Permutation Equivariant Graph Framelets for Heterophilous Graph Learning

Jianfei Li, Ruigang Zheng, Han Feng, Ming Li, *Member, IEEE*, and Xiaosheng Zhuang

***Abstract*— The nature of heterophilous graphs is significantly different from that of homophilous graphs, which causes difficulties in early graph neural network (GNN) models and suggests aggregations beyond the one-hop neighborhood. In this article, we develop a new way to implement multiscale extraction via constructing Haar-type graph framelets with desired properties of permutation equivariance, efficiency, and sparsity, for deep learning tasks on graphs. We further design a graph framelet neural network model permutation equivariant graph framelet augmented network (PEGFAN) based on our constructed graph framelets. The experiments are conducted on a synthetic dataset and nine benchmark datasets to compare the performance with other state-of-the-art models. The result shows that our model can achieve the best performance on certain datasets of heterophilous graphs (including the majority of heterophilous datasets with relatively larger sizes and denser connections) and competitive performance on the remaining.**

***Index Terms*— Graph framelets/wavelets, graph neural networks (GNNs), heterophily, permutation equivariance.**

I. Introduction

GRAPHS are ubiquitous data structures for a variety of real-life entities, such as traffic networks, social networks, citation networks, and chemoinformatics and bioinformatics networks. With the abstraction via graphs, many real-world problems that are related to networks and communities can be cast into a unified framework and solved by exploiting its underlying rich and deep mathematical theory as well as tremendously efficient computational techniques. In recent years, graph neural networks (GNNs) for graph learning, such as node classification [1], link prediction [2], and graph classification [3], have demonstrated their powerful learning ability and achieved remarkable performance [4], [5], [6], [7], [8]. In the particular field of node classification, many GNN models follow the *homophily* assumption, that is, the majority of edges connect nodes from the same classes (e.g., researchers in a citation network tend to cite each other from the same area), yet graphs with *heterophily*, that is, the majority of edges connect nodes from different classes [9], do exist in many real-world scenarios. A typical example is in a cyber network where a phishing attacker usually sends fraudulent messages to a large population of normal users (victims) in order to obtain sensitive information. We refer to [10] and [11] for the limitations of early GNNs on homophilous graphs and a recent survey paper [9] on GNNs for heterophilous graphs.

Heterophilous graphs differ from homophilous graphs not only *spatially* in terms of distribution beyond the one-hop neighborhood but also *spectrally* with larger oscillation in terms of the frequency distribution of graph signals under the graph Laplacian. Such properties bring challenges to learning on heterophilous graphs and demand new GNNs the ability to extract intrinsic information in order to achieve high performance. To enhance the influence of nodes from the same classes that are outside of one-hop neighborhoods, one common approach is based on the *multihop aggregation* to leverage information of k-hop neighborhoods, $k \geq 2$. Its effectiveness for heterophilous graphs is emphasized and theoretically verified in [12]. A common way to perform multihop aggregation is to utilize the powers of the adjacency matrix. Repeatedly applying Laplacian smoothing many times, prompted by using higher powers of adjacency matrix, can result in a convergence of vertex features within each connected component of the graph toward uninformative or identical values, a phenomenon referred to as oversmoothing [10], [13]. Moreover, they may lead to dense matrices and cause computation and storage burdens. To seek further improvement, it is thus desirable to consider an alternative spatial resolution of graphs other than the k-hop neighborhood. To answer this question, we work on the theory of wavelet/framelet systems on graphs, which brings a notion of *scale* on graphs and wavelets/framelets corresponding to such scales. In this article, we introduce and integrate a dedicated graph framelet system so as to perform *multiscale extraction* on graphs.

Actually, classical wavelets/framelets in the Euclidean domains, see, e.g., [14], [15], are well-known examples of multiscale representation, which have been extended to irregular domains such as graphs and manifolds under similar principles in recent years, see, e.g., [16], [17], [18], [19].

Manuscript received 26 June 2023; revised 16 October 2023 and 11 January 2024; accepted 23 February 2024. Date of publication 14 March 2024; date of current version 4 September 2024. The work of Han Feng was supported in part by the Research Grants Council of Hong Kong Special Administrative Region, China, under Project CityU 11303821 and Project CityU 11315522. The work of Ming Li was supported in part by the Key Research and Development Program of Zhejiang Province under Grant 2024C03262, in part by the National Natural Science Foundation of China under Grant 62172370 and Grant U21A20473, and in part by Zhejiang Provincial Natural Science Foundation under Grant LY22F020004. The work of Xiaosheng Zhuang was supported in part by the Research Grants Council of Hong Kong Special Administrative Region, China, under Project CityU 11309122 and Project CityU 11302023. *(Jianfei Li and Ruigang Zheng contributed equally to this work.) (Corresponding authors: Ming Li; Xiaosheng Zhuang.)*

Jianfei Li, Ruigang Zheng, Han Feng, and Xiaosheng Zhuang are with the Department of Mathematics, City University of Hong Kong, Hong Kong, SAR, China (e-mail: jianfeili2-c@my.cityu.edu.hk; ruigzheng2-c@my.cityu.edu.hk; hanfeng@cityu.edu.hk; xzhuang7@cityu.edu.hk).

Ming Li is with the Key Laboratory of Intelligent Education Technology and Application of Zhejiang Province, Zhejiang Normal University, Jinhua 321017, China (e-mail: mingli@zjnu.edu.cn).

Digital Object Identifier 10.1109/TNNLS.2024.3370918

Some graph wavelets/framelets systems are also proposed and applied in GNNs for node and graph classifications [20], [21], [22]. When the graph is reordered, it is natural to expect the produced wavelets/framelets to be reordered in the same way for robust learning. However, most of the graph wavelets/framelets do not possess such a property of *permutation equivariance*, that is, up to certain permutations, the constructed graph wavelet/framelet systems should be the "same" regardless of the underlying node orderings. The work on Haar-type graph wavelets/framelets [17], [20], [23], [24], [25] is "piecewise constant" functions on graphs that depend on a given tree with certain underlying node ordering. If new orderings are given, though the underlying graph and graph data are the same, the newly resulting graph wavelets/framelets are no longer the same. Without the property of permutation equivariance, the network outputs could vary with respect to graph reordering and thus lead to instability of the GNNs.

In this article, we provide a novel and general method to construct Haar-type graph framelets having the permutation equivariance property, which further implies the permutation equivariance of our graph framelet neural network model permutation equivariant graph framelet augmented network (PEGFAN). Our Haar-type graph framelets are constructed spatially with respect to a hierarchical structure on the underlying graph. Scales in such systems correspond to the levels in the hierarchical structure in which higher levels are associated with larger groups of nodes. Multiscale extractions via such graph framelets are regarded as alternatives and supplements for the usual multihop aggregations. Moreover, we show that our graph framelets possess sparse representation property, which leads to the sparsity property of the orthogonal projection matrix (framelet matrix) formed by stacking those framelet vectors at certain scales. This is in contrast to the high powers of adjacency matrices and their nonsparse nature. Furthermore, we apply our graph framelets in the neural network architecture design by using the framelet matrices at different scales as well as the adjacency matrices to form multichannel input and perform multiscale extraction through attention and concatenation. The state-of-the-art node classification accuracies on several benchmark datasets validate the effectiveness of our model.

In summary, the contribution of this article is as follows: 1) we propose a novel and general method to construct Haar-type graph framelets that have properties of permutation equivariance, sparse representation, efficient computation, and so on; 2) we apply our Haar-type graph framelet system to extract multiscale information and integrate it into a GNN architecture; and 3) we demonstrate the effectiveness of our model for node classification on synthetic and benchmark datasets via extensive comparisons with several state-of-the-art GNN models.

II. Related Work

A. Node Classification on Heterophilous Graphs

Early work on node classification includes [1], [26], [27], which are some of the earliest examples of spectral and spatial GNNs. GEOM-GCN [28] is the first work that aims at heterophilous graphs. Topology augmentation graph convolutional network (TA-GCN) [29] is proposed under the guidance of a neighborhood class consistency (NCC) metric. To enhance the performance of GNNs on the heterophily datasets, conv-agnostic GNN (CAGNN) [30] is developed by learning the neighbor effect for each node. From the relation-based frequency point of view, relation-based frequency adaptive GNN (RFA-GNN) [31] aims to adaptively pick up signals of different frequencies in each corresponding relation space in the message-passing process. In [12], a set of key designs is discussed, which can boost learning under heterophily. To counter the limit imposed by node-level assortativity (homophily), in [32], a computation graph with proximity and structural information is proposed, which is converted from the input graph. A new generalized PageRank [33], which is jointly optimized with node features and topological information extraction, works for graphs regardless of homophily or heterophily. Two novel fully differentiable and inductive rewiring layers are introduced in [34] to mitigate the problems of oversmoothing, oversquashing, and underreaching on both homophilous and heterophilous graphs. Adopting a homophily-oriented deep heterogeneous graph rewiring method to increase the meta-paths subgraph homophily ratio, heterogeneous GNN (HGNN) [35] improves the performance on heterophilous graphs. In [36], a random-edge dropping mechanism for increasing heterophily of graphs is proposed, aiming at enhancing fairness in GNNs' predictions. We refer to [9] for a comprehensive review of GNNs for graphs with heterophily.

B. Multihop Aggregation in GNNs

Papers of [12], [37], [38], and [39] are GNNs that adopt hidden layer concatenation and multihop aggregation and involve the powers of adjacency matrices. Thus, they resemble each other in terms of the neural network architecture. The difference is that works [37], [38] mainly deal with homophilous datasets. On the other hand, with emphasis on the heterophilous setting, the work [12] theoretically shows the importance of concatenation of aggregation beyond the one-hop neighborhood, with an addition on the importance of ego- and neighbor-embedding separation. Such a nonlocal neighborhood aggregation is also emphasized in [27] and [37]. The current state-of-the-art model feature selection GNN (FSGNN) [39] is different from the previous ones by, in our interpretation, viewing the semi-supervised setting as a supervised setting in which multihop aggregation is regarded as input of different feature channels from different hops and were not applied in the following layers. As a result, its network architecture basically consists of a *mix-hop* [38] layer and fully connected layers with attention weights for different channels being applied before the concatenation. It is worth mentioning that a recent work [40] on large-scale heterophilous node classification is very similar to [39], in which input channels were limited to the zero-hop and the one-hop.

C. Graph Wavelets/Framelets

Papers of [16], [17], [18], [20], [23], [24], [41], and [25] are work of graph wavelets/framelets in which [16] and [18] are spectral-type and the rest are Haar-type. A framelet system differs from the classical (orthogonal) wavelet system by being a frame in a Hilbert space and offering redundant representation. The Haar-type wavelet system in [41] is defined

for different nodes as centers. In [17], [20], [23], [24], and [25], graph wavelets/framelets are defined under a given tree and they differ in the interpretation and generation of the tree. In [23], it is applied to trees from graphs. In [17], the tree is represented as a filtration on $[0, 1]$, and the wavelet system is equivalent to an orthogonal basis of tree polynomials. Similar to [17], the trees are further generalized to hierarchical partitions of $[0, 1]^2$ in [24] and apply to directed graphs, and a Haar-type wavelet system for directed graphs is thus constructed. To further generalize [24], [42], the constructions of Haar-type framelet systems on any compact set in $\mathbb{R}^d$ is considered under a given hierarchical partition and adapt the construction of directed graph framelets to such cases [25].

III. Permutation Equivariant Graph Framelets

In this section, we develop Haar-type graph framelet systems and the binary Haar graph framelets, with properties of tightness, sparsity, efficiency, and permutation equivariance, which yield robustness and effective algorithms for the model PEGFAN to be introduced in Section IV. All proofs of the main results in this article are postponed to Appendix A.

A. Preliminaries

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph, where $\mathcal{V} = \{v_1, \ldots, v_n\}$ is the vertex set containing n vertices (or equivalently, we simply identify $\mathcal{V} = \{1, 2, \ldots, n\}$) and $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ is the edge set of ordered pairs (i, j). The adjacency matrix $\boldsymbol{A} : \mathcal{V} \times \mathcal{V} \to \mathbb{R}$ of $\mathcal{G}$ is a matrix of size $n \times n$ such that its (i, j)-entry a_{ij} is the weight on edge (i, j) and $a_{ij} = 0$ if $(i, j) \notin \mathcal{E}$. We consider only undirected graphs in this article, i.e., $\boldsymbol{A}^\top = \boldsymbol{A}$. We denote $\tilde{\boldsymbol{A}} := \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}$ with $\boldsymbol{D}$ being the diagonal degree matrix of $\mathcal{G}$, whose diagonal elements defined as $d_{ii} = \sum_{j=1}^n a_{ij}$. A signal $\boldsymbol{f} = [f_1, \ldots, f_n]^\top$ on the graph is defined as $\boldsymbol{f} : \mathcal{V} \to \mathbb{R}$ with ℓ_2 norm $\|\boldsymbol{f}\|^2 = \sum_{i=1}^n |f_i|^2 < \infty$. All such ℓ_2 signals on $\mathcal{G}$ form a Hilbert space $L_2(\mathcal{G})$ under the usual inner product. A collection $\{\boldsymbol{e}_m : m \in [M]\} \subset L_2(\mathcal{G})$ is a *tight frame* of $L_2(\mathcal{G})$ if $\boldsymbol{f} = \sum_{m=1}^M \langle \boldsymbol{f}, \boldsymbol{e}_m \rangle \boldsymbol{e}_m$ for all $\boldsymbol{f} \in L_2(\mathcal{G})$, where $\langle \cdot, \cdot \rangle$ is the inner product, and we denote $[M] := \{1, \ldots, M\}$. We denote the ith column vector and row vector of a matrix $\boldsymbol{M}$, by $\boldsymbol{M}_{:i}$ and $\boldsymbol{M}_{i:}$, respectively.

For $K \geq 2$, we call a sequence $\mathcal{P}_K := \{\mathcal{V}_j : j = 1, \ldots, K\}$ of sets as a K-*hierarchical clustering* of $\mathcal{V}$ if each $\mathcal{V}_j := \{s_{\boldsymbol{\Lambda}} \subset \mathcal{V} : \dim(\boldsymbol{\Lambda}) = j\}$ is a partition of $\mathcal{V}$, i.e., $\mathcal{V} = \cup_{\boldsymbol{\Lambda}} s_{\boldsymbol{\Lambda}}$, and $\mathcal{V}_j$ is a refinement of $\mathcal{V}_{j-1}$, where we use the index vector $\boldsymbol{\Lambda} = (\lambda_1, \ldots, \lambda_j) \in \mathbb{N}^j$ to encode position, level j, and parent–children relationship, of the clusters $s_{\boldsymbol{\Lambda}}$. Fig. 1 gives an example of a K-hierarchical clustering. Let us denote $\mathcal{V}_1 = \{s_{(1)} = \mathcal{V} = \{1, 2, \ldots, 8\}\}$. Then, according to parent–children relationship in Fig. 1, we have $\mathcal{V}_2 = \{s_{(1,1)}, s_{(1,2)}\}$, where $s_{(1,1)} = \{1, \ldots, 4\}$ and $s_{(1,2)} = \{5, \ldots, 8\}$. Similarly, we have $\mathcal{V}_3 = \{s_{(1,1,1)}, s_{(1,1,2)}, s_{(1,2,1)}, s_{(1,2,2)}\}$, where $s_{(1,1,1)} = \{1, 2\}$, $s_{(1,1,2)} = \{3, 4\}$, $s_{(1,2,1)} = \{5, 6\}$, and $s_{(1,2,2)} = \{7, 8\}$, and thus, $s_{(1,1,1,1)} = \{1\}$ and $s_{(1,1,1,2)} = \{2\}$. Here, $\dim(\boldsymbol{\Lambda})$ denotes the length of the index vector. If $s_{\boldsymbol{\Lambda}} \in \mathcal{V}_j$ is a parent, then the index vectors of its children are appended with an integer, i.e., $(\boldsymbol{\Lambda}, i)$, indicating its ith child, and thus, the child is denoted by $s_{(\boldsymbol{\Lambda},i)} \in \mathcal{V}_{j+1}$. Then, we have the parent–children relationship $s_{(\boldsymbol{\Lambda},i)} \subset s_{\boldsymbol{\Lambda}}$. We denote the number of children of $s_{\boldsymbol{\Lambda}}$ by $L_{\boldsymbol{\Lambda}}$. Unless specified, we consider K-hierarchical clustering $\mathcal{P}_K$ with $\mathcal{V}_K = \{\{1\}, \ldots, \{n\}\}$ and $\mathcal{V}_1 = \{[n]\}$ being a singleton, i.e., $\mathcal{P}_K$ is a *tree*.

