Simplicial Complex Neural Networks

Hanrui Wu, Andy Yip, Jinyi Long, Jia Zhang, *Member, IEEE*, and Michael K. Ng, *Senior Member, IEEE*

***Abstract*—Graph-structured data, where nodes exhibit either pair-wise or high-order relations, are ubiquitous and essential in graph learning. Despite the great achievement made by existing graph learning models, these models use the direct information (edges or hyperedges) from graphs and do not adopt the underlying indirect information (hidden pair-wise or high-order relations). To address this issue, in this paper, we propose a general framework named Simplicial Complex Neural (SCN) network, in which we construct a simplicial complex based on the direct and indirect graph information from a graph so that all information can be employed in the complex network learning. Specifically, we learn representations of simplices by aggregating and integrating information from all the simplices together via layer-by-layer simplicial complex propagation. In consequence, the representations of nodes, edges, and other high-order simplices are obtained simultaneously and can be used for learning purposes. By making use of block matrix properties, we derive the theoretical bound of the simplicial complex filter learnt by the propagation and establish the generalization error bound of the proposed simplicial complex network. We perform extensive experiments on node (0-simplex), edge (1-simplex), and triangle (2-simplex) classifications, and promising results demonstrate the performance of the proposed method is better than that of existing graph and hypergraph network approaches.**

***Index Terms*—Block matrices, edge classification, generalization error, graph learning networks, high-order simplex classification, node classification, simplicial complex.**

Manuscript received 30 December 2022; revised 30 August 2023; accepted 3 October 2023. Date of publication 13 October 2023; date of current version 5 December 2023. This work was supported in part by the National Natural Science Foundation of China under Grants 62206111, 62276115, and 62106084, in part by the Young Talent Support Project of Guangzhou Association for Science and Technology under Grant QT-2023-017, in part by Guangzhou Basic and Applied Basic Research Foundation under Grants 2023A04J1058 and 202201010498, in part by China Postdoctoral Science Foundation under Grants 2022M721343 and 2022M711331, in part by the National Natural Science Foundation of Guangdong, China under Grants 2019A1515012175 and 2022A1515010468, in part by the Outstanding Youth Project of Guangdong Natural Science Foundation of China under Grant 2021B1515020076, in part by Guangdong Provincial Key Laboratory of Traditional Chinese Medicine Informatization under Grant 2021B1212040007, in part by the Science and Technology Planning Project of Guangdong Province under Grant 2023A0505050092, and in part by the Fundamental Research Funds for Central Universities under Grant 21622326. The work of Michael K. Ng was supported in part by Hong Kong Research Grant Council GRF under Grants 17201020, 17300021, CRF C7004-21GF, and Joint NSFC-RGC N-HKU76921. Recommended for acceptance by X. Hu. *(Corresponding author: Michael K. Ng.)*

Hanrui Wu and Jia Zhang are with the College of Information Science and Technology, Jinan University, Guangzhou, Guangdong 510006, China (e-mail: hrwu@outlook.com; jiazhang@jnu.edu.cn).

Jinyi Long is with the College of Information Science and Technology, Guangdong Key Lab of Traditional Chinese Medicine Information Technology, Jinan University, Guangzhou, Guangdong 510006, China, and also with Pazhou Lab, Guangzhou, Guangdong 510006, China (e-mail: jinyil@jnu.edu.cn).

Andy Yip and Michael K. Ng are with the Department of Mathematics, Hong Kong Baptist University, Hong Kong, China (e-mail: mhyipa@hotmail.com; michael-ng@hkbu.edu.hk).

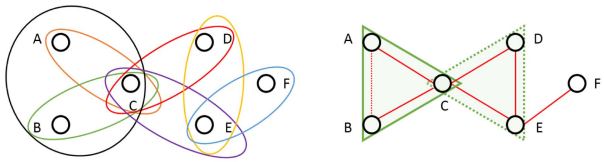
Digital Object Identifier 10.1109/TPAMI.2023.3323624

I. Introduction

Graph learning [1], [2], [3], [4], [5], [6] has received increasing attention because of its ability to handle non-euclidean data. Recent studies tend to extend deep learning into graphs and achieve great improvement, especially in the application of semi-supervised node classification [7], [8], [9]. In [7], Kipf et al. proposed a classical model named Graph Convolutional Network (GCN), which introduces a first-order approximation of ChebNet [10]. Particularly, GCN effectively handles the standard graphs, where the correlations between nodes are pair-wise. However, in many practical applications, such as co-citation and co-authorship networks, the data relations are more complex than pair-wise. Consequently, as a generalization of graphs, hypergraphs [11], [12], [13], [14] emerge and play an important role in modeling such complex high-order relations since one hyperedge can link more than two nodes.

In [12], Feng et al. proposed a Hypergraph Neural Network (HGNN) model, which generalizes the convolution operation of standard graphs, i.e., GCN, to deal with the hypergraph situation by constructing an adjacency matrix based on the hypergraph. Nevertheless, although this kind of generalization is intuitive and simple to achieve, Dong et al. [15] have argued that it does not fully exploit the hypergraph structure. To address the issue, they proposed alternately learning the node and hyperedge representations layer-by-layer. Furthermore, in [16], Wu et al. proposed studying the hypergraph structure in both nodes and hyperedges and developed a Hypergraph Collaborative Network (HCoN) model. When learning node and hyperedge representations, HCoN collaboratively aggregates information from both nodes and hyperedges of the previous layer.

Despite the great success of exploiting data relations achieved by graph and hypergraph learning, most existing models focus on the direct information (edges or hyperedges) and do not consider the indirect information (hidden pair-wise or high-order relations). Below, we use a high-order correlation example to illustrate the motivation of this paper. As shown in Fig. 1(a), where A, B, ..., and F are six authors of seven papers with co-authorship (A, C), (B, C), (C, D), (A, B, C), (C, E), (D, E), and (E, F) for papers. In this example, the direct information is represented by hyperedges, i.e., the papers. According to the hypergraph, the pairs (A, B), (A, D), (A, E), and (A, F) do not have any papers, and there are no hyperedges among these pairs. We observe that (A, C), (B, C), and (A, B, C) have papers and three hyperedges exist among them. The pair (A, B) can be viewed as a hidden pair-wise relation (the indirect information from the hypergraph). That is, the degree of association (in terms of paper collaboration) between A and B is significantly larger



(a) Hypergraph (b) Simplicial complex

Fig. 1. Motivation of direct and indirect information. (a) a hypergraph and (b) a simplicial complex. The black circles are the nodes indicating authors (A, B, C, D, E, F). The ovals in Fig. 1(a) are the hyperedges in a hypergraph, and different colors mean different hyperedges. Hyperedges are papers, i.e., (A, C), (B, C), (C, D), (A, B, C), (C, E), (D, E), and (E, F), which are the direct information. The black circles, red line segments, and green triangles in Fig. 1(b) represent 0-, 1-, and 2-simplices, respectively. Compared with a hypergraph, a simplicial complex further involves indirect information, e.g., the relation between A and B, and the relation among C, D, and E, indicated by a dotted line and a dotted triangle, respectively.

than A and D, A and E, and A and F. Moreover, there are three hyperedges (C, D), (C, E), and (D, E) in the hypergraph. They are direct information corresponding to the three papers. Even C, D, and E do not have a paper together, i.e., there is no hyperedge among C, D, and E in the hypergraph, the triplet (C, D, E) can be viewed as a hidden high-order relation (the indirect information from the hypergraph). In other words, the degree of association among C, D, and E is very strong compared with other triplets. Based on the above two cases, the indirect information (A, B) and (C, D, E) is indeed important. However, such information is usually not used in existing hypergraph models.

To fully leverage such indirect information, we introduce a simplicial complex [17] into graph learning. A simplicial complex[1] is built as a union of nodes, edges, triangles, tetrahedron, and higher-dimensional polytopes, namely simplices [18], [19]. That is, a 0-simplex is a point (a node), a 1-simplex is a line segment (an edge) linking between two points (nodes), and a 2-simplex is a filled triangle containing three nodes, and so on. Consequently, the indirect relation between A and B, and the indirect relation among C, D, and E are naturally formulated by a 1-simplex and a 2-simplex in Fig. 1(b) using a dotted line and a dotted triangle, respectively.

Based on the above discussion, in this paper, we propose leveraging simplex-level information to investigate both direct and indirect underlying graph information. Moreover, considering the promising performance achieved by the integrated learning strategy proposed in [16], we develop a general framework, namely **S**implicial **C**omplex **N**eural (SCN) network, which simultaneously learns all the simplex representations by aggregating and integrating the information from all simplices of the previous layer. For example, given a graph, we first extract its 0-, 1-, and 2-simplices. Subsequently, we build three incidence matrices indicating the relationships between 0- and 1-simplices, 0- and 2-simplices, and 1- and 2-simplices, respectively. Through constructing the incidence matrices, both direct and indirect graph information is captured. Then we learn the representations of the simplices by aggregating the information from all simplices of the previous layer. Consequently, the proposed model is a general framework that produces effective and informative representations of all simplices simultaneously. We highlight the main contributions of this paper as follows.

- We present a novel general graph learning model called Simplicial Complex Neural (SCN) network, which exploits simplex-level information to study both direct and indirect underlying graph information to achieve informative representations.
- SCN learns the representations of all simplices simultaneously by fully integrating all simplices of the previous layer. As a consequence, the relationships between different simplices are well-explored.
- We derive theoretical analyses, including a bound of the proposed simplicial complex filter and a bound of the generalization gap of SCN, to establish some desirable properties of the proposed model.
- Since SCN is capable of learning representations of every simplex, we perform experiments on node (0-simplex) classification, edge (1-simplex) classification, and triangle (2-simplex) classification to validate the effectiveness of the proposed model.

The remainder of the paper is organized as follows. We review important studies regarding graph learning in Section II. Then, we introduce our proposed method in Section III. The theoretical analysis of the proposed model is given in Section IV. We conduct extensive experiments and discuss the experimental results in Section V. In Section VI, we make a conclusion on the paper.

II. Related Work

In this section, we mainly review several studies about deep learning on graphs, especially the models proposed for classification tasks. Particularly, we discuss two situations, i.e., standard graph learning and hypergraph learning, and focus on convolutional operation-based models.

A. Standard Graph Learning

Convolutional neural networks [20] have received rapid development in the past ten years. Recent studies found that extending convolutional neural networks to graph-structured data is attractive and powerful. However, due to the lacking of grid structures of graphs, such extension of convolution operation is challenging [1]. In [10], Defferrard et al. proposed ChebNet to generalize convolutional neural networks to graph learning by Chebyshev polynomial approximation. Furthermore, Kipf et al. proposed Graph Convolutional Network (GCN) [7], an improved version of ChebNet that updates a node's representation by aggregating the information of its neighbors. Since then, plenty of GCN-based variants were developed, such as [21], [22], [23], [24], and have shown superior performance in various applications like computer vision [25], [26], [27]. For instance, Graph Attention Network (GAT) [21] adopts the attention mechanism to determine the weights of the node's neighbors. To prevent the over-smoothing [28] problem, Graph Convolutional

[1]Formally, we use abstract simplicial complex [18] and drop "abstract" for easier reading.

Network via Initial residual and Identity mapping (GCNII) [23] introduces initial residual connections and identity mappings into the graph learning. In addition, Orthogonal Graph Convolution (Ortho-GConv) [24] proposes augmenting the GCN backbone to stabilize the model training and improve the model's generalization performance by preserving the orthogonality of the feature transformation. Note that these models are proposed for standard graphs depicting pair-wise relations and do not exploit high-order graph structure information.

