# HL-HGAT: Heterogeneous Graph Attention Network via Hodge-Laplacian Operator

Jinghan Huang, Qiufeng Chen, *Senior Member, IEEE*, Pengli Zhu, *Member, IEEE*, Yijun Bian, Nanguang Chen, Moo K. Chung, and Anqi Qiu, *Senior Member, IEEE*

***Abstract*—Graph neural networks (GNNs) have proven effective in capturing relationships among nodes in a graph. This study introduces a novel perspective by considering a graph as a simplicial complex, encompassing nodes, edges, triangles, and $k$-simplices, enabling the definition of graph-structured data on any $k$-simplex. We design a novel Hodge-Laplacian heterogeneous graph attention network (HL-HGAT) to learn heterogeneous signal representations across $k$-simplices. The HL-HGAT incorporates three key components: HL convolutional filters (HL-filters), simplicial projection (SP), and simplicial attention pooling (SAP) operators, applied to $k$-simplices. HL-filters leverage the unique topology of $k$-simplices encoded by the Hodge-Laplacian (HL) operator, operating within the spectral domain of the $k$-th HL operator. To address computation challenges, we introduce a polynomial approximation for HL-filters, exhibiting spatial localization properties. Additionally, we propose a pooling operator to coarsen $k$-simplices, combining features through simplicial attention mechanisms of self-attention and cross-attention via transformers and SP operators, capturing topological interconnections across multiple dimensions of simplices. The HL-HGAT is comprehensively evaluated across diverse graph applications, including NP-hard problems, graph multi-label and classification challenges, and graph regression tasks in logistics, computer vision, biology, chemistry, and neuroscience. The results demonstrate the model's efficacy and versatility in handling a wide range of graph-based scenarios.**

***Index Terms*—Graph neural network (GNN), graph transformer, Hodge-Laplacian filters, simplex, graph pooling.**

## I. Introduction

GRAPH-structured data, embodying entities as nodes and their relationships as edges, offer a robust framework for modeling and analyzing complex interdependencies. To harness the potent representations of these data, graph neural networks (GNNs) have been engineered, primarily focusing on node-centric message aggregation to discern features defined on graph nodes [1], [2]. Such an approach has proved valuable across multiple domains, illuminating insights and driving informed decisions [3], [4]. The versatility and utility of GNNs stand out in predicting molecular generation and chemical properties [5], [6], analyzing brain networks [7], [8], forecasting traffic flow in transport systems [9], [10], and numerous other applications [11], [12], [13], [14], [15]. However, the real-world frequently exhibits heterogeneous data defined on nodes, edges, and more intricate connections among these graph elements [16], [17]. Such scenarios reveal an extant gap: the need for a generic GNN capable of comprehending and analyzing higher-order relationships and dependencies within heterogeneous graph-structured data.

The architecture of existing GNNs, analogous to conventional convolutional neural networks (CNNs) [18], encompasses two principal components: convolution over graph nodes and graph pooling for spatial dimensionality reduction and multi-scale learning [2]. However, the irregular topology of graphs poses challenges to the shifting operation inherent in the convolution on a graph. Current GNNs attempt to navigate this in two ways, through the spatial and spectral domains. In the spatial domain, GNNs endeavor to unearth valuable information from node neighborhoods to facilitate message aggregation for node convolution. Graph convolutional network (GCN) [3] and dynamic graph convolutional network (dGCN) [19] initiate graph convolution with isotropic normalization filters, but they lack adaptive node weighting. Some elaborate mechanisms are introduced into convolutional layers to allow for adaptive aggregation of neighborhood information into target nodes [4], [20], [21], [22], such as clustering-based embedding mechanisms in BrainGNN [20], attention mechanisms in graph attention network (GAT) [4], general, powerful, and scalable (GPS) graph transformer [22], and gating mechanisms in GatedGCN (gated

Received 11 December 2023; revised 2 October 2024; accepted 27 July 2025. Date of publication 31 July 2025; date of current version 5 November 2025. This work was supported in part by STI 2030 – Major Project under Grant 2022ZD0209000, in part by the National Research Foundation, Singapore, in part by the Agency for Science Technology and Research (A*STAR), Singapore, through Prenatal/Early Childhood under Grant H22P0M0007, in part by RGC GRF Project under Grant 15201124, and in part by Hong Kong Global STEM Scholar Scheme. Recommended for acceptance by V. Pavlovic. *(Jinghan Huang and Qiufeng Chen contributed equally to this work.) (Corresponding author: Anqi Qiu.)*

This work involved human subjects or animals in its research. Approval of all ethical and experimental procedures and protocols was granted by the Central Institutional Review Board (CIRB) at the University of California, San Diego for the ABCD Research and the IRB of Washington University School of Medicine for the OASIS-3 Projects.

Jinghan Huang, Pengli Zhu, Yijun Bian, and Nanguang Chen are with the Department of Biomedical Engineering, National University of Singapore, Singapore 119077.

Qiufeng Chen is with the College of Computer and Information Science, Fujian Agriculture and Forestry University, Fuzhou 353302, China.

Moo K. Chung is with the Department of Biostatistics and Medical Informatics, The University of Wisconsin-Madison, Madison, WI 53706 USA.

Anqi Qiu is with the Department of Biomedical Engineering, National University of Singapore, Singapore 119077, also with the Department of Health Technology and Informatics, The Hong Kong Polytechnic University, Kowloon Hong Kong 999077, also with the Mental Health Research Center, The Hong Kong Polytechnic University, Kowloon Hong Kong 999077, and also with the Department of Biomedical Engineering, Johns Hopkins University, Baltimore, MD 21218 USA (e-mail: an-qi.qiu@polyu.edu.hk).

This article has supplementary downloadable material available at https://doi.org/10.1109/TPAMI.2025.3594226, provided by the authors.

Digital Object Identifier 10.1109/TPAMI.2025.3594226

graph convolutional network) [21]. Especially, the attention mechanism via a transformer [23] perceives each node as a token and learning weights over nodes of a graph for node-centric message aggregation [4], [22].

In the spectral domain, graph convolution is achieved through the design of spectral filters via the graph Laplacian, where filter values represent the importance of neighborhood nodes [24]. For computational efficiency, Chebyshev polynomials and other polynomial forms are used to approximate spectral filters for GNNs [25], [26]. When dealing with larger graphs, spectral graph convolution with polynomial approximation emerges as a computationally efficient and spatially localized approach [26].

Despite their widespread use, node-centric convolutions are not adept at managing high-order dimensional relationships between nodes, edges, and beyond. Thus far, CensNet [27] and Hypergraph NN [28] have employed techniques, such as line graph transformation and dual hypergraph transformation, to interchange the roles of nodes and edges within a graph. Spatial graph convolution on simplicial complexes, including nodes and edges, has been introduced using the Hodge-Laplacian operator [29], [30], [31], [32], [33], [34], [35]. The Hodge-Laplacian operator generalizes the graph Laplacian to higher-dimensional simplices, encoding the neighboring relationship between higher-dimensional simplices. Bodnar et al. [33] proposed a message-passing neural network for simplicial complexes by firstly aggregating signals from neighboring simplices of the same dimension followed by a nonlinear update function. This approach was further extended by incorporating multihop neighbors through the powers of the Hodge-Laplacian operator [31], [32]. Meanwhile, Goh et al. [34] and Giusti et al. [35] introduced a self-attention mechanism into the aggregation operation, extending GAT to develop the Simplicial Attention Network (SAT). These simplex-based GNNs have enabled traditional GNN convolution to be applied beyond nodes, achieving significant success in social networks, molecular science, and neuroscience [27], [28], [36], [37], demonstrating the advantages of edge-based and simplex-based convolution when signals are intrinsically linked to higher-dimensional simplices. However, these models often neglect interactions between simplices of different dimensions, and a comprehensive pooling method for reducing the spatial scale of simplices across various dimensions remains lacking.

The second crucial component of GNNs is graph pooling, which serves to reduce data dimensionality and learn multi-scale spatial information [38]. Currently, most graph pooling methods first coarsen a graph by clustering nodes and then pool signals by taking their average or maximum across the nodes within each cluster [25], [39]. An alternative approach is to perform graph pooling by discarding nodes based on their attention scores [40], [41], or further extending the pooling method from nodes to edges [42]. However, to the best of our knowledge, none of the existing graph pooling methods consider higher-order interactions between nodes, edges, or other parts in the signal pooling process. Therefore, there is an urgent need for efficient and innovative graph pooling methods that can utilize heterogeneous information about nodes, edges, and beyond.

In this study, we present a pioneering approach, named the Hodge-Laplacian heterogeneous graph attention network (HL-HGAT). This model interprets a graph as a simplicial complex, encompassing nodes, edges, triangles, and more, enabling the definition of graph-structured data on any $k$-simplex. In this setting, HL-HGAT aims to learn heterogeneous signal representations using Hodge-Laplacian (HL) operators on $k$-simplices while capturing their intricate complex relationships across $k$-simplices. We propose the HL-HGAT with three key elements: convolutional filters constructed by HL operators, multi-simplicial interaction (MSI) by introducing simplicial projection operators, and simplicial attention pooling (SAP) via simplicial transformers. Notably, we utilize the $k$-th boundary operator encoding the relationship between the $(k-1)$-simplices and $k$-simplices, enabling the construction of the $k$-th HL operator on $k$-simplices. We then develop convolutional filters on $k$-simplices in the spectral domain of the $k$-th HL operator, termed HL-filters. This necessitates the computation of the $k$-th HL eigenfunctions, which can be computationally demanding for larger graphs. To mitigate this, we introduce a generic polynomial approximation of the HL-filters, which exhibits a spatial localization property relative to the polynomial order, representing a significant advancement in the field. Furthermore, we design a simplicial projection operator that facilitates the conversion of signals from $k_1$-simplices to $k_2$-simplices, enabling their fusion and thereby enabling the learning of their interactions. In addition, we define a pooling operator by coarsening the $k$-simplices and consolidating features associated with these simplices using attention mechanisms. These mechanisms encompass self-attention and cross-attention through the simplicial projection operators and transformers, collectively referred to as simplicial transformers. The simplicial transformers ascertain the importance of each simplex by learning its weight while gathering signals from its topologically connected simplices, thereby assessing their relevance to a downstream task. Therefore, our innovative solution—HL-HGAT—encompasses the development of HL-filters, simplicial projection operators, and simplicial transformers, providing a comprehensive framework for capturing complex relationships in graph-structured data.

To rigorously assess the HL-HGAT model, we conducted comprehensive evaluations across a diverse range of graph applications, including NP-hard, graph multi-label and classification, and graph regression problems in the domains of logistics, computer vision, biology, chemistry, and neuroscience. We compared the HL-HGAT performance with state-of-the-art GNN models, such as GCN [3], GAT [4], GatedGCN [21], GPS [22], and SAT [34], as well as GNN models specialized for brain network data, such as BrainGNN [20], dGCN [19], Hypergraph NN [28], using six benchmark datasets. Our results include interpretable attention maps that demonstrate the HL-HGAT's ability to learn meaningful representations at nodes, edges, and beyond.

The rest of this paper is organized as follows. We summarize the related work in Section II, detail each component of the proposed HL-HGAT in Section III, and evaluate the performance of HL-HGAT in Section IV, respectively.

## II. Related Work

### A. Convolution on a Graph

*Spatial Domain:* Graph convolution in the spatial domain is a prominent technique in graph neural networks, providing a mechanism to map signals on nodes into latent features by leveraging information from their respective neighborhoods. The concept of graph convolution was first introduced using isotropic normalization filters to aggregate neighborhood signals with equal weight, as illustrated by early works such as graph convolutional network (GCN) [3] and dynamic graph convolutional network (dGCN) [19]. These methods were relatively straightforward and relied on the premise that every node in the neighborhood contributes equally to the feature representation of the target node. Although it provided a starting point for graph convolution, it lacked the capability to capture the varying importance of different nodes.

Recognizing the limitation of equal node contribution, subsequent development introduced various weight updating strategies, accounting for signal differences between nodes. These strategies incorporated attention or gating mechanisms to discern the differences between nodes and assign weights accordingly [4], [21]. The inclusion of these mechanisms allowed for the realization of anisotropic convolutions, leading to more nuanced and effective aggregation of neighborhood information. For instance, the gated graph convolutional network (GatedGCN) developed by Bresson and Laurent (2017) leverages a gating mechanism to introduce anisotropy into the convolution process [21]. The gating mechanism, inspired by recurrent neural networks (RNNs), facilitates dynamic weight assignment based on the relationships between nodes, ensuring a more accurate and adaptive aggregation process. Similarly, the graph attention network (GAT) utilizes a transformer architecture to learn attention coefficients that assign different weights to neighboring nodes based on their importance [4]. The attention mechanism allows the model to focus on relevant nodes, leading to more efficient feature extraction. Simplicial Attention Network (SAT) [34], an extension of GAT to higher-dimensional simplices, performs spatial convolution by aggregating signals from neighboring simplices. Unlike previous methods that aggregate neighborhood signals with equal weights [31], [32], [33], SAT employs a self-attention mechanism to assign different weights to the target simplex and its neighboring simplices. This mechanism enables SAT to effectively model signals on high-dimensional simplices, allowing for more comprehensive feature extraction from complex structures.

In addition to the previously mentioned techniques, there are other innovative approaches such as Graph Transformer, which treats each node as a token and encodes its position along with signals on nodes [23]. This method allows for the capture of global dependencies between nodes, enhancing the expressive power of graph convolution. A more recent approach is the general, powerful, and scalable (GPS) graph transformer [22]. This model blends the strengths of traditional transformers with spatial graph convolution in each layer to effectively learn signal representation over nodes. This innovative approach encapsulates the benefits of spatial-based graph convolution techniques, providing a robust tool for handling graph-structured data defined on nodes of graphs.

