3D geometric surfaces represented with triangular meshes (Figure 3). They may also be used to represent more abstract relations; however, there is a risk of introducing spurious connections if the strict geometric constraints of an SC are not respected by the data—a point we elaborate on in Section 2.1.2.

Cellular complexes (CCs) generalize SCs such that cells are not limited to simplexes: faces can involve more than three nodes, volumes more than four faces, and so on. This flexibility endows CCs with greater expressivity than SCs (Bodnar et al., 2021b). A practitioner should consider employing this domain when studying a system that features part-whole interactions between more than three nodes, such as a molecule with benzene rings (Figure 3).

Hypergraphs (HGs) extend graphs in that their edges, called *hyperedges*, can connect more than two nodes. Connections in HGs represent *set-type relationships*, in which participation in an interaction is not implied by any other relation in the system. This makes HGs an ideal choice for data with abstract and arbitrarily large interactions of equal importance, such as semantic text and citation networks. Protein interaction networks (Figure 3) also exhibit this property: an interaction between

proteins requires a precise set of molecules—no more and no less. The interaction of Proteins A, B, and C does not imply an interaction between A and B on their own.

Combinatorial complexes (CCCs) generalize CCs and HGs to incorporate both *part-whole* and *set-type* relationships (Hajij et al., 2022a, 2023). The benefit of this can be observed in the example of molecular representation. The strict geometric constraints of simplicial and cellular complexes are too rigid for capturing much of hierarchical structure observed in molecules. By contrast, the flexible but hierarchically ranked hyperedges of a combinatorial complex can capture the full richness of molecular structure, as depicted in Figure 3. This is the most recent and most general topological domain, introduced in 2022 by Hajij et al. (2022a) and further theoretically established in Hajij et al. (2023).

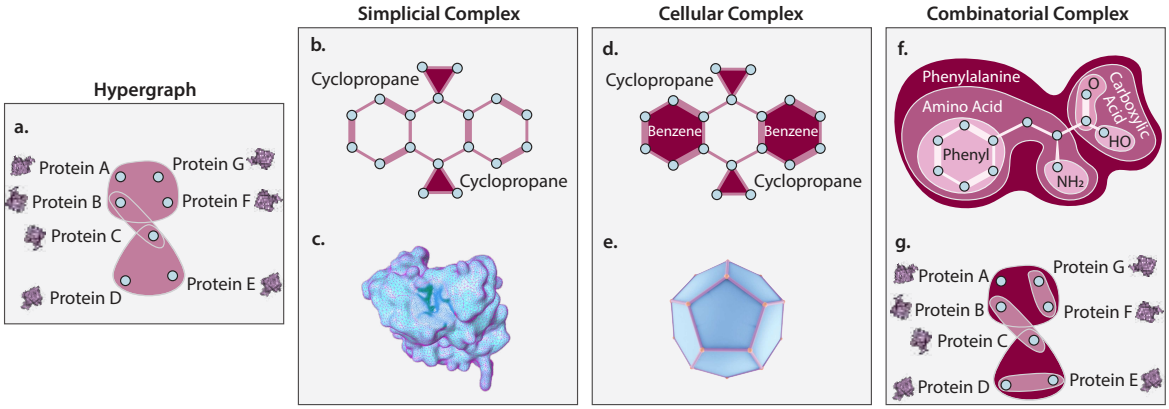

Figure 3: Examples of Data on Topological Domains. (a) Higher-order interactions in protein networks. (b) Limited molecular representation: rings can only contain three atoms. (c) Triangular mesh of a protein surface. (d) More flexible molecular representation, permitting the representation of any ring-shaped functional group. (e) Flexible mesh which includes arbitrarily shaped faces. (f) Fully flexible molecular representation, permitting the representation of the complex nested hierarchical structure characteristic of molecules and other natural systems. (g) Hierarchical higher-order interactions in protein networks.

2.1.1 Terminology

Across discrete domains, we use the term *cell* to denote any node or relation between nodes such as (hyper)edges, faces, or volumes. Cells possess two attributes: *size*—the number of cells it contains—and *rank*—where nodes are said to have rank 0, edges and hyperedges rank 1, faces rank 2, and so on. The part-whole relationships of simplicial and cellular complexes impose a relationship between the *rank* of a cell and its *size*: cells of rank r contains exactly (resp. at least) $r+1$ cells of rank $r-1$: faces ($r=2$) contain exactly (resp. at least) three edges ($r-1=1$). By contrast, hypergraph cells do not encode part-whole relations and hyperedges may have any size. However, hypergraph cells are limited to ranks 0 and 1. A combinatorial complex is unrestricted in both rank and size: nodes have rank 0 and cells of any size >1 can have any rank.

There is an important distinction between the inherent domain of the data (the *data domain*) and the domain in which the data will be processed within a TNN: the *computational domain*. Data defined on a graph, for example, may be "lifted" (Figure 4) to an alternative domain through a pre-processing stage (Figure 1). For instance, a protein originally given as the graph of its atoms (nodes) and covalent bounds (edges) may be lifted into a CC computational domain that explicitly represents its rings (faces). In this review, *domain* refers to the computational domain. Additionally, the computational domain may be *dynamic*, changing from layer to layer in a TNN.

Dynamic Domains

Static vs. Dynamic: In a TNN, a *static* domain is identical for each layer. For example, all three layers in Figure 1 operate on the same CCC, only features evolve across layers. A *dynamic* domain changes from layer to layer. Nodes can be added or removed, edges can be rewired, and so on.

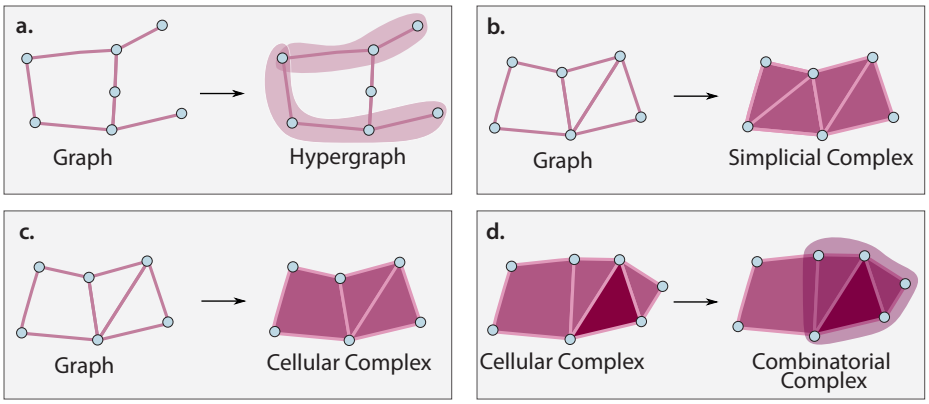

Figure 4: Lifting Topological Domains. (a) A graph is "lifted" to a hypergraph by adding hyperedges that connect groups of nodes. (b) In lifting a graph to a simplicial complex, a pairwise edge is required to bound triangular faces. (c) A graph can be converted to a cellular complex by adding faces of any shape. (d) Hyperedges can be added to a cellular complex to lift the structure to a combinatorial complex. Figure adopted from Hajij et al. (2023).

2.1.2 Limitations

An important limitation of both SCs and CCs is that faces (and analogous higher-order structures) can only form rings; the nodes on the boundary of the face must be connected in pairs. In many cases, this requirement is too stringent and can introduce artificial connections to the domain (Yang et al., 2022a). For instance, lifting a citation network into an SC necessarily requires that any set of three co-authors having written a paper together (A, B, C) are also pairwise connected (A and B, A and C, B and C), even if no paper was ever exclusively authored by authors A and B, authors A and C, or authors B and C. Yang et al. (2022a) propose a "relaxed" definition of the SC that remedies this. They show how training a TNN on such a modified domain increases performance. We note that even with artificial connections, SCs and CCs allow TNNs to leverage richer topological structure and avoid computational problems faced by GNNs (Rusch et al., 2023). We further note that any topological domain is mathematically equivalent to a (possibly larger) graph (Veličković, 2022). We choose to express domains in their form above in order to provide better intuition to newcomers and reflect the widely adopted approaches in the literature.

2.1.3 Features on a Domain

Consider a domain, denoted $\mathcal{X}$, encoding relationships between components of a system. Data on the domain are represented as features supported on the domain's cells. Typically, features are vectors in $\mathbb{R}^d$ that encode attributes of each cell. For example, features may encode the atom (node), bond (edge), and functional group (face) types in a molecule. A feature associated with the interaction between a set of drugs (hyperedge) could indicate the probability of adverse reaction.