B. Hypergraph Learning

Hypergraphs can be treated as a generalization of standard graphs. In recent years, learning on hypergraphs has attracted increasing attention since it effectively investigates high-order structure information. The capability and flexibility in modeling complex data relations make hypergraphs popular, and hypergraph learning has been successfully applied to lots of fields, such as recommendation systems [29], [30], [31], [32] and computer vision [33], [34]. To perform learning on hypergraphs, one simple and direct strategy is to generalize the convolution operation in GCN to a hypergraph by building the adjacency matrix based on the hypergraph, such as [12], [35]. For example, HyperGraph Neural Networks (HGNN) [12] is the first model approximating and introducing the hypergraph Laplacian into the deep hypergraph learning paradigm, thus extending the convolution operation to hypergraphs. HGNN$^+$ [36] further improves HGNN from the hypergraph modeling and hypergraph convolution aspects. As a result, the modeling of complex relationships is promoted and the convolution operation is flexibly defined. In [13], Yadati et al. proposed HyperGCN to transform a hypergraph into a weighted graph, to which the GCN is applied to learn node representations. HyperGCN exhibits its effectiveness not only in semi-supervised node classification where hyperedges encode similarity but also in combinatorial optimization where hyperedges do not encode similarity. Dong et al. proposed Hypergraph Networks with Hyperedge Neurons (HNHN) in [15] to adequately utilize the hypergraph structure by an alternative learning method, thus alternatively updating the node and hyperedge representations. HNHN also provides a flexible approach to normalization. In [16], Wu et al. further incorporated both node and hyperedge information of the previous layer collaboratively into the representation learning of nodes and hyperedges. Besides, Unified Graph Neural Network (UniGNN) [37] interprets the message passing process in graphs and hypergraphs, thus adapting the graph neural network-based variants into hypergraphs. Additionally, Line Expansion Graph Convolutional Network (LEGCN) [38] proposes line expansion to reduce a hypergraph to a standard graph, which makes existing graph neural networks work with high-order structure possible. All the above-mentioned methods seek to study the complex correlations in hypergraphs. In spite of the powerful ability of these models to exploit high-order relations to learn hypergraph representations, they barely consider both direct and indirect underlying graph information. Different from existing models, in this paper, we propose to leverage the simplex-level information to fully consider the direct and indirect information.

Fig. 2. From left to right: 0-, 1-, 2-, and 3-simplex.

III. Methodology

In this section, we first give the notations and their descriptions. Then, we provide the details of the proposed Simplicial Complex Neural (SCN) network.

A. Notations

Let $\mathcal{V}$ be the set of nodes in a graph. Denote a simplicial complex as $\mathcal{S}$, which can be viewed as a generalization of a graph containing high-order relationships between nodes. A k-simplex [39] is a subset of $\mathcal{V}$ of cardinality $k+1$ and can be denoted by $\{v_0, \ldots, v_i, \ldots, v_k\}$ where each $v_i \in \mathcal{V}$. As shown in Fig. 2, a 0-simplex is a point, a 1-simplex is a pair of points connected with a line segment, a 2-simplex is a filled triangle, and so on. More formally, a simplicial complex is a finite collection of simplices [19], i.e., points, edges, triangles, tetrahedron, and higher-dimensional polytopes, such that every face of a simplex of $\mathcal{S}$ belongs to $\mathcal{S}$ and the intersection of any two simplices of $\mathcal{S}$ is a common face of both of them.

Moreover, the weights of the k-simplices are given by a positive diagonal matrix $\mathbf{W}_k$. When there is no information about the simplices, each simplex is assigned with value 1, indicating the diagonal entries in $\mathbf{W}_k$ are all equal to 1. The incidence matrix $\mathbf{A}_{p,q}$ between p-simplices and q-simplices (with $p<q$) is of size $n_p \times n_q$ by considering p-simplices are incident to which q-simplices:[2]

$$\mathbf{A}_{p,q}(i,j) = \begin{cases} 1, & \text{if } \mathcal{S}_p^j \text{ is on the boundary of } \mathcal{S}_q^i, \\ 0, & \text{otherwise,} \end{cases} \tag{1}$$

where $\mathcal{S}_q^i$ and $\mathcal{S}_p^j$ are the i-th and j-th simplices in $\mathcal{S}_q$ and $\mathcal{S}_p$, respectively. Let $\mathbf{Z}_{p,q}$ and $\mathbf{Z}_{q,p}$ be the degrees of p-simplices and q-simplices, respectively. Let n_p and n_q be the numbers of p-simplices and q-simplices, respectively. Then, $\mathbf{Z}_{p,q}$ and $\mathbf{Z}_{q,p}$ are diagonal matrices with sizes being $n_p \times n_p$ and $n_q \times n_q$, respectively. Specifically, we have weighted degrees

$$\mathbf{Z}_{p,q}(i,i) = \sum_{j \in \mathcal{S}_q} \mathbf{W}_q(j,j)\mathbf{A}_{p,q}(i,j)$$

and unweighted degrees

$$\mathbf{Z}_{q,p}(j,j) = \sum_{i \in \mathcal{S}_p} \mathbf{A}_{p,q}(i,j).$$

Particularly, if a diagonal entry of $\mathbf{Z}_{p,q}$ is zero, then it is replaced with a one to ensure that $\mathbf{Z}_{p,q}^{-\frac{1}{2}}$ is well-defined. Similar replacements are done to $\mathbf{Z}_{q,p}$. Therefore, a rescaled version of $\mathbf{A}_{p,q}$ can be defined by:

$$\widehat{\mathbf{A}}_{p,q} = \mathbf{Z}_{p,q}^{-\frac{1}{2}}\mathbf{A}_{p,q}\mathbf{Z}_{q,p}^{-\frac{1}{2}}. \tag{2}$$

[2]Refer to [19] for the definition of *boundary*.

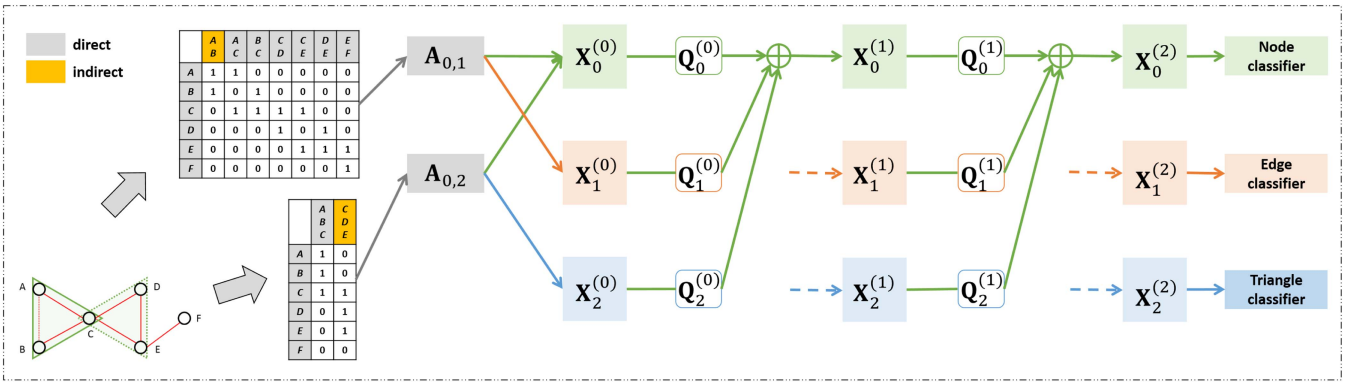


Fig. 3. Illustration of the proposed SCN model for learning the 0-simplex representations. The superscripts "(0)", "(1)", and "(2)" represent different layers. For clarity, we only depict the learning of 0-simplices. The solid lines show the learning procedure and the dotted lines indicate the procedure is omitted. When learning the representations of 0-simplices, we care about the graph structure information in $\mathbf{A}_{0,1}$ and $\mathbf{A}_{0,2}$, and the representations of 0-, 1-, and 2-simplices of the previous layer are aggregated.

B. Simplicial Complex Neural Networks

1) Overview: Here, we present the proposed SCN model and show how it exploits the simplex-level information to leverage both direct and indirect underlying graph information. Particularly, SCN learns representation of simplices by aggregating information from all the simplices of the previous layer. The aggregation procedure of the proposed model is illustrated in Fig. 3, where we use the example in Section I and only depict the learning of 0-simplices for clarity. The solid lines show the learning procedure and the dotted lines indicate omitted procedures. In this example, we have 0-, 1-, and 2-simplices in a graph. Fig. 3 shows that the learning of 0-simplices in one layer is based on the information of simplices in the previous layer.

Below, we take the above example to elaborate on the proposed model. Specifically, we introduce three encoders for learning the representations of 0-, 1-, and 2-simplices, respectively. Subsequently, we propose a general framework to integrate encoders for a simplicial complex of an arbitrary order, thus producing effective and informative representations of simplices of various orders simultaneously.

2) 0-Simplex Encoder: Here, we describe the encoder for 0-simplices. Denote the representations of 0-, 1-, and 2-simplices as $\mathbf{X}_0$, $\mathbf{X}_1$, and $\mathbf{X}_2$, respectively. Since we utilize the simplex-level information, we consider every combination of relationships. As a result, we present the relationship between 0- and 1-simplices by $\mathbf{A}_{0,1}$ and the relationship between 0- and 2-simplices by $\mathbf{A}_{0,2}$. For instance, $\mathbf{A}_{0,1}$ is an incidence matrix where the rows and columns are 0- and 1-simplices, respectively.

First, considering the relations in $\mathbf{A}_{0,1}$, the convolution operation on 0-simplices can be formulated as:

$$\mathbf{K}_{\mathrm{i}} = \mathbf{A}_{0,1}\mathbf{W}_1\mathbf{A}_{0,1}^{\top}\mathbf{X}_0\mathbf{Q}_0, \tag{3}$$

where $\mathbf{Q}_0$ is the trainable weight matrix. To avoid the possibility of vanishing gradients and numerical instabilities when used in a deep neural network model, we apply the normalization formulation in (2) to (3) as follows:

$$\mathbf{K}_{\mathrm{i}} = \widehat{\mathbf{A}}_{0,1}\mathbf{W}_1\widehat{\mathbf{A}}_{0,1}^{\top}\mathbf{X}_0\mathbf{Q}_0. \tag{4}$$

According to [16], incorporating 1-simplices into the learning of 0-simplices could make full use of the graph structure. Hence, based on $\mathbf{A}_{0,1}$, we extract the 1-simplices information as follows:

$$\mathbf{K}_{\mathrm{ii}} = \mathbf{A}_{0,1}\mathbf{W}_1\mathbf{X}_1\mathbf{Q}_1, \tag{5}$$

where $\mathbf{Q}_1$ is the trainable weight matrix. We further normalize (5) as follows:

$$\mathbf{K}_{\mathrm{ii}} = \widehat{\mathbf{A}}_{0,1}\mathbf{W}_1^{\frac{1}{2}}\mathbf{Z}_{1,0}^{-\frac{1}{2}}\mathbf{X}_1\mathbf{Q}_1. \tag{6}$$

Second, considering the relations in $\mathbf{A}_{0,2}$, the normalized convolution operation on 0-simplices is similar to (4):

$$\mathbf{K}_{\mathrm{iii}} = \widehat{\mathbf{A}}_{0,2}\mathbf{W}_2\widehat{\mathbf{A}}_{0,2}^{\top}\mathbf{X}_0\mathbf{Q}_0. \tag{7}$$

And the extraction on the 2-simplices can be formulated as follows:

$$\mathbf{K}_{\mathrm{iv}} = \mathbf{A}_{0,2}\mathbf{W}_2\mathbf{X}_2\mathbf{Q}_2, \tag{8}$$

where $\mathbf{Q}_2$ is the trainable weight matrix. The normalized version of (8) is:

$$\mathbf{K}_{\mathrm{iv}} = \widehat{\mathbf{A}}_{0,2}\mathbf{W}_2^{\frac{1}{2}}\mathbf{Z}_{2,0}^{-\frac{1}{2}}\mathbf{X}_2\mathbf{Q}_2. \tag{9}$$

For the above-mentioned normalization, we give corresponding analyses in Section IV. In summary, taking the simplex-level information into account, the structure in $\mathbf{A}_{0,1}$ is utilized by Eqs. (4) and (6) and the structure in $\mathbf{A}_{0,2}$ is leveraged by Eqs. (7) and (9). Therefore, we incorporate Eqs. (4), (6), (7), and (9) to obtain the latent representation of 0-simplices as follows:

$$\begin{aligned}\widetilde{\mathbf{X}}_0 &= \sigma(\alpha(\mathbf{K}_{\mathrm{i}} + \mathbf{K}_{\mathrm{iii}}) + (1-\alpha)(\mathbf{K}_{\mathrm{ii}} + \mathbf{K}_{\mathrm{iv}})) \\ &= \sigma\Big(\alpha(\widehat{\mathbf{A}}_{0,1}\mathbf{W}_1\widehat{\mathbf{A}}_{0,1}^{\top} + \widehat{\mathbf{A}}_{0,2}\mathbf{W}_2\widehat{\mathbf{A}}_{0,2}^{\top})\mathbf{X}_0\mathbf{Q}_0 \\ &\quad + (1-\alpha)\Big(\widehat{\mathbf{A}}_{0,1}\mathbf{W}_1^{\frac{1}{2}}\mathbf{Z}_{1,0}^{-\frac{1}{2}}\mathbf{X}_1\mathbf{Q}_1 \\ &\quad + \widehat{\mathbf{A}}_{0,2}\mathbf{W}_2^{\frac{1}{2}}\mathbf{Z}_{2,0}^{-\frac{1}{2}}\mathbf{X}_2\mathbf{Q}_2\Big)\Big),\end{aligned} \tag{10}$$

where α is a trade-off parameter ranging in (0, 1) and $\sigma(\cdot)$ is a nonlinear activation function such as ReLU.