*Spectral Domain:* In the spectral domain, graph convolution can be achieved via the graph Laplacian that captures the graph's topological characteristics [24]. The groundbreaking work of Defferrard et al. revolutionized the application of convolution operations to graphs in the spectral domain [25]. They developed a fast localized convolutional filter that propagates information within the $k$-hop neighborhood of a node. Here, a "hop" signifies the adjacency relationship between nodes: two nodes are within one hop of each other if they are directly connected by an edge. The $k$-hop neighborhood of a node encompasses all nodes reachable within $k$ hops. To enhance the efficiency of this convolution operation, a parameterization based on Chebyshev polynomials is proposed [25], [26]. Polynomials (e.g., Chebyshev, Hermite, Laguerre) are a sequence of orthogonal polynomials, which are well-suited to approximation tasks [26]. By expressing the graph convolution operation in terms of polynomials, the graph convolution achieves linear computation complexity. This development was a significant contribution, as it made the convolution operation computationally feasible for large-scale graphs.

However, the state-of-the-art in graph convolution is not without limitations. The convolution operations focus primarily on nodes and the messages are passed solely among these entities. This methodology overlooks the rich information contained in higher-dimensional simplices, such as edges (1-simplices) and triangles (2-simplices). Edges, for instance, embody crucial relationships between nodes, and their information may be valuable in many graph-related tasks. Hence, future research in the spectral domain of graph convolution could benefit significantly from exploring methods that can effectively utilize information from these higher-dimensional simplices, which is the focus of the present study.

### B. Graph Pooling

Graph pooling is a crucial operation in GNNs that mirrors the function of pooling layers in traditional CNNs [18]. Its primary objective is to reduce the spatial dimensionality of graph data while preserving the essential structural information. One common approach to graph pooling is to group nodes into clusters and then aggregate (or pool) the signals within each cluster. This aggregation can be achieved by taking the average or the maximum signal value within each cluster, which effectively reduces the dimensionality while capturing the intra-cluster relationships [25]. Similarly, MinCutPool, inspired by spectral clustering, encourages nodes within the same cluster to be strongly connected and have similar features by utilizing a minCUT loss [43].

Differentiable graph pooling (DIFFPOOL) softens the assignment of nodes to clusters by using a learned assignment matrix that probabilistically maps nodes to clusters in a coarsened graph [39]. This process makes the pooling operation differentiable and facilitates end-to-end training of the GNN. However, this approach can be both memory and computationally expensive as it requires learning a dense assignment matrix.

Another approach to graph pooling is to discard a fixed proportion of nodes based on some criterion. For example,

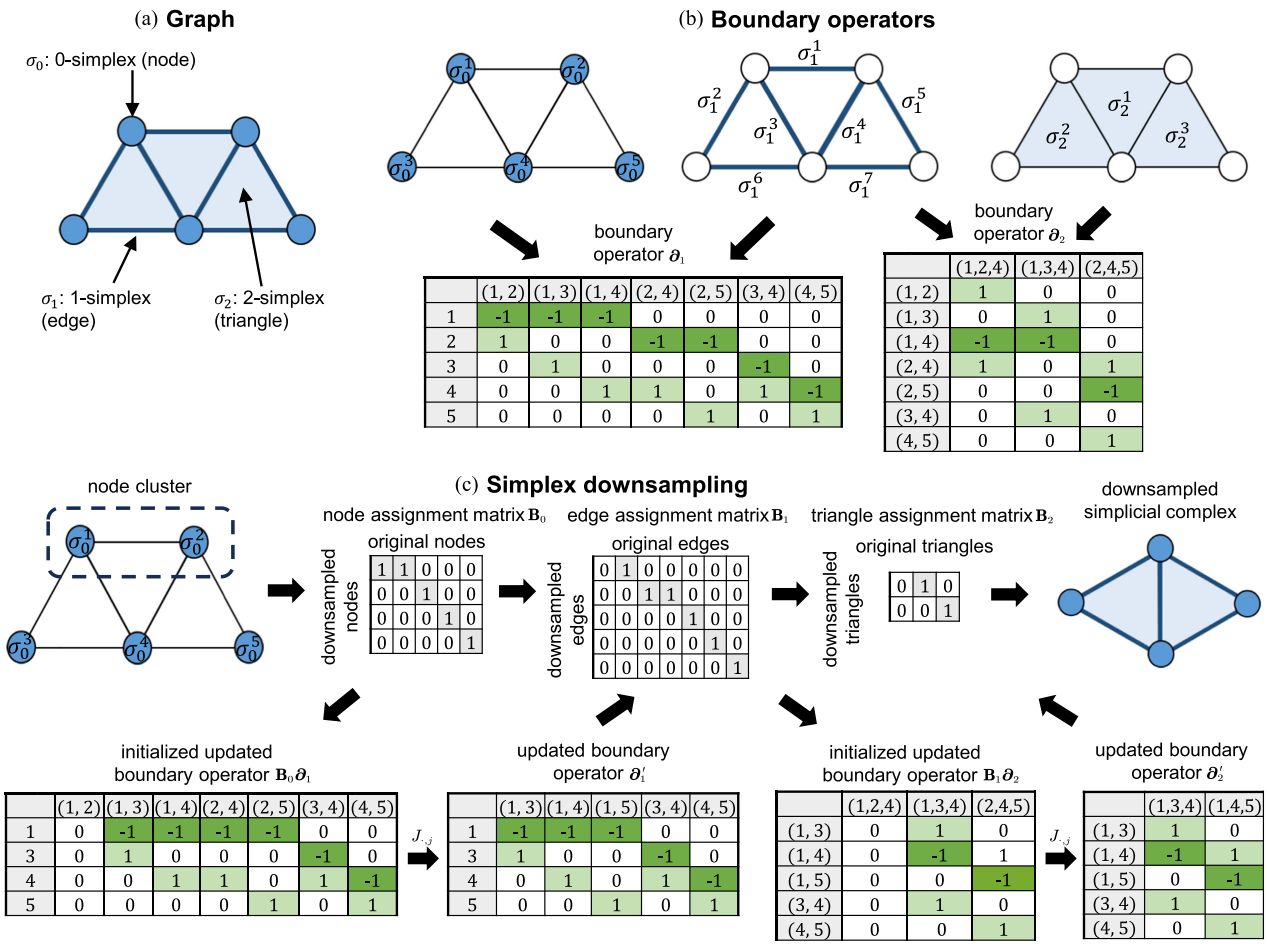



Fig. 1. Illustration of simplices, boundary operators, and simplex downsampling. Panel (a) illustrates a graph with the 0-, 1-, 2-simplices, while Panel (b) displays its corresponding 1-st and 2-nd boundary operators. Panel (c) demonstrates an example of the simplex downsampling and the update of the corresponding boundary operators. In the simplex assignment matrix, each row corresponds to the $k$-simplex of the downsampled graph, and each column corresponds to the $k$-simplex of the original graph.

TopKPool applies a linear layer to project node features into scalars and then ranks them, selecting the $k$ nodes with the largest values [40]. Similarly, topology-aware pooling (TAPool) generates scores from both local and global voting and selects nodes based on these scores [41].

Despite the considerable progress in graph pooling, a notable gap remains in applying these techniques to $k$-simplices, which are generalizations of nodes (0-simplex), edges (1-simplex), triangles (2-simplex), and so on. These higher-order structures contain rich information about the topological properties of the graph, which is underutilized in current graph pooling methods. The challenge of effectively pooling such complex, multi-dimensional structures is our research to generalizing GNNs on $k$-simplices.

## III. Methods

In this section, we provide a comprehensive exploration of the mathematical foundation that underlies Hodge-Laplacian spectral filters (HL-filters) and simplicial attention pooling (SAP) operators when applied to $k$-simplices, as well as multi-simplicial interaction (MSI) across multi-dimensional simplices. Within the overarching framework of HL-HGAT, we shed light on the HL-filters' association with $k$-simplices through the utilization of the $k$-th HL operator, accompanied by an in-depth discussion of their polynomial approximations. Subsequently, we introduce the simplicial projection (SP) operator, designed to facilitate signal transformation between simplices of different dimensions, while also introducing the concept of MSI to enable the learning of signal representations spanning across multi-dimensional simplices. This discussion is further enriched with the introduction of simplicial attention pooling, a novel approach that leverages attention mechanisms to distill features, both intrinsically within a specific dimensional simplex and interdimensionally across a multitude of simplices.

### A. Spectral Filters Via the Hodge-Laplacian Operator (HL-Filters)

Let's denote a graph by $\mathcal{G}$ as a simplicial complex comprising nodes, edges, triangles, and $k$-simplices, as shown in Fig. 1(a).

A node on $\mathcal{G}$ corresponds to a 0-simplex, denoted as $\sigma_0$; an edge on $\mathcal{G}$, connecting two nodes, represents a 1-simplex, denoted as $\sigma_1$; a $k$-simplex $\sigma_k$ is defined as a $k$-th polytope with $(k+1)$ nodes. In this study, we utilize the $k$-th boundary operator $\boldsymbol{\partial}_k$ to characterize the relationship between the $(k-1)$-simplex and $k$-simplex on $\mathcal{G}$. For $k \in \mathbb{Z}^+$, this $k$-th boundary operator is expressed in a matrix form as

$$[\boldsymbol{\partial}_k]_{ij} = \begin{cases} 1\,, & \text{if } \sigma_{k-1}^i \text{ is positively oriented w.r.t. } \sigma_k^j\,; \\ -1\,, & \text{if } \sigma_{k-1}^i \text{ is negatively oriented w.r.t. } \sigma_k^j\,; \\ 0\,, & \text{otherwise}\,, \end{cases} \tag{1}$$

The element of $\boldsymbol{\partial}_k$ in the $i$-th row and $j$-th column indicates whether the $i$-th $(k-1)$-simplex $\sigma_{k-1}^i$ belongs to the $j$-th $k$-simplex $\sigma_k^j$. An illustrative example is given in Fig. 1(b) to present the construction of the boundary operators.

Specifically, if we assume that a graph $\mathcal{G}$ has $n_0$ nodes, it is worth noting that the element $[\boldsymbol{\partial}_1]_{ij}$ encodes whether the $i$-th and $j$-th nodes (i.e., 0-simplices) are connected to form an edge (i.e., a 1-simplex) [44]. In graph theory [45], the 1-st boundary operator $\boldsymbol{\partial}_1$ is equivalent to a traditional incidence matrix with size of $n_0 \times \frac{n_0 \times (n_0-1)}{2}$, where nodes are indexed over rows and edges are indexed over columns. When $[\boldsymbol{\partial}_1]_{ij}$ is equal to $\pm 1$, the $i$-th and $j$-th nodes form an edge that connects these two nodes. Similarly, the 2-nd boundary operator $\boldsymbol{\partial}_2$ encodes how 1-simplices (i.e., nodes) are connected to form a 2-simplex (i.e., a triangle).

We now design spectral filters via the $k$-th Hodge-Laplacian (HL) operator on the $k$-simplices. The $k$-th HL operator is defined as

$$\boldsymbol{\mathcal{L}}_k = \boldsymbol{\partial}_{k+1}\boldsymbol{\partial}_{k+1}^\top + \boldsymbol{\partial}_k^\top \boldsymbol{\partial}_k\,. \tag{2}$$

Note that a special case of $k = 0$ is the 0-th HL operator over nodes, that is,

$$\boldsymbol{\mathcal{L}}_0 = \boldsymbol{\partial}_1 \boldsymbol{\partial}_1^\top\,. \tag{3}$$

This special case is equivalent to the standard graph Laplacian operator, i.e., $\boldsymbol{\mathcal{L}}_0 \equiv \Delta$. By solving the eigensystem of the $k$-th HL operator, we can obtain a set of orthonormal bases $\{\psi_k^0, \psi_k^1, \psi_k^2, \cdots\}$ on the $k$-simplices, that is,

$$\boldsymbol{\mathcal{L}}_k \psi_k^j = \lambda_k^j \psi_k^j\,, \tag{4}$$

where $\lambda_k^j$ represents the $j^{th}$ eigenvalue corresponding to the $j^{th}$ eigenvector $\psi_k^j$, and $\boldsymbol{\lambda}_k = [\lambda_k^0, \lambda_k^1, \lambda_k^2, \cdots]^\top$ is the spectrum of the matrix $\boldsymbol{\mathcal{L}}_k$.

Now we consider an HL-filter $\boldsymbol{h}(\cdot)$ with spectrum as

$$\boldsymbol{h}(\cdot,\cdot) = \sum_{j=0}^{\infty} \boldsymbol{h}(\lambda_k^j)\psi_k^j(\cdot)\psi_k^j(\cdot). \tag{5}$$

A generic form of convolution of a signal $\boldsymbol{x}$ with a HL-filter $\boldsymbol{h}(\cdot)$ on the heterogeneous graph $\mathcal{G}$ can be defined by firstly transforming the signal to the spectral domain and then filtering it based on the corresponding spectrum of $\boldsymbol{h}$ [26], [46], that is,

$$\boldsymbol{x}'(\cdot) = \boldsymbol{h} * \boldsymbol{x}(\cdot) = \sum_{j=0}^{\infty} \boldsymbol{h}(\lambda_k^j) c_k^j \psi_k^j(\cdot)\,, \tag{6}$$

where $\boldsymbol{x}(\cdot) = \sum_{j=0}^{\infty} c_k^j \psi_k^j(\cdot)$. Specifically, the signal $\boldsymbol{x}$ is defined on the nodes of the graph $\mathcal{G}$ when $k = 0$.