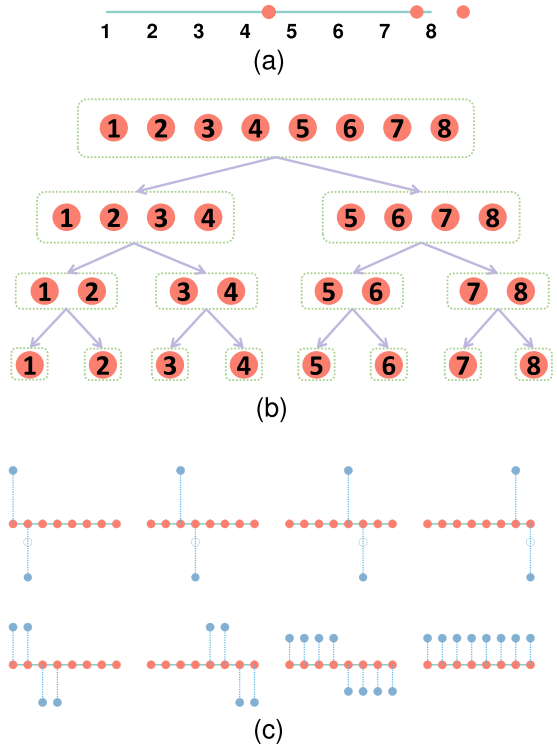

Fig. 1. (a) Graph framelets (bottom) with respect to $\mathcal{G}_1$ and (b) hierarchical partition $\mathcal{P}_4$ (middle). The blue points in (c) are the values of graph framelets on each node. (c) Height of blue points represents the value of graph framelets $\psi_{\boldsymbol{\Lambda}}$ (eight subfigures). The blue points are above the corresponding nodes if values are positive and below otherwise.

In classical wavelet/framelet theory, an important concept is the multiresolution analysis (MRA). One of the most important ideas is to find a sequence of subspaces $\{V_j\} \subset L_2(\mathbb{R})$ such that $V_j \subset V_{j+1}$ and $\cup_{j\in\mathbb{Z}} V_j = L_2(\mathbb{R})$. If there exists $\phi(t) \in V_0$ such that $\{\phi(t - b)\}_{b\in\mathbb{Z}}$ forms an orthonormal basis of V_0 and $f(t) \in V_j$ if and only if $f(2t) \in V_{j+1}$, then we can find a mother wavelet $\psi(t)$ such that $\{2^{j/2}\psi(2^j t - b)\}_{j,b\in\mathbb{Z}}$ forms an orthonormal basis for $L_2(\mathbb{R})$. For example, if $\phi(t) = \chi_{[0,1)}(t)$ and $\psi(t) = \phi(2t) - \phi(2t - 1)$, then the resulting wavelet is the so-called Haar wavelet. However, the translation and dilation operators are not naturally defined for graph signals. Fortunately, if we look at the support of Haar wavelets, we can find that the union of the support of elements in $\mathfrak{W}_j := \{2^{j/2}\psi(2^j t - b)\}_{b\in\mathbb{N}}$ is equal to $\mathbb{R}$ for a fixed j and the collection of the support of elements in $\mathfrak{W}_{j+1}$ is a refinement of that of $\mathfrak{W}_{j+1}$. Hence, these supports actually form a hierarchical partition. Based on this observation, a natural way to define translations on the graph is to generalize hierarchical partitions to the graph, see [19], [25]. For example, when mapping each node in a graph to an interval on $[0, 1]$, then based on the hierarchical partition on $[0, 1]$, graph framelets can be constructed similarly as classical Haar wavelet [25]. In the following, we provide general conditions in Theorem 1 for constructing Haar graph framelets based on a K-hierarchical clustering.

B. Main Construction

Given $\mathcal{P}_K$, we define the unit scaling vectors $\boldsymbol{\phi}_{\boldsymbol{\Lambda}}$ (similar to scaling functions for V_j's on an MRA) iteratively from

$\dim(\mathbf{\Lambda}) = K$ to $\dim(\mathbf{\Lambda}) = 1$. When $\dim(\mathbf{\Lambda}) = K$, each cluster (node) $s_{\mathbf{\Lambda}}$ contains only one vertex in graph $\mathcal{G}$ (see Fig. 1), and thus, we define $\boldsymbol{\phi}_{\mathbf{\Lambda}} = \boldsymbol{I}_{:i}$, where $i \in s_{\mathbf{\Lambda}} \subset \mathcal{V}$ and $\boldsymbol{I}_{:i}$ is the ith column of the identity matrix $\boldsymbol{I} \in \mathbb{R}^{n\times n}$. When $\dim(\mathbf{\Lambda}) < K$, we define

$$\boldsymbol{\phi}_{\mathbf{\Lambda}} := \sum_{\ell\in[L_{\mathbf{\Lambda}}]} p_{(\mathbf{\Lambda},\ell)}\boldsymbol{\phi}_{(\mathbf{\Lambda},\ell)} \tag{1}$$

where $\boldsymbol{p}_{\mathbf{\Lambda}} = [p_{(\mathbf{\Lambda},1)}, \ldots, p_{(\mathbf{\Lambda},L_{\mathbf{\Lambda}})}]^\top \in \mathbb{R}^{L_{\mathbf{\Lambda}}}$ and $\|\boldsymbol{p}_{\mathbf{\Lambda}}\| = 1$. Obviously, $\boldsymbol{\phi}_{\mathbf{\Lambda}}$ is with support $\operatorname{supp}\boldsymbol{\phi}_{\mathbf{\Lambda}} = s_{\mathbf{\Lambda}}$ and $\|\boldsymbol{\phi}_{\mathbf{\Lambda}}\| = 1$. For framelet vectors on the graph, we define $\boldsymbol{\psi}_{(\mathbf{\Lambda},m)}$, $m \in [M_{\mathbf{\Lambda}}]$ for some $M_{\mathbf{\Lambda}} \in \mathbb{N}$ by

$$\boldsymbol{\psi}_{(\mathbf{\Lambda},m)} := \sum_{\ell\in[L_{\mathbf{\Lambda}}]} (\boldsymbol{B}_{\mathbf{\Lambda}})_{m,\ell}\boldsymbol{\phi}_{(\mathbf{\Lambda},\ell)} \tag{2}$$

from some matrices $\boldsymbol{B}_{\mathbf{\Lambda}} \in \mathbb{R}^{M_{\mathbf{\Lambda}}\times L_{\mathbf{\Lambda}}}$. Theorem 1 characterizes when $\boldsymbol{\phi}_{\mathbf{\Lambda}}$ and $\boldsymbol{\psi}_{(\mathbf{\Lambda},m)}$ form a tight frame of $L_2(\mathcal{G})$.

Theorem 1 (General Characterization): Let $\mathcal{P}_K$ be a K-hierarchical clustering on a graph $\mathcal{G}$. Then, the matrices $\boldsymbol{B}_{\mathbf{\Lambda}}$ and vectors $\boldsymbol{p}_{\mathbf{\Lambda}}$ satisfy $\boldsymbol{B}_{\mathbf{\Lambda}}\boldsymbol{B}_{\mathbf{\Lambda}}^\top\boldsymbol{B}_{\mathbf{\Lambda}} = \boldsymbol{B}_{\mathbf{\Lambda}}$, $\boldsymbol{B}_{\mathbf{\Lambda}}\boldsymbol{p}_{\mathbf{\Lambda}} = \boldsymbol{0}$, and $\operatorname{Rank}(\boldsymbol{B}_{\mathbf{\Lambda}}) = L_{\mathbf{\Lambda}} - 1$ for all $\mathbf{\Lambda}$ with $\dim(\mathbf{\Lambda}) = j_0, \ldots, K$ if and only if for any $j_0 \in [K]$, the collection $\mathcal{F}_{j_0}(\mathcal{P}_K) := \{\boldsymbol{\phi}_{\mathbf{\Lambda}} : \dim(\mathbf{\Lambda}) = j_0\} \cup \{\boldsymbol{\psi}_{\mathbf{\Lambda}} : \dim(\mathbf{\Lambda}) = j\}_{j=j_0+1}^{K}$ defined by (1) and (2) is a tight frame of $L_2(\mathcal{G})$.

We remark that Theorem 1 provides a more general sufficient and necessary condition than that in [19], for all graph framelets having the form (1) and (2) to be a tight frame. When we use the Haar graph framelets to extract frequency features of graph signals, general graph wavelets/framelets [see (1) and (2)] can be viewed as multiscale representation systems in which the notion of "scale" is different from the usual k-hop neighborhood in graphs and serve as an alternative to capture long-range information.

Besides, the given $\mathcal{P}_K$ in the proposed construction is not specified. The advantage of the generality of this definition is that there is no constraint on how $\mathcal{P}_K$ is generated: one can use the edges solely or combine the edges and node features to generate $\mathcal{P}_K$ and so on. Thus, this provides great potential in theories and applications. As shown in experiments, clustering graph nodes based only on adjacency matrices are capable of providing nice graph framelets that help improve the learning abilities of neural networks on node classification tasks.

The following example shows a close relationship between our framelet systems and the traditional Haar graph basis.

Example 1 (Path Graph and Haar Basis): Given a path graph $\mathcal{G}_1$ with eight nodes $\mathcal{V} = \{1, 2, \ldots, 8\}$. If we choose hierarchical clustering $\mathcal{P}_4 = \{\mathcal{V}_1, \mathcal{V}_2, \mathcal{V}_3, \mathcal{V}_4\}$ with $\mathcal{V}_1 = \{s_{(1)}\}$, $\mathcal{V}_2 = \{s_{(1,1)}, s_{(1,2)}\}$, $\mathcal{V}_3 = \{s_{(1,1,1)}, s_{(1,1,2)}, s_{(1,2,1)}, s_{(1,2,2)}\}$, $\mathcal{V}_4 = \{s_{(1,1,1,1)}, s_{(1,1,1,2)}, s_{(1,1,2,1)}, s_{(1,1,2,2)}, s_{(1,2,1,1)}, s_{(1,2,1,2)}, s_{(1,2,2,1)}, s_{(1,2,2,2)}\}$, and $\boldsymbol{p}_{\mathbf{\Lambda}} = [(1/(2)^{1/2}), (1/(2)^{1/2})]^\top$ and $\boldsymbol{B}_{\mathbf{\Lambda}} = [(1/(2)^{1/2}), -(1/(2)^{1/2})]$ for all $\mathbf{\Lambda}$ (note that each parent has exactly two children $L_{\mathbf{\Lambda}} = 2$), then the graph framelet system $\mathcal{F}_{j_0}(\mathcal{P}_K)$ with $K = 4$ as in Theorem 1 is a Haar basis for any $j_0 \in [K]$ (see Fig. 1 for illustration).

Based on the general conditions in Theorem 1, we further investigate the specific structure of $\boldsymbol{B}_{\mathbf{\Lambda}}$. We give the following proposition that completely characterizes the structure of matrices $\boldsymbol{B}_{\mathbf{\Lambda}}$ in Theorem 1.

Proposition 1: Let $\boldsymbol{p}$ be a unit vector of length $L \geq 1$, that is, $\|\boldsymbol{p}\| = 1$. Assume that $\boldsymbol{B} \in \mathbb{R}^{M\times L}$ with $M \geq L - 1$ is a matrix such that $\boldsymbol{B}\boldsymbol{p} = 0$ and $\operatorname{Rank}(\boldsymbol{B}) = L - 1$. Then, $\boldsymbol{B}\boldsymbol{B}^\top\boldsymbol{B} = c\boldsymbol{B}$ for some constant c if and only if $\boldsymbol{B}^\top\boldsymbol{B} = c(\boldsymbol{I} - \boldsymbol{p}\boldsymbol{p}^\top)$. In particular, if $c \neq 0$, then $\boldsymbol{P} := [\boldsymbol{p}, (1/(c)^{1/2})\boldsymbol{B}^\top]$ satisfies $\boldsymbol{P}\boldsymbol{P}^\top = \boldsymbol{I}$.

Proposition 1 shows that $\boldsymbol{B}_{\mathbf{\Lambda}}$ is from the (matrix) splitting of a rank $L - 1$ matrix $\boldsymbol{I} - \boldsymbol{p}_{\mathbf{\Lambda}}\boldsymbol{p}_{\mathbf{\Lambda}}^\top$. Notice that the role of elements in $\boldsymbol{p}_{\mathbf{\Lambda}}$ in (1) is to give weights to each cluster $s_{(\mathbf{\Lambda},\ell)}$. One typical scenario is that each child cluster is of equal importance, which means that the vector $\boldsymbol{p}_{\mathbf{\Lambda}}$ is a vector with all equal elements. On the other hand, it could be too involved to use matrix splitting techniques [43], [44], [45], [46] for obtaining the matrix $\boldsymbol{B}_{\mathbf{\Lambda}}$. We next show that we can obtain matrices $\boldsymbol{B}_{\mathbf{\Lambda}}$ by simply permuting a fixed vector $\boldsymbol{w}$ such that each of its elements appears with equal chance. Under this hypothesis of equal importance and equal chance, in the following result, we introduce a binary Haar graph framelet system by a careful design of the matrices $\boldsymbol{B}_{\mathbf{\Lambda}}$ and $\boldsymbol{p}_{\mathbf{\Lambda}}$. The word *binary* here is chosen since each nonzero coefficient of high-frequency framelets in (2) only takes from $\{1, -1\}$ (without normalization). We show that such graph framelet systems $\mathcal{F}_{j_0}(\mathcal{P}_K)$ have many desirable properties, including permutation equivariance.

For each pair (ℓ_1, ℓ_2) with $1 \leq \ell_1 < \ell_2 \leq L_{\mathbf{\Lambda}}$, define a vector $\boldsymbol{w}_{\mathbf{\Lambda}}^m$ of size $L_{\mathbf{\Lambda}} \times 1$ by

$$\left(\boldsymbol{w}_{\mathbf{\Lambda}}^m\right)_\tau = \begin{cases} \dfrac{1}{\sqrt{L_{\mathbf{\Lambda}}}}, & \tau = \ell_1 \\ \dfrac{-1}{\sqrt{L_{\mathbf{\Lambda}}}}, & \tau = \ell_2 \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

where $m := m(\ell_1, \ell_2, L_{\mathbf{\Lambda}}) := ((2L_{\mathbf{\Lambda}} - \ell_1)(\ell_1 - 1)/2) + \ell_2 - \ell_1$ is ranging from 1 to $M_{\mathbf{\Lambda}} := (L_{\mathbf{\Lambda}}(L_{\mathbf{\Lambda}} - 1)/2)$ for all possible pairs (ℓ_1, ℓ_2) with $1 \leq \ell_1 < \ell_2 \leq L_{\mathbf{\Lambda}}$. Note that $\boldsymbol{w}_{\mathbf{\Lambda}}^m$ has only two nonzero entries locating at the ℓ_1th and ℓ_2th positions, respectively. Such $\boldsymbol{w}_{\mathbf{\Lambda}}^m$ will be used as the mth row of the matrix $\boldsymbol{B}_{\mathbf{\Lambda}}$.

Corollary 1 (Binary Haar Graph Framelets): Let $\mathcal{P}_K$ be a K-hierarchical clustering on a graph $\mathcal{G}$. Let $\boldsymbol{p}_{\mathbf{\Lambda}} = (1/(L_{\mathbf{\Lambda}})^{1/2})\boldsymbol{1}$ be a constant vector of size $L_{\mathbf{\Lambda}} \times 1$ and $\boldsymbol{B}_{\mathbf{\Lambda}} := [\boldsymbol{w}_{\mathbf{\Lambda}}^1, \ldots, \boldsymbol{w}_{\mathbf{\Lambda}}^{M_{\mathbf{\Lambda}}}]^\top$ with $\boldsymbol{w}_{\mathbf{\Lambda}}^m$ being given as in (3). Define $\mathcal{F}_{j_0}(\mathcal{P}_K)$ as in Theorem 1. Then, $\mathcal{F}_{j_0}(\mathcal{P}_K)$ is a tight frame for $L_2(\mathcal{G})$ for any $j_0 \in [K]$.

Remark 1: In fact, the framelets obtained in Example 1 belong to binary Haar graph framelets (see Fig. 1 for illustration). The matrix $\boldsymbol{B}_{\mathbf{\Lambda}}$ is formed by permuting 1 and -1 of the specific type of vectors $\boldsymbol{w} = [1, -1, 0, \ldots, 0]$ to all possible positions. In fact, more general types of vectors $\boldsymbol{w}$ can be served to form the matrix $\boldsymbol{B}_{\mathbf{\Lambda}}$ through permutations.

We next focus on the sparsity, efficiency, and permutation equivariance of the binary Haar graph framelets constructed in Corollary 1.

C. Sparsity

Notice that if each row of the matrix $\boldsymbol{B}_{\mathbf{\Lambda}}$ is sparse, then the produced $\psi_{(\mathbf{\Lambda},m)}$ is also sparse. For the binary Haar graph framelets, each row of $\boldsymbol{B}_{\mathbf{\Lambda}}$ only contains two nonzero values. If $L_j := \max_{\dim(\mathbf{\Lambda})=j} L_{\mathbf{\Lambda}}$, then it is easy to see that the number $\|\boldsymbol{\psi}_{(\mathbf{\Lambda},m)}\|_0$ of nonzero entries of $\boldsymbol{\psi}_{(\mathbf{\Lambda},m)}$ satisfies $\|\boldsymbol{\psi}_{(\mathbf{\Lambda},m)}\|_0 \leq 2L_{j+1}$, for all $\dim(\mathbf{\Lambda}) = j$. When the hierarchical clustering is balanced and $\dim(\mathbf{\Lambda})$ is large, high-level framelets $\boldsymbol{\psi}_{(\mathbf{\Lambda},m)}$ are well-localized and thus sparse.