We denote with $\mathbf{h}_x^{t,(r)}$ a feature supported on the cell $x \in \mathcal{X}$ at layer t of the TNN, with r indicating the rank of x (Figure 5). The domain is decomposed into ranks, with $X^{(r)}$, or *r-skeleton*, referring to all cells of rank r. Features can be categorical or quantitative. If the feature dimension varies across skeletons, the domain is *heterogeneous*.

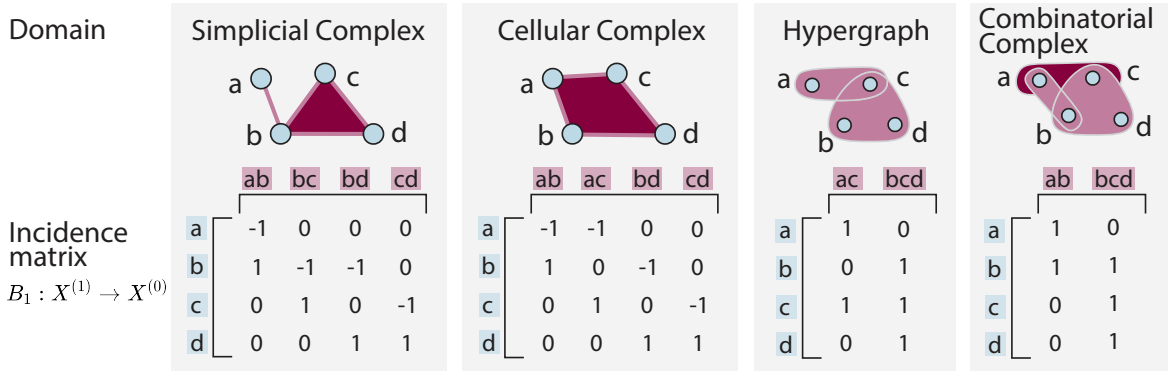

Figure 6: Incidence Matrices. Examples of an SC, a CC, an HG, and a CCC with their corresponding boundary matrices B_1 which map from 1-cells to 0-cells. The SC and CC maps are signed to encode edge orientation: the node appearing first in the arbitrary ordering (a,b,c,d) is always assigned -1.

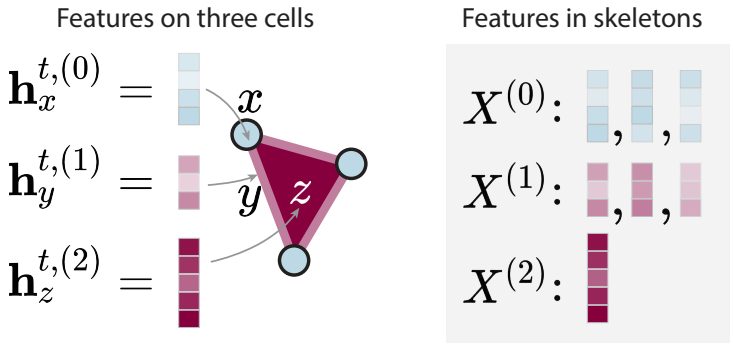

Figure 5: Features on a Domain. Left: Features onto three cells—x, y, and z. Right: Skeletons for the entire complex: $X^{(0)}$ contains node features, $X^{(1)}$ contains edge features, and so on.

Heterogeneous Domains

Homogeneity vs. Heterogeneity: In a *heterogeneous* domain, the dimension d_r of a feature $\mathbf{h}_x^{(r)}$ depends on the rank r of the cell x supporting it. A *homogeneous* domain uses the same dimensionality d for all ranks.

The features assigned to each cell may come directly from the data or be hand-designed by the practitioner. Alternatively, features can be assigned in a pre-processing stage using *embedding* methods, which compute cell feature vectors that encode the local structure of the space. For graphs, methods such as DeepWalk (Perozzi et al., 2014) and Node2Vec (Grover and Leskovec, 2016) are commonly used to embed nodes. Recent works have generalized these approaches to topological domains: Hyperedge2Vec (Sharma et al., 2018) and Deep Hyperedge (Payne, 2019) for hypergraphs, Simplex2Vec (Billings et al., 2019) and k-Simplex2Vec (Hacker, 2020) for simplicial complexes, and Cell2Vec (Hajij et al., 2020) for cellular complexes.

2.2 Neighborhood Structure

A TNN successively updates cell features throughout its layers by using a notion of *nearness* between cells: the *neighborhood structure* (Figure 1). Neighborhood structures are defined by **boundary relations**, which describe how cells of different ranks relate to each other. A cell y of rank r is said to be on the **boundary** of cell x of rank R if it is connected to x and rank $r < R$. This relation is expressed as $y \prec x$. For example, a node connected to an edge is said to be on the boundary of that edge.

Boundary relations are encoded in **incidence matrices**. Specifically, we denote with B_r the matrix that records which (regular) cells of rank $r-1$ bound which cells of rank r (Figure 6). Formally, B_r is a matrix of size $n_{r-1} \times n_r$, with n_r denoting the number of cells of rank $r \geq 1$, defined:

$$(B_r)_{i,j} = \begin{cases} \pm 1 & x_i^{(r-1)} \prec x_j^{(r)} \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where $x_i^{(r-1)}$, $x_j^{(r)}$ are two cells of ranks $r-1$ and r respectively. The ± 1 sign encodes a notion of orientation required for SCs and CCs (Aschbacher, 1996; Klette, 2000), and is always $+1$ for HGs and CCCs.

Incidence matrices can be used to encode the four most common neighborhood structures used in the literature, which we define in the text box below. Here, $L_{\uparrow,0}$ denotes the typical graph Laplacian. Its higher order generalization, the r-Hodge Laplacian, is $H_r = L_{\downarrow,r} + L_{\uparrow,r}$ (Barbarossa and Sardellitti, 2020; Schaub et al., 2021). $D_r \in \mathbb{N}^{n_r \times n_r}$ denotes the degree matrix, a diagonal matrix representing the number of connections of r-cells with $(r+1)$-cells.

Neighborhood Structures

Boundary Adjacent Neighborhood $\mathcal{B}(y) = \{x \mid x \prec y\}$:
The set of y-connected x cells of next lower rank. The neighborhood is specified with the *boundary matrix* B_r. *Example: The set of nodes x connected to edge y.*

Co-Boundary Adjacent Neighborhood $\mathcal{C}(y) = \{x \mid y \prec x\}$:
The set of y-connected x cells of next higher rank. The neighborhood is specified with the *co-boundary matrix* B_r^T. *Example: The set of edges x connected to node y.*

Lower Adjacent Neighborhood $\mathcal{L}_{\downarrow}(y) = \{x \mid \exists z \text{ s.t. } z \prec y \text{ and } z \prec x\}$:
The set of x cells that share a boundary z with y. The neighborhood is specified with either the *lower Laplacian matrix* $L_{\downarrow,r} = B_r B_r^T$ or the *lower adjacency matrix* $A_{\downarrow,r} = D_r - L_{\downarrow,r}$. *Example: the set of edges x that connect to any of the nodes z that touch edge y.*

Upper Adjacent Neighbors $\mathcal{L}_{\uparrow}(y) = \{x \mid \exists z \text{ s.t. } y \prec z \text{ and } x \prec z\}$:
The set of x cells that share a co-boundary z with y. The neighborhood is specified with either the *upper Laplacian matrix* $L_{\uparrow,r} = B_{r+1}^T B_{r+1}$ or the *upper adjacency matrix* $A_{\uparrow,r} = D_r - L_{\uparrow,r}$.
Example: The set of nodes x that touch any of the edges z that touch node y.