### B. Polynomial Approximation of HL-Filters

The shape of a HL-filter (e.g., $\boldsymbol{h}(\cdot)$ in (6)) determines how many simplices are aggregated in the filter process. When $\mathcal{G}$ is large, the computation of $\lambda_k^j$ and $\psi_k^j$ is intensive. To mitigate the costly computation, in this subsection, we propose to approximate a HL-filter by the expansion of polynomials, that is, $\{T_p \mid p = 0, 1, 2, \ldots, P-1\}$, such that

$$\boldsymbol{h}(\boldsymbol{\lambda}_k) = \sum_{p=0}^{P-1} \theta_p T_p(\boldsymbol{\lambda}_k)\,, \tag{7}$$

where $\theta_p$ is the expansion coefficient associated with the $p^{\text{th}}$-order polynomial. The coefficients $\{\theta_p \mid p = 0, 1, \ldots, P-1\}$ are the parameters of the HL-filter for optimization.

We can rewrite the convolution in (6) as:

$$\boldsymbol{x}'(\cdot) = \boldsymbol{h} * \boldsymbol{x}(\cdot) = \sum_{p=0}^{P-1} \theta_p T_p(\boldsymbol{\mathcal{L}}_k)\boldsymbol{x}(\cdot)\,, \tag{8}$$

where $T_p$ can be any polynomial that is represented by a recursive equation. In this study, we select the Laguerre polynomial [47], [48], which can be computed from a recurrence relation, that is,

$$T_{p+1}(\boldsymbol{\lambda}_k) = \frac{(2p+1-\boldsymbol{\lambda}_k)T_p(\boldsymbol{\lambda}_k) - pT_{p-1}(\boldsymbol{\lambda}_k)}{p+1}\,, \tag{9}$$

with $T_0(\boldsymbol{\lambda}_k) = 1$ and $T_1(\boldsymbol{\lambda}_k) = 1 - \boldsymbol{\lambda}_k$.

Fig. 2(a) and (b) provide clear illustrations of filtered node and edge pulse signals, showcasing the results of approximating Laguerre polynomials of different orders, such as the 1st, 2nd, and 3rd-orders. This visualization effectively demonstrates that the spatial localization property of the HL-filters is determined by the order of Laguerre polynomials. Notably, alternative polynomials with a recurrence relation can also be employed for approximating the HL-filter as shown in Huang et al. [26].

### C. Simplicial Projection Operator

Signals on simplices of different dimensions may be relevant to each other. However, the HL-filters described above work exclusively on one specific-dimensional simplex, in other words, they do not operate across multiple dimensions. To address this problem, we define two projection operators, that is, $\mathbf{T}_{\mathrm{k}\downarrow}$ and $\mathbf{T}_{\uparrow\mathrm{k}}$. The first projection operator $\mathbf{T}_{\mathrm{k}\downarrow}$ is supposed to map signals from the $k$-simplices to the $(k-1)$-simplices. The design of this operator relies on the topological relationship between the $k$- and $(k-1)$-simplices of a graph $\mathcal{G}$, and this relationship is captured by the $k$-th boundary operator of $\mathcal{G}$. As a result, $\mathbf{T}_{\mathrm{k}\downarrow}$ is expressed as

$$[\mathbf{T}_{\mathrm{k}\downarrow}]_{ij} = \left|[\boldsymbol{\partial}_k]_{ij}\right| = \begin{cases} 1\,, & \text{if } \sigma_{k-1}^i \text{ is incident with } \sigma_k^j\,; \\ 0\,, & \text{otherwise}\,, \end{cases} \tag{10}$$

where the notation of $|\cdot|$ represents the absolute value function. Similarly, the second projection operator $\mathbf{T}_{\uparrow\mathrm{k}}$ is supposed to map signals from the $(k-1)$-simplices to the $k$-simplices, and it can

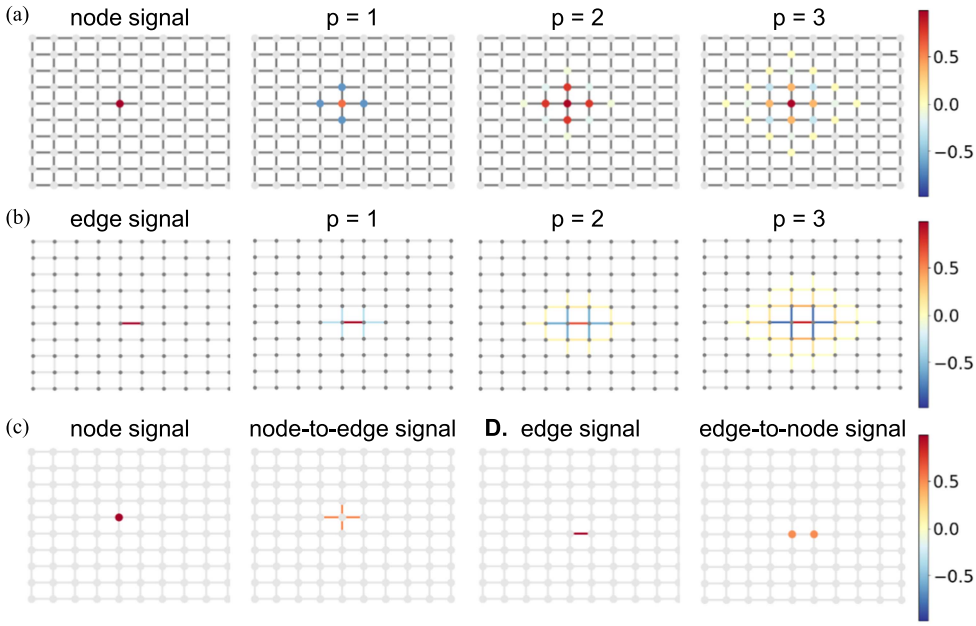



Fig. 2. HL-filters and simplicial projection operators. (a–b) The series of panels, from left to right, display a pulse signal associated with either a node or an edge, followed by the corresponding signals filtered through HL-filters employing approximations based on the 1st, 2nd, and 3rd-order Laguerre polynomials. (c–d) The left and right panels respectively depict a pulse signal defined on either a node or an edge and their resulting signals after projection through a simplicial projection operator.

be rewritten as the transpose of $\mathbf{T}_{\mathrm{k}\downarrow}$, that is,

$$\mathbf{T}_{\uparrow\mathrm{k}} = \mathbf{T}_{\mathrm{k}\downarrow}^{\top}. \tag{11}$$

The pair of these two projection operators will serve as the fundamental operators to facilitate the interaction of signals across multi-dimensional simplices.

We now define a generic projection operator to map signals from the $k_2$-simplices to the $k_1$-simplices, where $k_1 < k_2$. This mapping is performed iteratively by projecting signals: first from $k_2$ to $(k_2 - 1)$, then from $(k_2 - 1)$ to $(k_2 - 2)$, and so on. Then this iterative projection could be denoted by

$$\mathbf{T}_{\mathrm{k}_2\downarrow\mathrm{k}_1} = \mathbf{T}_{\mathrm{k}_1+1\downarrow}\cdots\mathbf{T}_{\mathrm{k}_2-1\downarrow}\mathbf{T}_{\mathrm{k}_2\downarrow} = \prod_{k=k_1+1}^{k_2}\mathbf{T}_{\mathrm{k}\downarrow}. \tag{12}$$

Conversely, to project signals from $k_1$-simplices to $k_2$-th ones, we use the operator

$$\mathbf{T}_{\mathrm{k}_1\uparrow\mathrm{k}_2} = \mathbf{T}_{\uparrow\mathrm{k}_2}\cdots\mathbf{T}_{\uparrow\mathrm{k}_1+2}\mathbf{T}_{\uparrow\mathrm{k}_1+1} = \prod_{k=k_2}^{k_1+1}\mathbf{T}_{\uparrow\mathrm{k}} = \mathbf{T}_{\mathrm{k}_2\downarrow\mathrm{k}_1}^{\top}. \tag{13}$$

Fig. 2(c,d) provide a practical simulation of projecting a pulse signal defined on a node to its neighboring edges and a pulse signal defined on an edge to its neighboring nodes, respectively. The proposed simplicial projection operator can map any simplex's signal with any dimensional neighboring simplices' signals, which enables us to learn signal interactions and cross-attentions across multiple dimensional simplices introduced in Section III-D.

### D. Multi-Simplicial Interaction (MSI)

Based on the simplicial projection operator, we define a multi-simplicial interaction layer. This layer is tailored to learn signal representations on the $k$-simplices, incorporating influences from signals on other dimensional simplices. Assume we have a signal $\boldsymbol{x}$ defined on the $k$-simplices, which are derived from the HL-filters, as described in (8). In the context of a discrete framework, this signal can be represented as a matrix, denoted by $\boldsymbol{X}_k \in \mathbb{R}^{n_k\times d}$, where $n_k$ is the number of $k$-simplices and $d$ is the number of features. To capture the interaction and integration between $\boldsymbol{X}_{k_1}$ and $\boldsymbol{X}_{k_2}$ $(k_1 < k_2)$, we utilize two fully connected layers, with a ReLU activation layer followed by the first fully connected layer, which is written as

$$\tilde{\boldsymbol{X}}_{k_1} = \mathrm{ReLU}\left(\left(\boldsymbol{X}_{k_1}\|\mathbf{T}_{\mathrm{k}_2\downarrow\mathrm{k}_1}\boldsymbol{X}_{k_2}\right)\boldsymbol{W}'_{k_1}\right)\boldsymbol{W}_{k_1}, \tag{14a}$$

$$\tilde{\boldsymbol{X}}_{k_2} = \mathrm{ReLU}\left(\left(\boldsymbol{X}_{k_2}\|\mathbf{T}_{\mathrm{k}_1\uparrow\mathrm{k}_2}\boldsymbol{X}_{k_1}\right)\boldsymbol{W}'_{k_2}\right)\boldsymbol{W}_{k_2}, \tag{14b}$$

where $(\boldsymbol{X}_a\|\boldsymbol{X}_b)$ indicates the concatenation of $\boldsymbol{X}_a$ and $\boldsymbol{X}_b$, and both $\boldsymbol{W}'_k \in \mathbb{R}^{2d\times d}$ and $\boldsymbol{W}_k \in \mathbb{R}^{d\times d}$ denote weight matrices. Note that weight matrices are usually viewed as learnable parameters in the training.

### E. Simplicial Attention Pooling (SAP)

We introduce an innovative simplicial attention pooling technique, encompassing self- and cross-attention mechanisms (Fig. 3(c)), simplex downsampling (Fig. 3(b)), and attention-weighted pooling.