3) 1-Simplex Encoder: We develop an encoder for learning the representation for 1-simplices, which is similar to the 0-simplex encoder. Specifically, when learning on 1-simplices, we care about the structures in the incidence matrices $\mathbf{A}_{0,1}$ and $\mathbf{A}_{1,2}$. For clarity, we give the formulation of 1-simplex encoder as follows:

$$\begin{aligned}\widetilde{\mathbf{X}}_1 = \sigma\Big(&\alpha(\widehat{\mathbf{A}}_{1,2}\mathbf{W}_2\widehat{\mathbf{A}}_{1,2}^\top + \widehat{\mathbf{A}}_{0,1}^\top\mathbf{W}_0\widehat{\mathbf{A}}_{0,1})\mathbf{X}_1\mathbf{Q}_1\\ &+(1-\alpha)\Big(\widehat{\mathbf{A}}_{0,1}^\top\mathbf{W}_0^{\frac{1}{2}}\mathbf{Z}_{0,1}^{-\frac{1}{2}}\mathbf{X}_0\mathbf{Q}_0\\ &+\widehat{\mathbf{A}}_{1,2}\mathbf{W}_2^{\frac{1}{2}}\mathbf{Z}_{2,1}^{-\frac{1}{2}}\mathbf{X}_2\mathbf{Q}_2\Big)\Big). \end{aligned} \tag{11}$$

4) 2-Simplex Encoder: Similarly, given the structures in the incidence matrices $\mathbf{A}_{0,2}$ and $\mathbf{A}_{1,2}$, we have the 2-simplex encoder as follows:

$$\begin{aligned}\widetilde{\mathbf{X}}_2 = \sigma\Big(&\alpha(\widehat{\mathbf{A}}_{0,2}^\top\mathbf{W}_0\widehat{\mathbf{A}}_{0,2} + \widehat{\mathbf{A}}_{1,2}^\top\mathbf{W}_1\widehat{\mathbf{A}}_{1,2})\mathbf{X}_2\mathbf{Q}_2\\ &+(1-\alpha)\Big(\widehat{\mathbf{A}}_{0,2}^\top\mathbf{W}_0^{\frac{1}{2}}\mathbf{Z}_{0,2}^{-\frac{1}{2}}\mathbf{X}_0\mathbf{Q}_0\\ &+\widehat{\mathbf{A}}_{1,2}^\top\mathbf{W}_1^{\frac{1}{2}}\mathbf{Z}_{1,2}^{-\frac{1}{2}}\mathbf{X}_1\mathbf{Q}_1\Big)\Big). \end{aligned} \tag{12}$$

5) A General Framework: Based on the above encoders, we can obtain the representations of 0-, 1-, and 2-simplices according to Eqs. (10), (11), and (12), respectively. To learn representations of all simplices in a general simplicial complex simultaneously, suppose a graph contains 0-, 1-, ..., and s-simplices where s is the largest order of simplices, we propose the following Simplicial Complex Neural (SCN) networks:

$$\begin{aligned}\widetilde{\mathbf{S}} = \sigma\Bigg(&\alpha\sum_{p=0}^{s}\sum_{q=p+1}^{s}((\mathbf{\Phi}_{p,q}^\top\mathbf{\Lambda}\mathbf{\Phi}_{p,q})\odot\mathbf{J})\mathbf{S_1}\\ &+(1-\alpha)\sum_{p=0}^{s}\sum_{q=p+1}^{s}\mathbf{\Phi}_{p,q}^\top\mathbf{\Lambda}^{\frac{1}{2}}\mathbf{\Gamma}_{p,q}\mathbf{S_2}\Bigg), \end{aligned} \tag{13}$$

where $\odot$ means the Hadamard product operation,

$$\widetilde{\mathbf{S}} = \begin{bmatrix}\widetilde{\mathbf{X}}_0\\ \widetilde{\mathbf{X}}_1\\ \vdots\\ \widetilde{\mathbf{X}}_s\end{bmatrix},\quad \mathbf{J} = \begin{bmatrix}\mathbf{J_0} & \mathbf{0} & \cdots & \mathbf{0}\\ \mathbf{0} & \mathbf{J}_1 & \cdots & \mathbf{0}\\ \vdots & \vdots & \ddots & \vdots\\ \mathbf{0} & \cdots & \cdots & \mathbf{J}_s\end{bmatrix},$$

$$\mathbf{S_1} = \begin{bmatrix}\mathbf{X_0Q_0}\\ \mathbf{X}_1\mathbf{Q}_1\\ \vdots\\ \mathbf{X}_s\mathbf{Q}_s\end{bmatrix},\quad \mathbf{S_2} = \begin{bmatrix}\mathbf{X_sQ_s}\\ \mathbf{X}_{s-1}\mathbf{Q}_{s-1}\\ \vdots\\ \mathbf{X_0Q_0}\end{bmatrix},$$

$$\mathbf{\Lambda} = \begin{bmatrix}\mathbf{W_0} & \mathbf{0} & \cdots & \mathbf{0}\\ \mathbf{0} & \mathbf{W}_1 & \cdots & \mathbf{0}\\ \vdots & \vdots & \ddots & \vdots\\ \mathbf{0} & \cdots & \cdots & \mathbf{W}_s\end{bmatrix},$$

$\mathbf{J}_i$ is an all-ones matrix, $\mathbf{\Phi}_{p,q}$ is a zero matrix except $(p+1,q+1)$-th block given by $\widehat{\mathbf{A}}_{p,q}$ and $(q+1,p+1)$-th block given by $\widehat{\mathbf{A}}_{p,q}^\top$, and $\mathbf{\Gamma}_{p,q}$ is an anti-diagonal zero matrix except $(p+1)$-th anti-diagonal block given by $\mathbf{Z}_{p,q}^{-\frac{1}{2}}$ and $(q+1)$-th anti-diagonal block given by $\mathbf{Z}_{q,p}^{-\frac{1}{2}}$.

Note that the learning of representations of 0-, 1-, and 2-simplices based on Eqs. (10), (11), and (12) are integrated by setting s in (13) to be 2, i.e., $\widetilde{\mathbf{S}}^\top = [\widetilde{\mathbf{X}}_0^\top, \widetilde{\mathbf{X}}_1^\top, \widetilde{\mathbf{X}}_2^\top]$. Consequently, (13) is a general framework to exploit simplex-level information for graph learning. Such an integration is used in each hidden layer to produce the outputs $\widetilde{\mathbf{X}}_0^{(l)}$, $\widetilde{\mathbf{X}}_1^{(l)}$, and $\widetilde{\mathbf{X}}_2^{(l)}$ to be used as the inputs to the next layer. Moreover, the direct and indirect information is effectively investigated by the proposed model to achieve representations of all simplices simultaneously.

6) Objective Function: In this paper, we deal with the semi-supervised multi-class classification problem. Therefore, given a task of making predictions on p-simplices, we evaluate the cross-entropy error on all labeled p-simplices as follows:

$$\mathcal{L} = -\sum_{l\in\mathcal{Y}_p^L}\sum_{y=1}^{C_p}\mathbf{Y}_p(l,y)\ln\widetilde{\mathbf{X}}_p(l,y), \tag{14}$$

where C_p represents the number of classes of p-simplices, $\mathbf{Y}_p$ is the label matrix of p-simplices, and $\mathcal{Y}_p^L$ is the set of p-simplex indices that have labels.

C. Relationship to Hypergraph Models

It is worthwhile to point out some differences between SCN and hypergraph models. In hypergraph models, such as HGNN [12] and HCoN [16], a hypergraph is specified by a binary incidence matrix $\mathbf{H}$ where each row represents a node and each column represents a hyperedge. The edge-degrees are given by the column sums of $\mathbf{H}$. Suppose that the highest edge-degree is d and the columns of $\mathbf{H}$ have been sorted by the edge-degrees in ascending order. Then, the matrix $\mathbf{H}$ can be partitioned into blocks according to the edge-degrees:

$$\mathbf{H} = [\mathbf{H}_1\ \mathbf{H}_2\ \ldots\ \mathbf{H}_d].$$

To see the structural differences between hypergraph and SCN models, it is clearer to look at the adjacency matrices of their multipartite graph representations. The adjacency matrix for a hypergraph is given by:

$$\begin{bmatrix}\mathbf{0} & \mathbf{H}\\ \mathbf{H}^\top & \mathbf{0}\end{bmatrix} = \begin{bmatrix}\mathbf{0} & \mathbf{H}_1 & \cdots & \mathbf{H}_d\\ \mathbf{H}_1^\top & \mathbf{0} & \cdots & \mathbf{0}\\ \vdots & \vdots & \ddots & \vdots\\ \mathbf{H}_d^\top & \mathbf{0} & \cdots & \mathbf{0}\end{bmatrix}.$$

In the case of a SCN with simplices of order up to s, the adjacency matrix is given by:

$$\begin{bmatrix}\mathbf{0} & \mathbf{A}_{0,1} & \cdots & \mathbf{A}_{0,s}\\ \mathbf{A}_{0,1}^\top & \mathbf{0} & \ddots & \vdots\\ \vdots & \ddots & \ddots & \mathbf{A}_{s-1,s}\\ \mathbf{A}_{0,s}^\top & \cdots & \mathbf{A}_{s-1,s}^\top & \mathbf{0}\end{bmatrix}.$$

Thus, we see that a SCN reduces to a hypergraph model if we set $s = d-1$, $\mathbf{A}_{p,q} = \mathbf{0}$ for $q > p > 0$, and $\mathbf{A}_{0,q} = \mathbf{H}_{q+1}$ for $1 \le q \le s$. However, in SCN, 1) we systematically construct

the adjacency matrix $\mathbf{A}_{p,q}$ via (1) to exploit relationships between simplices; 2) the framework has the flexibility to handle additional simplices not directly observed in the data; 3) the order s of the simplicial complex can be set different from $d-1$. In our experiments, we found superior results with $s=2$ compared with hypergraph models. Intuitively, if the data has a hyperedge representation with degree higher than 3, then a SCN with $s=2$ will represent it with multiple inter-connecting simplices of orders 0, 1, and 2.

IV. Theoretical Analysis

In this section, we present some fundamental properties of the core layer of a SCN network. First, we show that the simplicial complex filter underlies a SCN network has a uniform bound independent of the network size. It is a consequence of the filter normalization (2) and it shows that the network gradient is under control. Moreover, it forms the basis for the algorithmic stability of a SCN layer. Next, we present a bound of the generalization gap of the core layer of a SCN network when the network parameters are updated via the stochastic gradient descent (SGD). We adopt the approach in [40], [41], [42] to develop the bound, where a network that is uniformly stable in the sense of [43] has a generalization gap converges towards zero as the size of the training set increases. The convergence of the bound establishes the consistency between the training error and the test error, and thus confirms the usefulness of the SCN model.

A. A Bound of the Simplicial Complex Filter

The layer in (13) can be expressed as

$$\widetilde{\mathbf{X}} = \sigma(\mathbf{B}\mathbf{X}\mathbf{Q}), \tag{15}$$

where

$$\widetilde{\mathbf{X}} = \begin{bmatrix} \widetilde{\mathbf{X}}_0 \\ \vdots \\ \widetilde{\mathbf{X}}_s \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} \mathbf{B}_{00} & \cdots & \mathbf{B}_{0\,s} \\ \vdots & & \vdots \\ \mathbf{B}_{s0} & \cdots & \mathbf{B}_{ss} \end{bmatrix},$$

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_0 & & \\ & \ddots & \\ & & \mathbf{X}_s \end{bmatrix}, \quad \mathbf{Q} = \begin{bmatrix} \mathbf{Q}_0 \\ \vdots \\ \mathbf{Q}_s \end{bmatrix},$$

$$\mathbf{B}_{pp} = \alpha \sum_{i=0}^{p-1} \widehat{\mathbf{A}}_{i,p}^{\top} \mathbf{W}_i \widehat{\mathbf{A}}_{i,p} + \alpha \sum_{j=p+1}^{s} \widehat{\mathbf{A}}_{p,j} \mathbf{W}_j \widehat{\mathbf{A}}_{p,j}^{\top}, \tag{16}$$

and for $p<q$

$$\mathbf{B}_{pq} = (1-\alpha)\widehat{\mathbf{A}}_{p,q} \mathbf{W}_q^{\frac{1}{2}} \mathbf{Z}_{q,p}^{-\frac{1}{2}}, \tag{17}$$

$$\mathbf{B}_{qp} = (1-\alpha)\widehat{\mathbf{A}}_{p,q}^{\top} \mathbf{W}_p^{\frac{1}{2}} \mathbf{Z}_{p,q}^{-\frac{1}{2}}. \tag{18}$$

The bound of simplicial complex filter $\mathbf{B}$ is given as follows. It is independent of the size of the simplicial network. It depends only on s, the highest order of the simplex used. Let $\|\cdot\|_2$ denote the 2-norm of a matrix.