Besides the sparsity of the framelets, we also want to know when the framelet coefficients of a signal are sparse, which is the desired property of sparse representation of framelets. Different coefficients represent different scales. The sparse representation property plays an important role in feature extraction and representation for classification tasks. In node classification, piecewise constant signals, e.g., one-hot label encoding, are of great importance in practice due to their simplicity [47]. Hence, it is valuable to study the framelet coefficients of the piecewise constant signals. Let $\mathcal{F}_{j_0}(\mathcal{P}_K) := \{\boldsymbol{\phi}_{\boldsymbol{\Lambda}}, \dim(\boldsymbol{\Lambda}) = j_0\} \cup \{\boldsymbol{\psi}_{\boldsymbol{\Lambda}}, \dim(\boldsymbol{\Lambda}) = j\}_{j=j_0+1}^{K} =: \{\boldsymbol{u}_i\}_{i=1}^{M_{\mathcal{G}}}$ be a binary Haar graph framelet system with $M_{\mathcal{G}}$ elements and define $\hat{\boldsymbol{f}} \in \mathbb{R}^{M_{\mathcal{G}}}$ to be the *framelet coefficient vector* with its ith element $(\hat{\boldsymbol{f}})_i := \langle \boldsymbol{f}, \boldsymbol{u}_i \rangle$ for a signal $\boldsymbol{f}$. In what follows, we denote $\boldsymbol{F} := [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{M_{\mathcal{G}}}]$ to be the matrix representation of the graph framelet system $\mathcal{F}_{j_0}(\mathcal{P}_K)$. Then, $\hat{\boldsymbol{f}} = \boldsymbol{F}^\top \boldsymbol{f}$. We have the following result regarding the sparsity of $\hat{\boldsymbol{f}}$.

Theorem 2 (Binary Haar Graph Framelet Transform Preserving Sparsity): Let $\mathcal{F}_{j_0}(\mathcal{P}_K)$ be a binary Haar graph framelet system defined as in Corollary 1. Assume that $\max_{\dim(\boldsymbol{\Lambda})>0} L_{\boldsymbol{\Lambda}} \le h$. Then, for a signal $\boldsymbol{f} \in \mathbb{R}^n$, the framelet coefficient vector $\hat{\boldsymbol{f}}$ satisfies $\|\hat{\boldsymbol{f}}\|_0 \le (K-1)(h-1)\|\boldsymbol{f}\|_0$.

Remark 2: If for all $\boldsymbol{\Lambda}$, we have $L_{\boldsymbol{\Lambda}} = h$ for some integer $h \ge 2$, then $K = O(\log_h n)$ and, hence, $\|\hat{\boldsymbol{f}}\|_0 = \|\boldsymbol{f}\|_0 \cdot O(h \log_h n)$, which shows that our binary Haar graph transform preserves sparsity for sparse signals. In fact, the total number $M_{\mathcal{G}}$ of elements in $\mathcal{F}_{j_0}(\mathcal{P}_K)$ in this case is of order $O(nh)$. When $\|f\|_0 \ll n$, we see that $\|\hat{\boldsymbol{f}}\|_0 \ll O(nh) = M_{\mathcal{G}}$. Theorem 2 can be extended to other type of matrices $\boldsymbol{B}_{\boldsymbol{\Lambda}}$ that is row-wise sparse.

D. Efficiency

Graph Fourier basis based on graph Laplacian is of great importance in GNNs. However, the computational complexity and space complexity of generating graph Fourier basis could be as large as $O(n^3)$ and $O(n^2)$, respectively. Hence, these reasons prevent it from being more flexible in practice when n is large and the graph Laplacian is not sparse. On the other hand, when using our binary Haar graph framelets, we have an efficient way to compute our framelets as well as the framelet coefficient vector via sparse computation. For the rest of this article, when we discuss computational complexity, we assume that all matrix/vector operations are done by using sparse operations, i.e., the operations are evaluated only on nonzero entries.

Theorem 3: Let $h > 1$ be a positive integer. Assume that the K-hierarchical clustering $\mathcal{P}_K$ satisfies $n = O(h^{K-1})$ with $h := \max_{\dim(\boldsymbol{\Lambda})>0} L_{\boldsymbol{\Lambda}}$. For $j_0 \in [K]$, let $\boldsymbol{F} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{M_{\mathcal{G}}}]$ be the framelet matrix with respect to the binary Haar graph framelet system $\mathcal{F}_{j_0}(\mathcal{P}_K)$ as given in Corollary 1. Then, for all $j_0 \in [K]$, the number $M_{\mathcal{G}}$ of framelet vectors in $\mathcal{F}_{j_0}(\mathcal{P}_K)$ is of order $O(nh)$, the computational complexity of generating all $\boldsymbol{u}_m$, $m = 1, \ldots, M_{\mathcal{G}}$, in $\boldsymbol{F}$ is of order $O(nh \log_h n)$, and the total number $\mathrm{nnz}(\boldsymbol{F})$ of nonzero entries in $\boldsymbol{F}$ is of order $O(nh \log_h n)$.

Remark 3: In practice, h is usually small (e.g., 2, 4, or 8), and hence, $\boldsymbol{F}$ is a sparse matrix. Theorem 3 shows that our binary Haar graph framelet systems are efficient in processing datasets with large graphs. Moreover, the framelet coefficient vector $\hat{\boldsymbol{f}}$ can be computed with the computational complexity of order $O(nh)$ as well. See Theorem 5 for the fast decomposition and reconstruction algorithms using our graph framelet systems.

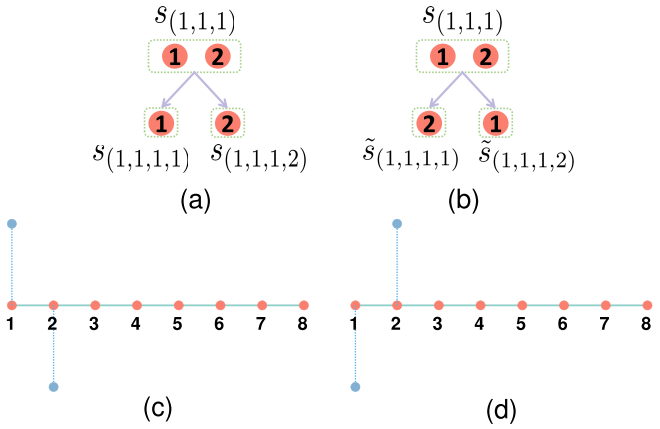

Fig. 2. Partition permutation. Consider the graphs $\mathcal{G}_1$ and $\mathcal{P}_4$ in Fig. 1. Let us permute the order of children of $s_{(1,1,1)} = \{1, 2\}$ in $\mathcal{P}_4$ only while keeping other part of $\mathcal{P}_4$, $\boldsymbol{p}_{\boldsymbol{\Lambda}}$, and $\boldsymbol{B}_{\boldsymbol{\Lambda}}$. The original framelet is given by $\boldsymbol{\psi}_{(1,1,1)}$ [with respect to $s_{(1,1,1)}$] and the new framelet is given by $\tilde{\boldsymbol{\psi}}_{(1,1,1)}$ [with respect to $\tilde{s}_{(1,1,1)}$]. (a) $s_{(1,1,1)}$ of $\mathcal{P}_4$. (b) Permute children of $s_{(1,1,1)}$. (c) $\boldsymbol{\psi}_{(1,1,1)}$. (d) $\tilde{\boldsymbol{\psi}}_{(1,1,1)}$.

E. Permutation Equivariance

Fix $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and $\mathcal{P}_K$. Denote our construction of graph framelets in Theorem 1 by $\mathcal{A}$, where it is provided with a graph $\mathcal{G}$ and a corresponding hierarchical partition $\mathcal{P}_K$ and then builds the graph framelet system, i.e., $\mathcal{A}(\mathcal{G}, \mathcal{P}_K) = \mathcal{F}_{j_0}(\mathcal{P}_K)$. Let $\pi : \mathcal{V} \to \mathcal{V}$ be a reordering (relabeling and bijection) of $\mathcal{V} = \{1, 2, \ldots, n\}$, i.e., π is with respect to a *node permutation* on $[n]$ with $\pi(V) = \{\pi(1), \ldots, \pi(n)\}$. We denote $\pi(\mathcal{G}) = (\pi(V), \pi(\mathcal{E}))$ with $\pi(\mathcal{E}) := \{(\pi(i), \pi(j)) : (i, j) \in \mathcal{E}\}$. The corresponding signal $\boldsymbol{f}$ on the graph $\mathcal{G}$ is reordered to be $\pi(\boldsymbol{f})$ under the newly ordered graph $\pi(\mathcal{G})$. In other words, given a π, there exists a permutation matrix $\boldsymbol{P}_\pi$ of size $n \times n$ such that $\pi(\boldsymbol{f}) = \boldsymbol{P}_\pi \boldsymbol{f}$. For each node permutation π, the construction $\mathcal{A}$ is called (node) *permutation equivariant* if $\mathcal{A}(\pi(\mathcal{G}), \mathcal{P}_K) = \pi(\mathcal{A}(\mathcal{G}, \mathcal{P}_K))$, where $\pi(\boldsymbol{u}_m) = \boldsymbol{P}_\pi \boldsymbol{u}_m$ for $\boldsymbol{u}_m \in \mathcal{F}_{j_0}(\mathcal{P}_K)$.

Note that $\mathcal{P}_K$ is a *tree* and that the children nodes in a parent–children subtree are ordered according to the last integer in the index vectors $\boldsymbol{\Lambda}$. The order of nodes in such subtrees and the order of nodes in $\mathcal{V}$ are separately defined. This means that a reordering of nodes in $\mathcal{V}$ does not affect the order in subtrees in $\mathcal{P}_K$ and vice versa. On the other hand, the reordering of tree nodes $\boldsymbol{\Lambda}$ may result in different graph framelets. Fig. 2 shows a simple example. Thus, it is necessary to analyze the relationship of the graph framelets under such types of permutations. We say that π_p is a *partition permutation* on $\mathcal{P}_K$ if the permutation is on the children of each tree node $\boldsymbol{\Lambda}$ only. For each partition permutation π_p, the construction $\mathcal{A}$ is called *partition permutation equivariant* if $\mathcal{A}(\mathcal{G}, \pi_p(\mathcal{P}_K)) = \pi_p(\mathcal{A}(\mathcal{G}, \mathcal{P}_K))$, that is, there exists a permutation π^* on $[M_{\mathcal{G}}]$ associated with π_p such that for each $\boldsymbol{u}_m \in \mathcal{F}_{j_0}(\mathcal{P}_K)$, $\pi_p(\boldsymbol{u}_m) = c_m \boldsymbol{u}_{\pi^*(m)}$ for some $c_m \in \{-1, +1\}$. We have the following theorem regarding the permutation equivariance on both node and partition permutations.

Theorem 4: Let $\mathcal{A}(\mathcal{G}, \mathcal{P}_K)$ be the construction of the binary Haar graph framelet systems in Corollary 1 for $j_0 \in [K]$. Then, the following three statements hold.

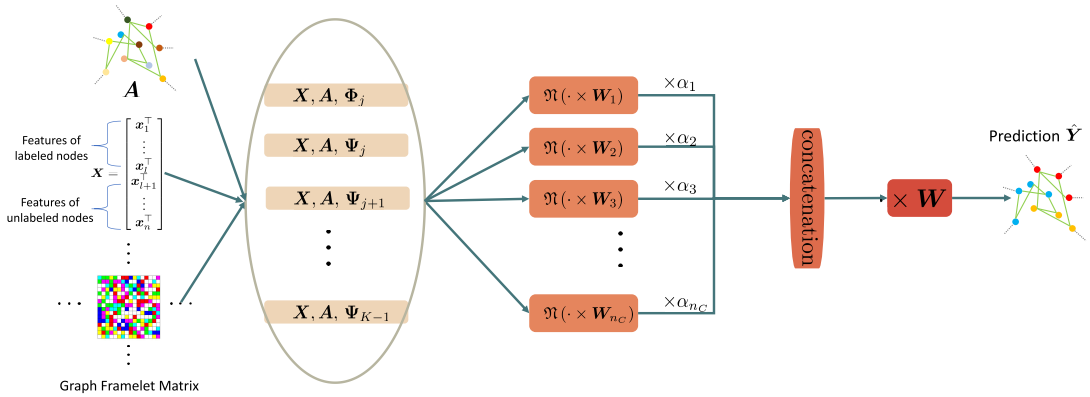

Fig. 3. Neural network architecture. The input is the feature matrix $\boldsymbol{X}$. The network operations are determined by the underlying adjacency matrix $\boldsymbol{A}$ and the constructed binary Haar graph framelet system $\{\boldsymbol{F}_0, \ldots, \boldsymbol{F}_{K-1}\}$. The operator $\mathfrak{N}(\cdot)$ is defined as normalizing each row of any given matrix.

1) For any node permutation π, we have $\mathcal{A}(\pi(\mathcal{G}), \mathcal{P}_K) = \pi(\mathcal{A}(\mathcal{G}, \mathcal{P}_K))$.
2) For any partition permutation π_p, we have $\mathcal{A}(\mathcal{G}, \pi_p(\mathcal{P}_K)) = \pi_p(\mathcal{A}(\mathcal{G}, \mathcal{P}_K))$.
3) For any node permutation π and partition permutation π_p, we have $\mathcal{A}(\pi(\mathcal{G}), \pi_p(\mathcal{P}_K)) = \pi_p(\pi(\mathcal{A}(\mathcal{G}, \mathcal{P}_K))) = \pi(\pi_p(\mathcal{A}(\mathcal{G}, \mathcal{P}_K)))$.

Remark 4: Theorem 4 shows that our binary framelet system $\mathcal{F}_{j_0}(\mathcal{P}_K)$ is permutation equivariant when reordering node or the tree indices. By applying Theorem 4, we show that our proposed graph framelet neural network model PEGFAN has the property of permutation equivariance. See Proposition 2 in Section IV.

Permutation equivariance is a subtle property that most of the GNNs in the literature possess since they generally employ operation that only involves the adjacency matrices, the graph Laplacians, summation, and concatenation. Nonetheless, there are works [48], [49] that theoretically investigate the permutation equivariance of general and specific GNNs, which is highly related to the graph classification and the importance of the topic of the expressiveness of GNNs [50], [51] as permutation is one of the most basic types of isomorphism on graphs. In this article, we confine ourselves to the output consistency that permutation equivariance derives as this is coherent to our context of node classification. On the contrary, graph wavelets/framelets, especially Haar-type graph wavelets/framelets, are more complicatedly generated mathematical tools and the discussion of such property is missing in both the mathematical literature and the recent works of GNNs that apply graph wavelets/framelets. In some of the works of Haar-type graph wavelets/framelets [20], [23], [24], it is obvious that the permutation equivariance is violated if there are no further constraints.

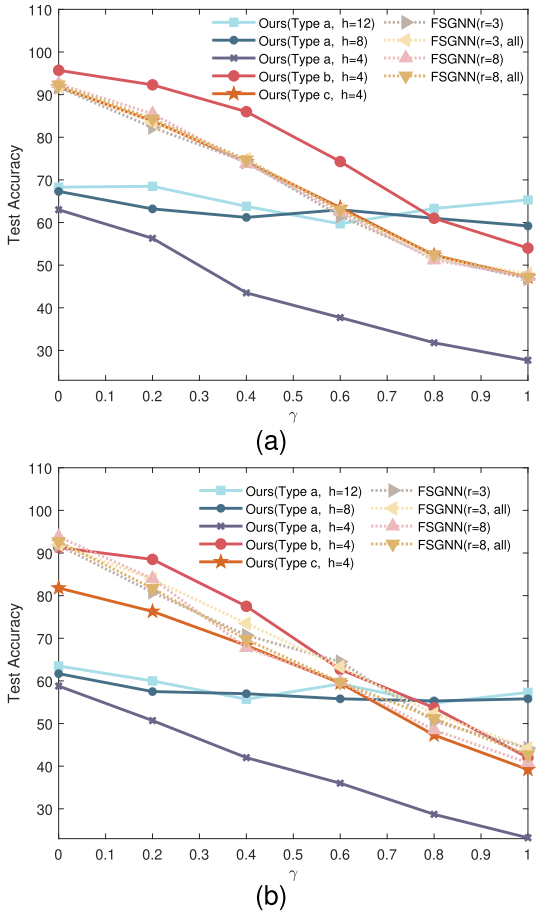

Fig. 4. Demonstration of statistical performance comparison. (a) Results in Table I. (b) Results in Table II.