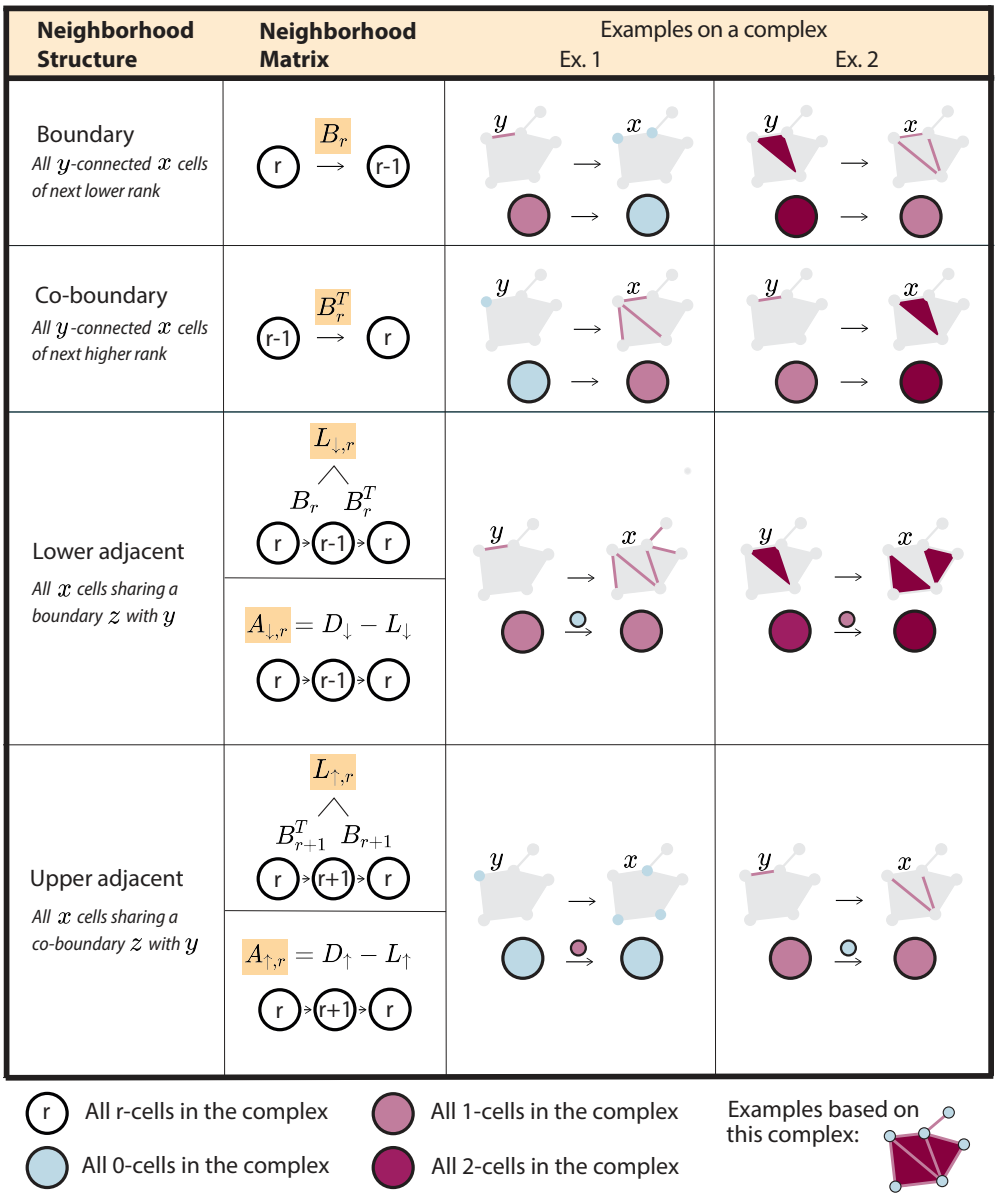

Figure 7: Neighborhood Structures: their neighborhood matrices and illustrations for a cell x in the neighborhood of a cell y.

2.3 Message Passing

Message passing defines the computation performed by a single layer t of the TNN. During message passing, each cell's feature $\mathbf{h}_{\mathbf{x}}^{\mathbf{t},(\mathbf{r})}$ is updated to incorporate: (1) the features associated with cells in its neighborhood and (2) the layer's learnable parameters denoted Θ^t. The term "message passing" reflects that a signal is "traveling" through the network, passing between cells on paths laid out by the neighborhood structure. The output $\mathbf{h}^{t+1}$ of layer t becomes the input to layer $t+1$. In this way, deeper layers incorporate information from more distant cells, as information diffuses through the network.

2.3.1 The Steps of Message Passing

We decompose message passing into four steps, adopted from the framework of Hajij et al. (2023). Each step is represented with a different color—*red*, *orange*, *green*, or *blue*—illustrated in Figure 8.

The Steps of Message Passing

1. Message: First, a message $m_{y\to x}^{(r'\to r)}$ travels from a r'-cell y to a r-cell x through a neighborhood k of x denoted $\mathcal{N}_k(x)$:

$$m_{y\to x}^{(r'\to r)} = M_{\mathcal{N}_k}\left(\mathbf{h}_x^{t,(r)}, \mathbf{h}_y^{t,(r')}, \Theta^t\right). \tag{2}$$

via the function $M_{\mathcal{N}_k}$ depicted in red in Figure 8. Here, $\mathbf{h}_x^{t,(r)}$ and $\mathbf{h}_y^{t,(r')}$ are features of dimension d_r and $d_{r'}$ on cells y and x respectively, and Θ^t are learnable parameters. In the simplest case, this step looks like a neighborhood matrix M propagating a feature $\mathbf{h}_y^{t,(r')}$ on r'-cell y to r-cell x as:

$$m_{y\to x}^{(r'\to r)} = M_{xy} \cdot \mathbf{h}_y^{t,(r')} \cdot \Theta^t, \tag{3}$$

where M_{xy} is the scalar entry of matrix M at the row corresponding to cell x and column corresponding to cell y and $m_{y\to x}^{(r'\to r)}$ and Θ is a $d_{r'} \times d_r$ matrix. If y is not in the neighborhood structure of x, then M_{xy} will be 0, and x cannot receive any message from y.

2. Within-Neighborhood Aggregation: Next, messages are aggregated across all cells y belonging to the neighborhood $\mathcal{N}_k(x)$:

$$m_x^{(r'\to r)} = AGG_{y\in\mathcal{N}_k(x)} m_{y\to x}^{(r'\to r)}, \tag{4}$$

resulting in the *within-neighborhood aggregated message* $m_x^{(r'\to r)}$. Here, AGG is an aggregation function, depicted in orange in Figure 8, analogous to pooling in standard convolutional networks.

3. Between-Neighborhood Aggregation: Then, messages are aggregated across neighborhoods in a neighborhood set $\mathcal{N}$:

$$m_x^{(r)} = AGG_{\mathcal{N}_k\in\mathcal{N}} m_x^{(r'\to r)}, \tag{5}$$

where AGG is a (potentially different) aggregation function depicted in green in Figure 8, and $m_x^{(r)}$ is the message received by cell x that triggers the update of its feature.

4. Update: Finally, the feature on cell x is updated via a function U depicted in blue in Figure 8, which may depend on the previous feature $\mathbf{h}_x^{t,(r)}$ on cell x:

$$\mathbf{h}_x^{t+1,(r)} = U\left(\mathbf{h}_x^{t,(r)}, m_x^{(r)}\right), \tag{6}$$

The result $\mathbf{h}_x^{t+1,(r)}$ is the updated feature on cell x that is input to layer $t+1$.

In this review, we decompose the structure of TNN architectures proposed in the literature into these four message passing steps—a unified notational framework adopted from Hajij et al. (2023) that allows us to contrast existing approaches. Many architectures repeat steps and/or modify their order. We note that this conceptualization of message passing as a local, cell-specific operation is called the *spatial approach* (Gilmer et al., 2017). In GNNs and TNNs alike, message passing can alternatively be expressed in its dual *spectral* form, using global Fourier analysis over the domain. For this review, we choose to write all equations in spatial form for intuitiveness and generality (Bodnar et al., 2021a; Hajij et al., 2022a; Heydari and Livi, 2022).

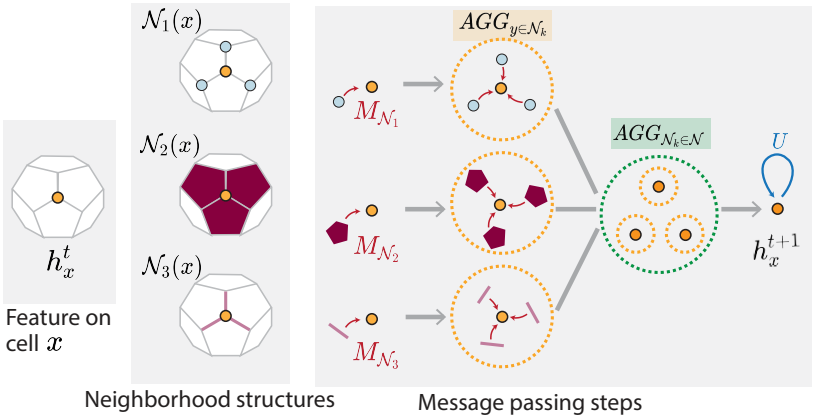

Figure 8: Message passing steps: 1: Message (red), 2: Within-neighborhood aggregation (orange), 3: Between-neighborhood aggregation (green), 4: Update (blue). The scheme updates a feature $\mathbf{h}_x^{t,(r)}$ on a r-cell x at layer t (left column) into a new feature $\mathbf{h}_x^{t+1,(r)}$ on that same cell at the next layer $t+1$ (right column). Here, the scheme uses four neighborhood structures $\mathcal{N}_k$ for $k \in \{1,2,3,4\}$ (middle column). Figure adapted from (Hajij et al., 2023).