Theorem 1: For any $s \geq 1$, we have

$$\|\mathbf{B}\|_2 \leq s.$$

B. Proof of Theorem 1

The approach to prove Theorem 1 consists two main steps. First, in Lemma 1–Corollary 2, we devise a bound of each block matrix in (16)–(18). Then, we obtain the main result by considering a splitting of $\mathbf{B}$ into a block diagonal matrix plus the remaining part.

Lemma 1: For $p<q$, we have

$$\left\| \widehat{\mathbf{A}}_{p,q} \mathbf{W}_q \widehat{\mathbf{A}}_{p,q}^{\top} \right\|_2 \leq 1 \tag{19}$$

$$\left\| \widehat{\mathbf{A}}_{p,q}^{\top} \mathbf{W}_p \widehat{\mathbf{A}}_{p,q} \right\|_2 \leq 1. \tag{20}$$

Proof: We shall prove only (19); the proof for (20) is similar. The matrix $\widehat{\mathbf{A}}_{p,q} \mathbf{W}_q \widehat{\mathbf{A}}_{p,q}^{\top}$ is symmetric positive semi-definite. Hence, we have

$$\left\| \widehat{\mathbf{A}}_{p,q} \mathbf{W}_q \widehat{\mathbf{A}}_{p,q}^{\top} \right\|_2 = \rho\left(\widehat{\mathbf{A}}_{p,q} \mathbf{W}_q \widehat{\mathbf{A}}_{p,q}^{\top} \right),$$

where $\rho(\cdot)$ is the spectral radius (largest absolute eigenvalue). Next, note that the following matrices are similar so that their spectral radii are equal:

$$\begin{aligned} \widehat{\mathbf{A}}_{p,q} \mathbf{W}_q \widehat{\mathbf{A}}_{p,q}^{\top} &= \mathbf{Z}_{p,q}^{-\frac{1}{2}} \mathbf{A}_{p,q} \mathbf{W}_q \mathbf{Z}_{q,p}^{-1} \mathbf{A}_{p,q}^{\top} \mathbf{Z}_{p,q}^{-\frac{1}{2}} \\ &\sim \mathbf{Z}_{p,q}^{-1} \mathbf{A}_{p,q} \mathbf{W}_q \mathbf{Z}_{q,p}^{-1} \mathbf{A}_{p,q}^{\top}, \end{aligned}$$

where $\mathbf{A} \sim \mathbf{B}$ denotes that $\mathbf{A}$ and $\mathbf{B}$ are similar. Since $\rho(\cdot) \leq \|\cdot\|_{\infty}$, we have

$$\left\| \widehat{\mathbf{A}}_{p,q} \mathbf{W}_q \widehat{\mathbf{A}}_{p,q}^{\top} \right\|_2 = \rho\left(\mathbf{Z}_{p,q}^{-1} \mathbf{A}_{p,q} \mathbf{W}_q \mathbf{Z}_{q,p}^{-1} \mathbf{A}_{p,q}^{\top} \right) \tag{21}$$

$$\leq \left\| \mathbf{Z}_{p,q}^{-1} \mathbf{A}_{p,q} \mathbf{W}_q \mathbf{Z}_{q,p}^{-1} \mathbf{A}_{p,q}^{\top} \right\|_{\infty} \tag{22}$$

$$\leq \left\| \mathbf{Z}_{p,q}^{-1} \mathbf{A}_{p,q} \mathbf{W}_q \right\|_{\infty} \left\| \mathbf{Z}_{q,p}^{-1} \mathbf{A}_{p,q}^{\top} \right\|_{\infty}. \tag{23}$$

The row sums of $\mathbf{A}_{p,q}\mathbf{W}_q$ are either the diagonal entries of $\mathbf{Z}_{p,q}$ or zero. Therefore, the row sums of $\mathbf{Z}_{p,q}^{-1}\mathbf{A}_{p,q}\mathbf{W}_q$ are either 1 or 0. Thus, we have $\|\mathbf{Z}_{p,q}^{-1}\mathbf{A}_{p,q}\mathbf{W}_q\|_{\infty} \leq 1$. Likewise, the row sums of $\mathbf{A}_{p,q}^{\top}$ are either the diagonal entries of $\mathbf{Z}_{q,p}$ or zero, so that $\|\mathbf{Z}_{q,p}^{-1}\mathbf{A}_{p,q}^{\top}\|_{\infty} \leq 1$. It follows that

$$\left\| \widehat{\mathbf{A}}_{p,q} \mathbf{W}_q \widehat{\mathbf{A}}_{p,q}^{\top} \right\|_2 \leq 1.$$

Corollary 1: For $p=0,\ldots,s$, we have

$$\|\mathbf{B}_{pp}\|_2 \leq \alpha s.$$

Proof: By (16), we have

$$\begin{aligned} &\|\mathbf{B}_{pp}\|_2 \\ &= \left\| \alpha \sum_{j=p+1}^{s} \widehat{\mathbf{A}}_{p,j} \mathbf{W}_j \widehat{\mathbf{A}}_{p,j}^{\top} + \alpha \sum_{i=0}^{p-1} \widehat{\mathbf{A}}_{i,p}^{\top} \mathbf{W}_i \widehat{\mathbf{A}}_{i,p} \right\|_2 \\ &\leq \alpha \sum_{j=p+1}^{s} \left\| \widehat{\mathbf{A}}_{p,j} \mathbf{W}_j \widehat{\mathbf{A}}_{p,j}^{\top} \right\|_2 + \alpha \sum_{i=0}^{p-1} \left\| \widehat{\mathbf{A}}_{i,p}^{\top} \mathbf{W}_i \widehat{\mathbf{A}}_{i,p} \right\|_2 \\ &\leq \alpha s. \end{aligned}$$

Lemma 2: For $p < q$, we have

$$\left\|\widehat{\mathbf{A}}_{p,q}\mathbf{W}_q^{\frac{1}{2}}\mathbf{Z}_{q,p}^{-\frac{1}{2}}\right\|_2 \leq 1 \tag{24}$$

$$\left\|\widehat{\mathbf{A}}_{p,q}^{\top}\mathbf{W}_p^{\frac{1}{2}}\mathbf{Z}_{p,q}^{-\frac{1}{2}}\right\|_2 \leq 1. \tag{25}$$

Proof: We shall prove only (24); the proof for (25) is similar.

$$\left\|\widehat{\mathbf{A}}_{p,q}\mathbf{W}_q^{\frac{1}{2}}\mathbf{Z}_{q,p}^{-\frac{1}{2}}\right\|_2 \leq \left\|\widehat{\mathbf{A}}_{p,q}\mathbf{W}_q^{\frac{1}{2}}\right\|_2 \left\|\mathbf{Z}_{q,p}^{-\frac{1}{2}}\right\|_2 \leq \left\|\widehat{\mathbf{A}}_{p,q}\mathbf{W}_q^{\frac{1}{2}}\right\|_2.$$

The last inequality is due to the fact that the unweighted degrees in the diagonal of $\mathbf{Z}_{q,p}$ are at least 1. Finally, note that $\|\widehat{\mathbf{A}}_{p,q}\mathbf{W}_q^{\frac{1}{2}}\|_2$ equals to the largest singular value of the (rectangular) matrix $\widehat{\mathbf{A}}_{p,q}\mathbf{W}_q^{\frac{1}{2}}$. Hence,

$$\left\|\widehat{\mathbf{A}}_{p,q}\mathbf{W}_q^{\frac{1}{2}}\right\|_2 = \sqrt{\left\|\widehat{\mathbf{A}}_{p,q}\mathbf{W}_q\widehat{\mathbf{A}}_{p,q}^{\top}\right\|_2} \leq 1.$$

The last inequality is due to Lemma 1.

Corollary 2: For $p < q$, we have

$$\|\mathbf{B}_{pq}\|_2 \leq 1 - \alpha$$
$$\|\mathbf{B}_{qp}\|_2 \leq 1 - \alpha.$$

Proof: The results are direct consequences of Lemma 2 and (17), (18).

Proof (Proof of Theorem 1): We write

$$\mathbf{B} = \mathbf{B}_1 + \mathbf{B}_2 = \begin{bmatrix} \mathbf{B}_{00} & & & \\ & \mathbf{B}_{11} & & \\ & & \ddots & \\ & & & \mathbf{B}_{ss} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{B}_{01} & \cdots & \mathbf{B}_{0\,s} \\ \mathbf{B}_{10} & \mathbf{0} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{B}_{s-1,s} \\ \mathbf{B}_{s0} & \cdots & \mathbf{B}_{s,s-1} & \mathbf{0} \end{bmatrix}.$$

By Corollary 1, we have

$$\|\mathbf{B}_1\|_2 = \max_p \|\mathbf{B}_{pp}\| \leq \alpha s.$$

Let $\mathbf{x} = (\mathbf{x}_0^{\top}, \mathbf{x}_1^{\top}, \ldots, \mathbf{x}_s^{\top})^{\top}$ be a unit vector. By Corollary 2, we have

$$\|\mathbf{B}_2\mathbf{x}\|_2^2 = \sum_{p=0}^{s}\left\|\sum_{q\neq p}\mathbf{B}_{pq}\mathbf{x}_q\right\|_2^2 \leq \sum_{p=0}^{s}\left(\sum_{q\neq p}\|\mathbf{B}_{pq}\|_2\|\mathbf{x}_q\|_2\right)^2$$
$$\leq \sum_{p=0}^{s}(1-\alpha)^2\left(\sum_{q\neq p}\|\mathbf{x}_q\|_2\right)^2$$
$$= (1-\alpha)^2\left[\sum_{p=0}^{s}\|\mathbf{x}_p\|_2^2 + (s-1)\left(\sum_{q=0}^{s}\|\mathbf{x}_q\|_2\right)^2\right]$$
$$\leq (1-\alpha)^2\left[\sum_{p=0}^{s}\|\mathbf{x}_p\|_2^2 + (s-1)(s+1)\sum_{q=0}^{s}\|\mathbf{x}_q\|_2^2\right]$$
$$= (1-\alpha)^2\left[1 + (s-1)(s+1)\right] = (1-\alpha)^2 s^2.$$

Thus, we have $\|\mathbf{B}_2\|_2 \leq (1-\alpha)s$. Finally, we have

$$\|\mathbf{B}\|_2 \leq \|\mathbf{B}_1\|_2 + \|\mathbf{B}_2\|_2 \leq \alpha s + (1-\alpha)s = s.$$

C. A Bound of the Generalization Gap

In this part, we present a bound of the generalization gap on the core layer of a SCN model. Consider a single-layer SCN model for a given simplicial complex $\mathcal{S}$, representations $\mathbf{X}$, and labels $\mathbf{Y}$. A sample is a pair $z = (\mathcal{S}_p^i, y)$ consisting of a simplex $\mathcal{S}_p^i \in \mathcal{S}$ and the associated label y for $\mathcal{S}_p^i$. The set of all samples is denoted by D. Let $\mathsf{T} \subset \mathsf{D}$ be a training set of size m consisting of i.i.d. samples drawn from D, where $m \leq \sum_{p=0}^{s} n_p$. The SCN model is trained with T using SGD. Since SGD is a randomized algorithm, we denote by A a particular randomization used. The loss function value with respect to a sample z is denoted by $\ell(\mathsf{T}, \mathsf{A}, z)$. The *generalization error* is defined by

$$R(\mathsf{T}, \mathsf{A}) := \mathbb{E}_{z\sim\mathsf{D}}[\ell(\mathsf{T}, \mathsf{A}, z)].$$

The *empirical risk* is defined by:

$$R_{\text{emp}}(\mathsf{T}, \mathsf{A}) := \frac{1}{m}\sum_{i=1}^{m}\ell(\mathsf{T}, \mathsf{A}, z_i),$$

where z_i is the i-th sample in T. The *generalization gap* is given by

$$G(\mathsf{T}) = \mathbb{E}_{\mathsf{A}}[R(\mathsf{T}, \mathsf{A}) - R_{\text{emp}}(\mathsf{T}, \mathsf{A})].$$

Theorem 2 (Generalization Gap): Consider a single-layer simplicial complex model trained on a dataset T of size m using the SGD algorithm for T iterations. The following expected generalization gap holds for all $0 < \delta < 1$, with probability at least $1 - \delta$,

$$G(\mathsf{T}) \leq \frac{\kappa}{m} + \frac{2\kappa + \gamma_\ell}{\sqrt{m}} \cdot \sqrt{\frac{\ln\frac{1}{\delta}}{2}}. \tag{26}$$

Here,

$$\kappa = \frac{2\alpha_\ell^2\alpha_\sigma^2\left\{\left[1 + \eta(\alpha_\ell\nu_\sigma + \nu_\ell\alpha_\sigma^2)\|\mathbf{B}\|_2^2\|\mathbf{X}\|_2^2\right]^{\top} - 1\right\}}{\alpha_\ell\nu_\sigma + \nu_\ell\alpha_\sigma^2},$$

γ_ℓ is the upper bound of the loss function ℓ, α_ℓ and α_σ are the Lipschitz constants of the loss function and activation function, respectively, ν_ℓ and ν_σ are the smoothness constants of the loss function and activation function, respectively, and η is the learning rate.