IV. Graph Framelet Neural Networks

We introduce the graph framelet neural network model that integrates our constructed binary Haar graph framelets, which we call PEGFAN, see Fig. 3.

Semi-supervised learning is characterized by involving both unlabeled and labeled data to infer a discriminative function f. In contrast, in supervised learning, only labeled data are utilized in obtaining f. In a (semi-supervised) node classification task, we assume that the first l nodes are labeled. Each node $i \in \mathcal{V}$ is associated with a feature vector $\boldsymbol{x}_i \in \mathbb{R}^{n_f}$ and a one-hot $\boldsymbol{y}_i \in \mathbb{R}^{n_c}$ indicating the ground truth of labels, where n_f and n_c are the numbers of features and classes, respectively. Stacking these vectors gives a feature matrix $\boldsymbol{X} \in \mathbb{R}^{n \times n_f}$ and a label matrix $\boldsymbol{Y} \in \mathbb{R}^{n \times n_c}$ (the first l elements are given labels and the rest part has no label and need to be predicted). Suppose that there are n_C channels, associating a series of matrices $\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_{n_C}$ for each channel, and $\boldsymbol{X}_i \in \mathbb{R}^{n \times d_i}$, $1 \le i \le n_C$. Our model is a two-layer neural

TABLE I
CLASSIFICATION ACCURACY ON SYNTHETIC DATASET WITH FEATURES SAMPLED FROM $6(-0.75+0.5\,c)+\xi$, WHERE $\xi \sim N(0,1)$, $c \in \{0,1,2,3\}$

γ	0	0.2	0.4	0.6	0.8	1
Ours(Type a, $h=12$)	68.3	68.5	63.8	59.7	63.3	65.3
Ours(Type a, $h=8$)	67.3	63.2	61.2	63	61	59.2
Ours(Type a, $h=4$)	63	56.3	43.5	37.7	31.8	27.7
Ours(Type b, $h=4$)	95.7	92.3	86	74.3	61	54
Ours(Type c, $h=4$)	91.8	83.8	74.2	63.5	52.3	47.2
FSGNN($r=3$)	91.7	82.3	74.2	61.5	51.8	46.8
FSGNN($r=3$, all)	92	83.5	74.8	63	51.7	48
FSGNN($r=8$)	92.5	85.5	73.8	62.5	51.2	47.3
FSGNN($r=8$, all)	92.3	84.3	74.7	63	52.5	47.2

TABLE II
CLASSIFICATION ACCURACY ON SYNTHETIC DATASET WITH FEATURES SAMPLED FROM $(-0.75+0.5\,c)+\xi$, WHERE $\xi \sim N(0,1)$, $c \in \{0,1,2,3\}$

γ	0	0.2	0.4	0.6	0.8	1
Ours(Type a, $h=12$)	63.5	60	55.7	59.3	54.8	57.3
Ours(Type a, $h=8$)	61.7	57.5	57	55.8	55.3	55.8
Ours(Type a, $h=4$)	58.8	50.7	42	36	28.7	23.2
Ours(Type b, $h=4$)	91.3	88.5	77.5	62.7	53.7	42
Ours(Type c, $h=4$)	81.8	76.3	68.3	59.5	47.3	39.2
FSGNN($r=3$)	91.8	80.8	70.7	64.5	50.5	44.2
FSGNN($r=3$, all)	92	83.8	73.5	63.3	52.3	44.2
FSGNN($r=8$)	93.8	84	67.8	59.8	48.5	40.8
FSGNN($r=8$, all)	92.7	81.7	69.7	59.7	51.2	42.7

network, which is defined as

$$\boldsymbol{H}_1 = \mathop{\Vert}_{i=1}^{n_C} \alpha_i \cdot \mathfrak{N}(\boldsymbol{X}_i \boldsymbol{W}_i) \tag{4}$$

$$\hat{\boldsymbol{Y}} = \text{softmax}(\text{ReLU}(\boldsymbol{H}_1)\boldsymbol{W}) \tag{5}$$

where $\Vert$ denotes the concatenation operation, α_i are trainable attention weights satisfying $\alpha_i \in (0,1)$ and $\sum_i \alpha_i = 1$, $\mathfrak{N}(\cdot)$ is the row normalization operation, and $\boldsymbol{W}_i \in \mathbb{R}^{d_i \times n_h}$ and $\boldsymbol{W} \in \mathbb{R}^{n_C n_h \times n_c}$ are trainable parameters. Our model comprises several input channels at the beginning and, subsequently, several fully connected layers. Therefore, it is easy to be extended with more layers. As usual, we minimize the cross entropy of the labeled nodes using the first l columns of $\hat{\boldsymbol{Y}}$ and $\boldsymbol{Y}$.

Given the binary Haar graph framelet system $\mathcal{F}_{j_0}(\mathcal{P}_K)$, we also use $\boldsymbol{\Phi}_j = (\boldsymbol{\phi}_{\boldsymbol{\lambda}})_{\dim(\boldsymbol{\lambda})=j} \in \mathbb{R}^{N_j \times n}$ and $\boldsymbol{\Psi}_j = (\boldsymbol{\psi}_{(\boldsymbol{\lambda},m)})_{\dim(\boldsymbol{\lambda})=j, m\in[M_{\boldsymbol{\lambda}}]} \in \mathbb{R}^{M_j \times n}$ to be the matrix representations of the scaling vectors and framelet vectors at scale j, respectively. We denote $\boldsymbol{F}_0(\boldsymbol{M}) := \boldsymbol{\Phi}_1^\top \boldsymbol{\Phi}_1 \boldsymbol{M}$ and $\boldsymbol{F}_j(\boldsymbol{M}) := \boldsymbol{\Psi}_j^\top \boldsymbol{\Psi}_j \boldsymbol{M}$, $1 \le j \le K-1$. For our model PEGFAN, we select three options for $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_{n_C}\}$ of feature matrices for graphs with homophily and heterophily.

For *homophilous graphs*, we have three types.

1) *Type a:* $n_C = 1 + K, \{\boldsymbol{X}, \boldsymbol{F}_0(\boldsymbol{X}), \boldsymbol{F}_1(\boldsymbol{X}), \ldots, \boldsymbol{F}_{K-1}(\boldsymbol{X})\}$.
2) *Type b:* $n_C = 1 + r + K, \{\boldsymbol{X}, \tilde{\boldsymbol{A}}\boldsymbol{X}, \tilde{\boldsymbol{A}}^2\boldsymbol{X}, \ldots, \tilde{\boldsymbol{A}}^r\boldsymbol{X}, \boldsymbol{F}_0(\boldsymbol{X}), \boldsymbol{F}_1(\boldsymbol{X}), \ldots, \boldsymbol{F}_{K-1}(\boldsymbol{X})\}$.
3) *Type c:* $n_C = 1 + r + K, \{\boldsymbol{X}, \tilde{\boldsymbol{A}}\boldsymbol{X}, \tilde{\boldsymbol{A}}^2\boldsymbol{X}, \ldots, \tilde{\boldsymbol{A}}^r\boldsymbol{X}, \boldsymbol{F}_0(\tilde{\boldsymbol{A}}\boldsymbol{X}), \boldsymbol{F}_1(\tilde{\boldsymbol{A}}\boldsymbol{X}), \ldots, \boldsymbol{F}_{K-1}(\tilde{\boldsymbol{A}}\boldsymbol{X})\}$.

For *heterophilous graphs*, we have three types.

1) *Type a:* $n_C = 1 + K, \{\boldsymbol{X}, \boldsymbol{F}_0(\boldsymbol{X}), \boldsymbol{F}_1(\boldsymbol{X}), \ldots, \boldsymbol{F}_{K-1}(\boldsymbol{X})\}$.
2) *Type b:* $n_C = 1 + r + K, \{\boldsymbol{X}, \boldsymbol{A}\boldsymbol{X}, \boldsymbol{A}^2\boldsymbol{X}, \ldots, \boldsymbol{A}^r\boldsymbol{X}, \boldsymbol{F}_0(\boldsymbol{X}), \boldsymbol{F}_1(\boldsymbol{X}), \ldots, \boldsymbol{F}_{K-1}(\boldsymbol{X})\}$.
3) *Type c:* $n_C = 1 + r + K, \{\boldsymbol{X}, \boldsymbol{A}\boldsymbol{X}, \boldsymbol{A}^2\boldsymbol{X}, \ldots, \boldsymbol{A}^r\boldsymbol{X}, \boldsymbol{F}_0(\boldsymbol{A}\boldsymbol{X}), \boldsymbol{F}_1(\boldsymbol{A}\boldsymbol{X}), \ldots, \boldsymbol{F}_{K-1}(\boldsymbol{A}\boldsymbol{X})\}$.

With the permutation equivariance of our graph Haar framelets, now we can formally state the permutation equivariance of our graph framelet neural network model PEGFAN.

Proposition 2: Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph with feature matrix $\boldsymbol{X}$, adjacency matrix $\boldsymbol{A}$, and a K-hierarchical partition $\mathcal{P}_K$. Let $\boldsymbol{P}$ be a permutation matrix with respect to a node permutation π on $\mathcal{V}$. If the permuted feature matrix $\boldsymbol{P}\boldsymbol{X}$, adjacency matrix $\boldsymbol{P}\boldsymbol{A}\boldsymbol{P}^\top$, and binary Haar graph framelet system $\pi(\mathcal{A}(\mathcal{G}, \mathcal{P}_K))$ are used in forming Type a, b, and c channels for PEGFAN, then the new output $\hat{\boldsymbol{Y}}_{\boldsymbol{P}}$ differs from the original one by a permutation matrix, i.e., $\hat{\boldsymbol{Y}}_{\boldsymbol{P}} = \boldsymbol{P}\hat{\boldsymbol{Y}}$.

Remark 5: In contrast to our PEGFAN, the model FSGNN [39] adopts the two-layer network model with the following three options of input channels.

1) *Homophily:* $n_C = 1 + r, \{\boldsymbol{X}, \tilde{\boldsymbol{A}}\boldsymbol{X}, \tilde{\boldsymbol{A}}^2\boldsymbol{X}, \ldots, \tilde{\boldsymbol{A}}^r\boldsymbol{X}\}$.
2) *Heterophily:* $n_C = 1 + r, \{\boldsymbol{X}, \boldsymbol{A}\boldsymbol{X}, \boldsymbol{A}^2\boldsymbol{X}, \ldots, \boldsymbol{A}^r\boldsymbol{X}\}$.
3) *All:* $n_C = 1 + 2r, \{\boldsymbol{X}, \boldsymbol{A}\boldsymbol{X}, \tilde{\boldsymbol{A}}\boldsymbol{X}, \boldsymbol{A}^2\boldsymbol{X}, \tilde{\boldsymbol{A}}^2\boldsymbol{X}, \ldots, \boldsymbol{A}^r\boldsymbol{X}, \tilde{\boldsymbol{A}}^r\boldsymbol{X}\}$.

As shown in Fig. 3, our network model differs from existing GNNs using graph wavelets/framelets in the sense that we fully utilize the multiscale property of our Haar graph framelets as well as the powers of the adjacency matrix as the multichannel inputs. In such a way, short-range information and long-range information of the graph are fully exploited for the training of the network model. On the contrary, neural network architectures of other existing GNNs using graph wavelets/framelets are similar to classical spectral GNNs, which are essentially different from ours in exploiting multiscale information.

V. EXPERIMENTS

A. Experiment on Synthetic Dataset

In [52], it has been theoretically shown that for a linear classifier, using $\boldsymbol{A}_{rw} := \boldsymbol{D}^{-1}\boldsymbol{A}$ to aggregate features has a lower probability of misclassifying under the condition that the "neighborhood class distributions" are distinguishable. To elaborate, it assumes that for each node i of class $y_i = c$, the neighbors of i are sampled from a distribution $\mathcal{D}_{y_i}$, and the distribution $\mathcal{D}_c$'s are different. For heterophilous graphs, it is possible to fit the aforementioned condition as long as for each node of some class, the connection pattern with nodes from each class is different from the patterns of nodes of a different class. In other words, using simple neighborhood aggregation such as $\boldsymbol{A}_{rw}\boldsymbol{X}$ in GNNs still has the chance to achieve good performance for heterophilous graphs and the experiments in [52] has empirically validated this statement.

Following their observation, we are interested in how the neighborhood distribution $\mathcal{D}_c$ affects the performance of FSGNN and PEGFAN. We follow the way in [52] and generate four-class heterophilous graphs with 3000 nodes, fixed Gaussian features, and different neighborhood class distributions. The proportion of training, validation, and test set was set to 48%, 32%, and 20%, respectively. We compare the performance of PEGFAN with FSGNN to demonstrate the ability of multiscale extraction when our binary Haar graph framelet system is added. To emphasize the difference between graph framelets and K-hop aggregation, we excluded

the feature matrix channel $\boldsymbol{X}$ in the overall channels. A hyperparameter $\gamma \in [0, 1]$ indicates the tendency to sample edges from uniform neighborhood class distribution. Consequently, larger γ results in more indistinguishable neighborhood class distributions. Implementation details are the same as shown in this section for the benchmark datasets except that the hyperparameter search range is reduced and h is set to 4, 8, and 12 (cf. Theorem 3). More details of the synthetic dataset experiment are given in Appendix C.

Table I [see also Fig. 4(a)] collects the results from the experiment following the procedure defined in Appendix C. Table II [see also Fig. 4(b)] contains the results of replacing features sampled Gaussian distributions with closer means, which are more similar for different classes and more difficult to classify.

B. Comments on the Synthetic Dataset Experiment

For the synthetic dataset, it can be seen from Tables I and II that in most of the cases, the performance decreases with respect to γ in all cases (if $\gamma \to 1$, then the graph is close to the case of being generated by uniform neighborhood class distribution). However, in Table I, *Type b channels* input, in which graph framelets project the original feature matrix $\boldsymbol{X}$, show a large improvement compared with FSGNN. In some cases, the increment can reach over 10%. This evidently shows the effectiveness of multiscale extractions via graph framelets when combined with adjacency matrix aggregations. There are also drawbacks, which can be seen from the results of *Type a channels input* in both tables. It shows that the results are sensitive to hyperparameter h and that using graph framelets alone is not enough. Indeed, we chose a rather simple and unsupervised way to generate hierarchical clustering. This process altered the representation of the connectivity among nodes and caused a loss of information. Therefore, it is better to combine fine-scale information using one-to-three-hop aggregation and coarse-scale projection via graph framelets. However, the performance of *Type a* channels has less variation across different γ's and is better when γ is closed to 1. It is also obvious that the neighborhood distributions affect our model performance for *Type b* and *Type c* given the theory in [52], in which the model accuracy decreased as the neighborhood distributions approached the same and indistinguishable uniform distribution. As for *Type c channels* in both tables, the channels are affected by the adjacency matrix before multiscale extraction, and thus, they perform similarly compared with FSGNN. In Table II, since it is more difficult to correctly classify nodes, *Type b channels* gain less improvement compared to Table I.

C. Experiments on Benchmark Datasets

We conducted experiments on nine datasets, including three homophilous citation networks and six heterophilous datasets, and followed the public data splits provided in [28]. We define the density of a graph as $\|\boldsymbol{A}\|_0/n^2$, which is the proportion between the number of nonzero terms in $\boldsymbol{A}$ and the numbers of terms of $\boldsymbol{A}$. The statistics is summarized in the top rows of Table III. To generate a series of partitions for each dataset, we applied *sknetwork.hierarchy.Ward* and *sknetwork.hierarchy.cut_balanced* from python package *scikit-network*[1] to form intermediate clusters and control the hyperparameter h in Theorem 3. h is set to 4 and 8 and the values are indicated in Table III. Once new partition $\mathcal{V}''$ of clusters is formed from a graph $G' = (\mathcal{V}', \boldsymbol{A}')$, we define as follows the new adjacency matrix $\boldsymbol{A}''$ to form the graph $G'' = (\mathcal{V}'', \boldsymbol{A}'')$ for next level clustering:

$$\boldsymbol{A}''_{ij} = \sum_{p=1}^{n'} \sum_{q=p+1}^{n'} \boldsymbol{A}'^{pq} \delta(\mathrm{ID}(p), i)\delta(\mathrm{ID}(q), j)$$

where $\#\mathcal{V}' = n'$, $\#\mathcal{V}'' = m'$, $\mathrm{ID}(p)$ and $\mathrm{ID}(q)$ are the indices of clusters that nodes p and q belong to, and $\delta(a, b)$ takes 1 when $a = b$. For heterophilous graphs, we iterate for a few steps until the final graph has less than $h = 4$ or $h = 8$ nodes. For *Pubmed*, when $h = 4$, we constrained the number of steps of generating hierarchical clustering to be 6 so as to reduce input channels.