2.3.2 Tensor Diagrams

We visually represent message passing schemes with an adapted version of the *tensor diagram* introduced in Hajij et al. (2022a) and further developed in Hajij et al. (2023). A tensor diagram provides a graphical representation of a TNN architecture. Figure 8 explains the recipe for constructing a tensor diagram from message passing steps.

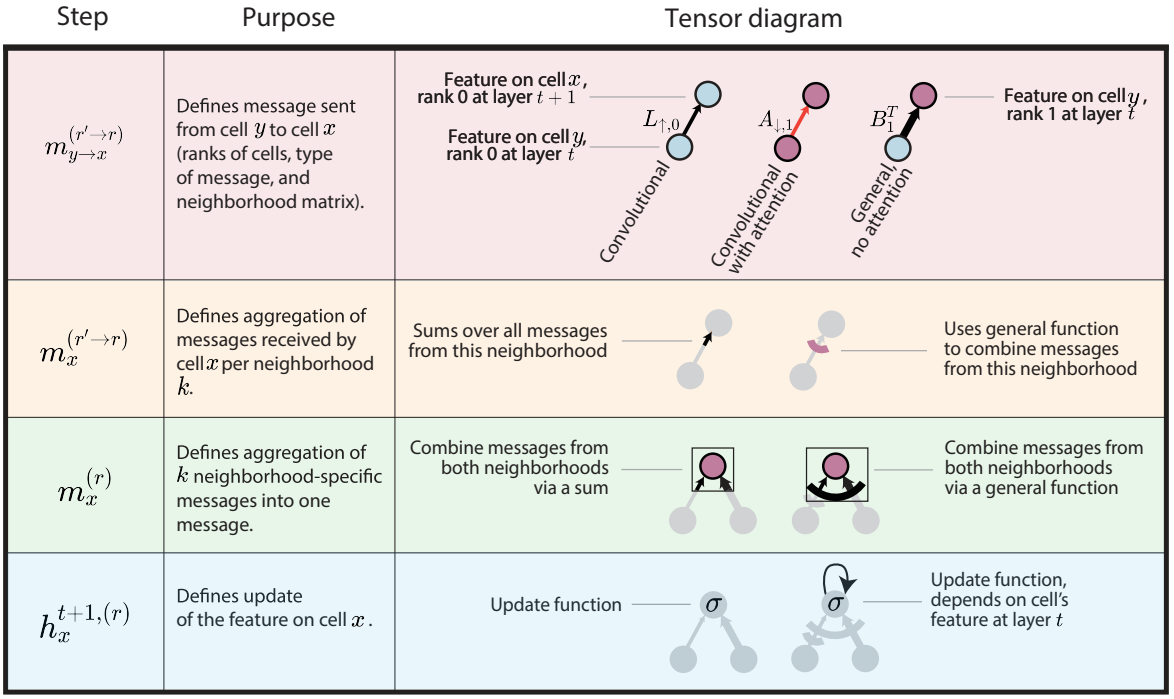

Figure 9: Tensor Diagrams: a graphical notation for the four steps of a message passing scheme. A diagram depicts how a feature on cell y at layer t, $\mathbf{h}_y^{(t)}$, becomes a feature on cell x at layer $t+1$, $\mathbf{h}_x^{(t+1)}$.

2.3.3 Types of Message Passing Functions

The message passing function $M_{\mathcal{N}_k}$ employed in Step 1 is defined by the practitioner. There are three kinds of functions commonly used in the literature, as outlined in Figure 10 (Bronstein, 2022). The variety used determines how layer parameters weight each incoming message from cell y to cell x. The *standard convolutional* case multiplies each message by some learned scalar. The *attentional convolutional* case weights this multiplication depending on the features of the cells involved. The

general case implements a potentially non-linear function that may or may not incorporate attention. Some schemes also make use of fixed, non-learned weights to assign different levels of importance to higher-order cells. Figure 10 illustrates each type with tensor diagrams.

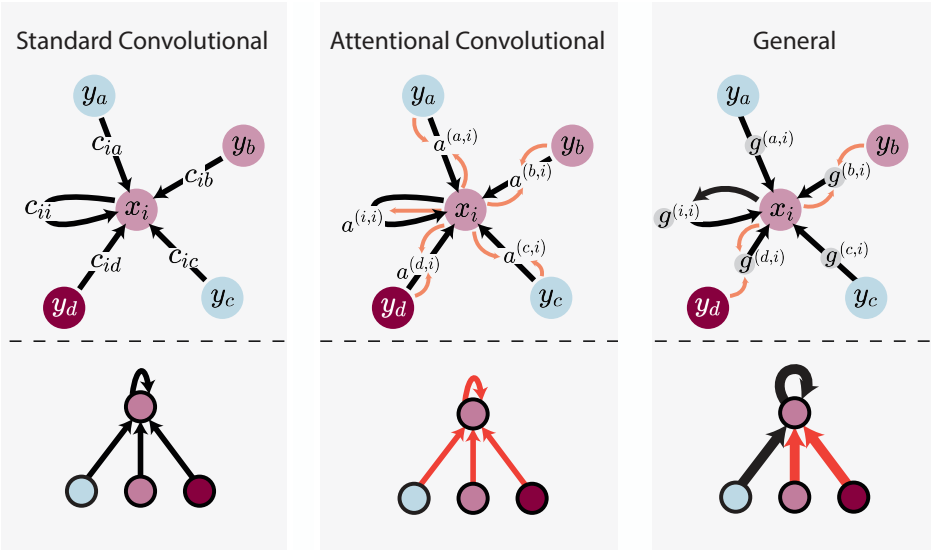

Figure 10: Types of Message Passing Functions. In each case, a cell x_i (an edge) receives information from its various neighbors, cells y_j (two nodes, an edge, and a face). The message received by cell x_i from cell y_j is determined by a specific function $c(x_i, y_j)$, $a(x_i, y_j)$, or $g(x_i, y_j)$. Top: Each neighborhood cell y_j sends a message to cell x_i. (Inspired by P. Veličković and (Bronstein, 2022)). Bottom: Illustration of the message-passing scheme above using tensor diagrams (Hajij et al., 2023).

3 Literature Review

We now review the literature on topological neural networks (TNNs) over hypergraphs, simplicial complexes, cellular complexes, and combinatorial complexes, using the conceptual framework of Section 2. We summarize and compare the TNNs in terms of their architectures (Section 3.1), the machine learning tasks to which they have been applied (Section 3.2), and their geometric properties (Section 3.3).

3.1 Architectures

Figure 11 summarizes TNN architectures according to the fundamental concepts introduced in Section 2, with the *domain* on the vertical axis, the *message passing type* on the horizontal axis, *neighborhood structures* and *message passing equations* visually represented with tensor diagrams. We share complete message passing equations for each architecture—decomposed according to the four steps introduced in Section 2.3.1 and rewritten in unifying notations —at github.com/awesome-tnns.

3.1.1 Hypergraphs

Of the domains considered here, hypergraph neural networks have been most extensively researched, and have been surveyed previously (Ling et al., 2021; Gao et al., 2022; Hu et al., 2021; Wang et al., 2021a; Fischer et al., 2021). Many papers in the early literature do not use hypergraphs as the computational domain. Rather, algorithms like clique-expansion (Zien et al., 1999; Agarwal et al., 2005; Zhou et al., 2006) are used to reduce hypergraphs to graphs, which are then processed by the model. This reduction adversely affects performance, as structural information is lost (Hein et al., 2013; Li et al., 2013; Chien et al., 2019). Many such graph-based models—including HGNN (Feng et al., 2019), HyperConv(Bai et al., 2021), HyperGCN (Yadati et al., 2019), and HNHN (Dong et al., 2020)—are used as benchmarks for more recent models that *do* computationally operate on hypergraphs. Here, we focus on models that preserve hypergraph structure during learning.