Theorem 2 shows that the bound in (26) reduces at the rate of $O(1/\sqrt{m})$ as m increases. The filter $\mathbf{B}$ is designed so that $\|\mathbf{B}\|_2$ is independent of $n := \sum_{p=0}^{s} n_p$, the size of the simplicial complex. Thus, when $\|\mathbf{X}\|_2 = \max\{\|\mathbf{X}_0\|_2, \ldots, \|\mathbf{X}_s\|_2\}$

is bounded independent of n (e.g., through data normalization or the nature of the input features), the bound in (26) will be uniform to all simplicial complexes of order s and converge to zero as $m \to \infty$. In particular, the proof of the bound of the generalization gap of SCN in Theorem 2 can be adapted from the proof in [42] by replacing the hypergraph filter in [42] with the simplicial complex filter in (15) and using the newly developed bound in Theorem 1.

V. Experiment

In this section, extensive experiments are conducted to estimate the performance of the proposed model. We first describe the specifications of the learning tasks and datasets. Then, we introduce the compared baseline approaches. Subsequently, we show and discuss the experimental results. More detailed experimental analyses are also included.

A. Tasks and Datasets

We evaluate the proposed model using different learning tasks, i.e., node classification, edge classification, and triangle classification, on several datasets under the semi-supervised setting. The detailed specifications of tasks and datasets are listed as follows.

- *Node (0-simplex) Classification:*
  We adopt the Citeseer [44], Cora [45], Citeseer co-citation [13], Pubmed co-citation [13], Cora co-authorship [13], ACMv9 [46], Citationv1 [46], and DBLPv7 [46] datasets for semi-supervised node classifications. In Citeseer and Cora, the relations are pair-wise where nodes are documents and edges are citation links. In Citeseer co-citation, Pubmed co-citation, and Cora co-authorship, the relations are high-order where each node is a paper connecting with a hyperedge that contains papers citing this node for co-citation data or linking with a hyperedge meaning an author for co-authorship data. Particularly, we preprocess ACMv9, Citationv1, and DBLPv7 as co-citation data. Each paper in ACMv9 and Citationv1 belongs to one of the following categories according to its research topic, including "Artificial Intelligent", "Computer Vision", and "Information Security". And the classes in DBLPv7 contain "Database", "Artificial Intelligent", "Computer Vision", "Information Security", and "Networking". In addition, the papers are represented by the sparse bag-of-words features extracted from the paper title. We build two settings on the Citeseer and Cora datasets. In the first setting, we employ the graph obtained from [7] and name the datasets Citeseer-I and Cora-I for the sake of distinction. To check the impact of the amount of graph information on performance, in the second setting, we pick up graphs with node-degree in the range [1, 4] based on the graphs obtained from [7] and obtain datasets named Citeseer-II and Cora-II. It is clear that the graphs from Citeseer-II and Cora-II are sparser than those from Citeseer-I and Cora-I, respectively.
  Likewise, we construct two settings on the Citeseer co-citation, Pubmed co-citation, and Cora co-authorship datasets. Based on the graphs given in [13], the first setting samples graphs with edge-degree equaling 2 and calls the datasets Citeseer co-I, Pubmed co-I, and Cora co-I. The second setting selects graphs with edge-degree in the range [2, 3] and obtains datasets named Citeseer co-II, Pubmed co-II, and Cora co-II. Again the graphs from Citeseer co-II, Pubmed co-II, and Cora co-II are sparser than those from Citeseer co-I, Pubmed co-I, and Cora co-I, respectively. Similar operations are conducted on the ACMv9, Citationv1, and DBLPv7 datasets. We name the datasets in the first setting ACM co-I, Citation co-I, and DBLP co-I and those in the second setting ACM co-II, Citation co-II, and DBLP co-II.
  In each dataset constructed, we randomly pick up 20 nodes per class as training data and keep the remaining as test data.
- *Edge (1-simplex) Classification:*
  We use the Brain [47] and DAWN [48] datasets to perform semi-supervised edge classifications, where an edge connects two nodes. In Brain, the nodes are brain regions from an MRI scan while the edges connect either regions with high fMRI correlation (labeled fMRI correlation) or regions with similar activation patterns (labeled Jaccard index similarity). We select 2000 and 3000 edges per class and remove the duplicate ones, thus obtaining graphs with edge-size being 3906 and 5778, named Brain-edge-I and Brain-edge-II, respectively. In DAWN, the nodes represent drugs, and edges are combinations of drugs taken by a patient prior to an emergency room visit. The edge classes indicate the patient disposition. We pick up four classes and 200 and 500 edges per class, thus achieving graphs with edge-size being 800 and 2000, called DAWN-edge-I and DAWN-edge-II, respectively. Specifically, in each dataset, we randomly select 100 edges per class as training data and employ the remaining as test data.
- *Triangle (2-simplex) Classification:*
  Regarding the semi-supervised triangle classification, we employ the MAG [48], [49] and DAWN [48] datasets to construct tasks, where a triangle contains three nodes. MAG is a subset of the Microsoft Academic Graph where nodes are authors and hyperedges correspond to a publication from those authors. The class labels of hyperedges are computer science conferences. Specifically, we select the hyperedges with exactly three authors to form triangles. We pick up four classes (i.e., KDD, WWW, ICML, and NeurIPS) and 200 and 500 triangles per class, thus obtaining graphs with edge-size being 800 and 2000, called MAG-tri-I and MAG-tri-II, respectively. For DAWN, we select four classes and 200 and 500 triangles per class, thus attaining graphs with edge-size being 800 and 2000, called DAWN-tri-I and DAWN-tri-II, respectively. Specifically, we randomly sample 100 triangles per class as training data and utilize the remaining as test data in each dataset.

More information about the datasets refers to the corresponding references. Table I also presents the statistical information on the constructed datasets. Specifically, for node classification tasks, the initial node features are preprocessed in advance

TABLE I
STATISTICAL INFORMATION ON THE CONSTRUCTED DATASETS

Task	Dataset	Graph type	#size	#feature	#class	#training (test) sample	#0-simplex	#1-simplex	#2-simplex
Node classification	Citeseer-I	pair-wise	3327 × 3327	3703	6	120 (3207)	3327	4552	307
	Cora-I		2708 × 2708	1433	7	140 (2568)	2708	5278	220
	Citeseer-II		2371 × 2371	3703	6	120 (2251)	2371	1740	9
	Cora-II		1446 × 1446	1433	7	140 (1306)	1446	1085	2
	Citeseer co-I	high-order	775 × 504	3703	6	120 (655)	775	504	4
	Pubmed co-I		2764 × 3136	500	3	60 (2704)	2764	3136	91
	Cora co-I		5460 × 2925	1000	10	200 (5260)	5460	2925	3
	ACM co-I		2216 × 1464	4205	3	60 (2156)	2216	1464	14
	Citation co-I		1494 × 995	3190	3	60 (1434)	1494	995	12
	DBLP co-I		1891 × 1176	3605	5	100 (1791)	1891	1176	8
	Citeseer co-II		1103 × 755	3703	6	120 (983)	1103	755	279
	Pubmed co-II		3294 × 7161	500	3	60 (3234)	3294	7161	2192
	Cora co-II		8141 × 4137	1000	10	200 (7941)	8141	6253	1235
	ACM co-II		3285 × 2188	5145	3	60 (3225)	3285	3520	830
	Citation co-II		2218 × 1471	3954	3	60 (2158)	2218	2346	524
	DBLP co-II		2908 × 1823	4525	5	100 (2808)	2908	3014	710
Edge classification	Brain-edge-I	high-order	638 × 3906	-	2	200 (3706)	638	3906	1557
	DAWN-edge-I		581 × 800	-	4	400 (400)	581	800	15
	Brain-edge-II		638 × 5778	-	2	200 (5578)	638	5778	5591
	DAWN-edge-II		892 × 2000	-	4	400 (1600)	892	2000	179
Triangle classification	MAG-tri-I	high-order	2177 × 800	-	4	400 (400)	2177	2348	800
	DAWN-tri-I		408 × 800	-		400 (400)	408	1970	800
	MAG-tri-II		4923 × 2000	-		400 (1600)	4923	5867	2000
	DAWN-tri-II		644 × 2000	-		400 (1600)	644	4127	2000

#Size means the size of given graph and #0-simplex, #1-simplex, and #2-simplex indicate the numbers of 0-simplex, 1-simplex, and 2-simplex, respectively, based on the given graph.

or obtained from [7], [13]. While for the edge and triangle classifications, the initial features of edges and triangles are not available. Therefore, for these two tasks, we use identity matrices as the initial feature representations.

B. Baseline Methods

- *NN:* 1-Nearest Neighbor (NN) is an algorithm that stores all available samples and classifies new samples based on a similarity measure. NN is a method without considering the graph structure, and we adopt the NN tool provided by *sklearn*.[3]
- *SVM:* We apply Support Vector Machine (SVM) [50] to the labeled samples to train a classifier, which is then used to make predictions for the unlabeled samples. We use the SVM tool provided by *sklearn*. Similarly, SVM is treated as a method without considering the graph structure.
- *GCN:* Graph Convolutional Network (GCN) [7] is a classical graph-based convolutional network performing convolution operations on a standard graph.
- *GAT:* Graph Attention Network (GAT) [21] is a convolution-style neural network, which leverages masked self-attentional layers to attend over the neighborhoods' features of nodes.
- *Ortho-GConv;* Orthogonal Graph Convolution (Ortho-GConv) [24] is a graph-based convolutional network maintaining the orthogonality of the feature transformation, thus augmenting graph neural network backbones to stabilize the model training.
- *HGNN:* HyperGraph Neural Networks (HGNN) [12] is a classical hypergraph-based convolutional network adopting hypergraph Laplacian and truncated chebyshev polynomials to generalize the convolution operations to hypergraph learning.
- *HNHN:* Hypergraph Networks with Hyperedge Neurons (HNHN) [15] is a hypergraph convolution network updating the representations of nodes and hyperedges alternatively.
- *UniGNN:* Unified Graph Neural Network (UniGNN) [37] is a unified framework for graph and hypergraph neural networks. Here, we adopt the generalized UniSAGE model in the experiments.
- *LEGCN:* Line Expansion Graph Convolutional Network (LEGCN) [38] is a hypergraph expansion, which attempts to study the symmetric nature of data co-occurrence.
- *HCoN:* Hypergraph Collaborative Network (HCoN) [16] obtains latent node and hyperedge representations collaboratively based on both nodes and hyperedges from the previous layer.

Among these baseline models, NN and SVM are methods ignoring the graph structure information. Specifically, GCN, GAT, and Ortho-GConv are representative approaches for standard graphs. In addition, HGNN, HNHN, UniGNN, LEGCN, and HCoN could be treated as state-of-the-art hypergraph learning models.