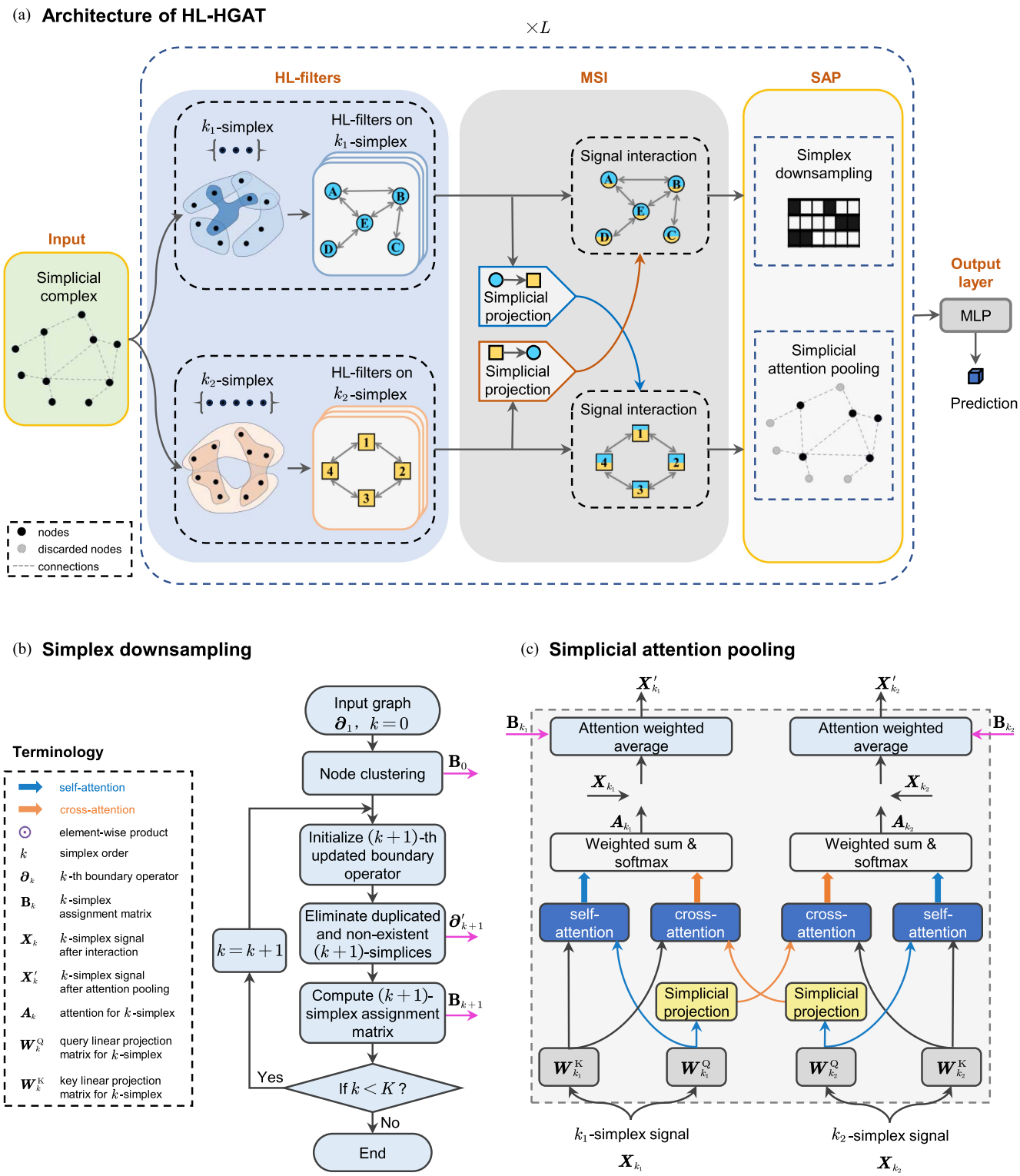



Fig. 3. HL-HGAT architecture. (a) Schematic representation of the Hodge-Laplacian Heterogeneous Graph Attention Network (HL-HGAT) architecture showcasing three key innovations: HL-filters, multi-simplicial interaction (MSI), and simplicial attention pooling (SAP). In each processing block, we initiate the workflow by applying HL-filters to signals from the $k_1$- and $k_2$-simplices from the preceding block. Subsequently, an MSI layer is employed to capture signal interactions between the $k_1$- and $k_2$-simplices. Following this, we implement an SAP layer, which involves updating the boundary operator and feature consolidation based on simplex attention. Finally, an output layer is designed for prediction. (b) Flow chart outlining the proposed simplex downsampling algorithm in Section III-D. We employ the Graclus clustering algorithm [49] to derive the node assignment matrix. This is followed by an iterative three-step process (depicted in Fig. 1(c)): 1) Initialization of the updated boundary operator for the $(k+1)$-th iteration; 2) Removal of non-existent $(k+1)$-simplices that contain nodes within the same node clusters and duplicated $(k+1)$-simplices; 3) Computation of the $(k+1)$-simplex assignment matrix using the updated boundary operator. (c) Schematic diagram illustrating the architecture of the Simplicial Attention Pooling (SAP). Within this framework, we compute self-attention and cross-attention for each simplex. The simplex signals are then modulated by attention mechanisms and pooled based on the assignment matrices.

Our self- and cross-attention mechanisms are designed to evaluate the significance of a signal within a specific-dimensional simplex while simultaneously recognizing its topological relevance across different-dimensional simplices. For self-attention, we adopt a conventional transformer [23], which calculates key and query weights to facilitate the learning of signal representations within specific-dimensional simplices.

Furthermore, by drawing upon the projection operators described in (12) and (13), we construct a transformer to assess the similarity between signals on $k_1$-th and $k_2$-simplices. As a result, the overall weights assigned to the signals on the $k_1$-th and $k_2$-simplices can be expressed as a combination of their respective learned weights from both self-attention and cross-attention mechanisms. These can be represented as:

$$\boldsymbol{A}_{k_1} = \operatorname{diag}\Bigg(S\Bigg(\alpha_{k_1}\frac{\boldsymbol{X}_{k_1}\boldsymbol{W}^{\mathrm{Q}}_{k_1}(\boldsymbol{X}_{k_1}\boldsymbol{W}^{\mathrm{K}}_{k_1})^{\top}}{\sqrt{d_k}} + (1-\alpha_{k_1})\frac{\mathbf{T}_{\mathrm{k}_2\downarrow\mathrm{k}_1}(\boldsymbol{X}_{k_2}\boldsymbol{W}^{\mathrm{Q}}_{k_2})(\boldsymbol{X}_{k_1}\boldsymbol{W}^{\mathrm{K}}_{k_1})^{\top}}{\sqrt{d_k}}\Bigg)\Bigg), \tag{15a}$$

$$\boldsymbol{A}_{k_2} = \operatorname{diag}\Bigg(S\Bigg(\alpha_{k_2}\frac{\boldsymbol{X}_{k_2}\boldsymbol{W}^{\mathrm{Q}}_{k_2}(\boldsymbol{X}_{k_2}\boldsymbol{W}^{\mathrm{K}}_{k_2})^{\top}}{\sqrt{d_k}} + (1-\alpha_{k_2})\frac{\mathbf{T}_{\mathrm{k}_1\uparrow\mathrm{k}_2}(\boldsymbol{X}_{k_1}\boldsymbol{W}^{\mathrm{Q}}_{k_1})(\boldsymbol{X}_{k_2}\boldsymbol{W}^{\mathrm{K}}_{k_2})^{\top}}{\sqrt{d_k}}\Bigg)\Bigg), \tag{15b}$$

where $\boldsymbol{W}^{\mathrm{Q}}_{k}$ and $\boldsymbol{W}^{\mathrm{K}}_{k}$ represent the query weight matrix and key weight matrix, both with dimensions of $d \times d_k$, respectively. Here, $S(\cdot)$ denotes a softmax function; $\alpha_k$ is a scalar determining the relative importance of self- and cross-attention; $\boldsymbol{A}_{k_1}$ and $\boldsymbol{A}_{k_2}$ are used as weights in the following pooling operation.

We now introduce a genetic pooling procedure for a simplicial complex denoted as $\mathcal{G}$, enabling simplex downsampling, boundary operator update, and attention-weighted pooling. The downsampled complex is represented as $\mathcal{G}'$, preserving simplices up to the $K$-th order. As shown in Fig. 1(c), our downsampling procedure initially clusters the nodes in $\mathcal{G}$ (i.e., 0-simplex), which can be achieved through existing node clustering algorithms. In this study, we utilize the Graclus multi-level clustering algorithm [49], grouping the nodes in $\mathcal{G}$ based on a local normalized cut. The nodes in the same cluster are merged into a single node in $\mathcal{G}'$ and its corresponding features are computed as the weighted average of the features defined on these nodes. These weights are determined by $\boldsymbol{A}_0$ as defined in (15).

For further downsampling of higher-dimensional simplices, we introduce a simplex assignment matrix denoted as $\mathbf{B}_k$. This matrix has rows corresponding to the $k$-simplex in $\mathcal{G}'$ and columns corresponding to the $k$-simplex in $\mathcal{G}$. Each element within $\mathbf{B}_k$ is binary, indicating whether or not the $k$-simplex in $\mathcal{G}'$ corresponds to one $k$-simplex in $\mathcal{G}$. Thus, $\mathbf{B}_k$ characterizes the relationship of the $k$-simplex between $\mathcal{G}$ and $\mathcal{G}'$.

To update the boundary operator $\boldsymbol{\partial}_k$, we denote the updated boundary operator as $\boldsymbol{\partial}'_k$ and define it as follows:

$$\boldsymbol{\partial}'_k = \mathbf{J}_{\cdot,j}\left(\mathbf{B}_{k-1}\boldsymbol{\partial}_k\right), \tag{16}$$

where $\mathbf{J}_{\cdot,j}(A)$ is an operator that removes the $j$-th column from a matrix $A$. (16) can be applied on any $k$-simplices by three steps: 1) initializing the $k$-th updated boundary operator $\boldsymbol{\partial}'_k$ to $\mathbf{B}_{k-1}\boldsymbol{\partial}_k$; 2) eliminating the $k$-simplices that contain nodes in the same node clusters; and 3) eliminating the replicated $k$-simplices. Fig. 1(c) illustrates an example of the simplex downsampling up to $2-$simplex. In this process, while removing the $k$-simplex, its features are aggregated into the simplex connecting to it through weighted averaging, where the weights are defined via attention mechanisms in (15). While $\mathbf{B}_0$ is computed by the node clustering algorithm, the higher-dimensional simplex assignment matrix $\mathbf{B}_k$ in (16) can be derived from the updated boundary operator iteratively. We calculate the element of $\mathbf{B}_k$ based on $\boldsymbol{\partial}'_k$, $\boldsymbol{\partial}_k$, and $\mathbf{B}_{k-1}$, that is,

$$[\mathbf{B}_k]_{ij} = \begin{cases} 1\,, & \text{if } \sum_{q=1}^{n'_{k-1}} \left|\left[\boldsymbol{\partial}'_k\right]_{qi}\right| \left|\left[\mathbf{B}_{k-1}\boldsymbol{\partial}_k\right]_{qj}\right| = k+1\,; \\ 0\,, & \text{otherwise}\,, \end{cases} \tag{17}$$

where $n'_{k-1}$ is the number of the $(k-1)$-simplices in the coarsened $\mathcal{G}'$. In sum, the proposed simplex downsampling algorithm can be easily implemented by first clustering nodes and computing $\mathbf{B}_0$. Subsequently, we update the boundary operator by (16) and then compute the simplex assignment matrix with (17) iteratively until $k$ reaches $K$.

### F. Architecture of HL-HGAT

HL-HGAT is an architecture characterized by a harmonious integration of three essential components: Hodge Laplacian spectral filters (HL-filters), multi-simplicial interaction (MSI), and simplicial attention pooling (SAP), each contributing significantly to its functionality.

Fig. 3(a) illustrates a specific design of the HL-HGAT architecture, which can be constructed with multiple blocks, each composed of these three components. Within each block, there are convolutional branches, each equipped with HL-filters meticulously designed to process signals on dedicated dimensional simplices. Subsequently, the MSI module plays a pivotal role in orchestrating the interaction of signals across distinct dimensional simplices, facilitating a more comprehensive and holistic understanding of the data. Finally, within the architecture, SAP serves a dual purpose: it efficiently reduces the spatial dimension of simplices while simultaneously performing information pooling, as illustrated in Fig. 3(b) and (c). It is worth noting that SAP may become particularly beneficial when dealing with large graph sizes.

HL-HGAT optimally leverages the combination of these three components, working in synergy within a multi-layered architecture. This approach is empowered to perform effective signal processing and feature extraction, making it a potent tool for a wide range of applications.

### G. Implementation

As depicted in Fig. 3, the HL-HGAT architecture comprises an output layer and multiple blocks, of which each combines

HL-filter layers, MSI, and SAP. To enhance the model's effectiveness, we incorporate positional encoding for simplices of individual dimensions, a practice commonly employed in existing graph transformers [22], [50]. Specifically, we take the Laplacian node positional encoding with sign flipping [22] as our node position encoder. Similarly, for edge positional encoding, we use the first eight eigenvectors of the 1-st HL operator. This approach can be readily extended to accommodate positional encoding for simplices of any order.

HL-HGAT relies on several essential model parameters, including: 1) the number of blocks, 2) the number of convolutional layers, 3) the number of the HL-filters per convolutional layer, 4) the polynomial order for the HL-filter approximation, 5) the size of query and key matrices in the SAP layer, 6) the number of fully connected layers in the output layer, 7) the number of points in the fully connected layers, and 8) a dropout rate. For consistency across all experiments in this study, we have set the dropout rate for each convolutional layer and fully connected layer to be 0.25. Activation functions are chosen from either a standard ReLU or a Leaky ReLU with a leak rate of 0.1. Furthermore, the dimension of the query and key matrices in the SAP layer is consistently configured as $32 \times 32$. The remaining model parameters are determined through a greedy search specific to each individual application, detailed in Table 1 of the Supplementary Material.

All experiments adhere to a consistent set of training parameters, which includes an initial learning rate of 0.001 and a weight decay parameter of 0.001. We use the ADAM optimizer [51] with a mini-batch size of 1,000 for the ZINC dataset and 64 for the other datasets, although we may reduce the mini-batch size to 32 in cases where memory demands exceed GPU capacity.

HL-HGAT is implemented using Python version 3.9.13 and relies on the PyTorch 1.12.1 framework [52] along with the PyTorch Geometric 2.1.0 library [53]. The execution of HL-HGAT is carried out on an NVIDIA Tesla V100SXM2 GPU boasting 32 GB of RAM.

## IV. Experiments

In the following, we demonstrate the use of the proposed HL-HGAT in NP-hard combinatorial optimization problems, graph classification, and multi-label tasks, as well as graph regression tasks. To illustrate the versatility of HL-HGAT, we conduct experiments using established open datasets across various domains, including logistics, computer vision, biology, chemistry, and neuroscience. These demonstrations showcase the adaptability and effectiveness of HL-HGAT in diverse real-world applications. We compare our HL-HGAT with baseline models, including GCN [3], GAT [4], GatedGCN [21], GPS [22], dGCN [19], BrainGNN [20], Hypergraph NN [28], and SAT [34]. The setup of these baseline models for each experiment below is listed in Table 2 of the Supplementary Material.

### A. Traveling Salesman Problem

The Traveling Salesman Problem (TSP) represents a classic NP-hard challenge within combinatorial optimization. It revolves around determining the most efficient route to visit all cities exactly once on a map, given a list of cities and the distances between each pair of them. The importance of solving the TSP lies in its wide array of practical applications, such as route planning for logistics and delivery services [54]. The TSP also serves as a fundamental model for more complex optimization problems, making advancements in its solution methods applicable to a broader range of challenges [55].

In this study, we model each map as a graph, with cities represented as nodes and the connections between them as edges. A total of 12,000 maps[1] were employed in this research, each featuring a varying number of nodes, ranging from 50 to 500 nodes. Each node is connected to 25 other nodes through edges. The HL-HGAT approach tackles the TSP by framing it as an edge classification task. This is achieved through the minimization of a focal loss [56], which effectively quantifies the weighted entropy of edge probabilities. To estimate the probability of each edge within the shortest path, the HL-HGAT architecture omits the SAP component. Input features encompass the coordinates of cities assigned to nodes and the distances between all pairs of cities assigned to edges. We meticulously follow the dataset-splitting strategy outlined in the benchmarking paper by Dwivedi et al. [50], which includes 10,000 training graphs, 1,000 validation graphs, and 1,000 test graphs.