Many hypergraph models use a message passing scheme comprised of two phases, with information flowing from nodes to their hyperedges and then back to the nodes. We call this the *two-phase*

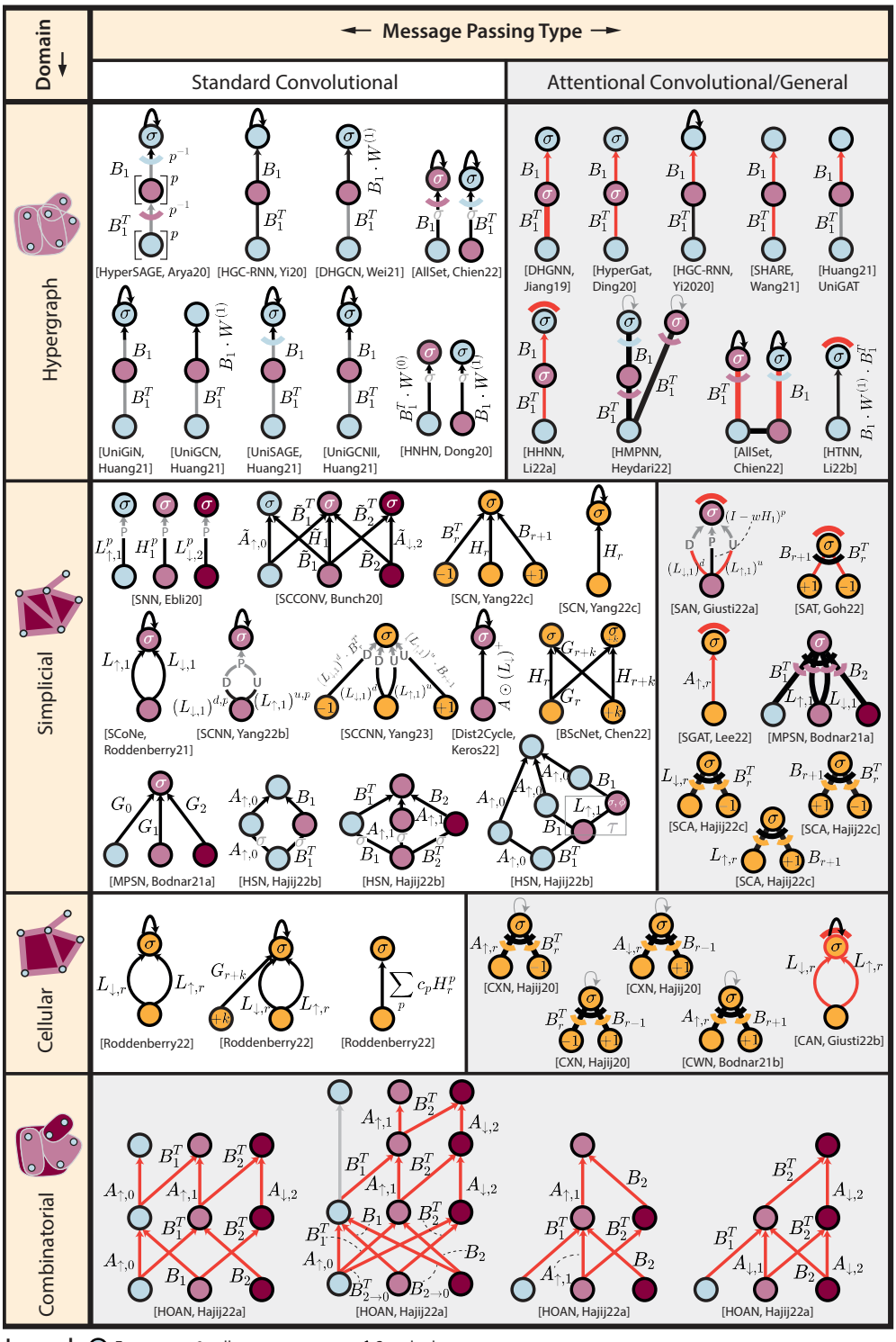

Figure 11: Topological Neural Networks (TNNs): A Graphical Literature Review. We organize TNNs according to the *domain* (rows) and the *message passing type* (columns).

scheme. The scheme appears in many tensor diagrams of Figure 11 where information flows from blue to pink (phase 1) and then from pink to blue (phase 2). The scheme is used in models with both standard and attentional message passing.

Standard

Of those using a standard message passing, the models from Arya et al. (2020), Yi and Park (2020), Wei et al. (2021), and Huang and Yang (2021) use the two-phase scheme. Yi and Park (2020) is unique in using a learnable weight matrix in the first phase of message passing. On the second phase, Wei et al. (2021) and the UniGCN model from Huang and Yang (2021) are unique in using a fixed weight matrix on top of learnable weights. In Arya et al. (2019), Yi and Park (2020), and the UniGNN, UniSAGE, and UniGCNII models from Huang and Yang (2021), the initial feature on each node is recurrently used to update each incoming message—denoted with a looped black arrow in Figure 11. We note that Huang and Yang (2021) systematically generalizes some of the most popular GNN architectures to hypergraphs with its unifying framework: UniGNN.

In Dong et al. (2020), fixed weights are used on both the node to hyperedge and hyperedge to node phases. The paper AllSet (Chien et al., 2022) uses a similar structure while incorporating fully learnable multi-set functions for neighborhood aggregation, which imbues its TNNs with high expressivity and generality. EHNN (Kim et al., 2022) (excluded from Figure 11 for its complexity; see written equations) proposes a maximally expressive model using sparse symmetric tensors to process data on hypergraphs with uniformly sized hyperedges.

Attentional / General

The models from Jiang et al. (2019), Ding et al. (2020), Yi and Park (2020), Wang et al. (2021a), and the UniGAT model from Huang and Yang (2021) employ the two-phase scheme in concert with attentional message passing. The architectures of Li et al. (2022a) and Li et al. (2022b) apply multi-head attention. Heydari and Livi (2022) adapts the two-phase scheme in order to update node features through two parallel paths. Chien et al. (2022) and Kim et al. (2022) offer transformer-based variants of their standard architectures, concurrently with Li et al. (2022b).

3.1.2 Simplicial Complexes

Simplicial complexes were first explored from a signal processing perspective (Battiston et al., 2020), with initial focus on edge flows (Jiang et al., 2011; Schaub and Segarra, 2018), Hodge Laplacians (Barbarossa and Sardellitti, 2020; Schaub et al., 2020), and convolution (Yang et al., 2021, 2022b; Isufi and Yang, 2022). As a precursor to deep learning, Roddenberry and Segarra (2019) introduced $\mathcal{L}_{\downarrow,1}$ in HodgeNet to learn convolutions on edge features on graphs. This contrasts with former GNN approaches processing node features.

Standard

Ebli et al. (2020) (SNN) and Bunch et al. (2020) (SCCONV) first generalized the convolutional approach of Roddenberry and Segarra (2019) to features supported on faces and cells of higher ranks. Unlike HodgeNet, SNN and SCCONV use both $\mathcal{L}_{\downarrow,1}$ and $\mathcal{L}_{\uparrow,1}$. In SNN (Ebli et al., 2020) messages are not passed between adjacent ranks. By contrast, SCCONV uses independent boundary and co-boundary neighborhoods, hence incorporating features from adjacent ranks. Yang et al. (2022c) also makes use of this multi-neighborhood scheme for updating and classifying edge features. They also propose a single neighborhood scheme with an update including the initial cell's feature. Roddenberry et al. (2021) and Yang et al. (2022d) devise schemes where messages coming from $\mathcal{L}_{\downarrow,1}$ and $\mathcal{L}_{\uparrow,1}$ are weighted separately, providing greater learning flexibility. Yang et al. (2022d) allows features to travel multiple hops through the domain by using a polynomial form of the neighborhood structures, leveraging the simplicial convolutional filter fromYang et al. (2022b). Yang and Isufi (2023) extend this multiple-hop model with additional neighborhood structures. Keros et al. (2022) used a modified version of $\mathcal{L}_{\downarrow}$ to find signals coiled around holes in the complex. BSCNet (Chen et al., 2022) combines node and edge-level shifting to predict links between nodes. This was the first model to pass messages between arbitrary ranks, leveraging a pseudo Hodge Laplacian.

MPSN (Bodnar et al., 2021a) explicitly details their message-passing scheme in the spatial domain, subsuming previous models described from a spectral approach. Hajij et al. (2022b) introduces High

Skip Networks (HSNs), in which each layer updates features through multiple sequential convolutional steps, "skipping" it through higher ranks as a generalization of skip-connections in conventional neural networks.