C. Common Experimental Settings

We follow [36] similarly and set some common experimental settings for fair comparisons. For the graph learning models, we adopt two-layer models with the final output dimension being the number of classes and use the cross-entropy error on labeled samples as the objective function. Specifically, we set the latent feature dimension, learning rate, and L2 regularization factor

[3]https://scikit-learn.org/

TABLE II
RESULTS (%) OF DIFFERENT GRAPH LEARNING METHODS FOR SEMI-SUPERVISED NODE CLASSIFICATION ON PAIR-WISE GRAPHS

Method	Citeseer-I		Cora-I		Citeseer-II		Cora-II	
	Acc.	p	Acc.	p	Acc.	p	Acc.	p
NN	35.63 ± 3.78	0.00	29.95 ± 12.22	0.00	34.65 ± 3.83	0.00	29.92 ± 13.52	0.00
SVM	44.95 ± 3.62	0.00	37.20 ± 3.50	0.00	38.74 ± 5.31	0.00	35.50 ± 3.75	0.00
GCN [7]	61.64 ± 2.16	0.00	68.57 ± 3.46	0.00	59.42 ± 1.79	0.00	64.15 ± 2.12	0.00
GAT [21]	63.57 ± 2.11	0.00	71.26 ± 1.84	0.00	60.53 ± 1.22	0.00	63.14 ± 1.19	0.00
Ortho-GConv [24]	63.72 ± 1.11	0.00	70.95 ± 2.64	0.00	59.34 ± 2.63	0.00	62.99 ± 3.26	0.00
HGNN [12]	63.78 ± 2.20	0.00	73.63 ± 1.75	0.00	58.51 ± 1.63	0.00	60.75 ± 2.20	0.00
UniGNN [37]	64.61 ± 1.88	0.00	73.64 ± 2.29	0.02	57.23 ± 1.98	0.00	60.23 ± 1.78	0.00
HCoN [16]	65.74 ± 1.46	0.00	74.16 ± 1.60	0.00	58.88 ± 1.53	0.00	61.13 ± 1.95	0.00
SCN (ours)	**67.68 ± 1.53**		**74.93 ± 1.57**		**62.57 ± 1.42**		**68.52 ± 1.82**	

TABLE III
RESULTS (%) OF DIFFERENT GRAPH LEARNING METHODS FOR SEMI-SUPERVISED NODE CLASSIFICATION ON HIGH-ORDER GRAPHS

Method	Citeseer co-I		Pubmed co-I		Cora co-I		Citeseer co-II		Pubmed co-II		Cora co-II	
	Acc.	p	Acc.	p	Acc.	p	Acc.	p	Acc.	p	Acc.	p
NN	38.95 ± 4.19	0.00	35.99 ± 3.88	0.00	8.96 ± 3.90	0.00	28.88 ± 9.55	0.00	35.72 ± 4.24	0.00	7.83 ± 3.83	0.00
SVM	47.32 ± 4.59	0.00	34.88 ± 7.49	0.00	27.43 ± 9.21	0.00	45.64 ± 6.85	0.00	33.96 ± 7.55	0.00	30.49 ± 8.48	0.00
HGNN [12]	64.27 ± 1.62	0.00	40.36 ± 2.05	0.00	45.84 ± 3.14	0.00	64.06 ± 1.64	0.00	43.61 ± 2.22	0.00	45.26 ± 1.69	0.00
HNHN [15]	64.79 ± 0.86	0.00	39.40 ± 2.38	0.00	43.75 ± 0.83	0.00	62.99 ± 0.80	0.00	46.12 ± 1.13	0.00	44.56 ± 1.12	0.00
UniGNN [37]	64.95 ± 1.65	0.00	39.91 ± 2.27	0.00	45.65 ± 2.51	0.00	64.28 ± 3.17	0.00	43.73 ± 2.60	0.00	46.21 ± 1.84	0.00
LEGCN [38]	64.32 ± 1.47	0.00	40.44 ± 1.21	0.00	45.38 ± 2.54	0.00	64.68 ± 1.64	0.00	40.77 ± 0.61	0.00	46.24 ± 1.54	0.00
HCoN [16]	65.94 ± 1.64	0.00	43.33 ± 2.07	0.00	46.64 ± 2.79	0.08	65.96 ± 1.50	0.02	46.88 ± 2.09	0.00	46.80 ± 1.70	0.03
SCN (ours)	**67.15 ± 1.64**		**45.40 ± 1.95**		**46.90 ± 2.65**		**66.26 ± 1.49**		**48.60 ± 1.77**		**47.31 ± 1.87**	
	ACM co-I		Citation co-I		DBLP co-I		ACM co-II		Citation co-II		DBLP co-II	
Method	Acc.	p	Acc.	p	Acc.	p	Acc.	p	Acc.	p	Acc.	p
NN	41.99 ± 4.13	0.00	46.51 ± 3.68	0.00	40.52 ± 4.19	0.00	43.81 ± 2.52	0.00	43.59 ± 7.70	0.00	39.23 ± 3.65	0.00
SVM	49.58 ± 4.67	0.00	45.44 ± 8.79	0.00	49.94 ± 6.41	0.00	49.47 ± 4.88	0.00	47.74 ± 7.04	0.00	47.33 ± 8.51	0.00
HGNN [12]	64.39 ± 2.74	0.00	66.43 ± 3.73	0.00	66.97 ± 2.12	0.00	67.96 ± 3.22	0.00	71.18 ± 2.30	0.00	69.70 ± 2.18	0.00
HNHN [15]	60.77 ± 0.81	0.00	61.94 ± 0.91	0.00	64.28 ± 0.82	0.00	62.70 ± 0.77	0.00	65.00 ± 1.71	0.00	66.16 ± 0.90	0.00
UniGNN [37]	64.20 ± 2.62	0.00	64.07 ± 3.54	0.00	65.82 ± 2.11	0.00	67.97 ± 2.70	0.00	70.05 ± 2.85	0.00	69.27 ± 2.47	0.00
LEGCN [38]	62.61 ± 2.70	0.00	63.29 ± 2.60	0.00	65.40 ± 1.69	0.00	66.03 ± 2.68	0.00	69.32 ± 1.05	0.00	68.32 ± 2.06	0.00
HCoN [16]	64.69 ± 2.55	0.00	66.72 ± 3.64	0.00	67.39 ± 2.14	0.00	68.20 ± 3.17	0.00	71.38 ± 2.29	0.00	69.99 ± 2.13	0.00
SCN (ours)	**65.19 ± 2.54**		**67.59 ± 3.42**		**67.95 ± 2.19**		**69.20 ± 3.31**		**72.50 ± 2.44**		**71.48 ± 2.32**	

to be 512, 0.01, and 0.0001, respectively, for fair comparisons. We train the models for 200 epochs and use the ReLU as the activation function $\sigma(\cdot)$. Particularly, HNHN, HCoN, and the proposed SCN model contain extra parameters. According to the suggestion in [15], we tune the parameters α and β of HNHN in a range $\{-3, -2.5, -2, -1.5, -1, -0.5, 0, 0.5, 1\}$. Besides, for a fair comparison with HCoN, we set $\alpha = \beta$, whose optimal value is searched in the range space $\{0.1, 0.2, ..., 0.9\}$. For the proposed SCN model, α is tuned in the range space $\{0.1, 0.2, ..., 0.9\}$.

Since SCN leverages simplex-level information based on a given graph, we describe the construction of simplices as follows. For simplicity, we set $s = 2$, i.e., the extracted simplices consist of 0-, 1-, and 2-simplices only in experiments. In other words, suppose a hyperedge from the given graph contains four nodes and these nodes are connected to each other, i.e., they form a tetrahedron, we break this tetrahedron into triangles. We show that using SCN with just $s = 2$ can produce good results already in experiments. Specifically, for the node (or edge or triangle) classification task, we keep the input 0-simplices (or 1-simplices or 2-simplices) of SCN as the same as those of baseline methods, and the remaining simplices are built by considering both direct and indirect connections.

For each model on each dataset, we repeat the experiments for 20 trials to obtain the mean and standard deviation results. Moreover, we carry out significance tests between baseline models and SCN to evaluate the significance of the proposed method. Specifically, the significance level is set to be 0.05, meaning that the proposed SCN model significantly performs better than a baseline model if the p-value is smaller than 0.05.

D. Results on Node Classification

Tables II and III list the results of different methods for semi-supervised node classification on pair-wise and high-order graphs, respectively. Specifically, for graph learning on the pair-wise graphs, we compare the proposed model with NN, SVM, GCN, GAT, Ortho-GConv, HGNN, UniGNN, and HCoN. For graph learning on the high-order graphs, we make comparisons with NN, SVM, HGNN, HNHN, UniGNN, LEGCN, and HCoN. Generally, the proposed SCN method obtains better performance than the baseline models. Based on the significance tests, SCN significantly outperforms the baseline methods on

TABLE IV
RESULTS (%) OF DIFFERENT GRAPH LEARNING METHODS FOR SEMI-SUPERVISED EDGE CLASSIFICATION

	Brain-edge-I		DAWN-edge-I		Brain-edge-II		DAWN-edge-II	
Method	Acc.	p	Acc.	p	Acc.	p	Acc.	p
HGNN [12]	56.95 ± 0.99	0.00	28.46 ± 1.79	0.00	56.09 ± 0.95	0.00	29.99 ± 1.03	0.00
HNHN [15]	55.94 ± 0.42	0.00	30.30 ± 1.81	0.00	52.08 ± 0.35	0.00	30.31 ± 0.51	0.00
UniGNN [37]	57.39 ± 1.09	0.00	31.14 ± 2.15	0.00	57.29 ± 1.15	0.00	31.99 ± 1.06	0.00
LEGCN [38]	56.83 ± 1.10	0.00	31.98 ± 1.79	0.00	56.65 ± 1.01	0.00	32.32 ± 1.09	0.00
HCoN [16]	58.70 ± 1.02	0.00	34.10 ± 1.49	0.00	58.65 ± 0.90	0.00	33.70 ± 0.83	0.00
SCN (ours)	**74.61 ± 0.90**		**65.09 ± 1.76**		**75.64 ± 0.82**		**59.42 ± 0.94**	

most tasks except Cora co-I, demonstrating the effectiveness of the proposed model. We draw several interesting observations as follows.

- NN and SVM perform worse than graph-based models, which indicates that learning on graph-structured data requires specific models exploiting the graph structure information.
- On the Citeseer-I and Cora-I datasets, hypergraph learning models work better than standard graph ones. However, this phenomenon is opposite on the Citeseer-II and Cora-II datasets. The reason may be that Citeseer-II and Cora-II are sampled based on Citeseer-I and Cora-I, respectively, and the structure information may not be very sufficient for hypergraph learning models after sampling.
- The performance of the tested methods on the Citeseer-I and Cora-I datasets is generally better than that on the Citeseer-II and Cora-II datasets, which we conjecture that the information of Citeseer-II and Cora-II is less than Citeseer-I and Cora-I since the former ones are sampled from the latter ones. Nevertheless, SCN performs quite good compared with the baseline models on the Citeseer-II and Cora-II datasets, the reason is that SCN further investigates the indirect relations among samples to aggregate more information.
- Compared with non-graph-based, standard graph-based, and hypergraph-based approaches, the proposed SCN achieves the best performance, verifying the effectiveness of exploiting simplex-level structure information for graph learning.
- HGNN is a classical hypergraph learning method, and HCoN further introduces edge information into hypergraph convolutional networks for collaborative learning. In particular, compared with HGNN and HCoN, SCN performs better, meaning that the proposed model is more effective in fully leveraging the graph structures because it exploits the simplex-level information to study direct and indirect information.

E. Results on Edge Classification

We present the results of different models for semi-supervised edge classification on the Brain-edge-I, DAWN-edge-I, Brain-edge-II, and DAWN-edge-II datasets in Table IV. Since the edge features are not available, we omit the baseline methods NN and SVM. Specifically, we compare the proposed model with HGNN, HNHN, UniGNN, LEGCN, and HCoN because these models can make use of high-order relations exhibited in the datasets. From Table IV, we observe that the proposed method significantly works better than baseline models and achieves considerable performance improvement. Concretely, SCN gains improvements of 15.91%, 30.99%, 16.99%, and 25.72% compared with the second best method, i.e., HCoN, on the Brain-edge-I, DAWN-edge-I, Brain-edge-II, and DAWN-edge-II datasets, respectively. Such achievement demonstrates the superior performance of the proposed model.

F. Results on Triangle Classification

Table V reports the results of different models for semi-supervised triangle classification on the MAG-tri-I, DAWN-tri-I, MAG-tri-II, and DAWN-tri-II datasets. Similar to the edge classification, we adopt HGNN, HNHN, UniGNN, LEGCN, and HCoN as baseline approaches, and the proposed model achieves better performance than these methods. SCN significantly outperforms the compared methods except on the MAG-tri-II dataset, further verifying the effectiveness of the proposed model.

G. Ablation Study

We exploit 0-, 1-, and 2-simplices in the proposed SCN model. Here, we perform ablation studies on the Pubmed co-I and Pubmed co-II datasets for node classification to evaluate the impact of different simplices. The results are listed in Table VI, where $\mathbf{X}_0$, $\mathbf{X}_1$, and $\mathbf{X}_2$ indicate the representations of 0-, 1-, and 2-simplices, respectively. We observe that different simplices play crucial roles and constitute important contributions to classification performance. Specifically, comparing the results in lines 4 and 7, the effectiveness of the learned representations is further enhanced by involving the 2-simplices. This phenomenon verifies the motivation and rationality of the proposed model.