Fig. 4 showcases the visualization of node and edge features every two layers, illustrating how the proposed HL-filters effectively eliminate irrelevant edge signals. As shown in Fig. 1 of the Supplementary Material, we present a range of TSP examples featuring varying numbers of cities and their optimized solutions generated by HL-HGAT, along with comparisons to several state-of-the-art GNN models, including GCN [3], GAT [4], GatedGCN [21], and SAT [34]. Visually, HL-HGAT offers solutions that closely align with the ground truth, outperforming the selected baseline models. This phenomenon can also be observed in the feature space learned from each method (i.e., Fig. 5(a)), suggesting a clear separation of edges belonging to the shortest path and edges not belonging to the path.

Quantitatively, Fig. 6(a) illustrates the edge classification accuracy evaluated by F1 score. Two-sample $t$-tests show that the F1 score from HL-HGAT is significantly greater than GCN, GAT, GatedGCN, and SAT ($p < 0.001$), suggesting the superior performance of the HL-HGAT over the state-of-art GNN methods.

### B. Image Classification

Image classification is a typical task in computer vision, with applications in numerous areas including medical imaging, surveillance, autonomous vehicles, etc [58]. The HL-HGAT approach treats classical natural image classification problems as graph multi-class classification tasks.

We employ the CIFAR10 dataset of RGB images (with 60,000 graphs and 10 distinct classes), which is a well-known dataset in the field of computer vision [59]. To facilitate graph-based representation, each superpixel, defined as a compact and contiguous region in images showcasing uniform intensity characteristics, is treated as a node [60], and they are connected to

[1] https://www.math.uwaterloo.ca/tsp/index.html

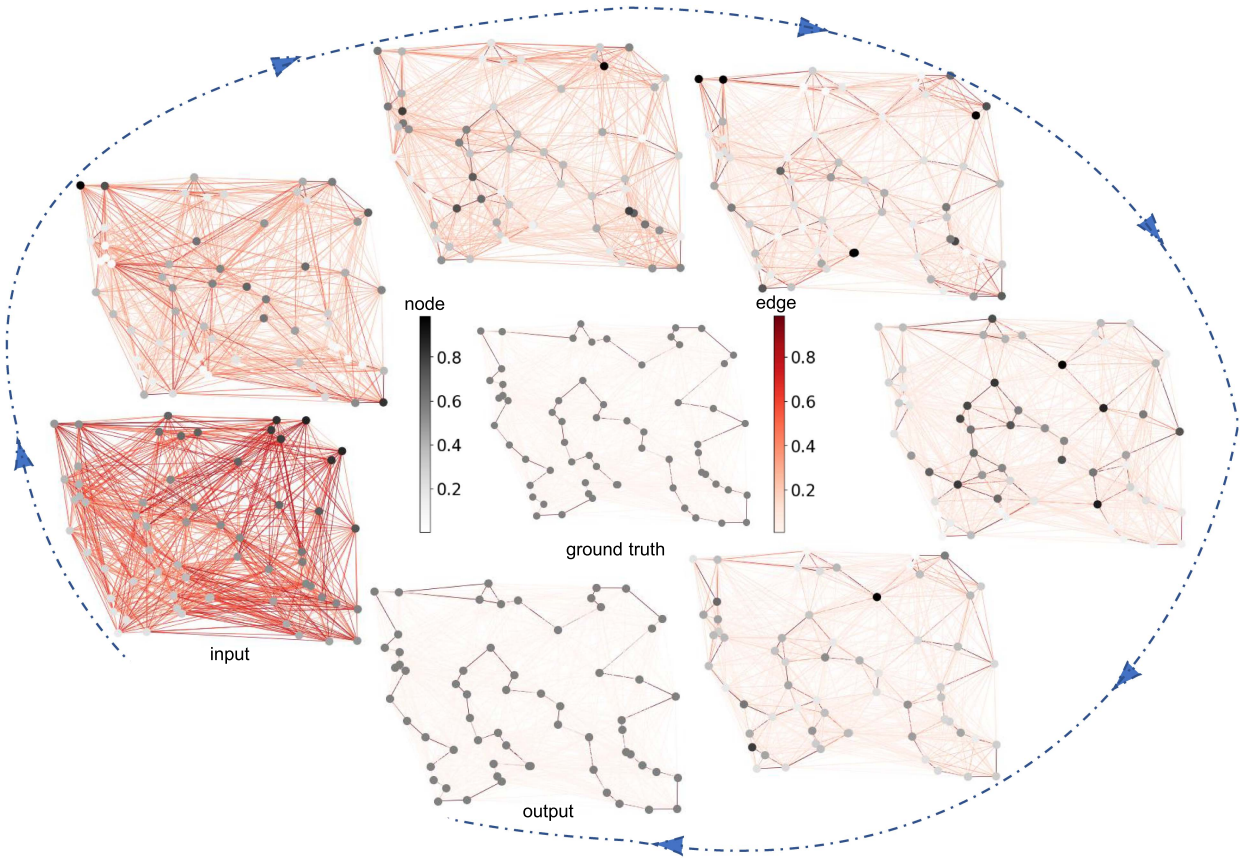



Fig. 4. Traveling Salesman Problem (TSP). Node and edge features at every two HL-filter layers are visualized. The ground truth is positioned at the center of the figure. Node features are represented in grey, while edge features are highlighted in red.

their eight nearest neighbors based on the Euclidean distance. On average, each graph in the dataset consists of 117 nodes. This forms a graph where superpixels are nodes, and edges connect neighboring superpixels. The nodes incorporate RGB values, and edges are established based on the absolute differences and Euclidean distances between corresponding RGB color values. These signal attributes enable the model to effectively capture both color information and spatial relationships among the superpixels within an image. In this experiment, we follow the dataset splitting strategy as described in Dwivedi et al. [50], which involves dividing the dataset into 45,000 training graphs, 5,000 validation graphs, and 10,000 test graphs. Notably, all three key components are integrated into the architecture of HL-HGAT for this specific application.

Fig. 5(b) illustrates the two-dimensional embedding of the features obtained through GCN [3], GAT [4], GatedGCN [21], GPS [22], SAT [34], and our HL-HGAT. Each point in this figure corresponds to an individual image, with color coding indicating its respective class. Notably, HL-HGAT achieves a more distinct separation of the 10 classes when compared to the other methods.

Quantitatively, Fig. 6(b) presents classification accuracy, which measures the proportion of correctly classified images relative to the total dataset size. The results of two-sample $t$-tests reveal that the accuracy achieved by HL-HGAT significantly surpasses those of other GNN models (such as GCN, GAT, GatedGCN, and SAT) as well as the graph transformer model (GPS) (all $p$-values $< 0.05$). These results suggest the effectiveness of HL-HGAT in computer vision tasks, establishing its superiority over the state-of-the-art GNN and graph transformer models.

### C. Biology and Chemistry

The study employs two widely-used molecular datasets, namely ZINC [61] and Peptide-func [62], to demonstrate the use of HL-HGAT in the fields of biology and chemistry.

The Peptide-func dataset with 15,535 graphs is specifically tailored for predicting peptide functions that are crucial for drug discovery, biological signaling, functional genomic discovery, etc [63]. The study considers this task as a multi-label graph classification problem that assigns each peptide with multiple functions, such as antibacterial, antiviral, intercellular communication, and more. Peptides, which are short chains of amino acids, are derived from SATPdb, a reference database [62]. In contrast to conventional approaches where amino acids typically serve as nodes in peptide graphs [64], this study employs heavy atoms as nodes, effectively expanding the size of the graph. On average, each graph in the dataset consists of 151 nodes. The edges within these graphs represent the bonds between these heavy atoms.

Similarly, ZINC is a curated repository of chemical compounds for virtual screenings of drug discovery, structural

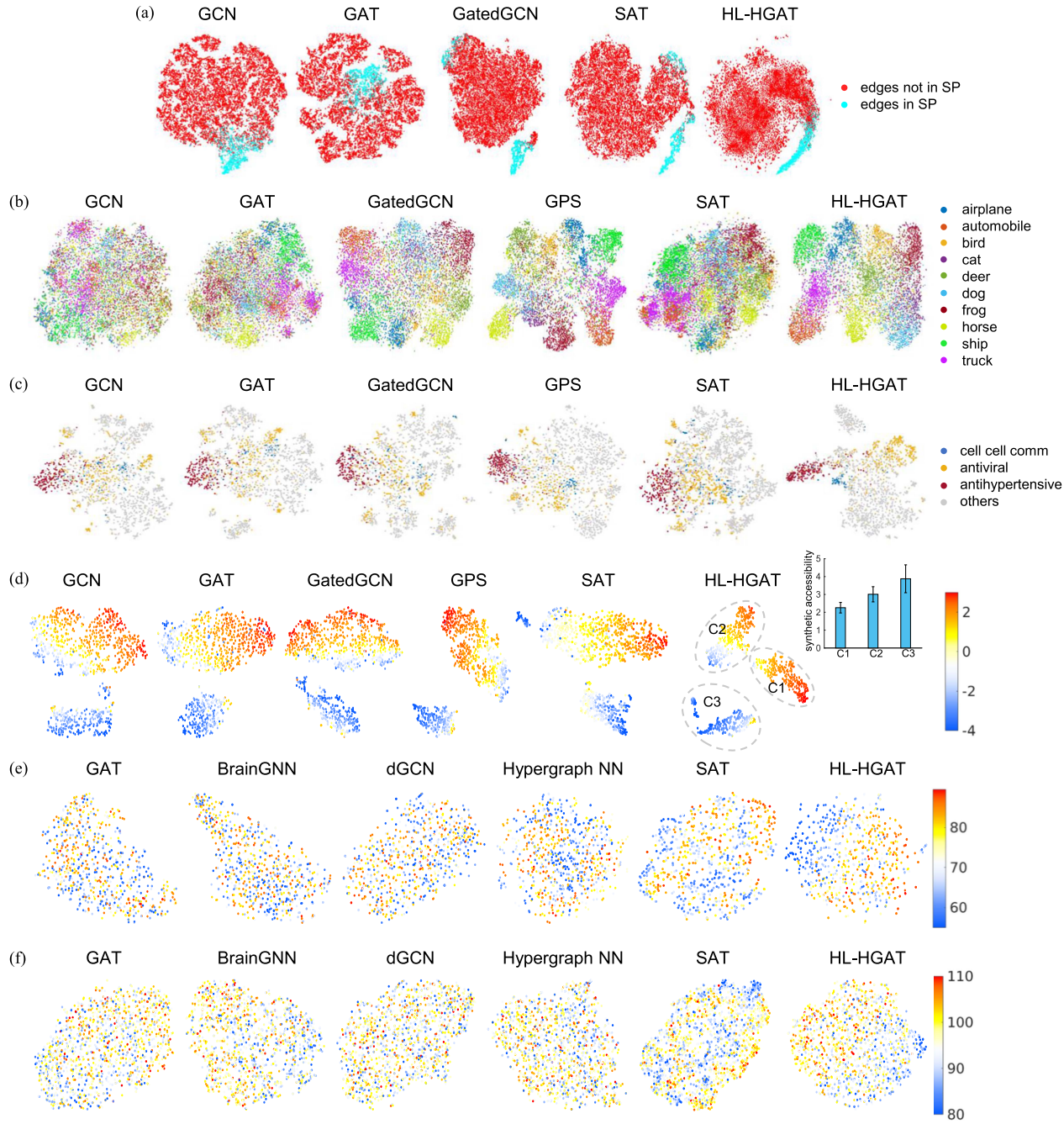



Fig. 5. Feature space. The feature space learned from each GNN method is projected into a two-dimensional space using t-distributed stochastic neighbor embedding (t-SNE) for visualization purposes. (a) In the feature space of the Traveling Salesman Problem, red points indicate edges that do not belong to the shortest path, while cyan points represent edges that are part of the shortest path. (b) The feature space of the CIFAR10 dataset visualizes the 10 classes of natural images. (c) In the feature space of the Peptide-func dataset, the colored points indicate peptides with corresponding functions, while grey points indicate peptides that do not have these functions. (d) In the feature space of the ZINC dataset, the points represent molecules colored by the corresponding constrained solubility. Additionally, the bar plot illustrates the average and standard deviation of synthetic accessibility across all molecules in the three clusters. (e–f) The feature spaces of brain images are color-coded by brain age and general intelligence. Each point in these plots represents an individual subject. Each column corresponds to the feature space obtained from various GNN models, including GCN [3], GAT [4], GatedGCN [21], GPS [22], dGCN [19], BrainGNN [20], Hypergraph NN [28], and SAT [34].

bioinformatics, and chemoinformatics research. We focus on a carefully selected subset of ZINC molecular graphs, consisting of 12,000 samples [61]. Our primary objective is to perform regression analysis on a molecular property referred to as "constrained solubility" that is crucial in determining drug efficacy and safety in pharmaceutical development [65]. Accurate solubility prediction enables the design of drugs with optimal absorption and bioavailability, ensuring effective treatment and minimizing potential side effects [66]. This task constitutes a typical graph regression problem, with our evaluation metric being mean absolute error (MAE). In each of these molecular graphs, the node features represent the types of heavy atoms