As for the implementation of the neural network,[2] we adopted the publicly released code of FSGNN[3] for integrating the graph framelet projections as detailed in our PEGFAN model. We use the same optimizer, hidden layer size, and so on, as those in FSGNN, and hence, the details are omitted. We noticed that the outcome of FSGNN was a bit different from those reported in [39] when we tried to reproduce the results. Therefore, we did a separate grid search for FSGNN and the results had slight changes. For our model, we set r of input channels to 3. Results of other models (Mixhop [38], GEOM-GCN [28], GCNII [53], H2GCN-1 [12], WRGAT [32], and GPRGNN [33]) are cited from [39] and the results of some of the top rows are omitted, which are not among the models with a relatively superior performance. All results are collected in Table III.

As a brief comparison, Table IV (see also Fig. 5) summarizes the average, maximum, and minimum training time of our model and FSGNN on *Chameleon* and *Squirrel* over 108 sets of hyperparameters shown in Table VI. Each training consists of ten individual training, each of which is on a single data split. All experiments in this article were conducted using an RTX 3090 graphics card.

D. Comments on Benchmark Dataset Experiments

We provide in Table III not only the performance of many state-of-the-art models but also their performance on both the homophilous and heterophilous graph datasets (nine datasets in total).

As pointed out in Section I, traditional models are usually with the underlying assumption of homophily. They perform well for homophilous graph datasets. One can clearly see from Table III that the best performances for the three typical homophilous datasets (*Cora*, *Citeseer*, and *Pubmed*) are given by GEOM-GCN, GCNII, and GPRGNN. For the homophilous datasets, their nature of being homophilous does not necessitate the need for further multiscale information, and thus, our method has a similar performance. The same drawback is shown in the results of synthetic data, where the results of *Type a channels* are not superior and sensitive to the hyperparameter h. It empirically shows that to use framelets alone, it is required to form sufficiently large clusters at the beginning of forming hierarchical partitions.

[1] https://scikit-network.readthedocs.io/en/v0.26.0/

[2] https://github.com/zrgcityu/PEGFAN

[3] https://github.com/sunilkmaurya/FSGNN/

TABLE III

DATASET STATISTICS, CLASSIFICATION ACCURACY, AND STANDARD DEVIATION. **BEST** IN BOLD AND SECOND BEST IN BLUE

	Cora	Citeseer	Pubmed	Texas	Wisconsin	Cornell	Actor	Chameleon	Squirrel	Avg.	Rank
Node	2,708	3,327	19,717	183	251	183	7,600	2,277	5,201	-	-
$\|\boldsymbol{A}\|_0$	10,556	9,228	88,651	325	515	298	30,019	36,101	217,073	-	-
Feature	1,433	3,703	500	1,703	1,703	1,703	931	2,325	2,089	-	-
Density	$1.44 \cdot 10^{-3}$	$8.34 \cdot 10^{-4}$	$2.28 \cdot 10^{-4}$	$9.70 \cdot 10^{-3}$	$8.17 \cdot 10^{-3}$	$8.90 \cdot 10^{-3}$	$5.20 \cdot 10^{-4}$	$6.96 \cdot 10^{-3}$	$8.02 \cdot 10^{-3}$	-	-
Class	6	7	3	5	5	5	5	5	5	-	-
Type	Homophily	Homophily	Homophily	Heterophily	Heterophily	Heterophily	Heterophily	Heterophily	Heterophily	-	-
Mixhop	87.61±0.85	76.26±1.33	85.31±0.61	77.84±7.73	75.88±4.90	73.51±6.34	32.22±2.34	60.50±2.53	43.80±1.48	68.10	14
GEOM-GCN	85.27	**77.99**	90.05	67.57	64.12	60.81	31.63	60.90	38.14	64.05	15
GCNII	88.01±1.33	77.13±1.38	**90.30±0.37**	77.84±5.64	81.57±4.98	76.49±4.37	-	62.48±2.74	-	-	-
H2GCN-1	86.92±1.37	77.07±1.64	89.40±0.34	84.86±6.77	86.67±4.69	82.16±4.80	35.86±1.03	57.11±1.58	36.42±1.89	70.72	13
WRGAT	88.20±2.26	76.81±1.89	88.52±0.92	83.62±5.50	86.98±3.78	81.62±3.90	**36.53±0.77**	65.24±0.87	48.85±0.78	72.93	11
GPRGNN	**88.49±0.95**	77.08±1.63	88.99±0.40	86.49±4.83	85.88±3.70	81.89±6.17	36.04±0.96	66.47±2.47	49.03±1.28	73.37	10
FSGNN($r = 3$)	86.92±1.66	77.18±1.27	89.71±0.45	84.51±4.71	**87.84±3.37**	84.86±4.56	35.26±1.01	78.60±0.71	73.93±2.00	77.65	5
FSGNN($r = 8$)	88.15±1.15	77.23±1.41	89.67±0.45	**86.76±3.72**	87.65±3.51	85.95±5.10	35.22±0.96	79.01±1.23	73.78±1.58	78.16	2nd
FSGNN($r = 3$, all)	87.59±1.03	76.91±1.60	89.68±0.37	84.60±5.41	86.67±2.75	86.22±6.78	35.51±0.89	77.68±1.10	73.79±2.32	77.63	6
FSGNN($r = 8$, all)	87.53±1.37	76.86±1.49	89.73±0.40	82.70±5.01	85.88±5.02	85.13±7.57	35.28±0.79	77.94±1.17	74.04±1.51	77.23	8
Ours($h = 4$, Type a)	79.48±2.68	71.29±2.01	88.46±0.35	83.78±5.54	86.08±4.34	85.95±5.51	34.96±1.24	65.83±2.05	51.98±1.98	71.97	12
Ours($h = 4$, Type b)	87.12±0.91	77.39±1.28	89.62±0.25	86.47±5.54	86.67±3.59	85.14±5.57	35.07±1.03	79.63±1.23	73.89±1.89	77.88	4
Ours($h = 4$, Type c)	87.36±1.09	76.78±1.51	89.55±0.32	85.14±4.05	87.65±4.02	**86.76±5.33**	35.41±0.82	79.65±1.33	74.58±2.07	78.10	3rd
Ours($h = 8$, Type a)	83.16±1.86	73.51±1.67	88.85±0.30	84.32±3.78	86.67±3.80	84.05±6.10	35.15±0.77	77.48±1.71	71.10±1.75	76.03	9
Ours($h = 8$, Type b)	87.22±1.21	76.76±1.40	89.73±0.40	84.87±5.70	85.69±3.29	84.60±5.41	35.34±0.81	79.21±1.09	73.09±1.66	77.39	7
Ours($h = 8$, Type c)	87.16±1.31	76.92±1.57	89.56±0.30	86.22±3.30	86.67±4.28	86.22±4.75	35.48±0.94	**80.31±1.10**	**75.06±1.72**	**78.18**	1st

TABLE IV

TRAINING TIME OVER 108 CONFIGURATIONS OF HYPERPARAMETERS. NUMBER OF CHANNELS: FSGNN ($r = 3$): 4, FSGNN ($r = 8$): 9, AND OURS (TYPE C, $h = 4$): 13 (CHAMELEON) AND 14 (SQURRIEL)

Chameleon	avg.	max.	min.
FSGNN($r = 3$)	20.71s	64.68s	11.21s
FSGNN($r = 8$)	37.41s	90.70s	17.75s
Ours(Type c, $h = 4$)	47.66s	125.25s	22.52s

Squirrel	avg.	max.	min.
FSGNN($r = 3$)	34.57s	83.24s	18.58s
FSGNN($r = 8$)	53.40s	141.24s	26.52s
Ours(Type c, $h = 4$)	59.82s	198.77s	32.87s

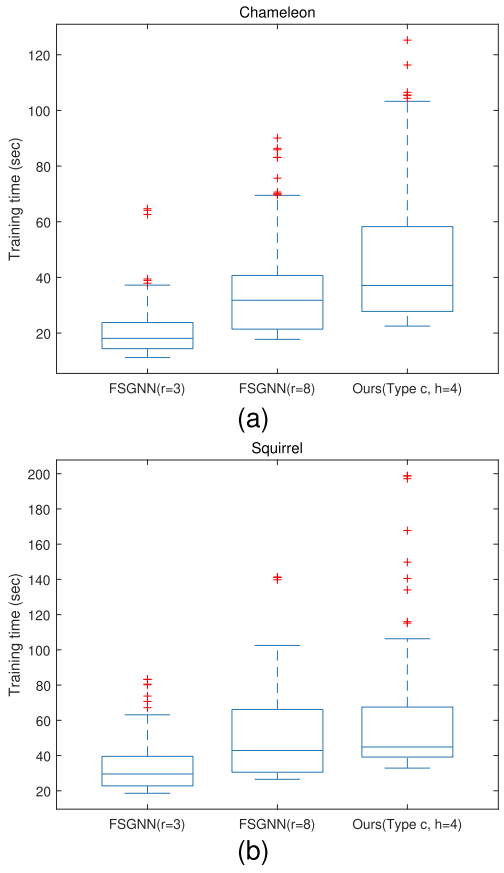

Fig. 5. Demonstration of training time comparison. (a) Chamemleon dataset. (b) Squirrel dataset.

While models, such as GEOM-GCN, GCNII, and GPRGNN, perform well in those homophilous datasets, they do not give the best performance for the other six heterophilous datasets. The models that give the best performance for heterophilous datasets are FSGNN and our PEGFAN.

Now, between FSGNN and our PEGFAN, from the above discussion, we only need to focus on the six heterophilous datasets: *Texas*, *Wisconsin*, *Cornell*, *Actor*, *Chamelon*, and *Squirrel*. We would like to emphasize that we follow the most common way that uses the public data splits in [28]. The proportions of train–validation–test splits are all 48%, 32%, and 20%, respectively. These six datasets can be considered as three groups discussed as follows.

The first group is the datasets of *Texas*, *Wisconsin*, and *Cornell*. They are similar datasets with a small number of nodes, edges, and features. Since the test sets are only 20% of the graph, they contain at most 51 nodes. A correctly predicted node accounts for at least 1.9% of accuracy. Hence, we can say that experiments on such datasets are relatively and statistically insignificant. Most of the models have very similar performance with at most seven nodes wrongly predicted. Nonetheless, we chose to follow the common conduct and report the results for completeness. PEGFAN is best for *Cornell* and ranks second for *Texas*. FSGNN is best for *Wisconsin* and *Texas* and ranks third for *Cornell*. The best or second-best performances of FSGNN and PEGFAN are without much difference. Since the number of nodes is too small, it is not reasonable to say that one is better than the other.

The second group is simply the dataset *Actor*. It is a large dataset with 7600 nodes. However, for this Actor dataset,

all models, including Mixhop, GEOM-GCN, and GCNII, do not give reasonable performance. They only give very low accuracy about 35%. The best performance is given by the model WRGAT. For this dataset, it is not reasonable to compare the performance among different methods.

The last group is the datasets of *Chameleon* and *Squirrel*. They are both big datasets in terms of nodes and edges. We can see that PEGFAN performs the best. Being heterophilous makes it necessary to gather multiscale information, and denser graphs facilitate forming better series of partitions and thus better graph framelets. This is also consistent with the empirical results of the synthetic dataset since, as shown in [52], the neighborhood distributions of *Chameleon* and *Squirrel* are distinguishable enough for different classes, while other heterophilous datasets either are small datasets that suffer from bias or do not fit such a condition.

Moreover, to compare the *overall performance* of each method on the nine benchmark datasets, we take the average of the performance of each method over the nine datasets. The average score of each method is given in the second last column (Avg.) of Table III. Our method with respect to $h = 8$ and *Type c* (the last row) ranks first among the comparing methods. In general, our *Type c* methods outperform other methods with high average overall performance (see the last column of Table III for the ranking of each method).

VI. Conclusion

This article proposes a novel and general method to construct Haar-type framelets on graphs that are permutation equivariant. It aims to serve as an alternative and supplement for multihop aggregations using powers of adjacency matrices. The results show that combining graph framelets and multihop aggregation increases the performance of node classification on heterophilous graphs in both synthetic and real-world data. Moreover, compared with using multihop aggregation alone, in the synthetic case, our model shows significant increases against the deterioration of neighborhood distribution and results show consistency between the synthetic and benchmark datasets in terms of the patterns of neighborhood distribution. The overall results validate the capability of our graph framelets to extract multiscale information under certain conditions and its superior performance. We would also like to mention that choosing a more sophisticated way to generate the hierarchical partitions has the potential to produce better graph framelets, which will be a future experimental direction to be explored. In addition, theoretical investigations on the impact of the heterophily ratio on the expressive capabilities of framelet-based GNNs are expected. Such studies could inspire more advanced GNNs tailored for heterophilous graphs. Building on our work, it would be beneficial to theoretically and empirically explore the potential interplay between key issues such as homophily versus heterophily, oversmoothing, and oversquashing, all through the lens of graph wavelets/framelets. We plan to delve into these significant directions in our subsequent research efforts.

Appendix

A. Proofs of Theoretical Results

Proof of Theorem 1: We denote $\boldsymbol{\Phi}_j = \{\boldsymbol{\phi}_{\boldsymbol{\Lambda}}\}_{\dim(\boldsymbol{\Lambda})=j}$ and $\boldsymbol{\Psi}_j = \{\boldsymbol{\psi}_{(\boldsymbol{\Lambda},m)}\}_{\dim(\boldsymbol{\Lambda})=j,m\in[M_{\boldsymbol{\Lambda}}]}$. Let $V_j := \operatorname{span}\boldsymbol{\Phi}_j$ and $W_j := \operatorname{span}\boldsymbol{\Psi}_j$. Note that supports of $\boldsymbol{\phi}_{\boldsymbol{\Lambda}}$ and $\boldsymbol{\phi}_{\boldsymbol{\Lambda}'}$ are disjoint if $\boldsymbol{\Lambda} \neq \boldsymbol{\Lambda}'$, so are $\boldsymbol{\psi}_{(\boldsymbol{\Lambda},m)}$ and $\boldsymbol{\psi}_{(\boldsymbol{\Lambda}',m')}$. Hence, by definition and $\|\boldsymbol{p}_{\boldsymbol{\Lambda}}\| = 1$, we can see that $\boldsymbol{\Phi}_j$ forms an orthonormal basis of V_j for each j. Thus, in [19, Lemma 1], the conditions $\boldsymbol{B}_{\boldsymbol{\Lambda}}\boldsymbol{B}_{\boldsymbol{\Lambda}}^\top\boldsymbol{B}_{\boldsymbol{\Lambda}} = \boldsymbol{B}_{\boldsymbol{\Lambda}}$, $\boldsymbol{B}_{\boldsymbol{\Lambda}}\boldsymbol{p}_{\boldsymbol{\Lambda}} = \boldsymbol{0}$, and $\operatorname{Rank}(\boldsymbol{B}_{\boldsymbol{\Lambda}}) = L_{\boldsymbol{\Lambda}} - 1$ are equivalent to that $V_{j+1} = V_j \oplus W_j$ and $\{\boldsymbol{\phi}_{\boldsymbol{\Lambda}}\}_{\dim(\boldsymbol{\Lambda})=j} \cup \{\boldsymbol{\psi}_{(\boldsymbol{\Lambda},m)}\}_{\dim(\boldsymbol{\Lambda})=j,m\in[M_{\boldsymbol{\Lambda}}]}$ is a tight frame of V_{j+1}. Iteratively, for $j_0 < j$, we deduce that $V_{j_0} \oplus W_{j_0} \oplus \cdots \oplus W_{j-1} = V_j$ and $\Phi_{j_0} \cup \Psi_{j_0} \cup \cdots \cup \Psi_{j-1}$ is a tight frame for V_j if and only if matrices $\boldsymbol{B}_{\boldsymbol{\Lambda}}$ and vectors $\boldsymbol{p}_{\boldsymbol{\Lambda}}$ satisfy $\boldsymbol{B}_{\boldsymbol{\Lambda}}\boldsymbol{B}_{\boldsymbol{\Lambda}}^\top\boldsymbol{B}_{\boldsymbol{\Lambda}} = \boldsymbol{B}_{\boldsymbol{\Lambda}}$, $\boldsymbol{B}_{\boldsymbol{\Lambda}}\boldsymbol{p}_{\boldsymbol{\Lambda}} = \boldsymbol{0}$, and $\operatorname{Rank}(\boldsymbol{B}_{\boldsymbol{\Lambda}}) = L_{\boldsymbol{\Lambda}} - 1$ for all $\boldsymbol{\Lambda}$ with $\dim(\boldsymbol{\Lambda}) = j_0, \ldots, j$. Now, the conclusion of the theorem follows by letting $j = K$ and noting that $\mathcal{F}_{j_0}(\mathcal{P}_K) = \Phi_{j_0} \cup \Psi_{j_0} \cup \cdots \cup \Psi_{K-1}$ as well as $V_K = L_2(\mathcal{G})$. □