H. Parameter Sensitivity Analysis

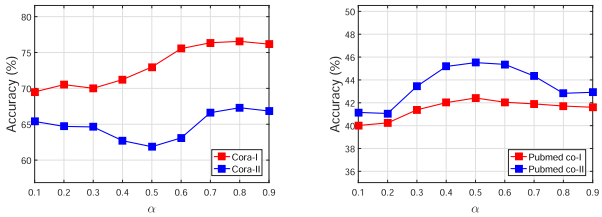
The proposed SCN model involves one parameter, i.e., α. To study the impact of α, we vary its value in the range $\{0.1, 0.2, ..., 0.9\}$ and perform node classification tasks on the Cora-I, Cora-II, Pubmed co-I, and Pubmed co-II datasets. The results of the one-trial experiment are shown in Fig. 4(a) and (b), where we see that the performance varies with α. Specifically, the best results are obtained on the Cora-I and Cora-II datasets when $\alpha = 0.8$. The trends of curves on the Pubmed co-I and Pubmed co-II datasets

TABLE V
RESULTS (%) OF DIFFERENT GRAPH LEARNING METHODS FOR SEMI-SUPERVISED TRIANGLE CLASSIFICATION

Method	MAG-tri-I		DAWN-tri-I		MAG-tri-II		DAWN-tri-II	
	Acc.	p	Acc.	p	Acc.	p	Acc.	p
HGNN [12]	31.11 ± 1.17	0.00	27.46 ± 2.07	0.00	32.99 ± 1.33	0.00	30.35 ± 1.58	0.00
HNHN [15]	32.59 ± 1.58	0.00	28.11 ± 1.96	0.00	33.78 ± 0.53	0.00	30.13 ± 0.37	0.00
UniGNN [37]	32.65 ± 1.71	0.00	29.82 ± 1.85	0.00	34.20 ± 0.95	0.00	31.59 ± 1.71	0.01
LEGCN [38]	31.26 ± 0.77	0.00	30.32 ± 1.67	0.02	32.33 ± 1.22	0.00	30.73 ± 1.26	0.00
HCoN [16]	33.69 ± 2.49	0.01	30.20 ± 2.05	0.00	35.22 ± 0.51	0.06	31.46 ± 1.55	0.00
SCN (ours)	$\mathbf{34.67 \pm 1.28}$		$\mathbf{30.93 \pm 1.89}$		$\mathbf{35.53 \pm 1.09}$		$\mathbf{32.24 \pm 1.45}$	

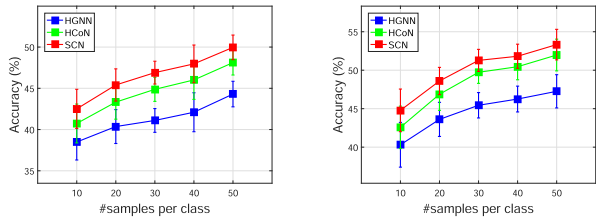
TABLE VI
ABLATION STUDY ON THE PUBMED CO-I AND PUBMED CO-II DATASETS

	$\mathbf{X}_0$	$\mathbf{X}_1$	$\mathbf{X}_2$	Pubmed co-I	Pubmed co-II
1	✓			41.23 ± 1.92	45.68 ± 1.64
2		✓		43.19 ± 1.94	43.31 ± 4.45
3			✓	36.18 ± 8.56	34.60 ± 9.38
4	✓	✓		43.67 ± 2.90	47.90 ± 1.48
5	✓		✓	41.71 ± 1.96	46.42 ± 1.57
6		✓	✓	43.67 ± 1.61	43.86 ± 2.87
7	✓	✓	✓	$\mathbf{45.40 \pm 1.95}$	$\mathbf{48.60 \pm 1.77}$



(a) Accuracy w.r.t. α (b) Accuracy w.r.t. α

Fig. 4. Parameter sensitivity study w.r.t. α of the proposed model.



(a) Pubmed co-I (b) Pubmed co-II

Fig. 5. Impact of the number of training samples on the performance.

TABLE VII
WALL-CLOCK TIME (S) OF DIFFERENT METHODS ON DIFFERENT DATASETS

Method	DAWN-edge-I		DAWN-edge-II	
	Time	Time per epoch	Time	Time per epoch
HGNN [12]	1.097	0.005 ± 0.045	1.168	0.006 ± 0.042
HNHN [15]	1.427	0.007 ± 0.044	1.573	0.008 ± 0.042
UniGNN [37]	1.070	0.005 ± 0.045	1.124	0.006 ± 0.044
LEGCN [38]	1.686	0.008 ± 0.058	1.812	0.009 ± 0.059
HCoN [16]	1.706	0.009 ± 0.044	1.833	0.009 ± 0.041
SCN (ours)	1.419	0.007 ± 0.041	1.883	0.009 ± 0.042

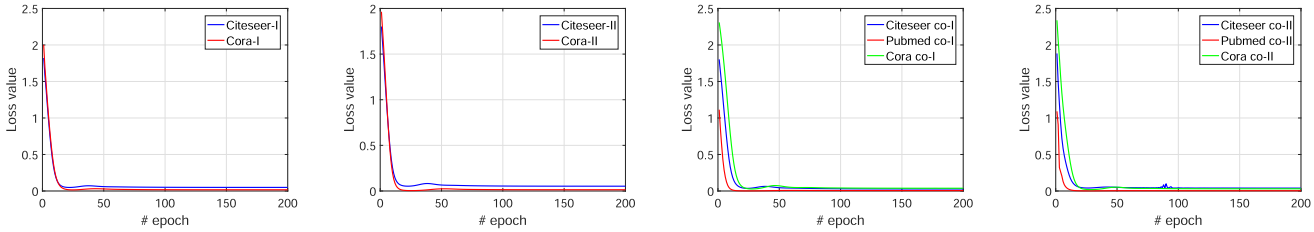
are similar, i.e., the performance improves and then drops as the value of α increases. The best performance is achieved when $\alpha = 0.5$ on these two datasets. Overall, the model is insensitive to the choice of α since the change in accuracy is within 6%. The parameter α facilitates the fine-tuning of the model without drastically altering its performance. This experiment validates the effectiveness of incorporating all simplex information for graph representation learning.

I. *Effect of Different Label Rates*

To estimate the impact of the number of training samples on the performance, we pick up different numbers of labeled samples randomly to perform experiments. Specifically, the experiments are carried out on the Pubmed co-I and Pubmed co-II datasets for node classifications. We set the numbers of labeled samples per class on the Pubmed co-I and Pubmed co-II datasets to be {10, 20, 30, 40, 50}. In particular, we compare SCN with HGNN and HCoN and run the experiments for 20 trials. The results are plotted in Fig. 5, where the performance of the test models improves with the number of labeled samples. The proposed SCN method consistently outperforms HGNN and HCoN, and the performance of SCN is improved with the increasing number of labeled samples, indicating the effectiveness of the proposed method.

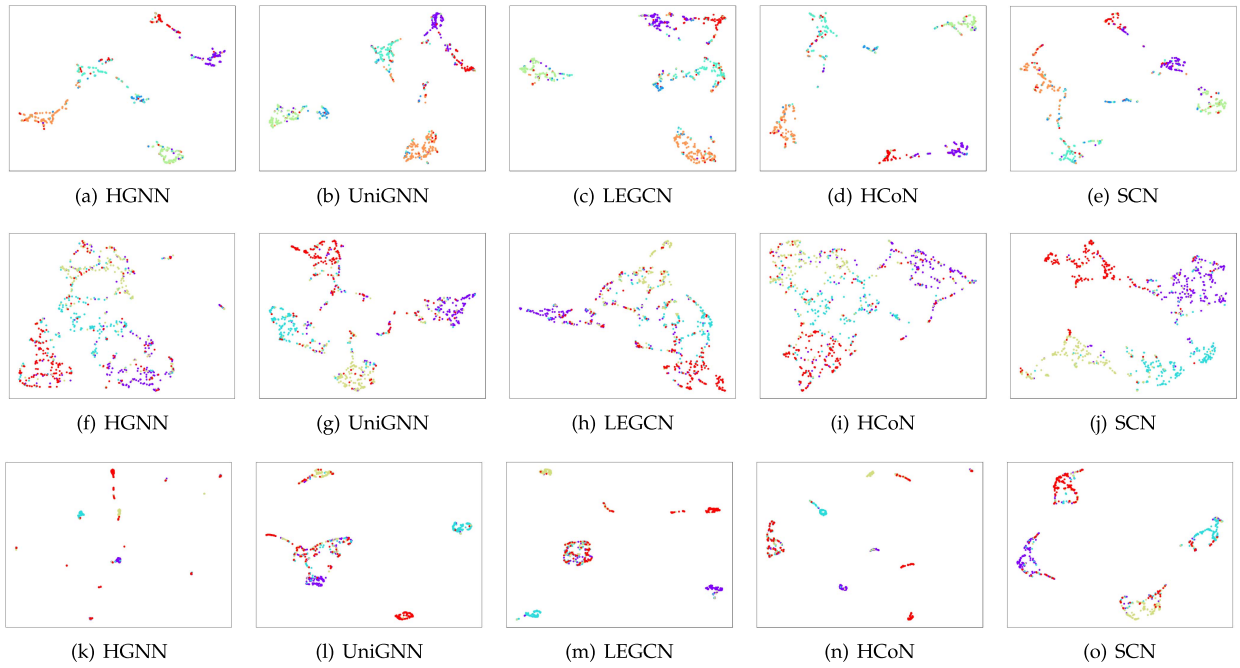
J. *Running Time Evaluation*

We evaluate the running time of the test methods by taking the DAWN-edge-I and DAWN-edge-II datasets as examples. The experiments are carried out on a server with NVIDIA GeForce RTX 3080Ti (12 GB) GPU cards. Table VII presents the experimental results measured in seconds wall-clock time. Following [7], we list the results w.r.t. the training time of all epochs and the mean training time per epoch, including forward pass, objective loss calculation, and backward pass. HGNN and UniGNN cost less running time since they mainly involve convolutional operations on nodes. The other models require slightly more running time than HGNN and UniGNN, the reason is that these models further incorporate edge information during the model training. Generally, the running time of the test models is roughly in the same order of magnitude. Note that the proposed model is able to obtain representations of all simplices simultaneously. Overall, SCN consumes reasonable training times and achieves superior performance.



(a) Loss value w.r.t. #epoch (b) Loss value w.r.t. #epoch (c) Loss value w.r.t. #epoch (d) Loss value w.r.t. #epoch

Fig. 6. Model learning curves depicting objective values versus the number of epochs of the proposed model.



(a) HGNN (b) UniGNN (c) LEGCN (d) HCoN (e) SCN

(f) HGNN (g) UniGNN (h) LEGCN (i) HCoN (j) SCN

(k) HGNN (l) UniGNN (m) LEGCN (n) HCoN (o) SCN

Fig. 7. Visualization of features represented by HGNN, UniGNN, LEGCN, HCoN, and SCN. The color indicates the class label in dataset. (a)–(e) are node features on the Citeseer co-I dataset, (f)–(j) are edge features on the DAWN-edge-I dataset, and (k)–(o) are triangle features on the MAG-tri-I dataset.

K. Convergence Property

To investigate the convergence of the training process of the proposed model, we perform experiments on several datasets used for node classifications. The curves of objective value versus the number of epochs are plotted in Fig. 6, where the curves flatten out quickly as the number of epochs increases, indicating the fast convergence of the algorithm. This experiment shows that SCN has a good convergence property that it converges within 20–30 epochs, and further verifies the effectiveness of the proposed method.

L. Visualization

We study qualitatively the effectiveness of the learned features of the proposed method by taking representative datasets for node, edge, and triangle classifications as examples. Specifically, we adopt the Uniform Manifold Approximation and Projection (UMAP) [51] tool to present discernible clusterings in a 2-D space to visualize the learned feature representations. The results are shown in Fig. 7, where we depict the representations obtained by HGNN, UniGNN, LEGCN, HCoN, and SCN. Fig. 7(a), (e), (f), and (j), and (k)–(o) exhibit the node, edge, and triangle features on the Citeseer co-I, DAWN-edge-I, and MAG-tri-I datasets, respectively. In general, the different classes in SCN are more separated than the other methods, i.e., the samples characterized by features learned by SCN have more clear data distribution boundaries, leading to better classification performance.