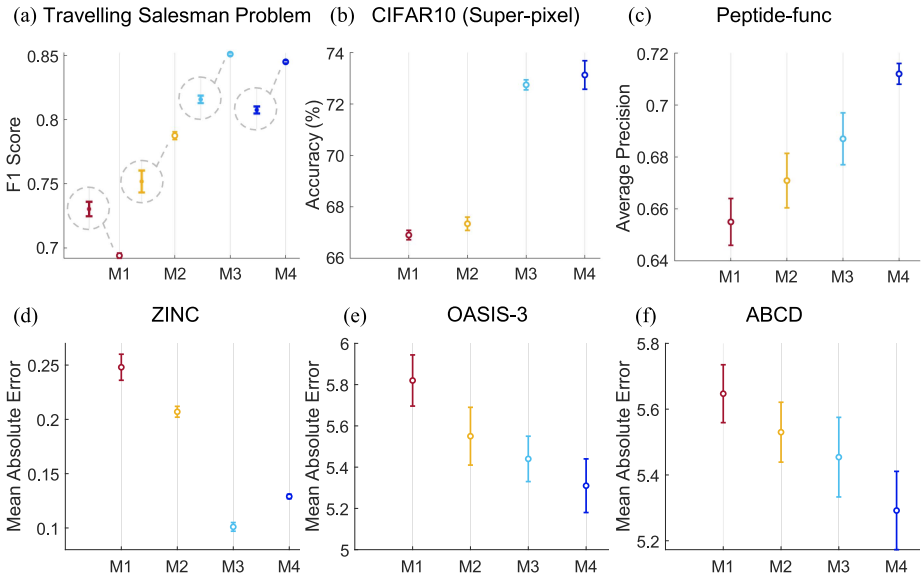



Fig. 6. Quantitative comparisons between HL-HGAT and state-of-the-art GNN methods across multiple datasets. Error bars illustrating quantitative metrics are presented in panels (a–f) for various datasets, including TSP, CIFAR10, Peptide-func, ZINC, OASIS-3, and ABCD. Notably, panel (a) zooms in on enlarged error bars enclosed within gray dashed circles. These quantitative metrics have been selected based on commonly reported measures in the existing literature [50], [57].

present, while the edge features characterize the types of bonds connecting these atoms. It is worth noting that the graphs in our dataset exhibit varying sizes, with node counts ranging from 9 to 37 and edge counts ranging from 16 to 84. We adhere to the dataset splitting strategy outlined in Dwivedi et al. [57], which involves a train-validation-test split ratio of 70%–15%–15%.

Fig. 5(c) provides a visual representation of the two-dimensional feature embeddings obtained through GCN [3], GAT [4], GatedGCN [21], GPS [22], SAT [34], and our HL-HGAT. In this plot, each point represents an individual peptide, with color coding indicating its corresponding peptide function.

Similarly, Fig. 5(d) provides a visual representation of the two-dimensional feature embeddings of the ZINC dataset obtained through GCN [3], GAT [4], GatedGCN [21], GPS [22], SAT [34], and our HL-HGAT. Each data point in this plot corresponds to an individual chemical compound, and its color code indicates its constrained solubility. Significantly, the embedding plot generated by HL-HGAT reveals three distinct clusters among these chemical compounds, whereas most of the other compared GNN and transformer models exhibit only two clusters. Further exploration of the three clusters identified by HL-HGAT reveals a connection to the synthetic accessibility score (SAS). The SAS evaluates the structural characteristics of drug-like molecules [67], as illustrated in the bar chart within the last panel of Fig. 5(d). This valuable insight into the SAS-related clustering dimension is a unique discovery attributed to HL-HGAT beyond the objective of this prediction task and is not observed in the other GNN models and GPS.

Quantitatively, the results of two-sample $t$-tests reveal that the average precision achieved for the Peptide-func dataset and the mean absolute error (MAE) obtained for the ZINC dataset through HL-HGAT demonstrate significantly superior classification or prediction accuracy when compared to GCN [3], GAT [4], GatedGCN [21], GPS [22], and SAT [34] (all $p$-values $< 0.01$).

### D. Brain Age and Intelligence

Two brain functional MRI datasets, derived from the Adolescent Brain Cognitive Development Study (ABCD)[2] [68] and Open Access Series of Imaging Studies (OASIS-3)[3] [69], are used in this study to predict intelligence [70] and brain age, respectively. The sample sizes of the ABCD and OASIS-3 datasets are 7,693 and 1,978, respectively. Brain age has been demonstrated to be a biomarker related to neurodegenerative diseases [71], while intelligence is a good predictor for academic outcomes in children.

For the purpose of constructing our brain functional organization graphs, we employ a total of 268 regions of interest (ROIs), as previously defined by Shen et al. (2017) [72]. These ROIs serve as the nodes within our graph, and we establish edges between them based on Pearson correlation calculated from the functional time series of each pair of ROIs. The sparsity level of our graph is set at 0.25. We employ the brain functional networks to predict brain age and intelligence. Our experimental setup closely follows the methodology outlined in Huang et al. [37].

Fig. 5(e) and (f) depict the two-dimensional feature embeddings obtained from GAT [4], BrainGNN [20], dGCN [19], Hypergraph NN [28], SAT [34], and our HL-HGAT. Notably, the selection of comparison graph models in this context differs from those mentioned earlier. BrainGNN [20], dGCN [19], and Hypergraph NN [28] are specialized techniques tailored for brain

[2]https://abcdstudy.org/
[3]https://www.oasis-brains.org

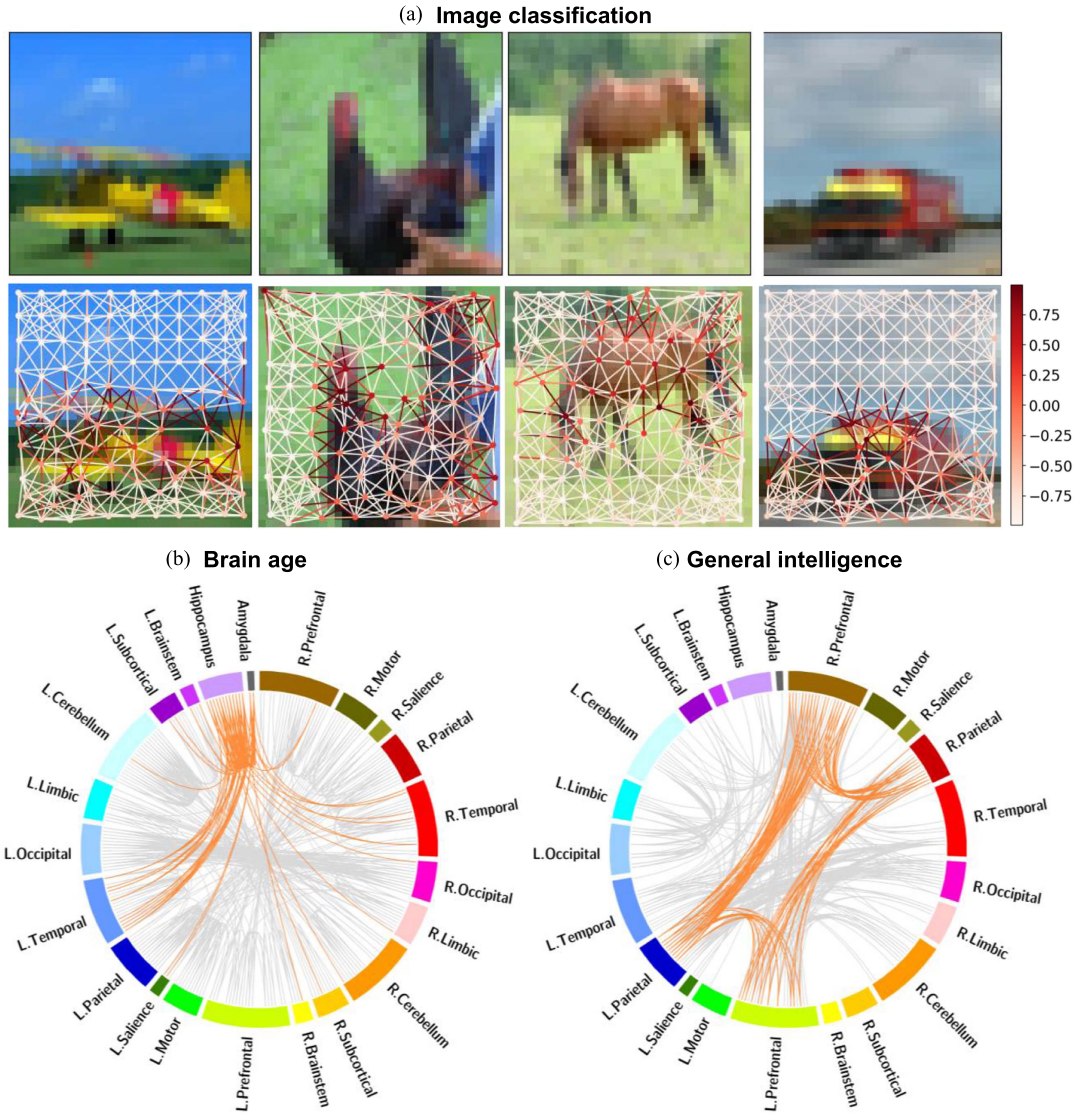



Fig. 7. Attention maps obtained from the SAP component in HL-HGAT for image classification. (a) Image Classification: Attention maps highlight nodes and edges with increased attention in red, emphasizing key features and boundaries crucial for image classification. (b) Brain Age Prediction: The attention maps visualize the functional connectivities (colored in orange) most contributing to age prediction. (c) General Intelligence prediction: Similarly, the functional connectivities colored in orange are most contributed to the prediction of general intelligence. Abbreviations: L, left; R, right.

functional networks, representing state-of-the-art approaches in the field of brain research. In these two sub-figures, each data point corresponds to an individual brain functional network, with color coding denoting their respective brain age and intelligence attributes.

The results, as depicted in Fig. 6(e) and (f), highlight the remarkable performance achieved by our HL-HGAT, positioning it at the forefront as one of the state-of-the-art solutions on both datasets when compared to the baseline models. For the OASIS-3 dataset, our HL-HGCNN attains state-of-the-art results in comparison to the baseline GNN models for node signals, such as GAT, BrainGNN, and dGCN (all $p$-values $< 0.01$). Additionally, our model surpasses the GNN model for edge signals, as evidenced by its superior performance relative to Hypergraph NN ($p = 0.02$). Similarly, in the case of the ABCD dataset, rigorous two-sample $t$-tests also reveal that the mean absolute errors (MAE) obtained through HL-HGAT significantly outshine those of GAT, BrainGNN, dGCN, and Hypergraph NN (all $p$-values $< 0.01$).

### E. SAP Provides Interpretable Attention Maps

The attention maps generated by the SAP component offer valuable interpretability to HL-HGAT, which is often considered opaque in existing GNN models. In this section, we demonstrate the interpretability of HL-HGAT through attention maps for the CIFAR10, ABCD, and OASIS datasets, originating from the SAP pooling layer.

When one SAP layer is implemented, attention is computed directly for each node and edge at the original spatial scale, making visualization and interpretation straightforward. However, when multiple SAP layers are used, we leverage the attention

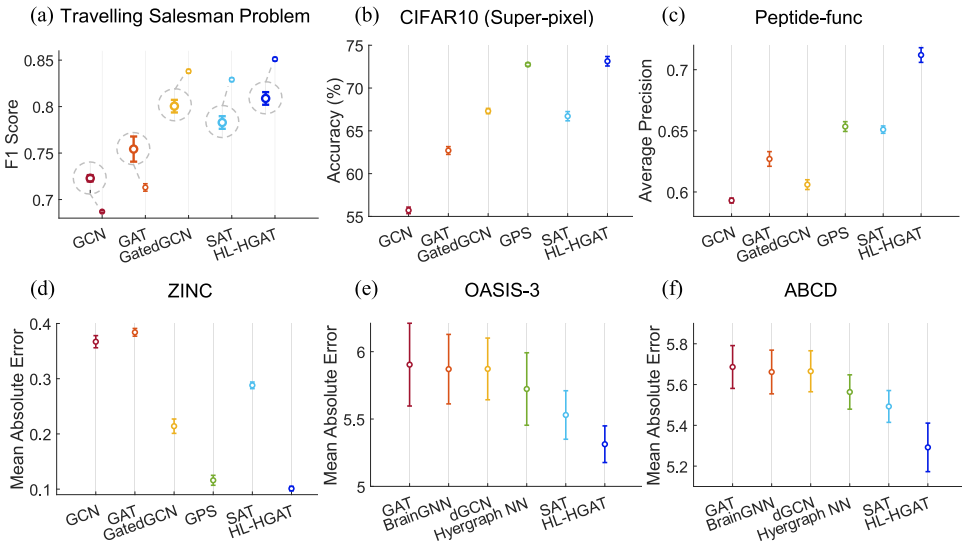



Fig. 8. Ablation Study. Panels (a–f) illustrate error bars for quantitative metrics across various datasets, including TSP, CIFAR10, Peptide-func, ZINC, OASIS-3, and ABCD. Panel (a) provides a close-up view of enlarged error bars enclosed within gray dashed circles. Abbreviations: M1 - HL-filters applied to node signals only; M2 - HL-filters applied to both node and edge signals; M3 - HL-filters applied to node and edge signals with MSI integration; M4 - The complete architecture encompassing all three key components as depicted in Fig. 3(a).

computed across all layers. By recording the simplex assignment matrices, we can map the attention from downsampled simplices to their corresponding simplices at the last spatial scale, thanks to the one-to-one correspondence between them (see Fig. 1(c)). This process is repeated until the attention from all SAP layers is projected back to the original spatial scale. Finally, we combine these attentions by multiplying them to produce the final value for visualization and interpretation.