Attentional / General

SAN (Giusti et al., 2022a), SAT (Goh et al., 2022), and SGAT (Lee et al., 2022) concurrently introduced attentional message passing networks on the simplicial domain. Each model makes use of a unique set of neighborhood structures and attention coefficients. SGAT is the only model as of yet developed for heterogeneous simplicial complexes of general rank. Hajij et al. (2022c) introduces a variety of general message passing schemes with two neighborhood structures. Bodnar et al. (2021a) uses all four neighborhood structures, endowing each with a separate learnable matrix and general aggregation function.

3.1.3 Cellular Complexes

Just as for simplicial complexes, cellular complex networks have been significantly influenced by work in signal processing (Barbarossa and Sardellitti, 2020; Sardellitti et al., 2021; Roddenberry et al., 2022). These works demonstrated that representing data in the CC domain yields substantially better results than the more rigid SC domain.

Standard

Roddenberry et al. (2022) proposes theoretically possible message passing schemes for CCs inspired by works in the SC domain. As of yet, these models have not been implemented.

Attentional / General

Hajij et al. (2020) introduces the first TNNs to be theoretically defined on the CC domain. Bodnar et al. (2021b) was the first to implement and evaluate such a model, and demonstrated that TNNs on CCs outperform state-of-the-art graph-based models in expressivity and classification tests. The CAN model from (Giusti et al., 2022b) adapts a modified version of the message passing scheme from Giusti et al. (2022a) onto the CC domain.

3.1.4 Combinatorial Complexes

The combinatorial complex domain was only recently mathematically defined by Hajij et al. (2022a). This work introduces four attentional message passing schemes for CCCs tailored to mesh and graph classification. A more extensive analysis is needed to quantify the advantages of this domain over other topological domains.

3.2 Tasks

Table 3.2 reviews the tasks studied by each paper proposing TNNs. Tasks are first categorized into: *node-level* tasks assigning labels to nodes, as in node classification, regression or clustering; *edge-level tasks* assigning labels to edges, as in edge classification or link prediction; and *complex-level* tasks assigning labels to each complex as a whole, as in hypergraph classification. Tasks are additionally labeled according to their purpose (e.g. classification, regression, prediction). We also indicate the extent of benchmarking performed on each model and code availability.

Domain	Model	Task Level			Task Purpose	Comparisons
		Node	*Edge*	*Complex*		
HG	HyperSage Arya et al. (2020)	✓			Classification (Inductive + Transductive)	GNN SOTA
	AllSet Chien et al. (2022)	✓			Classification	TNN SOTA
	HyperGat Ding et al. (2020)	✓			Classification	GNN SOTA
	HNHN Dong et al. (2020)	✓	✓		Classification, Dimensionality Reduction	GNN SOTA
	HMPNN* Heydari and Livi (2022)	✓			Classification	TNN SOTA
	UniGNN Huang and Yang (2021)	✓			Classification (Inductive + Transductive)	TNN SOTA
	DHGNN Jiang et al. (2019)	✓			Classification (Multimodal)	GNN SOTA
	EHNN Kim et al. (2022)	✓			Classification, Keypoint Matching	TNN SOTA
	HHNN Li et al. (2022a)	✓			Link prediction	TNN SOTA
	HTNN Li et al. (2022b)	✓			Classification	TNN SOTA
	SHARE* Wang et al. (2021a)	✓			Prediction	GNN SOTA
	DHGCN* Wei et al. (2021)			✓	Classification	GNN SOTA
	HGC-RNN* (Yi and Park, 2020)	✓			Prediction	GNN SOTA
SC	MPSN Bodnar et al. (2021a)		✓	✓	Classification, Trajectory Classification	GNN SOTA
	SCCONV Bunch et al. (2020)			✓	Classification	Graph
	BScNet Chen et al. (2022)		✓		Link prediction	GNN SOTA
	SNN Ebli et al. (2020)		✓		Imputation	None
	SAN Giusti et al. (2022a)		✓		Classification, Trajectory Classification	TNN SOTA
	SAT Goh et al. (2022)		✓	✓	Classification, Trajectory Classification	TNN SOTA
	HSN* Hajij et al. (2022b)	✓	✓	✓	Classification, Link prediction, Vector embedding	Graph
	SCA* Hajij et al. (2022c)			✓	Clustering	Graph
	Dist2Cycle Keros et al. (2022)		✓		Homology Localization	GNN SOTA
	SGAT Lee et al. (2022)	✓			Classification	GNN SOTA
	SCoNe Roddenberry et al. (2021)		✓		Trajectory Classification	TNN SOTA
	SCNN* Yang et al. (2022b)		✓		Imputation	TNN SOTA
	SCCNN Yang and Isufi (2023)		✓		Link prediction, Trajectory Classification	TNN SOTA
	SCN Yang et al. (2022c)		✓		Classification	TNN SOTA
CC	CWN Bodnar et al. (2021b)		✓	✓	Classification, prediction, regression	GNN SOTA
	CAN Giusti et al. (2022b)			✓	Classification	GNN SOTA
CCC	HOAN* Hajij et al. (2022a)		✓	✓	Classification	GNN SOTA

Table 1: Applications of Topological Neural Networks (TNNs). We organize papers according to domain and task level, task purpose, and extent of benchmark testing (Graph: compared to graph-based models, GNN SOTA: compared to GNN state-of-the-art, TNN SOTA: compared to state-of-the-art on topological domain). We exclude papers without implementation, and use * to indicate that an implementation has not been shared.

3.3 Symmetries and Geometric Properties

Topological domains possess symmetries and other geometric properties that should be respected to ensure the quality of the features learned by a TNN (Bronstein et al., 2017). Here, we outline such properties harnessed by models in the literature.

Hypergraphs. On hypergraphs, the following symmetries are desirable:

1. *Permutation Invariance:* Relabeling the nodes and applying the TNN yields an output that is identical to the original output obtained without relabeling. This requires the aggregation functions to be permutation invariant, such as a mean or a sum (Arya et al., 2020; Kim et al., 2022; Chien et al., 2022; Dong et al., 2020; Keros et al., 2022). This is also called *hypergraph isomorphism invariance*.
2. *Global Neighborhood Invariance:* The network's representation of a node is invariant to hyperedge cardinality: a hyperedge connecting many nodes is weighted the same as a hyperedge connecting less nodes (Arya et al., 2020).

Simplicial Complex. For simplicial complexes, the following symmetries have been considered:

1. *Permutation Invariance:* Invariance to node relabeling; the same as for HGs. (Schaub et al., 2021; Roddenberry et al., 2021; Bodnar et al., 2021a)
2. *Orientation Equivariance:* Changing the orientation of the simplicial complex (i.e. flipping the signs in the incidence matrix) re-orients the output of that network accordingly (Schaub et al., 2021; Roddenberry et al., 2021; Bodnar et al., 2021a).
3. *Simplicial Locality (geometric property):* In each layer, messages are only passed between r-cells and $(r \pm 1)$-cells (Schaub et al., 2021). If that property is not verified, and messages can pass between any r- and r'-cells, then the network has *extended simplicial locality*.

In addition, *simplicial awareness* can be imposed, such that message passing on a simplicial complex with maximum cell rank r depends on every rank $r' \leq r$ (Roddenberry et al., 2021).

Cellular Complex and Combinatorial Complex. Permutation invariance is defined for CCs (Bodnar et al., 2021b) and CCCs (Hajij et al., 2022a) just as for SCs and HGs. Beyond generalizing global neighborhood invariance to CCC, more research is required to understand the symmetries that can equip this general topological domain.

4 Discussion

Our literature review has revealed the diversity of TNN architectures as well as their main axes of comparison. Looking to the future, we highlight four salient opportunities for development.

Within-Domain and Between-Domain Benchmarking. Table 3.2 shows that the domain choice strongly correlates with a TNN's task level. This necessarily makes within-domain comparisons difficult, regardless of code sharing. We also emphasize that many TNNs are only benchmarked against graph-based models or early models in their respective domain, which makes between-domain comparisons equally difficult. As the field grows, improving within and between-domain benchmarking mechanisms will be critical to better informing model selection and quantifying progress.

TNN Architectures on General Domains. The diversity of implementations on HGs and SCs point to a strong potential for similar development in the cellular and combinatorial domains. For instance, only one attentional CC model has been proposed (Giusti et al., 2022b). Moreover, any previously developed HG/SC/CC model can be reproduced in the CCC domain and, if desirable, improved with greater flexibility. Evaluating the impact of this added flexibility will directly characterize utility of richer topological structure in deep learning.