Proof of Proposition 1: If $\boldsymbol{B}^\top\boldsymbol{B} = c(\boldsymbol{I} - \boldsymbol{p}\boldsymbol{p}^\top)$, then $\boldsymbol{B}\boldsymbol{B}^\top\boldsymbol{B} = c\boldsymbol{B}$ by direct computation and in view of $\boldsymbol{B}\boldsymbol{p} = \boldsymbol{0}$. Conversely, if $\boldsymbol{B}\boldsymbol{B}^\top\boldsymbol{B} = c\boldsymbol{B}$ for some constant c, then, by $\boldsymbol{B}(\boldsymbol{B}^\top\boldsymbol{B}) = c\boldsymbol{B} = c\boldsymbol{B}(\boldsymbol{I} - \boldsymbol{p}\boldsymbol{p}^\top)$ and $\boldsymbol{p}^\top(\boldsymbol{I} - \boldsymbol{p}\boldsymbol{p}^\top) = \boldsymbol{0}$, we have $\begin{bmatrix}\boldsymbol{p}^\top \\ \boldsymbol{B}\end{bmatrix}(\boldsymbol{B}^\top\boldsymbol{B} - c(\boldsymbol{I} - \boldsymbol{p}\boldsymbol{p}^\top)) = \boldsymbol{0}$. Consequently, by the full rank property of the matrix $\begin{bmatrix}\boldsymbol{p}, \boldsymbol{B}^\top\end{bmatrix}$, we conclude that $\boldsymbol{B}^\top\boldsymbol{B} = c(\boldsymbol{I} - \boldsymbol{p}\boldsymbol{p}^\top)$. The particular part is followed by direction evaluation. We are done. □

Proof of Corollary 1: We only need to show that $\boldsymbol{B}_{\boldsymbol{\Lambda}}$ and $\boldsymbol{p}_{\boldsymbol{\Lambda}}$ satisfy $\boldsymbol{B}_{\boldsymbol{\Lambda}}\boldsymbol{B}_{\boldsymbol{\Lambda}}^\top\boldsymbol{B}_{\boldsymbol{\Lambda}} = \boldsymbol{B}_{\boldsymbol{\Lambda}}$, $\boldsymbol{B}_{\boldsymbol{\Lambda}}\boldsymbol{p}_{\boldsymbol{\Lambda}} = \boldsymbol{0}$, and $\operatorname{Rank}(\boldsymbol{B}_{\boldsymbol{\Lambda}}) = L_{\boldsymbol{\Lambda}} - 1$. Obviously, $\boldsymbol{B}_{\boldsymbol{\Lambda}}\boldsymbol{p}_{\boldsymbol{\Lambda}} = \boldsymbol{0}$. Define $\boldsymbol{A}_{\boldsymbol{\Lambda}} := [\boldsymbol{p}_{\boldsymbol{\Lambda}}, \boldsymbol{B}_{\boldsymbol{\Lambda}}^\top]^\top$. By direct evaluations, one can show that the columns of $\boldsymbol{A}_{\boldsymbol{\Lambda}}$ satisfy $\|[\boldsymbol{A}_{\boldsymbol{\Lambda}}]_{:\ell_1}\| = 1$ and their inner product $\langle[\boldsymbol{A}_{\boldsymbol{\Lambda}}]_{:\ell_1}, [\boldsymbol{A}_{\boldsymbol{\Lambda}}]_{:\ell_2}\rangle = 0$ for all $\ell_1 \neq \ell_2$, that is, $\boldsymbol{A}_{\boldsymbol{\Lambda}}^\top\boldsymbol{A}_{\boldsymbol{\Lambda}} = \boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix of size $L_{\boldsymbol{\Lambda}}$. Consequently, we deduce that $\boldsymbol{B}_{\boldsymbol{\Lambda}}^\top\boldsymbol{B}_{\boldsymbol{\Lambda}} = \boldsymbol{A}_{\boldsymbol{\Lambda}}^\top\boldsymbol{A}_{\boldsymbol{\Lambda}} - \boldsymbol{p}_{\boldsymbol{\Lambda}}\boldsymbol{p}_{\boldsymbol{\Lambda}}^\top = \boldsymbol{I} - \boldsymbol{p}_{\boldsymbol{\Lambda}}\boldsymbol{p}_{\boldsymbol{\Lambda}}^\top$, which then implies that $\boldsymbol{B}_{\boldsymbol{\Lambda}}\boldsymbol{B}_{\boldsymbol{\Lambda}}^\top\boldsymbol{B}_{\boldsymbol{\Lambda}} = \boldsymbol{B}_{\boldsymbol{\Lambda}}(\boldsymbol{I} - \boldsymbol{p}_{\boldsymbol{\Lambda}}\boldsymbol{p}_{\boldsymbol{\Lambda}}^\top) = \boldsymbol{B}_{\boldsymbol{\Lambda}}$ in view of $\boldsymbol{B}_{\boldsymbol{\Lambda}}\boldsymbol{p}_{\boldsymbol{\Lambda}} = \boldsymbol{0}$. Now, $\operatorname{Rank}(\boldsymbol{B}_{\boldsymbol{\Lambda}}) = L_{\boldsymbol{\Lambda}} - 1$ directly follows from that $\boldsymbol{A}_{\boldsymbol{\Lambda}}$ is of full column rank and $\boldsymbol{B}_{\boldsymbol{\Lambda}}\boldsymbol{p}_{\boldsymbol{\Lambda}} = \boldsymbol{0}$. We are done. □

Proof of Theorem 2: We first consider the sparsity of $\langle\boldsymbol{I}_{:1}, \boldsymbol{\psi}_{(\boldsymbol{\Lambda},m)}\rangle$, $m = 1, \ldots, M_{\boldsymbol{\Lambda}}$. Notice that only when the node $1 \in s_{\boldsymbol{\Lambda}}$, can the term $\langle\boldsymbol{I}_{:1}, \boldsymbol{\psi}_{(\boldsymbol{\Lambda},m)}\rangle$ be nonzero. Thus, without loss of generality, we assume that $1 \in s_{\boldsymbol{\Lambda}}$. Thus, by our construction in Corollary 1, at most $h - 1$ framelets $\boldsymbol{\psi}_{(\boldsymbol{\Lambda},m)}$ that make $\langle\boldsymbol{I}_{:1}, \boldsymbol{\psi}_{(\boldsymbol{\Lambda},m)}\rangle \neq 0$. For each j, only one cluster $s_{\boldsymbol{\Lambda}}$ of $\mathcal{V}_j = \{s_{\boldsymbol{\Lambda}} : \dim(\boldsymbol{\Lambda}) = j\}$ contains node 1. Thus, $\boldsymbol{F}^\top\boldsymbol{I}_{:1}$ has at most $(h - 1)(K - 1)$ nonzero entries. Similar results hold for $\boldsymbol{I}_{:i}$. Hence, for $\boldsymbol{f} = [f_1, \ldots, f_n]^\top$, it is easy to show that $\|\hat{\boldsymbol{f}}\|_0 = \|\boldsymbol{F}^\top\boldsymbol{f}\|_0 = \|\sum_{i\in[n], f_i\neq 0}\boldsymbol{F}^\top\boldsymbol{I}_{:i}\|_0 \leq \sum_{i\in[n], f_i\neq 0}\|\boldsymbol{F}^\top\boldsymbol{I}_{:i}\|_0 \leq (h - 1)(K - 1)\|\boldsymbol{f}\|_0$. □

For generating framelets, we use Algorithm 1 [see (1) and (2)]. Its efficiency is discussed in Theorem 2.

Proof of Theorem 3: Note that we have $n \leq Ch^{K-1}$ and $\#\mathcal{V}_j = \#\{\boldsymbol{\Lambda} : \dim(\boldsymbol{\Lambda}) = j\} \leq Ch^{j-1}$ for some fixed constant $C > 0$. Moreover, $M_{\boldsymbol{\Lambda}} = (L_{\boldsymbol{\Lambda}}(L_{\boldsymbol{\Lambda}} - 1)/2) \leq (h(h - 1)/2)$. Therefore, there is no more than $C(h^{j_0-1} + \sum_{j=j_0}^{K-1}(1/2)h(h - 1)h^{j-1}) = O(nh)$ elements in the binary graph Haar framelet system $\mathcal{F}_{j_0}(\mathcal{P}_K)$ for any $j_0 \in [K]$. By (1) and (2), we have

$$\boldsymbol{\phi}_{\boldsymbol{\Lambda}}^\top := \boldsymbol{p}_{\boldsymbol{\Lambda}}^\top\begin{bmatrix}\boldsymbol{\phi}_{(\boldsymbol{\Lambda},1)}^\top \\ \vdots \\ \boldsymbol{\phi}_{(\boldsymbol{\Lambda},L_{\boldsymbol{\Lambda}})}^\top\end{bmatrix}, \quad \begin{bmatrix}\boldsymbol{\psi}_{(\boldsymbol{\Lambda},1)}^\top \\ \vdots \\ \boldsymbol{\psi}_{(\boldsymbol{\Lambda},M_{\boldsymbol{\Lambda}})}^\top\end{bmatrix} := \boldsymbol{B}_{\boldsymbol{\Lambda}}\begin{bmatrix}\boldsymbol{\phi}_{(\boldsymbol{\Lambda},1)}^\top \\ \vdots \\ \boldsymbol{\phi}_{(\boldsymbol{\Lambda},L_{\boldsymbol{\Lambda}})}^\top\end{bmatrix}. \quad (6)$$

By our construction, there is at most h^{K-j} nonzero entries for each $\boldsymbol{\phi}_{\boldsymbol{\Lambda}}$ and at most $2 \cdot h^{K-j-1}$ nonzero entries for each

Algorithm 1 Generating Framelets

Input: Node set $\mathcal{V}$, Partition $\mathcal{P}_K$, Vectors $\{\boldsymbol{p}_{\boldsymbol{\Lambda}}\}$, Matrices $\{\boldsymbol{B}_{\boldsymbol{\Lambda}}\}$, j_0

initialize $\mathcal{F}_{j_0}(\mathcal{P}_K)=\emptyset$, $\boldsymbol{\phi}_{\boldsymbol{\Lambda}}=\boldsymbol{I}_{:i}$ for any $s_{\boldsymbol{\Lambda}}=\{i\}$.

for $j=2$ **to** j_0-1 **do**

  for $\boldsymbol{\Lambda}\in\{\boldsymbol{\Lambda}:\dim(\boldsymbol{\Lambda})=j\}$ **do**

    $\boldsymbol{\phi}_{\boldsymbol{\Lambda}}:=\sum_{\ell\in[L_{\boldsymbol{\Lambda}}]}p_{(\boldsymbol{\Lambda},\ell)}\boldsymbol{\phi}_{(\boldsymbol{\Lambda},\ell)}$

    for $m=1$ **to** $M_{\boldsymbol{\Lambda}}$ **do**

      $\boldsymbol{\psi}_{(\boldsymbol{\Lambda},m)}:=\sum_{\ell\in[L_{\boldsymbol{\Lambda}}]}(\boldsymbol{B}_{\boldsymbol{\Lambda}})_{m,\ell}\boldsymbol{\phi}_{(\boldsymbol{\Lambda},\ell)}$

    update $\mathcal{F}_{j_0}(\mathcal{P}_K)\leftarrow\mathcal{F}_{j_0}(\mathcal{P}_K)\cup\{\boldsymbol{\phi}_{\boldsymbol{\Lambda}},\boldsymbol{\psi}_{(\boldsymbol{\Lambda},m),m=1,\ldots,M_{\boldsymbol{\Lambda}}}\}$

Output: $\mathcal{F}_{j_0}(\mathcal{P}_K)$

$\boldsymbol{\psi}_{(\boldsymbol{\Lambda},m)}$ for $\dim(\boldsymbol{\Lambda})=j$. Hence, the number of nonzero entries of $\boldsymbol{F}$ is at most $C(h^{K-j_0}\cdot h^{j_0-1}+\sum_{j=j_0}^{K-1}2\,h^{K-j-1}\cdot(h(h-1)/2)\cdot h^{j-1})\le C(K-1)h^K=O(nh\log_h n)$. Fix $\boldsymbol{\Lambda}$, which has size $\dim(\boldsymbol{\Lambda})=j$. Then, (9) implies at most $h\cdot h^{K-j-1}$ multiplication operations and $(h-1)\cdot h^{K-j-1}$ addition operations needed for $\boldsymbol{\phi}_{\boldsymbol{\Lambda}}$.For computing $\Psi_{(\boldsymbol{\Lambda},m)}$, $m=1,\ldots,M_{\boldsymbol{\Lambda}}$, we need at most $2\cdot h^{K-j-1}\cdot(h(h-1)/2)$ multiplication operations and $h^{K-j-1}\cdot(h(h-1)/2)$ addition operations. Notice that $\#\mathcal{V}_j\le Ch^{j-1}$. To compute the nonzero entries of $\boldsymbol{\phi}_{\boldsymbol{\Lambda}}$ and $\boldsymbol{\psi}_{(\boldsymbol{\Lambda},m)}$ for all $\dim(\boldsymbol{\Lambda})=j$ and $m=1,\ldots,M_{\boldsymbol{\Lambda}}$, from the above computation, one can see that it needs at most $2C(h^{K-j-1}\cdot h\cdot h^{j-1}+2h^{K-j-1}\cdot(h(h-1)/2)\cdot h^{j-1})=2C\cdot h^K$ evaluations of multiplications and additions. Hence, in total, to compute the nonzero entries of $\boldsymbol{\phi}_{\boldsymbol{\Lambda}}$ and $\boldsymbol{\psi}_{(\boldsymbol{\Lambda},m)}$ for all $\dim(\boldsymbol{\Lambda})=j_0,\ldots,K-1$ and $m=1,\ldots,M_{\boldsymbol{\Lambda}}$, it needs at most $2C\sum_{j=1}^{K-1}(h^{K-j-1}\cdot h\cdot h^{j-1}+2h^{K-j-1}\cdot(h(h-1)/2)\cdot h^{j-1})=2C(K-1)h^K=O(nh\log_h n)$ evaluations of multiplications and additions. □

Before showing the proof of Theorem 4, we want to give some comments on permutations. Notice that the construction of $\boldsymbol{p}_{\boldsymbol{\Lambda}}$ and $\boldsymbol{B}_{\boldsymbol{\Lambda}}$ only depends on $L_{\boldsymbol{\Lambda}}$. Hence, under node permutation (π or $\boldsymbol{P}_\pi$), it means that we have the following relationship between original $\boldsymbol{\phi}_{\boldsymbol{\Lambda}}$ and $\boldsymbol{\phi}^*_{\boldsymbol{\Lambda}}$, $\boldsymbol{\psi}_{\boldsymbol{\Lambda}}$ and $\boldsymbol{\psi}^*_{\boldsymbol{\Lambda}}$:

$$\left(\boldsymbol{\phi}^*_{\boldsymbol{\Lambda}}\right)^\top:=\boldsymbol{p}_{\boldsymbol{\Lambda}}^\top\begin{bmatrix}\boldsymbol{\phi}_{(\boldsymbol{\Lambda},1)}^\top\\ \vdots\\ \boldsymbol{\phi}_{(\boldsymbol{\Lambda},L_{\boldsymbol{\Lambda}})}^\top\end{bmatrix}\boldsymbol{P}_\pi \tag{7}$$

$$\begin{bmatrix}\left(\boldsymbol{\psi}^*_{(\boldsymbol{\Lambda},1)}\right)^\top\\ \vdots\\ \left(\boldsymbol{\psi}^*_{(\boldsymbol{\Lambda},M_{\boldsymbol{\Lambda}})}\right)^\top\end{bmatrix}:=\boldsymbol{B}_{\boldsymbol{\Lambda}}\begin{bmatrix}\boldsymbol{\phi}_{(\boldsymbol{\Lambda},1)}^\top\\ \vdots\\ \boldsymbol{\phi}_{(\boldsymbol{\Lambda},L_{\boldsymbol{\Lambda}})}^\top\end{bmatrix}\boldsymbol{P}_\pi. \tag{8}$$

Under partition permutation π_p, fixing $\boldsymbol{\Lambda}$ (there exists a permutation matrix $\boldsymbol{Q}_{\boldsymbol{\Lambda}}$ with respect to π_p at $\boldsymbol{\Lambda}$), we have the following relationship between original $\boldsymbol{\phi}_{\boldsymbol{\Lambda}}$ and $\boldsymbol{\phi}^*_{\boldsymbol{\Lambda}}$, and $\boldsymbol{\psi}_{\boldsymbol{\Lambda}}$ and $\boldsymbol{\psi}^*_{\boldsymbol{\Lambda}}$:

$$\left(\boldsymbol{\phi}^*_{\boldsymbol{\Lambda}}\right)^\top:=\boldsymbol{p}_{\boldsymbol{\Lambda}}^\top\boldsymbol{Q}_{\boldsymbol{\Lambda}}\begin{bmatrix}\boldsymbol{\phi}_{(\boldsymbol{\Lambda},1)}^\top\\ \vdots\\ \boldsymbol{\phi}_{(\boldsymbol{\Lambda},L_{\boldsymbol{\Lambda}})}^\top\end{bmatrix} \tag{9}$$

$$\begin{bmatrix}\left(\boldsymbol{\psi}^*_{(\boldsymbol{\Lambda},1)}\right)^\top\\ \vdots\\ \left(\boldsymbol{\psi}^*_{(\boldsymbol{\Lambda},M_{\boldsymbol{\Lambda}})}\right)^\top\end{bmatrix}:=\boldsymbol{B}_{\boldsymbol{\Lambda}}\boldsymbol{Q}_{\boldsymbol{\Lambda}}\begin{bmatrix}\boldsymbol{\phi}_{(\boldsymbol{\Lambda},1)}^\top\\ \vdots\\ \boldsymbol{\phi}_{(\boldsymbol{\Lambda},L_{\boldsymbol{\Lambda}})}^\top\end{bmatrix}. \tag{10}$$

Proof of Theorem 4: Let $\boldsymbol{\Phi}_{\boldsymbol{\Lambda}}:=[\boldsymbol{\phi}_{(\boldsymbol{\Lambda},1)},\ldots,\boldsymbol{\phi}_{(\boldsymbol{\Lambda},L_{\boldsymbol{\Lambda}})}]^\top$ and $\boldsymbol{\Psi}_{\boldsymbol{\Lambda}}:=[\boldsymbol{\psi}_{(\boldsymbol{\Lambda},1)},\ldots,\boldsymbol{\psi}_{(\boldsymbol{\Lambda},M_{\boldsymbol{\Lambda}})}]^\top$. Since the scaling vectors $\boldsymbol{\phi}_{\boldsymbol{\Lambda}}^\top=\boldsymbol{p}_{\boldsymbol{\Lambda}}^\top\boldsymbol{\Phi}_{\boldsymbol{\Lambda}}$ are defined iteratively for $\dim(\boldsymbol{\Lambda})$ decreasing from K to 1 through (1) and the framelets $\boldsymbol{\psi}_{(\boldsymbol{\Lambda},m)}$ are given by $\boldsymbol{\Psi}_{\boldsymbol{\Lambda}}=\boldsymbol{B}_{\boldsymbol{\Lambda}}\boldsymbol{\Phi}_{\boldsymbol{\Lambda}}$, we only need to prove the permutation equivariance properties for each $\boldsymbol{\Lambda}$.