For the image classification task, we anticipate that node attention will emphasize objects, while edge attention should accentuate object boundaries. To validate this notion, we randomly select four samples from the test set and overlay the attention onto superpixel images. Fig. 7(a) illustrates that SAP effectively identifies and highlights both objects and their boundaries across samples spanning various classes (more examples in Fig. 2 of the Supplementary Material). Additional examples of these attention maps are available in the supplementary materials.

In the context of brain age and general intelligence prediction, the attention maps derived from brain functional networks serve as informative biomarkers, shedding light on neural mechanisms linked to aging and intelligence. Fig. 7(b) and 7(c) depict functional connectivities corresponding to the top 5% group-averaged attention scores in the test set. Heightened attention is notably observed in the functional connectivity of the hippocampus and amygdala for brain age prediction, as these regions play pivotal roles in memory and emotional responses, both closely tied to aging [73], [74]. Furthermore, our attention maps unveil significant functional connectivities between prefrontal and parietal regions that contribute to the prediction of general intelligence, aligning with established research on neural activities in these regions [75], [76].

These findings underscore the interpretability of HL-HGAT, as it consistently provides meaningful attention maps across diverse applications, enriching our comprehension of complex data and underlying phenomena.

### F. Ablation Study

To comprehensively evaluate the significance of each component within HL-HGAT, we conduct an ablation study. We systematically increase the complexity of HL-HGAT by considering different architectural variations, each represented by a model denoted as follows: M1 (HL-filters applied to node signals only), M2 (HL-filters applied to both node and edge signals), M3 (HL-filters applied to node and edge signals with MSI integration), and M4 (the full architecture encompassing all three key components in Fig. 3(a)). It is worth noting that we maintain consistent hyperparameters, including the number of blocks, the number of HL-filters within each block, the order of Laguerre polynomials, and other learning parameters, across all experiments, as detailed in Section III-G.

Fig. 8 clearly demonstrates that increasing the complexity of HL-HGAT from M1 to M4 consistently results in improved classification accuracy for the CIFAR10 and Peptide-func datasets shown in Fig. 8(b) and (c), as well as enhanced prediction performance for brain age (i.e., OASIS-3, shown in Fig. 8(e)) and general intelligence (i.e., ABCD, shown in Fig. 8(f)). These findings underline the critical importance of HL-filters for handling heterogeneous signals, as well as the integration of MSI and SAP for effective multi-scale information processing. However, it is noteworthy that M4 does not consistently outperform M3 in the TSP and ZINC datasets. This discrepancy can be attributed to the significantly smaller graph size in the ZINC dataset, where the graph pooling operation may not necessarily enhance multi-scale processing. Furthermore, for the TSP problem, the nature of individual edge prediction may render pooling to be unnecessary.

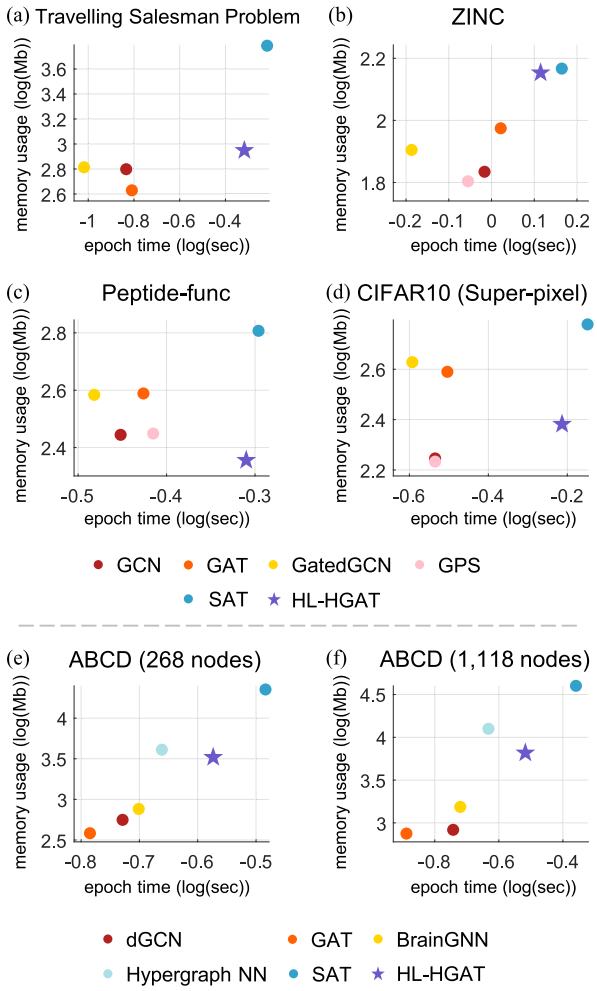



Fig. 9. Memory usage and computational time comparison between HL-HGAT and baseline models across all tasks. Warm-colored dots represent node-based GNNs or transformers, including GCN, GAT, GatedGCN, GPS, dGCN, and BrainGNN, while cool-colored dots represent edge- or simplex-based GNNs, including SAT, Hypergraph NN, and our approach.

Nonetheless, the SAP pooling operation stands out as a valuable addition, not only improving computational efficiency but also reducing memory usage. Table 3 of the Supplementary Material clearly indicates that our graph pooling operator significantly reduces both memory cost and computational time, with particular advantages evident for larger graphs. This efficiency enhancement is crucial for scaling HL-HGAT to handle larger and more intricate datasets.

### G. Scalability and Computational Complexity

To validate the scalability of HL-HGAT on large graphs, we conducted the general intelligence prediction task on the ABCD dataset, where the brain functional network was constructed using an atlas with 1,118 ROIs [77] instead of an atlas with 268 ROIs presented in the previous section. As shown in Fig. 3 of the Supplementary Material, the performance of HL-HGAT based on the brain functional networks of 1,118 nodes ($5.280 \pm 0.043$) is comparable with the result based on the brain functional networks of 268 nodes ($5.291 \pm 0.119$). In addition, HL-HGAT continued to achieve state-of-the-art results compared to the baseline models (all $p$-values $< 0.05$), demonstrating the superiority of HL-HGAT in modeling both medium and large brain functional networks.

As for the computational complexity, when only node and edge signals are considered, the computational complexity of HL-filters on nodes and edges is $O(n_1)$ and $O(n_1^2/n_0)$, respectively, where $n_0$ and $n_1$ represent the number of nodes and edges. The complexity of MSI and SAP modules is $O(n_1)$ for both nodes and edges. Real-world graphs, such as molecular and super-pixel graphs, are known for their sparsity, with the number of edges typically proportional to the number of nodes $n_1 = \Theta(n_0)$. In this scenario, the complexity of the proposed HL-HGAT is linear with respect to the number of nodes, i.e., $O(n_0)$. Fig. 9 illustrates the computational time and memory usage of HL-HGAT and baseline models across all five datasets, assessed during the testing. We omit the OASIS dataset because it has the same number of nodes and edges as the ABCD (268) dataset. When graph is sparse (sparsity level $< 0.1$), HL-HGAT exhibits lower memory usage compared to GAT, GatedGCN, and all simplex-based GNNs on the CIFAR10 dataset and shows the lowest memory usage among all node-based and simplex-based GNNs on the Peptide-func dataset. In contrast, when graph is less sparse (sparsity level $> 0.1$), simplex-based and edge-based GNNs generally consume significantly more memory than node-based GNNs on the OASIS-3 and ABCD datasets. Despite this, HL-HGAT still maintains the lowest memory usage among edge-based and simplex-based GNNs.

## V. Conclusion and Discussion

In this study, we introduce HL-HGAT, a versatile graph neural network designed to effectively model signals on nodes, edges, and higher-dimensional simplices within graphs. HL-HGAT harnesses the combined capabilities of HL-filters, MSI, and SAP to address complex problems involving graph-structured data. Our framework offers flexibility in combining these modules, providing a user-friendly workflow for a wide range of applications.

Our experiments, conducted on six diverse datasets spanning various domains—including NP-hard problems, graph classification, multi-label problems, and graph regression—consistently demonstrate HL-HGAT's superiority over state-of-the-art GNN and graph transformer models. This consistent performance highlights HL-HGAT as an advanced GNN model that excels in various scenarios. Furthermore, the ablation study underscores the significance of each component within HL-HGAT, emphasizing the importance of modeling heterogeneous signals, facilitating interactions across different-dimensional simplices, and leveraging the SAP layer for effective information aggregation. Additionally, the incorporation of the graph pooling operator significantly enhances the model's computational efficiency, making it well-suited for larger graph datasets. Furthermore, the interpretability provided by the attention maps generated by the SAP layer further enhances the model's utility. For instance, in the CIFAR10 dataset, the attention maps successfully capture objects and their boundaries, while in brain datasets, they

highlight task-related functional connectivity patterns. This interpretability holds promise for applications in medical imaging and computer vision.

However, our approach has a limitation in the form of the simplex downsampling method, which simplifies graphs solely based on their topology and may inadvertently remove significant simplices contributing to downstream tasks, and a more robust coarsening method that takes considers simplicial attention can be developed to address this limitation. Another limitation of HL-HGAT is its higher computational time and memory usage compared to conventional node-based GNNs, particularly on dense graphs, which is also a common limitation of simplex-based GNNs. One solution to mitigate this effect is to limit the number of neighboring simplices aggregated during convolution by defining the convolution operator within a specialized topological domain [78].

HL-HGAT is able to handle dynamic graphs, where the connections between nodes in the input graph vary over time, and this capability is realized by computing the boundary operator and the Hodge-Laplace operator. The application of HL-HGAT to dynamic graphs, including its potential usage in dynamic functional connectivity in neuroscience, will be the focus of our future studies.

In conclusion, HL-HGAT represents a powerful and flexible GNN model that is capable of solving diverse graph-based problems by effectively incorporating heterogeneous information, modeling complex interactions, and providing computational efficiency. Its interpretability further enhances its applicability across various fields, promising valuable contributions to graph-based machine learning applications.

## ACKNOWLEDGMENTS

We would like to extend our gratitude to Chenye Shen for his assistance with the atlas generation. Data used in the preparation of this article were obtained from the Adolescent Brain Cognitive Development (ABCD) Study (https://abcdstudy.org), held in the NIMH Data Archive (NDA).

## REFERENCES

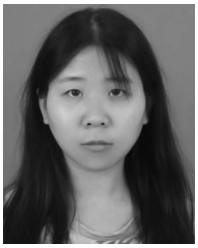
[1] J. Zhou et al., "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, 2020.
[2] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Jan. 2021.
[3] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representations*, 2017.
[4] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Representations*, 2018.
[5] P. Reiser et al., "Graph neural networks for materials science and chemistry," *Commun. Mater.*, vol. 3, no. 1, 2022, Art. no. 93.
[6] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1263–1272.
[7] S.-G. Huang, J. Xia, L. Xu, and A. Qiu, "Spatio-temporal directed acyclic graph learning with attention mechanisms on brain functional time series and connectivity," *Med. Image Anal.*, vol. 77, 2022, Art. no. 102370.
[8] H. Cui et al., "BrainGB: A benchmark for brain network analysis with graph neural networks," *IEEE Trans. Med. Imag.*, vol. 42, no. 2, pp. 493–506, Feb. 2023.
[9] H. Peng et al., "Spatial temporal incidence dynamic graph neural networks for traffic flow forecasting," *Inf. Sci.*, vol. 521, pp. 277–290, 2020.
[10] W. Jiang and J. Luo, "Graph neural network for traffic forecasting: A survey," *Expert Syst. Appl.*, vol. 207, 2022, Art. no. 117921.
[11] C. Huang et al., "Knowledge-aware coupled graph neural network for social recommendation," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, pp. 4115–4122.
[12] S. Wu, F. Sun, W. Zhang, X. Xie, and B. Cui, "Graph neural networks in recommender systems: A survey," *ACM Comput. Surv.*, vol. 55, no. 5, pp. 1–37, 2022.
[13] R. Hanocka, A. Hertz, N. Fish, R. Giryes, S. Fleishman, and D. Cohen-Or, "MeshCNN: A network with an edge," *ACM Trans. Graph.*, vol. 38, no. 4, pp. 1–12, 2019.
[14] S. Alonso-Monsalve et al., "Graph neural network for 3D classification of ambiguities and optical crosstalk in scintillator-based neutrino detectors," *Phys. Rev. D*, vol. 103, no. 3, 2021, Art. no. 032005.
[15] P. Pradhyumna and G. Shreya, "Graph neural network (GNN) in image and video understanding using deep learning for computer vision applications," in *Proc. Int. Conf. Electron. Sustain. Commun. Syst.*, 2021, pp. 1183–1189.
[16] J. Peng, C.-S. Kim, and C.-C. J. Kuo, "Technologies for 3D mesh compression: A survey," *J. Vis. Commun. Image Representation*, vol. 16, no. 6, pp. 688–733, 2005.
[17] M. Farazi, Z. Yang, W. Zhu, P. Qiu, and Y. Wang, "TetCNN: Convolutional neural networks on tetrahedral meshes," in *Proc. Int. Conf. Inf. Process. Med. Imag.*, 2023, pp. 303–315.
[18] Y. LeCun et al., "Handwritten digit recognition with a back-propagation network," in *Proc. Adv. Neural Inf. Process. Syst.*, D. Touretzky, Ed., San Mateo, CA, USA, 1989, vol. 2, pp. 396–404.
[19] K. Zhao, B. Duka, H. Xie, D. J. Oathes, V. Calhoun, and Y. Zhang, "A dynamic graph convolutional neural network framework reveals new insights into connectome dysfunctions in ADHD," *NeuroImage*, vol. 246, 2022, Art. no. 118774.
[20] X. Li et al., "BrainGNN: Interpretable brain graph neural network for fMRI analysis," *Med. Image Anal.*, vol. 74, 2021, Art. no. 102233.
[21] X. Bresson and T. Laurent, "Residual gated graph ConvNets," 2017, *arXiv:1711.07553*.
[22] L. Rampášek, M. Galkin, V. P. Dwivedi, A. T. Luu, G. Wolf, and D. Beaini, "Recipe for a general, powerful, scalable graph transformer," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, vol. 35, pp. 14501–14515.
[23] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, vol. 30.
[24] J. Bruna et al., "Spectral networks and deep locally connected networks on graphs," in *Proc. 2nd Int. Conf. Learn. Representations*, 2014.
[25] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, vol. 29.
[26] S.-G. Huang, M. K. Chung, A. Qiu, and A. D. N. Initiative, "Revisiting convolutional neural network on graphs with polynomial approximations of Laplace–Beltrami spectral filtering," *Neural Comput. Appl.*, vol. 33, pp. 13693–13704, 2021.
[27] X. Jiang, P. Ji, and S. Li, "CensNet: Convolution with edge-node switching in graph neural networks.," in *Proc. Int. Joint Conf. Artif. Intell.*, 2019, pp. 2656–2662.
[28] J. Jo, J. Baek, S. Lee, D. Kim, M. Kang, and S. J. Hwang, "Edge representation learning with hypergraphs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, vol. 34, pp. 7534–7546.
[29] O. T. Courtney and G. Bianconi, "Generalized network structures: The configuration model and the canonical ensemble of simplicial complexes," *Phys. Rev. E*, vol. 93, no. 6, 2016, Art. no. 062311.
[30] S. Barbarossa and S. Sardellitti, "Topological signal processing over simplicial complexes," *IEEE Trans. Signal Process.*, vol. 68, pp. 2992–3007, 2020.
[31] S. Ebli, M. Defferrard, and G. Spreemann, "Simplicial neural networks," 2020, *arXiv:2010.03633*.
[32] M. Yang, E. Isufi, and G. Leus, "Simplicial convolutional neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2022, pp. 8847–8851.
[33] C. Bodnar et al., "Weisfeiler and Lehman go topological: Message passing simplicial networks," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 1026–1037.
[34] C. W. J. Goh, C. Bodnar, and P. Lio, "Simplicial attention networks," in *Proc. ICLR Workshop Geometrical Topological Representation Learn.*, 2022.