Connecting to the Graph Literature. The HG field's ties to the graph community has led to GNN-based advancements not yet propagated to other domains. A first example are dynamic domains, successful with HGs for tasks like pose estimation (Liu et al., 2020), rail transit modeling (Wang et al., 2021b), and co-authorship prediction (Jiang et al., 2019). No work in other discrete domains has explored dynamism. In addition, outside of the HG domain, TNNs are largely implemented as homogeneous networks. This leaves room for heterogeneous and non-Euclidean generalizations.

Going Deeper. Over-smoothing occurs when a network is too effective at aggregating signal over multiple layers. This leads to very similar features across cells and poor performance on the downstream learning task. While this issue draws attention in the graph community (Chen et al., 2020a; Oono and Suzuki, 2020; Rusch et al., 2023), little of this work has been generalized to TNNs, causing them to remain mostly shallow. UniGCNII (Huang and Yang, 2021) achieves a 64-layer deep TNN by generalizing over-smoothing solutions from GNNs (Chen et al., 2020b) to the HG domain. HSNs (Hajij et al., 2022b) generalize skip connections to allow signal to propagate further, but are still implemented as shallow networks.

5 Conclusion

In this work, we have provided a comprehensive, intuitive and critical view of the advances in TNNs through unifying notations and graphical illustrations. We have characterized each neural network by its choice of data domain and its model, which we further specify through choice of neighboring structure(s) and message-passing scheme. We hope that this review will make this rich body of work more accessible to practitioners whose fields would benefit from topology-sensitive deep learning.

References

David Knoke and Song Yang. *Social network analysis*. SAGE publications, 2019.

Kanchan Jha, Sriparna Saha, and Hiteshi Singh. Prediction of protein–protein interaction using graph neural networks. *Scientific Reports*, 12(1):1–12, 2022.

Michael M Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021.

Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81, 2020.

Mustafa Hajij, Ghada Zamzmi, Theodore Papamarkou, Nina Miolane, Aldo Guzmán-Sáenz, Karthikeyan Natesan Ramamurthy, Tolga Birdal, Tamal Dey, Soham Mukherjee, Shreyas Samaga, Neal Livesay, Robin Walters, Paul Rosen, and Michael Schaub. Topological deep learning: Going beyond graph data. *arXiv preprint arXiv:1906.09068 (v3)*, 2023.

Cristian Bodnar. *Topological Deep Learning: Graphs, Complexes, Sheaves*. PhD thesis, Apollo - University of Cambridge Repository, 2022. URL https://www.repository.cam.ac.uk/handle/1810/350982.

Cristian Bodnar, Fabrizio Frasca, Yuguang Wang, Nina Otter, Guido F Montufar, Pietro Lio, and Michael Bronstein. Weisfeiler and Lehman Go Topological: Message Passing Simplicial Networks. In *International Conference on Machine Learning*, pages 1026–1037. PMLR, 2021a.

Cristian Bodnar, Fabrizio Frasca, Nina Otter, Yuguang Wang, Pietro Lio, Guido F Montufar, and Michael Bronstein. Weisfeiler and Lehman Go Cellular: CW Networks. *Advances in Neural Information Processing Systems*, 34:2625–2640, 2021b.

Jing Huang and Jie Yang. Unignn: a unified framework for graph and hypergraph neural networks. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, 2021.

Yihe Dong, Will Sawin, and Yoshua Bengio. Hnhn: Hypergraph networks with hyperedge neurons. *ICML Graph Representation Learning and Beyond Workshop*, 2020. URL https://arxiv.org/abs/2006.12278.

Mustafa Hajij, Ghada Zamzmi, Theodore Papamarkou, Nina Miolane, Aldo Guzmán-Sáenz, and Karthikeyan Natesan Ramamurthy. Higher-order attention networks. *arXiv preprint arXiv:2206.00606 (v1)*, 2022a.

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

Yuzhou Chen, Yulia R Gel, and H Vincent Poor. Bscnets: Block simplicial complex neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6333–6341, 2022.

Leo Torres, Ann S Blevins, Danielle Bassett, and Tina Eliassi-Rad. The why, how, and when of representations for complex systems. *SIAM Review*, 63(3):435–485, 2021.

Federico Battiston, Enrico Amico, Alain Barrat, Ginestra Bianconi, Guilherme Ferraz de Arruda, Benedetta Franceschiello, Iacopo Iacopini, Sonia Kéfi, Vito Latora, Yamir Moreno, et al. The physics of higher-order interactions in complex systems. *Nature Physics*, 17(10):1093–1098, 2021.

Felix Hensel, Michael Moor, and Bastian Rieck. A survey of topological machine learning methods. *Frontiers in Artificial Intelligence*, 4, 2021. ISSN 2624-8212. doi: 10.3389/frai.2021.681108. URL https://www.frontiersin.org/articles/10.3389/frai.2021.681108.

Ruochen Yang, Frederic Sala, and Paul Bogdan. Efficient representation learning for higher-order data with simplicial complexes. In Bastian Rieck and Razvan Pascanu, editors, *Proceedings of the First Learning on Graphs Conference*, volume 198 of *Proceedings of Machine Learning Research*, pages 13:1–13:21. PMLR, 09–12 Dec 2022a.

T. Konstantin Rusch, Michael M. Bronstein, and Siddhartha Mishra. A survey on oversmoothing in graph neural networks, 2023.

Petar Veličković. Message passing all the way up. *arXiv preprint arXiv:2202.11097*, 2022.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, 2014.

Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.

Ankit Sharma, Shafiq Joty, Himanshu Kharkwal, and Jaideep Srivastava. Hyperedge2vec: Distributed representations for hyperedges, 2018.

Josh Payne. Deep hyperedges: a framework for transductive and inductive learning on hypergraphs. *arXiv preprint arXiv:1910.02633*, 2019.

Jacob Charles Wright Billings, Mirko Hu, Giulia Lerda, Alexey N Medvedev, Francesco Mottes, Adrian Onicas, Andrea Santoro, and Giovanni Petri. Simplex2vec embeddings for community detection in simplicial complexes. *arXiv preprint arXiv:1906.09068*, 2019.

Celia Hacker. k-simplex2vec: a simplicial extension of node2vec. *arXiv preprint arXiv:2010.05636*, 2020.

Mustafa Hajij, Kyle Istvan, and Ghada Zamzmi. Cell Complex Neural Networks. *NeurIPS 2020 Workshop TDA and Beyond*, 2020.

Michael Aschbacher. Combinatorial cell complexes. In *Progress in Algebraic Combinatorics*, pages 1–80. Mathematical Society of Japan, 1996.

Reinhard Klette. Cell complexes through time. In *Vision Geometry IX*, volume 4117, pages 134–145. SPIE, 2000.

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.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.

Sajjad Heydari and Lorenzo Livi. Message passing neural networks for hypergraphs. In Elias Pimenidis, Plamen P. Angelov, Chrisina Jayne, Antonios Papaleonidas, and Mehmet Aydin, editors, *Proceedings of 31st International Conference on Artificial Neural Networks, Part II*, volume 13530 of *Lecture Notes in Computer Science*, pages 583–592. Springer, 2022. doi: 10.1007/978-3-031-15931-2_48.

Michael Bronstein. Beyond message passing: a physics-inspired paradigm for graph neural networks, May 2022. URL https://thegradient.pub/graph-neural-networks-beyond-message-passing-and-weisfeiler-lehman/.

Tian Ling, Zhang Jinchuan, Zhang Jinhao, Zhou Wangtao, and Zhou Xue. A review of knowledge graphs: Representation, construction, reasoning and knowledge hypergraph theory [j]. *Computer Applications*, 41(08):2161–2186, 2021.

Yue Gao, Zizhao Zhang, Haojie Lin, Xibin Zhao, Shaoyi Du, and Changqing Zou. Hypergraph learning: Methods and practices. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(5):2548–2566, 2022. doi: 10.1109/TPAMI.2020.3039374.

Bing-De Hu, Xin-Gen Wang, Xin-Yu Wang, Ming-Li Song, and Chun Chen. Survey on hypergraph learning: Algorithm classification and application analysis. *Journal of Software*, 33(2):498–523, 2021.

Jianling Wang, Kaize Ding, Ziwei Zhu, and James Caverlee. Session-based recommendation with hypergraph attention networks. In *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*, pages 82–90. SIAM, 2021a.

Maximilian T Fischer, Alexander Frings, Daniel A Keim, and Daniel Seebacher. Towards a survey on static and dynamic hypergraph visualizations. In *2021 IEEE visualization conference (VIS)*, pages 81–85. IEEE, 2021.