For Item 1), note that by (1) and Corollary 1, $\boldsymbol{\phi}_{(\boldsymbol{\Lambda},\ell)}:\mathcal{V}\to\mathbb{R}$ only depends on $\mathcal{G}$, $\mathcal{P}_K$, and $\boldsymbol{p}_{\boldsymbol{\Lambda}}=(1/(L_{\boldsymbol{\Lambda}})^{1/2})\mathbf{1}$. For any node permutation π, $\mathcal{P}_K$ is determined by the index vectors $\boldsymbol{\Lambda}$ according to a tree structure and is independent of the node permutation π. Moreover, the vectors $\boldsymbol{p}_{\boldsymbol{\Lambda}}$ are fixed constants. Hence, iteratively, after node permutation π acting on graph $\mathcal{G}$, the new scaling vector $\boldsymbol{\phi}^\pi_{(\boldsymbol{\Lambda},\ell)}:\pi(\mathcal{V})\to\mathbb{R}$ is given by $\boldsymbol{\phi}^\pi_{(\boldsymbol{\Lambda},\ell)}=\boldsymbol{P}_\pi\boldsymbol{\phi}_{(\boldsymbol{\Lambda},\ell)}$, where $\boldsymbol{P}_\pi$ is the permutation matrix with respect to π. Consequently, the new $\boldsymbol{\Phi}^\pi_{\boldsymbol{\Lambda}}$ and $\boldsymbol{\Psi}^\pi_{\boldsymbol{\Lambda}}$ on the permuted graph $\pi(\mathcal{G})$ are given by $\boldsymbol{\Phi}^\pi_{\boldsymbol{\Lambda}}=\boldsymbol{\Phi}_{\boldsymbol{\Lambda}}\boldsymbol{P}_\pi$ and $\boldsymbol{\Psi}^\pi_{\boldsymbol{\Lambda}}=\boldsymbol{B}_{\boldsymbol{\Lambda}}\boldsymbol{\Phi}^\pi_{\boldsymbol{\Lambda}}=\boldsymbol{B}_{\boldsymbol{\Lambda}}\boldsymbol{\Phi}_{\boldsymbol{\Lambda}}\boldsymbol{P}_\pi=\boldsymbol{\Psi}_{\boldsymbol{\Lambda}}\boldsymbol{P}_\pi$. This implies the conclusion in Item 1).

For Item 2), given a partition permutation π_p acting on $\mathcal{P}_K$, we denote $\pi_p(\mathcal{P}_K)$ the hierarchical clustering with respect to such a π_p. Let $\tilde{\boldsymbol{\Lambda}}:=\pi_p(\boldsymbol{\Lambda})$ be the permuted index vector $\pi_p(\mathcal{P}_K)$ from the index vector $\boldsymbol{\Lambda}$ in $\mathcal{P}_K$. Since the partition permutation acts on the children of each $\boldsymbol{\Lambda}$ only, we have $\pi_p(\boldsymbol{\Lambda},\ell)=(\tilde{\boldsymbol{\Lambda}},\pi_{\boldsymbol{\Lambda}}(\ell))$ for some permutation $\pi_{\boldsymbol{\Lambda}}$ on $[L_{\boldsymbol{\Lambda}}]$. Then, the matrix $\boldsymbol{\Phi}_{\tilde{\boldsymbol{\Lambda}}}$ is

$$\boldsymbol{\Phi}_{\tilde{\boldsymbol{\Lambda}}}:=\left[\boldsymbol{\phi}_{(\tilde{\boldsymbol{\Lambda}},\pi_{\boldsymbol{\Lambda}}(1))},\ldots,\boldsymbol{\phi}_{(\tilde{\boldsymbol{\Lambda}},\pi_{\boldsymbol{\Lambda}}(L_{\boldsymbol{\Lambda}}))}\right]^\top=\boldsymbol{P}_{\pi_{\boldsymbol{\Lambda}}}\boldsymbol{\Phi}_{\boldsymbol{\Lambda}}$$

with $\boldsymbol{P}_{\pi_{\boldsymbol{\Lambda}}}$ being the permutation matrix with respect to $\pi_{\boldsymbol{\Lambda}}$. Then, in view of $\boldsymbol{p}_{\boldsymbol{\Lambda}}^\top\boldsymbol{P}_{\pi_{\boldsymbol{\Lambda}}}=\boldsymbol{p}_{\boldsymbol{\Lambda}}^\top$, the permuted scaling vector $\boldsymbol{\phi}_{\tilde{\boldsymbol{\Lambda}}}$ is given by

$$\begin{aligned}\left(\boldsymbol{\phi}_{\tilde{\boldsymbol{\Lambda}}}\right)^\top&=\left(\boldsymbol{\phi}_{\pi_p(\boldsymbol{\Lambda})}\right)^\top=\boldsymbol{p}_{\boldsymbol{\Lambda}}^\top\left(\boldsymbol{P}_{\pi_{\boldsymbol{\Lambda}}}\boldsymbol{\Phi}_{\boldsymbol{\Lambda}}\right)\\&=\left(\boldsymbol{p}_{\boldsymbol{\Lambda}}^\top\boldsymbol{P}_{\pi_{\boldsymbol{\Lambda}}}\right)\boldsymbol{\Phi}_{\boldsymbol{\Lambda}}=\boldsymbol{p}_{\boldsymbol{\Lambda}}^\top\boldsymbol{\Phi}_{\boldsymbol{\Lambda}}=\boldsymbol{\phi}_{\boldsymbol{\Lambda}}^\top\end{aligned}$$

that is, the new scaling vectors in $\{\boldsymbol{\phi}_{\tilde{\boldsymbol{\Lambda}}}:\dim(\tilde{\boldsymbol{\Lambda}})=j\}$ are simply the recording of $\{\boldsymbol{\phi}_{\boldsymbol{\Lambda}}:\dim(\boldsymbol{\Lambda})=j\}$ under π_p for $j=0,\ldots,K$. Thus, all scaling vectors are invariant (up to index permutation) under the partition permutation π_p. Now, for the framelet vectors $\boldsymbol{\psi}_{(\boldsymbol{\Lambda},m)}$, by (2), we have

$$\boldsymbol{\Psi}_{\tilde{\boldsymbol{\Lambda}}}=\boldsymbol{B}_{\boldsymbol{\Lambda}}\boldsymbol{\Phi}_{\tilde{\boldsymbol{\Lambda}}}=\boldsymbol{B}_{\boldsymbol{\Lambda}}\boldsymbol{P}_{\pi_{\boldsymbol{\Lambda}}}\boldsymbol{\Phi}_{\boldsymbol{\Lambda}}.$$

We claim that there exist $M_{\boldsymbol{\Lambda}}\times M_{\boldsymbol{\Lambda}}$ permutation matrix $\boldsymbol{R}_{\boldsymbol{\Lambda}}$ and sign matrix $\boldsymbol{S}_{\boldsymbol{\Lambda}}=\operatorname{diag}(c_1,\ldots,c_{M_{\boldsymbol{\Lambda}}})$ with all $c_i\in\{-1,+1\}$ such that $\boldsymbol{B}_{\boldsymbol{\Lambda}}\boldsymbol{P}_{\pi_{\boldsymbol{\Lambda}}}=\boldsymbol{S}_{\boldsymbol{\Lambda}}\boldsymbol{R}_{\boldsymbol{\Lambda}}\boldsymbol{B}_{\boldsymbol{\Lambda}}$. Then, we have

$$\boldsymbol{\Psi}_{\tilde{\boldsymbol{\Lambda}}}=\boldsymbol{B}_{\boldsymbol{\Lambda}}\boldsymbol{P}_{\pi_{\boldsymbol{\Lambda}}}\boldsymbol{\Phi}_{\boldsymbol{\Lambda}}=\boldsymbol{S}_{\boldsymbol{\Lambda}}\boldsymbol{R}_{\boldsymbol{\Lambda}}\boldsymbol{B}_{\boldsymbol{\Lambda}}\boldsymbol{\Phi}_{\boldsymbol{\Lambda}}=\boldsymbol{S}_{\boldsymbol{\Lambda}}\boldsymbol{R}_{\boldsymbol{\Lambda}}\boldsymbol{\Psi}_{\boldsymbol{\Lambda}}$$

which then concludes Item 2). Noting that $\boldsymbol{B}_{\boldsymbol{\Lambda}}\boldsymbol{P}_{\pi_{\boldsymbol{\Lambda}}}$ is to reorder the columns of $\boldsymbol{B}_{\boldsymbol{\Lambda}}$ and regardless the sign, all elements appear in each column with the same times and 1 (or -1) appears in rows of $\boldsymbol{B}_{\boldsymbol{\Lambda}}$ once. In other words, $(\boldsymbol{B}_{\boldsymbol{\Lambda}}\boldsymbol{P}_{\pi_{\boldsymbol{\Lambda}}})_{r:}=\boldsymbol{w}^\top\boldsymbol{P}_r^\top\boldsymbol{P}_{\pi_{\boldsymbol{\Lambda}}}$, which is to permute $\boldsymbol{w}=[1,-1,0,\ldots,0]^\top$ (up to a constant) with respect to $\boldsymbol{P}_r^\top\boldsymbol{P}_{\pi_{\boldsymbol{\Lambda}}}$. Since $\operatorname{Rank}(\boldsymbol{B}_{\boldsymbol{\Lambda}})=L-1$ and $\boldsymbol{B}_{\boldsymbol{\Lambda}}\mathbf{1}=\mathbf{0}$, we have $\boldsymbol{P}\boldsymbol{w}\in\operatorname{span}\{\boldsymbol{P}_m\boldsymbol{w}\}_{m=1}^{M_{\boldsymbol{\Lambda}}}$ for any permutation matrix $\boldsymbol{P}$. Thus, for any r, there exists exactly one $j\in[M_{\boldsymbol{\Lambda}}]$ such that $(\boldsymbol{B}_{\boldsymbol{\Lambda}}\boldsymbol{P}_{\pi_{\boldsymbol{\Lambda}}})_{r:}=\boldsymbol{w}^\top\boldsymbol{P}_r^\top\boldsymbol{P}_{\pi_{\boldsymbol{\Lambda}}}=c\boldsymbol{w}^\top\boldsymbol{P}_j^\top$ where c is either 1 or -1. Hence, the claim holds. This completes the proof of Item 2).

The proof of Item 3) is a direct consequence of Items 1) and 2). □

Proof of Proposition 2: From Item 1) of Theorem 4, we see that the corresponding permuted versions of $\boldsymbol{\Phi}_1$ and $\boldsymbol{\Psi}_j$ are

$\boldsymbol{\Phi}_1\boldsymbol{P}$ and $\boldsymbol{\Psi}_j\boldsymbol{P}$. Thus, $\boldsymbol{F}_j(\boldsymbol{PX}) = \boldsymbol{PF}_j(\boldsymbol{X})$, $\boldsymbol{F}_j(\boldsymbol{PAP}^\top \boldsymbol{PX}) = \boldsymbol{PF}_j(\boldsymbol{AX})$, $\boldsymbol{F}_j(\boldsymbol{P\tilde{A}P}^\top \boldsymbol{PX}) = \boldsymbol{PF}_j(\boldsymbol{\tilde{A}X})$ for $j = 0, \ldots, K-1$. It is obvious that the remaining channels also differ by a permutation matrix $\boldsymbol{P}$. Since the row normalization and the softmax function are applied row-wise and the activation function is applied element-wise, it is straightforward to see that $\hat{\boldsymbol{Y}}_{\boldsymbol{P}} = \boldsymbol{P}\hat{\boldsymbol{Y}}$. □

B. Fast Decomposition and Reconstruction Algorithms

Given a K-hierarchical clustering $\mathcal{P}_K$, we consider graph Haar framelet transform between V_{j+1} and $V_j \oplus W_j$. Define $x_{(\boldsymbol{\Lambda},\ell)} := \langle \boldsymbol{f}, \boldsymbol{\phi}_{(\boldsymbol{\Lambda},\ell)} \rangle$ and $y_{(\boldsymbol{\Lambda},m)} := \langle \boldsymbol{f}, \boldsymbol{\psi}_{(\boldsymbol{\Lambda},m)} \rangle$ for a given graph signal $\boldsymbol{f}$. The transform algorithm is to evaluate $x_{(\boldsymbol{\Lambda},\ell)}$ and $y_{(\boldsymbol{\Lambda},m)}$ effectively. Let $\boldsymbol{C}_{\boldsymbol{\Lambda}} \in \mathbb{R}^{L_{\boldsymbol{\Lambda}} \times (1+M_{\boldsymbol{\Lambda}})}$ be a matrix satisfying $\boldsymbol{C}_{\boldsymbol{\Lambda}} \begin{pmatrix} \boldsymbol{p}_{\boldsymbol{\Lambda}}^\top \\ \boldsymbol{B}_{\boldsymbol{\Lambda}} \end{pmatrix} = \boldsymbol{I} \in \mathbb{R}^{L_{\boldsymbol{\Lambda}} \times L_{\boldsymbol{\Lambda}}}$. Then, work [19, Lemma 1] and (1) and (2) imply that

$$\begin{bmatrix} \boldsymbol{\phi}_{\boldsymbol{\Lambda}}^\top \\ \boldsymbol{\psi}_{(\boldsymbol{\Lambda},1)}^\top \\ \vdots \\ \boldsymbol{\psi}_{(\boldsymbol{\Lambda},M_{\boldsymbol{\Lambda}})}^\top \end{bmatrix} := \begin{pmatrix} \boldsymbol{p}_{\boldsymbol{\Lambda}}^\top \\ \boldsymbol{B}_{\boldsymbol{\Lambda}} \end{pmatrix} \begin{bmatrix} \boldsymbol{\phi}_{(\boldsymbol{\Lambda},1)}^\top \\ \vdots \\ \boldsymbol{\phi}_{(\boldsymbol{\Lambda},L_{\boldsymbol{\Lambda}})}^\top \end{bmatrix} \tag{11}$$

$$\begin{bmatrix} \boldsymbol{\phi}_{(\boldsymbol{\Lambda},1)}^\top \\ \vdots \\ \boldsymbol{\phi}_{(\boldsymbol{\Lambda},L_{\boldsymbol{\Lambda}})}^\top \end{bmatrix} := \boldsymbol{C}_{\boldsymbol{\Lambda}} \begin{bmatrix} \boldsymbol{\phi}_{\boldsymbol{\Lambda}}^\top \\ \boldsymbol{\psi}_{(\boldsymbol{\Lambda},1)}^\top \\ \vdots \\ \boldsymbol{\psi}_{(\boldsymbol{\Lambda},M_{\boldsymbol{\Lambda}})}^\top \end{bmatrix}. \tag{12}$$

For the decomposition algorithm, we are given a signal $\boldsymbol{f} \in V_{j+1}$, which means that

$$\boldsymbol{f} = \sum_{\dim(\boldsymbol{\Lambda})=j} \sum_{\ell \in [L_{\boldsymbol{\Lambda}}]} x_{(\boldsymbol{\Lambda},\ell)} \boldsymbol{\phi}_{(\boldsymbol{\Lambda},\ell)}.$$