VI. Conclusion

In this paper, we present a general framework named Simplicial Complex Neural (SCN) network to exploit simplex-level information, hence studying both direct and indirect graph information. Specifically, the proposed model is able to obtain representations of all simplices simultaneously, which is achieved by aggregating and integrating information from all the simplices of the previous layer. To establish some desirable properties of the

proposed model, we derive theoretical analyses on the proposed model. We show that the simplicial complex filter underlies the proposed method has a uniform bound independent of the network size. Besides, we present a bound of the generalization gap of the proposed model. In experiments, we compare the proposed model with state-of-the-art approaches on the node, edge, and triangle classifications. The promising results validate the effectiveness of the proposed method.

It is worth noting that the balance parameter α of the proposed model in (13) can be extended to a vector. In the future, we plan to develop an advanced SCN model with a parameter vector being adaptively learned. Besides, the learning tasks in this paper are mainly classification, thus we are going to apply SCN to unsupervised learning tasks such as clustering [52] and community identification [53].

REFERENCES

[1] F. Xia et al., "Graph learning: A survey," *IEEE Trans. Artif. Intell.*, vol. 2, no. 2, pp. 109–127, Apr. 2021.

[2] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 1, pp. 249–270, Jan. 2022.

[3] 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.

[4] H. Zhu and P. Koniusz, "Simple spectral graph convolution," in *Proc. Int. Conf. Learn. Representations*, 2021.

[5] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Jan. 2021.

[6] H. Yuan, H. Yu, S. Gui, and S. Ji, "Explainability in graph neural networks: A taxonomic survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 5, pp. 5782–5799, May 2023.

[7] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representations*, 2017.

[8] H. Wu and M. K. Ng, "Hypergraph convolution on nodes-hyperedges network for semi-supervised node classification," *ACM Trans. Knowl. Discov. Data*, vol. 16, no. 4, pp. 1–19, 2022.

[9] F. M. Bianchi, D. Grattarola, L. Livi, and C. Alippi, "Graph neural networks with convolutional ARMA filters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 7, pp. 3496–3507, Jul. 2022.

[10] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. Neural Inf. Process. Syst.*, 2016, pp. 3844–3852.

[11] D. Zhou, J. Huang, and B. Schölkopf, "Learning with hypergraphs: Clustering, classification, and embedding," *Neural Inf. Process. Syst.*, vol. 19, pp. 1601–1608, 2006.

[12] Y. Feng, H. You, Z. Zhang, R. Ji, and Y. Gao, "Hypergraph neural networks," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 3558–3565.

[13] N. Yadati, M. Nimishakavi, P. Yadav, V. Nitin, A. Louis, and P. Talukdar, "HyperGCN: A new method of training graph convolutional networks on hypergraphs," in *Proc. Neural Inf. Process. Syst.*, 2019, pp. 1511–1522.

[14] Y. Gao, Z. Zhang, H. Lin, X. Zhao, S. Du, and C. Zou, "Hypergraph learning: Methods and practices," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 5, pp. 2548–2566, May 2022.

[15] Y. Dong, W. Sawin, and Y. Bengio, "HNHN: Hypergraph networks with hyperedge neurons," in *Proc. Graph Representations Beyond Workshop Int. Conf. Mach. Learn.*, 2020.

[16] H. Wu, Y. Yan, and M. K.-P. Ng, "Hypergraph collaborative network on vertices and hyperedges," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 3, pp. 3245–3258, Mar. 2023.

[17] J. Jonsson, *Simplicial Complexes of Graphs*. Berlin, Germany: Springer, 2007.

[18] M. T. Schaub, A. R. Benson, P. Horn, G. Lippner, and A. Jadbabaie, "Random walks on simplicial complexes and the normalized Hodge 1-Laplacian," *SIAM Rev.*, vol. 62, no. 2, pp. 353–391, 2020.

[19] M. E. Aktas and E. Akbas, "Hypergraph Laplacians in diffusion framework," in *Proc. Int. Conf. Complex Netw. Appl.*, Springer, 2021, pp. 277–288.

[20] Y. LeCun et al., "Convolutional networks for images, speech, and time series," *Handbook Brain Theory Neural Netw.*, vol. 3361, no. 10, 1995, Art. no. 1995.

[21] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Representations*, 2018.

[22] R. Li, S. Wang, F. Zhu, and J. Huang, "Adaptive graph convolutional neural networks," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 3546–3553.

[23] M. Chen, Z. Wei, Z. Huang, B. Ding, and Y. Li, "Simple and deep graph convolutional networks," in *Proc. Int. Conf. Mach. Learn.*, PMLR, 2020, pp. 1725–1735.

[24] K. Guo, K. Zhou, X. Hu, Y. Li, Y. Chang, and X. Wang, "Orthogonal graph neural networks," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 3996–4004.

[25] H. Wu, Y. Yan, Y. Ye, M. K. Ng, and Q. Wu, "Geometric knowledge embedding for unsupervised domain adaptation," *Knowl.-Based Syst.*, vol. 191, 2020, Art. no. 105155.

[26] S. Zorzi, S. Bazrafkan, S. Habenschuss, and F. Fraundorfer, "Polyworld: Polygonal building extraction with graph neural networks in satellite images," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 1848–1857.

[27] Y. Dong, Q. Liu, B. Du, and L. Zhang, "Weighted feature fusion of convolutional neural network and graph attention network for hyperspectral image classification," *IEEE Trans. Image Process.*, vol. 31, pp. 1559–1572, 2022.

[28] Q. Li, Z. Han, and X.-M. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 3538–3545.

[29] H. Wu, J. Long, N. Li, D. Yu, and M. K. Ng, "Adversarial auto-encoder domain adaptation for cold-start recommendation with positive and negative hypergraphs," *ACM Trans. Inf. Syst.*, vol. 41, no. 2, pp. 1–25, 2023.

[30] V. La Gatta, V. Moscato, M. Pennone, M. Postiglione, and G. Sperlí, "Music recommendation via hypergraph embedding," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 10, pp. 7887–7899, Oct. 2023.

[31] Y. Yang, C. Huang, L. Xia, Y. Liang, Y. Yu, and C. Li, "Multi-behavior hypergraph-enhanced transformer for sequential recommendation," in *Proc. 28th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2022, pp. 2263–2274.

[32] H. Wu et al., "Cold-start next-item recommendation by user-item matching and auto-encoders," *IEEE Trans. Serv. Comput.*, vol. 16, no. 4, pp. 2477–2489, Jul./Aug. 2023.

[33] H. Shi et al., "Hypergraph-induced convolutional networks for visual classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 10, pp. 2963–2972, Oct. 2019.

[34] D. Di, J. Zhang, F. Lei, Q. Tian, and Y. Gao, "Big-hypergraph factorization neural network for survival prediction from whole slide image," *IEEE Trans. Image Process.*, vol. 31, pp. 1149–1160, 2022.

[35] S. Bai, F. Zhang, and P. H. Torr, "Hypergraph convolution and hypergraph attention," *Pattern Recognit.*, vol. 110, 2021, Art. no. 107637.

[36] Y. Gao, Y. Feng, S. Ji, and R. Ji, "HGNN+: General hypergraph neural networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 3, pp. 3181–3199, Mar. 2023.

[37] J. Huang and J. Yang, "UniGNN: A unified framework for graph and hypergraph neural networks," in *Proc. 30th Int. Joint Conf. Artif. Intell.*, 2021, pp. 2563–2569.

[38] C. Yang, R. Wang, S. Yao, and T. Abdelzaher, "Semi-supervised hypergraph node classification on hypergraph line expansion," in *Proc. 31st ACM Int. Conf. Inf. Knowl. Manage.*, 2022, pp. 2352–2361.

[39] M. E. Aktas, E. Akbas, and A. E. Fatmaoui, "Persistence homology of networks: Methods and applications," *Appl. Netw. Sci.*, vol. 4, no. 1, pp. 1–28, 2019.

[40] M. Hardt, B. Recht, and Y. Singer, "Train faster, generalize better: Stability of stochastic gradient descent," in *Proc. Int. Conf. Mach. Learn.*, PMLR, 2016, pp. 1225–1234.

[41] S. Verma and Z.-L. Zhang, "Stability and generalization of graph convolutional neural networks," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2019, pp. 1539–1548.

[42] M. K. Ng, H. Wu, and A. Yip, "Stability and generalization of hypergraph collaborative networks," 2023, *arXiv:2308.02347*.

[43] O. Bousquet and A. Elisseeff, "Stability and generalization," *J. Mach. Learn. Res.*, vol. 2, pp. 499–526, 2002.

[44] I. Bhattacharya and L. Getoor, "Collective entity resolution in relational data," *ACM Trans. Knowl. Discov. Data*, vol. 1, no. 1, pp. 5–es, 2007.

[45] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI Mag.*, vol. 29, no. 3, pp. 93–93, 2008.

[46] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, "ArnetMiner: Extraction and mining of academic social networks," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2008, pp. 990–998.
[47] N. A. Crossley et al., "Cognitive relevance of the community structure of the human brain functional coactivation network," in *Proc. Nat. Acad. Sci. USA*, vol. 110, no. 28, pp. 11 583–11 588, 2013.
[48] I. Amburg, N. Veldt, and A. Benson, "Clustering in graphs and hypergraphs with categorical edge labels," in *Proc. Web Conf.*, 2020, pp. 706–717.
[49] A. Sinha et al., "An overview of Microsoft Academic Service (MAS) and applications," in *Proc. 24th Int. Conf. World Wide Web*, 2015, pp. 243–246.
[50] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.
[51] L. McInnes, J. Healy, N. Saul, and L. Grossberger, "UMAP: Uniform manifold approximation and projection," *J. Open Source Softw.*, vol. 3, no. 29, 2018, Art. no. 861.
[52] M. Liu, Y. Liu, K. Liang, S. Wang, S. Zhou, and X. Liu, "Deep temporal graph clustering," 2023, *arXiv:2305.10738*.
[53] J. Yang and J. Leskovec, "Defining and evaluating network communities based on ground-truth," in *Proc. ACM SIGKDD Workshop Mining Data Semantics*, 2012, pp. 1–8.

Hanrui Wu received the BS and PhD degrees from the School of Software Engineering from South China University of Technology, China, in 2013 and 2020, respectively. He was a postdoctoral research fellow with the Department of Mathematics, The University of Hong Kong, from 2020 to 2021. He is currently an associate professor with the Department of Computer Science, Jinan University, Guangzhou, China. His research interests include transfer learning, hypergraph learning, and their applications in recommendation systems and brain-computer interactions.

Andy Yip received the PhD degree from the University of California at Los Angeles in 2005. He served as an assistant professor with the National University Singapore in 2005–2014 and a lecturer with Hong Kong Baptist University in 2014–2017. He worked in the data science and finance industries afterwards. His research interests include data science and image processing.

Jinyi Long received the PhD degree from the South China University of Technology, Guangzhou, China. From 2012 to 2013, he worked with South China University of Technology as Lecturer. From 2014 to 2016, he was with the Systems Neuroscience Institute, University of Pittsburgh and the Miami Project to Cure Paralysis Lois Pope Life Center, University of Miami, USA, as a research fellow. Since 2017, he has been with the Department of Computer Science, Jinan University, Guangzhou, as a full professor. He works in the field of machine learning, brain signal processing, brain computer interactions, and neural engineering.

Jia Zhang (Member, IEEE) received the PhD degree from the Department of Artificial Intelligence, Xiamen University, Xiamen, China, in 2020. He is currently a lecturer with the College of Information Science and Technology, Jinan University, Guangzhou, China. He is broadly interested in machine learning and data mining. He is currently working on multi-label learning, data fusion, feature selection, and weakly supervised learning.

Michael K. Ng (Senior Member, IEEE) received the BSc and MPhil degrees from the University of Hong Kong in 1990 and 1992, respectively, and the PhD degree from the Chinese University of Hong Kong in 1995. He was a research fellow of Computer Sciences Laboratory with Australian National University from 1995 to 1997, and an assistant/associate professor of the University of Hong Kong from 1997 to 2005. He was a professor/chair professor with the Department of Mathematics, Hong Kong Baptist University from 2006 to 2019. He was a chair professor with Research Division of Mathematical and Statistical Science, The University of Hong Kong from 2019 to 2023. He is currently a chair professor in Mathematics and a chair professor in Data Science with Hong Kong Baptist University. His research interests include bioinformatics, image processing, scientific computing, and data mining. He is selected for the 2017 Class of Fellows of the Society for Industrial and Applied Mathematics. He obtained the Feng Kang Prize for his significant contributions in scientific computing. He serves on the Editorial Board members of several international journals.