Jason Y Zien, Martine DF Schlag, and Pak K Chan. Multilevel spectral hypergraph partitioning with arbitrary vertex sizes. *IEEE Transactions on computer-aided design of integrated circuits and systems*, 18(9):1389–1399, 1999.

Sameer Agarwal, Jongwoo Lim, Lihi Zelnik-Manor, Pietro Perona, David Kriegman, and Serge Belongie. Beyond pairwise clustering. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 838–845. IEEE, 2005.

Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. Learning with hypergraphs: Clustering, classification, and embedding. *Advances in neural information processing systems*, 19, 2006.

Matthias Hein, Simon Setzer, Leonardo Jost, and Syama Sundar Rangapuram. The total variation on hypergraphs-learning on hypergraphs revisited. *Advances in Neural Information Processing Systems*, 26, 2013.

Guoyin Li, Liqun Qi, and Gaohang Yu. The z-eigenvalues of a symmetric tensor and its application to spectral hypergraph theory. *Numerical Linear Algebra with Applications*, 20(6):1001–1029, 2013.

I Eli Chien, Huozhi Zhou, and Pan Li. *hs*2: Active learning over hypergraphs with pointwise and pairwise queries. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2466–2475. PMLR, 2019.

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, pages 3558–3565, 2019.

Song Bai, Feihu Zhang, and Philip HS Torr. Hypergraph convolution and hypergraph attention. *Pattern Recognition*, 110:107637, 2021.

Naganand Yadati, Madhav Nimishakavi, Prateek Yadav, Vikram Nitin, Anand Louis, and Partha Talukdar. Hypergcn: A new method for training graph convolutional networks on hypergraphs. *Advances in neural information processing systems*, 32, 2019.

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

Jaehyuk Yi and Jinkyoo Park. Hypergraph convolutional recurrent neural network. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3366–3376, 2020.

Jinfeng Wei, Yunxin Wang, Mengli Guo, Pei Lv, Xiaoshan Yang, and Mingliang Xu. Dynamic hypergraph convolutional networks for skeleton-based action recognition. *arXiv preprint arXiv:2112.10570*, 2021.

Devanshu Arya, Stevan Rudinac, and Marcel Worring. Hyperlearn: a distributed approach for representation learning in datasets with many modalities. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 2245–2253, 2019.

Eli Chien, Chao Pan, Jianhao Peng, and Olgica Milenkovic. You are allset: A multiset function framework for hypergraph neural networks. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=hpBTIv2uy_E.

Jinwoo Kim, Saeyoon Oh, Sungjun Cho, and Seunghoon Hong. Equivariant hypergraph neural networks. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXI*, pages 86–103. Springer, 2022.

Jianwen Jiang, Yuxuan Wei, Yifan Feng, Jingxuan Cao, and Yue Gao. Dynamic hypergraph neural networks. In *Proceedings of International Joint Conferences on Artificial Intelligence*, 2019.

Kaize Ding, Jianling Wang, Jundong Li, Dingcheng Li, and Huan Liu. Be more with less: Hypergraph attention networks for inductive text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4927–4936, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.399. URL https://aclanthology.org/2020.emnlp-main.399.

Yongkang Li, Zipei Fan, Jixiao Zhang, Dengheng Shi, Tianqi Xu, Du Yin, Jinliang Deng, and Xuan Song. Heterogeneous hypergraph neural network for friend recommendation with human mobility. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 4209–4213, 2022a.

Mengran Li, Yong Zhang, Xiaoyong Li, Yuchen Zhang, and Baocai Yin. Hypergraph transformer neural networks. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2022b.

Federico Battiston, Giulia Cencetti, Iacopo Iacopini, Vito Latora, Maxime Lucas, Alice Patania, Jean-Gabriel Young, and Giovanni Petri. Networks beyond pairwise interactions: structure and dynamics. *Physics Reports*, 874:1–92, 2020.

Xiaoye Jiang, Lek-Heng Lim, Yuan Yao, and Yinyu Ye. Statistical ranking and combinatorial hodge theory. *Mathematical Programming*, 127(1):203–244, 2011.

Michael T Schaub and Santiago Segarra. Flow smoothing and denoising: Graph signal processing in the edge-space. In *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 735–739. IEEE, 2018.

Michael T Schaub, Austin R Benson, Paul Horn, Gabor Lippner, and Ali Jadbabaie. Random walks on simplicial complexes and the normalized hodge 1-laplacian. *SIAM Review*, 62(2):353–391, 2020.

Maosheng Yang, Elvin Isufi, Michael T Schaub, and Geert Leus. Finite impulse response filters for simplicial complexes. In *2021 29th European Signal Processing Conference (EUSIPCO)*, pages 2005–2009. IEEE, 2021.

Maosheng Yang, Elvin Isufi, Michael T Schaub, and Geert Leus. Simplicial convolutional filters. *IEEE Transactions on Signal Processing*, 70:4633–4648, 2022b.

Elvin Isufi and Maosheng Yang. Convolutional filtering in simplicial complexes. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5578–5582. IEEE, 2022.

T Mitchell Roddenberry and Santiago Segarra. Hodgenet: Graph neural networks for edge data. In *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, pages 220–224. IEEE, 2019.

Stefania Ebli, Michaël Defferrard, and Gard Spreemann. Simplicial neural networks. In *TDA & Beyond*, 2020.

Eric Bunch, Qian You, Glenn Fung, and Vikas Singh. Simplicial 2-complex convolutional neural networks. In *TDA & Beyond*, 2020.

Ruochen Yang, Frederic Sala, and Paul Bogdan. Efficient representation learning for higher-order data with simplicial complexes. In *Learning on Graphs Conference*, pages 13–1. PMLR, 2022c.

T Mitchell Roddenberry, Nicholas Glaze, and Santiago Segarra. Principled simplicial neural networks for trajectory prediction. In *International Conference on Machine Learning*, pages 9020–9029. PMLR, 2021.

Maosheng Yang, Elvin Isufi, and Geert Leus. Simplicial convolutional neural networks. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8847–8851. IEEE, 2022d.

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

Alexandros D Keros, Vidit Nanda, and Kartic Subr. Dist2cycle: A simplicial neural network for homology localization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 7133–7142, 2022.

Mustafa Hajij, Karthikeyan Natesan Ramamurthy, Aldo Guzmán-Sáenz, and Ghada Za. High Skip Networks: A Higher Order Generalization of Skip Connections. In *ICLR 2022 Workshop on Geometrical and Topological Representation Learning*, 2022b.

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

Christopher Wei Jin Goh, Cristian Bodnar, and Pietro Lio. Simplicial attention networks. *arXiv preprint arXiv:2204.09455*, 2022.

See Hian Lee, Feng Ji, and Wee Peng Tay. Sgat: Simplicial graph attention network. In *International Joint Conference on Artificial Intelligence*, 2022.

Mustafa Hajij, Ghada Zamzmi, Theodore Papamarkou, Vasileios Maroulas, and Xuanting Cai. Simplicial complex representation learning. In *Machine Learning on Graphs (MLoG) Workshop at 15th ACM International WSDM (2022) Conference, WSDM2022-MLoG ; Conference date: 21-02-2022 Through 25-02-2022*, January 2022c.

Stefania Sardellitti, Sergio Barbarossa, and Lucia Testa. Topological signal processing over cell complexes. In *2021 55th Asilomar Conference on Signals, Systems, and Computers*, pages 1558–1562. IEEE, 2021.

T Mitchell Roddenberry, Michael T Schaub, and Mustafa Hajij. Signal processing on cell complexes. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8852–8856. IEEE, 2022.

Lorenzo Giusti, Claudio Battiloro, Lucia Testa, Paolo Di Lorenzo, Stefania Sardellitti, and Sergio Barbarossa. Cell attention networks. *arXiv preprint arXiv:2209.08179*, 2022b.

Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

Shengyuan Liu, Pei Lv, Yuzhen Zhang, Jie Fu, Junjin Cheng, Wanqing Li, Bing Zhou, and Mingliang Xu. Semi-dynamic hypergraph neural network for 3d pose estimation. In *IJCAI*, pages 782–788, 2020.

Jingcheng Wang, Yong Zhang, Yun Wei, Yongli Hu, Xinglin Piao, and Baocai Yin. Metro passenger flow prediction via dynamic hypergraph convolution networks. *IEEE Transactions on Intelligent Transportation Systems*, 22(12):7891–7903, 2021b.

Deli Chen, Yankai Lin, Wei Li, Peng Li, Jie Zhou, and Xu Sun. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 3438–3445, 2020a.

Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=S1ldO2EFPr.

Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks. In *International conference on machine learning*, pages 1725–1735. PMLR, 2020b.