[35] C. Battiloro et al., "Generalized simplicial attention neural networks," *IEEE Trans. Signal Inf. Process. Over Netw.*, 2024.
[36] F. Monti, O. Shchur, A. Bojchevski, O. Litany, S. Günnemann, and M. M. Bronstein, "Dual-primal graph convolutional networks," 2018, *arXiv:1806.00770*.
[37] J. Huang, M. K. Chung, and A. Qiu, "Heterogeneous graph convolutional neural network via Hodge-Laplacian for brain functional data," in *Proc. Int. Conf. Inf. Process. Med. Imag.*, 2023, pp. 278–290.
[38] D. Grattarola, D. Zambon, F. M. Bianchi, and C. Alippi, "Understanding pooling in graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 2, pp. 2708–2718, Feb. 2024.
[39] Z. Ying, J. You, C. Morris, X. Ren, W. Hamilton, and J. Leskovec, "Hierarchical graph representation learning with differentiable pooling," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, vol. 31.
[40] H. Gao and S. Ji, "Graph U-Nets," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 2083–2092.
[41] H. Gao, Y. Liu, and S. Ji, "Topology-aware graph pooling networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 12, pp. 4512–4518, Dec. 2021.
[42] D. M. Cinque, C. Battiloro, and P. Di Lorenzo, "Pooling strategies for simplicial convolutional networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2023, pp. 1–5.
[43] F. M. Bianchi, D. Grattarola, and C. Alippi, "Spectral clustering with graph neural networks for graph pooling," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 874–883.
[44] H. Edelsbrunner, D. Letscher, and A. Zomorodian, "Topological persistence and simplification," in *Proc. Annu. Symp. Foundations Comput. Sci.*, 2000, pp. 454–463.
[45] H. Lee, M. K. Chung, H. Kang, and D. S. Lee, "Hole detection in metabolic connectivity of Alzheimer's disease using K-Laplacian," in *Med. Image Computing Comput.Assist. Interv.*, Berlin, Germany, Springer, 2014, pp. 297–304.
[46] M. Tan and A. Qiu, "Spectral Laplace-Beltrami wavelets with applications in medical images," *IEEE Trans. Med. Imag.*, vol. 34, no. 5, pp. 1005–1017, May 2015.
[47] F. Olver, D. Lozier, R. Boisvert, and C. Clark, *The NIST Handbook of Mathematical Functions*. New York, NY, USA: Cambridge Univ. Press, 2010.
[48] S.-G. Huang, I. Lyu, A. Qiu, and M. K. Chung, "Fast polynomial approximation of heat kernel convolution on manifolds and its application to brain sulcal and gyral graph pattern analysis," *IEEE Trans. Med. Imag.*, vol. 39, no. 6, pp. 2201–2212, Jun. 2020.
[49] I. S. Dhillon, Y. Guan, and B. Kulis, "Weighted graph cuts without eigenvectors a multilevel approach," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 11, pp. 1944–1957, Nov. 2007.
[50] V. P. Dwivedi, C. K. Joshi, A. T. Luu, T. Laurent, Y. Bengio, and X. Bresson, "Benchmarking graph neural networks," *J. Mach. Learn. Res.*, vol. 24, no. 43, pp. 1–48, 2023.
[51] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[52] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, vol. 32, 2019.
[53] M. Fey and J. E. Lenssen, "Fast graph representation learning with PyTorch geometric," in *Proc. Workshop Representation Learn. Graphs Manifolds*, 2019.
[54] B. Gavish and S. C. Graves, "The travelling salesman problem and related problems," Massachusetts Inst. Technol., Operations Res. Center, Working Paper OR 078-78, 1978.
[55] D. S. Johnson and L. A. McGeoch, "The traveling salesman problem: A case study in local optimization," *Local Search Combinatorial Optim.*, vol. 1, no. 1, pp. 215–310, 1997.
[56] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. Int. Conf. Comput. Vis.*, 2017, pp. 2980–2988.
[57] V. P. Dwivedi et al., "Long range graph benchmark," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, vol. 35, pp. 22326–22340.
[58] D. Lu and Q. Weng, "A survey of image classification methods and techniques for improving classification performance," *Int. J. Remote Sens.*, vol. 28, no. 5, pp. 823–870, 2007.
[59] A. Krizhevsky et al., "Learning multiple layers of features from tiny images," 2009. [Online]. Available: https://www.cs.toronto.edu/kriz/learningfeatures-2009-TR.pdf
[60] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels compared to State-of-the-art superpixel methods," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2274–2282, Nov. 2012.
[61] J. J. Irwin, T. Sterling, M. M. Mysinger, E. S. Bolstad, and R. G. Coleman, "ZINC: A free tool to discover chemistry for biology," *J. Chem. Inf. Model.*, vol. 52, no. 7, pp. 1757–1768, 2012.
[62] S. Singh et al., "SATPdb: A database of structurally annotated therapeutic peptides," *Nucleic Acids Res.*, vol. 44, no. D1, pp. D1119–D1126, 2016.
[63] O. Wieder et al., "A compact review of molecular property prediction with graph neural networks," *Drug Discov. Today: Technol.*, vol. 37, pp. 1–12, 2020.
[64] K. M. Borgwardt, C. S. Ong, S. Schönauer, S. Vishwanathan, A. J. Smola, and H.-P. Kriegel, "Protein function prediction via graph kernels," *Bioinformatics*, vol. 21, no. suppl_1, pp. i47–i56, 2005.
[65] H. Cumming and C. Rücker, "Octanol–water partition coefficient measurement by a simple $^{1}$H NMR method," *ACS Omega*, vol. 2, no. 9, pp. 6244–6249, 2017.
[66] M. Işik, D. Levorse, D. L. Mobley, T. Rhodes, and J. D. Chodera, "Octanol–water partition coefficient measurements for the SAMPL6 blind prediction challenge," *J. Comput.-Aided Mol. Des.*, vol. 34, no. 4, pp. 405–420, 2020.
[67] P. Ertl and A. Schuffenhauer, "Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions," *J. Cheminformatics*, vol. 1, pp. 1–11, 2009.
[68] D. M. Barch et al., "Demographic, physical and mental health assessments in the adolescent brain and cognitive development study: Rationale and description," *Develop. Cogn. Neurosci.*, vol. 32, pp. 55–66, 2018.
[69] D. S. Marcus, A. F. Fotenos, J. G. Csernansky, J. C. Morris, and R. L. Buckner, "Open access series of imaging studies: Longitudinal MRI data in nondemented and demented older adults," *J. Cogn. Neurosci.*, vol. 22, no. 12, pp. 2677–2684, 2010.
[70] N. Akshoomoff et al., "NIH toolbox cognition battery (CB): Composite scores of crystallized, fluid, and overall cognition," *Monographs Soc. Res. Child Develop.*, vol. 78, no. 4, pp. 119–132, 2013.
[71] K. Franke and C. Gaser, "Ten years of BrainAGE as a neuroimaging biomarker of brain aging: What insights have we gained?," *Front. Neurol.*, vol. 10, 2019, Art. no. 789.
[72] X. Shen et al., "Using connectome-based predictive modeling to predict individual behavior from brain connectivity," *Nature Protoc.*, vol. 12, no. 3, pp. 506–518, 2017.
[73] K. Nashiro, M. Sakaki, and M. Mather, "Age differences in brain activity during emotion processing: Reflections of age-related decline or increased emotion regulation," *Gerontol.*, vol. 58, no. 2, pp. 156–163, 2012.
[74] L. E. Bettio, L. Rajendran, and J. Gil-Mohapel, "The effects of aging in the hippocampus and cognitive decline," *Neurosci. Biobehavioral Rev.*, vol. 79, pp. 66–86, 2017.
[75] R. E. Jung and R. J. Haier, "The parieto-frontal integration theory (P-FIT) of intelligence: Converging neuroimaging evidence," *Behav. Brain Sci.*, vol. 30, no. 2, pp. 135–154, 2007.
[76] M. Song et al., "Brain spontaneous functional connectivity and intelligence," *Neuroimage*, vol. 41, no. 3, pp. 1168–1176, 2008.
[77] A. Qiu, L. Younes, and M. I. Miller, "Intrinsic and extrinsic analysis in computational anatomy," *Neuroimage*, vol. 39, no. 4, pp. 1803–1814, 2008.
[78] J. Huang, N. Chen, and A. Qiu, "Topological cycle graph attention network for brain functional connectivity," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Interv.*, Cham: Springer Nature Switzerland, 2024, pp. 723–732.

**Jinghan Huang** received the BS degree from the School of Physics and Astronomy, Shanghai Jiao Tong University, China. He is currently working towards the MEng degree in biomedical engineering with the National University of Singapore. His research interests are graph deep learning and its applications to medical image analysis.

**Qiufeng Chen** (Senior Member, IEEE) received the PhD degree from the School of Information Science and Engineering, Central South University, Changsha, in 2016. She was a visiting scholar with the School of Biomedical Engineering, Shanghai Tech University, in 2022, and with Department of Biomedical Engineering, National University of Singapore, in 2023. She is currently a lecturer with the College of Computer and Information Science, Fuzhou. Her research mainly focuses on machine learning in medical image analysis and computer-aided diagnosis.

**Pengli Zhu** (Member, IEEE) received the BEng, MEng, and PhD degrees from Dalian Maritime University, Dalian, China, in 2017, 2020, and 2024, respectively. From 2023 to 2024, he was supported by China Scholarship Council as a joint-training Ph.D. student with the with Department of Biomedical Engineering, National University of Singapore, Singapore. He is currently a research staff with the Department of Health Technology and Informatics, Hong Kong Polytechnic University, Hong Kong. His current research interests include intelligent image processing and machine learning.



**Yijun Bian** received the PhD degree in computer science and technology from the University of Science and Technology of China, Hefei, China, in 2020. She was a visiting research student with the Texas A&M University once and is currently a research fellow with the National University of Singapore, Singapore. Her research interests include ensemble methods, and fairness in machine learning.

**Nanguang Chen** received the BSc degree in electrical engineering from Hunan University and the MSc degree in physics Peking University, respectively, and the PhD in biomedical engineering from Tsinghua University. He is an associate professor of biomedical engineering with the National University of Singapore (NUS). His research interests include diffuse optical tomography, optical coherence tomography, novel fluorescence microscopic imaging methods, and image reconstruction.

**Moo K. Chung** received the PhD degree. He is currently a professor of biostatistics and medical informatics with the University of Wisconsin-Madison. He is also affiliated with the Waisman Laboratory for Brain Imaging and Behavior. His main research area is computational neuroimaging, where noninvasive brain imaging modalities such as magnetic resonance imaging (MRI) and diffusion tensor imaging (DTI) are used to map the spatiotemporal dynamics of the human brain.

**Anqi Qiu** (Senior Member, IEEE) received the PhD degree in electrical and computer engineering from Johns Hopkins University. She is currently a professor and global STEM scholar with the Department of Health Technology and Informatics, the Hong Kong Polytechnic University. She has been devoted to innovation in computational analyses of complex and informative datasets comprising of disease phenotypes, neuroimage, and genetic data to understand the origins of individual differences in health throughout the lifespan.