By (11), we have

$$\begin{aligned} \boldsymbol{f} &= \sum_{\dim(\boldsymbol{\Lambda})=j} \sum_{\ell \in [L_{\boldsymbol{\Lambda}}]} x_{(\boldsymbol{\Lambda},\ell)} \boldsymbol{\phi}_{(\boldsymbol{\Lambda},\ell)} \\ &= \sum_{\dim(\boldsymbol{\Lambda})=j} \sum_{\ell \in [L_{\boldsymbol{\Lambda}}]} x_{(\boldsymbol{\Lambda},\ell)} \left((C_{\boldsymbol{\Lambda}})_{\ell,1} \boldsymbol{\phi}_{\boldsymbol{\Lambda}} + \sum_{m \in [M_{\boldsymbol{\Lambda}}]} (C_{\boldsymbol{\Lambda}})_{\ell,m+1} \boldsymbol{\psi}_{(\boldsymbol{\Lambda},m)} \right) \\ &= \sum_{\dim(\boldsymbol{\Lambda})=j} \boldsymbol{\phi}_{\boldsymbol{\Lambda}} \sum_{\ell \in [L_{\boldsymbol{\Lambda}}]} x_{(\boldsymbol{\Lambda},\ell)} (C_{\boldsymbol{\Lambda}})_{\ell,1} \\ &\quad + \sum_{\dim(\boldsymbol{\Lambda})=j} \sum_{m \in [M_{\boldsymbol{\Lambda}}]} \boldsymbol{\psi}_{(\boldsymbol{\Lambda},m)} \sum_{\ell \in [L_{\boldsymbol{\Lambda}}]} x_{(\boldsymbol{\Lambda},\ell)} (C_{\boldsymbol{\Lambda}})_{\ell,m+1} \\ &= \sum_{\dim(\boldsymbol{\Lambda})=j} x_{\boldsymbol{\Lambda}} \boldsymbol{\phi}_{\boldsymbol{\Lambda}} + \sum_{\dim(\boldsymbol{\Lambda})=j} \sum_{m \in [M_{\boldsymbol{\Lambda}}]} y_{(\boldsymbol{\Lambda},m)} \boldsymbol{\psi}_{(\boldsymbol{\Lambda},m)} \end{aligned} \tag{13}$$

where we can represent the decomposition of $\boldsymbol{f}$ with respect to each $\boldsymbol{\Lambda}$ as

$$\begin{aligned} &\left[x_{\boldsymbol{\Lambda}},\ y_{(\boldsymbol{\Lambda},1)},\ \ldots,\ y_{(\boldsymbol{\Lambda},M_{\boldsymbol{\Lambda}})} \right] \\ &\quad = \left[x_{(\boldsymbol{\Lambda},1)},\ x_{(\boldsymbol{\Lambda},2)},\ \ldots,\ x_{(\boldsymbol{\Lambda},L_{\boldsymbol{\Lambda}})} \right] \boldsymbol{C}_{\boldsymbol{\Lambda}}. \end{aligned} \tag{14}$$

Conversely, if we have $\boldsymbol{f} \in V_j \oplus W_j$, which means that

$$\boldsymbol{f} = \sum_{\dim(\boldsymbol{\Lambda})=j} x_{\boldsymbol{\Lambda}} \boldsymbol{\phi}_{\boldsymbol{\Lambda}} + \sum_{\dim(\boldsymbol{\Lambda})=j} \sum_{m \in [M_{\boldsymbol{\Lambda}}]} y_{(\boldsymbol{\Lambda},m)} \boldsymbol{\psi}_{(\boldsymbol{\Lambda},m)}$$

then, by (11), for the reconstruction from $V_j \oplus W_j$ to V_{j+1}, we have

$$\begin{aligned} \boldsymbol{f} &= \sum_{\dim(\boldsymbol{\Lambda})=j} x_{\boldsymbol{\Lambda}} \boldsymbol{\phi}_{\boldsymbol{\Lambda}} + \sum_{\dim(\boldsymbol{\Lambda})=j} \sum_{m \in [M_{\boldsymbol{\Lambda}}]} y_{(\boldsymbol{\Lambda},m)} \boldsymbol{\psi}_{(\boldsymbol{\Lambda},m)} \\ &= \sum_{\dim(\boldsymbol{\Lambda})=j} \sum_{\ell \in [L_{\boldsymbol{\Lambda}}]} (\boldsymbol{r}_{\boldsymbol{\Lambda}})_\ell \boldsymbol{\phi}_{(\boldsymbol{\Lambda},\ell)} \\ &= \sum_{\dim(\boldsymbol{\Lambda})=j} \sum_{\ell \in [L_{\boldsymbol{\Lambda}}]} x_{(\boldsymbol{\Lambda},\ell)} \boldsymbol{\phi}_{(\boldsymbol{\Lambda},\ell)} \end{aligned} \tag{15}$$

where

$$\boldsymbol{r}_{\boldsymbol{\Lambda}} = x_{\boldsymbol{\Lambda}} \boldsymbol{p}_{\boldsymbol{\Lambda}}^\top + \boldsymbol{y}_{\boldsymbol{\Lambda}}^\top \boldsymbol{B}_{\boldsymbol{\Lambda}}, \text{ with } \boldsymbol{y}_{\boldsymbol{\Lambda}} = \left[y_{(\boldsymbol{\Lambda},1)}, \ldots, y_{(\boldsymbol{\Lambda},M_{\boldsymbol{\Lambda}})} \right]^\top \tag{16}$$

and $x_{(\boldsymbol{\Lambda},\ell)} := (r_{\boldsymbol{\Lambda}})_\ell$.

Algorithm 2 Fast Framelet Decomposition

Input: $\mathcal{P}_K$, $\{x_{\boldsymbol{\Lambda}} : \dim(\boldsymbol{\Lambda}) = j_0\}$, $\{\boldsymbol{C}_{\boldsymbol{\Lambda}}\}$, j_1
initialize $\hat{\boldsymbol{f}} = \emptyset$.
for $j = j_0 - 1$ **to** j_1 **do**
  for $\boldsymbol{\Lambda} \in \{\boldsymbol{\Lambda} : \dim(\boldsymbol{\Lambda}) = j\}$ **do**
    $[x_{\boldsymbol{\Lambda}}, y_{(\boldsymbol{\Lambda},1)}, \cdots, y_{(\boldsymbol{\Lambda},M_{\boldsymbol{\Lambda}})}] \leftarrow [x_{(\boldsymbol{\Lambda},1)}, x_{(\boldsymbol{\Lambda},2)}, \cdots, x_{(\boldsymbol{\Lambda},L_{\boldsymbol{\Lambda}})}]\boldsymbol{C}_{\boldsymbol{\Lambda}}$
    update $\hat{\boldsymbol{f}} \leftarrow \hat{\boldsymbol{f}} \cup \{x_{\boldsymbol{\Lambda}}, y_{(\boldsymbol{\Lambda},m)}, m = 1, \ldots, M_{\boldsymbol{\Lambda}}\}$
Output: $\mathcal{F}_{j_0}(\mathcal{P}_K)$

Algorithm 3 Fast Framelet Reconstruction

Input: $\mathcal{P}_K$, $\{x_{\boldsymbol{\Lambda}} : \dim(\boldsymbol{\Lambda}) = j_0\} \cup \{y_{(\boldsymbol{\Lambda},m)} : \dim(\boldsymbol{\Lambda}) = j_0, m \in [M_{\boldsymbol{\Lambda}}]\}$, $\{\boldsymbol{p}_{\boldsymbol{\Lambda}}, \boldsymbol{C}_{\boldsymbol{\Lambda}}\}$, j_1
initialize $\boldsymbol{f} = \emptyset$.
for $j = j_0 + 1$ **to** j_1 **do**
  for $\boldsymbol{\Lambda} \in \{\boldsymbol{\Lambda} : \dim(\boldsymbol{\Lambda}) = j\}$ **do**
    $\boldsymbol{r}_{\boldsymbol{\Lambda}} = x_{\boldsymbol{\Lambda}} \boldsymbol{p}_{\boldsymbol{\Lambda}}^\top + \boldsymbol{y}_{\boldsymbol{\Lambda}}^\top \boldsymbol{B}_{\boldsymbol{\Lambda}}$
    for $\ell = 1$ **to** $L_{\boldsymbol{\Lambda}}$ **do**
      $x_{(\boldsymbol{\Lambda},\ell)} \leftarrow (\boldsymbol{r}_{\boldsymbol{\Lambda}})_\ell$
    update $\boldsymbol{f} \leftarrow \boldsymbol{f} \cup \{x_{(\boldsymbol{\Lambda},\ell)}\}_{\ell \in [L_{\boldsymbol{\Lambda}}]}$
Output: $\boldsymbol{f}$

Hence, by using (14) iteratively from V_K, given a framelet system $\mathcal{F}_{j_0}(\mathcal{P}_K)$ and a graph signal $\boldsymbol{f}$, we get the coefficient vector $\hat{\boldsymbol{f}}$ consisting of coefficients from

$$\boldsymbol{f} \mapsto \{x_{\boldsymbol{\Lambda}} : \dim(\boldsymbol{\Lambda}) = j_0\} \cup \{y_{\boldsymbol{\Lambda}} : \dim(\boldsymbol{\Lambda}) = j\}_{j=j_0+1}^{K-1} \tag{17}$$

with respect to $V_{j_0} \oplus W_{j_0} \oplus \cdots \oplus W_{K-1}$. In the reconstruction process, we iteratively obtain the representation of $\boldsymbol{f}$ in V_K from coefficient vector $\hat{\boldsymbol{f}}$

$$\{x_{\boldsymbol{\Lambda}} : \dim(\boldsymbol{\Lambda}) = j_0\} \cup \{y_{\boldsymbol{\Lambda}} : \dim(\boldsymbol{\Lambda}) = j\}_{j=j_0+1}^{K-1} \mapsto \boldsymbol{f} \tag{18}$$

with respect to $V_{j_0} \oplus W_{j_0} \oplus \cdots \oplus W_{K-1}$.

From the above discussion, we observe that decomposition and reconstruction algorithms do not need to form the full framelet system explicitly, but $\boldsymbol{p}_{\boldsymbol{\Lambda}}$, $\boldsymbol{B}_{\boldsymbol{\Lambda}}$, and $\boldsymbol{C}_{\boldsymbol{\Lambda}}$, which implies efficiency in general applications that apply our framelet system.

Theorem 5: Under the same assumption as in Corollary 1, the decomposition algorithm to obtain the framelet coefficient vector $\hat{\boldsymbol{f}}$ from $\boldsymbol{f}$ and the reconstruction algorithm to obtain the graph signal $\boldsymbol{f}$ from $\hat{\boldsymbol{f}}$, as described in (17) and (18), are both with a computational complexity of order $O(nh)$.

Proof of Theorem 5: A fast decomposition algorithm is given by (13), which computes $\hat{\boldsymbol{f}}$ iteratively. In order to get $x_{\boldsymbol{\Lambda}}$ and $y_{(\boldsymbol{\Lambda},m)}$ for $\hat{\boldsymbol{f}}$, we need to compute $\sum_{\ell \in [L_{\boldsymbol{\Lambda}}]}$

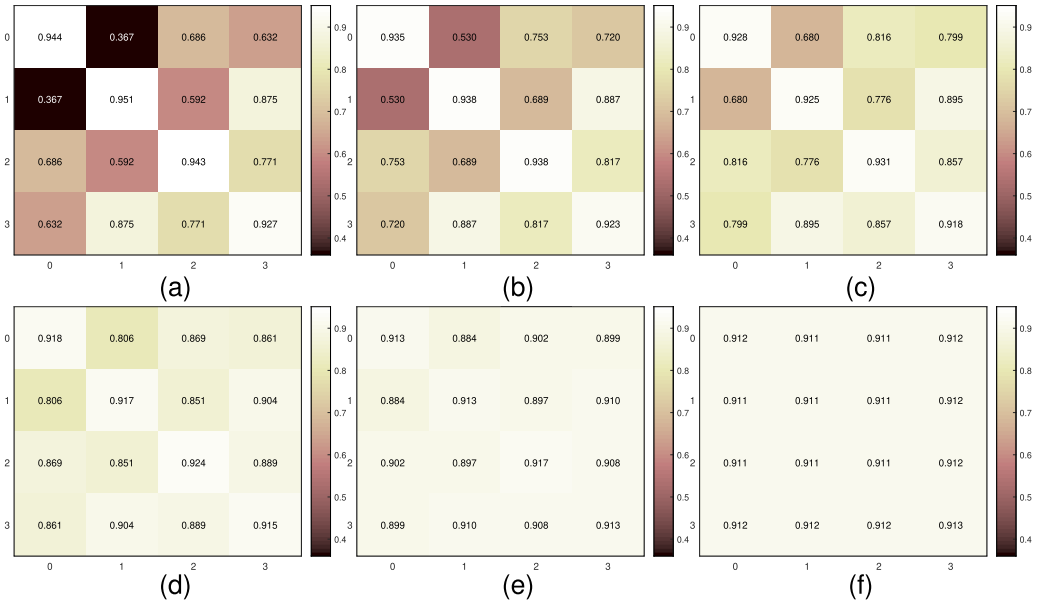

Fig. 6. CCNS on the synthetic graphs with different hyperparameters γ. (a) $\gamma = 0$. (b) $\gamma = 0.2$. (c) $\gamma = 0.4$. (d) $\gamma = 0.6$. (e) $\gamma = 0.8$. (f) $\gamma = 1$.

$x_{(\mathbf{\Lambda},\ell)}(\boldsymbol{C}_{\mathbf{\Lambda}})_{\ell,t} = (\boldsymbol{q}_{\mathbf{\Lambda}}^{\top}\boldsymbol{C}_{\mathbf{\Lambda}})_t$, where $t = 1, \ldots, M_{\mathbf{\Lambda}} + 1, \ell \in [L_{\mathbf{\Lambda}}]$ and $\boldsymbol{q}_{\mathbf{\Lambda}} := [x_{(\mathbf{\Lambda},1)}, \ldots, x_{(\mathbf{\Lambda},L_{\mathbf{\Lambda}})}]^{\top}$ [see (14)]. Note that for our binary graph Haar framelet system $\mathcal{F}_{j_0}(\mathcal{P}_K)$, the matrix $\boldsymbol{C}_{\mathbf{\Lambda}}$ in (11) is given by $\boldsymbol{C}_{\mathbf{\Lambda}} = [\boldsymbol{p}_{\mathbf{\Lambda}}, \boldsymbol{B}_{\mathbf{\Lambda}}^{\top}]$ and each row of $\boldsymbol{B}_{\mathbf{\Lambda}}$ has only two nonzero elements. Hence, for a given $\mathbf{\Lambda}$ with $\dim(\mathbf{\Lambda}) = j$, since $L_{\mathbf{\Lambda}} \le h$ and $M_{\mathbf{\Lambda}} \le (h(h-1)/2)$, the number of nonzero elements in $\boldsymbol{C}_{\mathbf{\Lambda}}$ is no more than $h + 2 \cdot (h(h-1)/2)$. Therefore, the computational complexity for obtaining $\boldsymbol{q}_{\mathbf{\Lambda}}^{\top}\boldsymbol{C}$ is of the same order as $h + 2 \cdot (h(h-1)/2)$. In total, observing that $\#\{\mathbf{\Lambda} : \dim(\mathbf{\Lambda}) = j\} \le h^{j-1}$, to get the full $\hat{\boldsymbol{f}}$, the computational complexity is of order the same as $\sum_{j=1}^{K-1}(h + 2 \cdot (h(h-1)/2))h^{j-1} = O(nh)$.

Fast reconstruction algorithm [see (15)], which computes $\boldsymbol{f}$ from $\hat{\boldsymbol{f}}$, only needs to compute $\boldsymbol{r}_{\mathbf{\Lambda}}$ iteratively. Let $\boldsymbol{Y}_{\mathbf{\Lambda}} := [x_{\mathbf{\Lambda}}, \boldsymbol{y}_{\mathbf{\Lambda}}^{\top}]^{\top} \in \mathbb{R}^{M_{\mathbf{\Lambda}}+1}$ and $\boldsymbol{P}_{\mathbf{\Lambda}} := \binom{p_{\mathbf{\Lambda}}^{\top}}{\boldsymbol{B}_{\mathbf{\Lambda}}}$. Then, $\boldsymbol{P}_{\mathbf{\Lambda}} = \boldsymbol{C}_{\mathbf{\Lambda}}^{\top}$ and $\boldsymbol{r}_{\mathbf{\Lambda}} = \boldsymbol{Y}_{\mathbf{\Lambda}}^{\top}\boldsymbol{P}_{\mathbf{\Lambda}}$. Following a similar calculation as for the fast decomposition algorithm, it is not hard to see that the computation complexity is of $\sum_{j=1}^{K-1} h^{j-1}(h + 2 \cdot (h(h-1)/2)) = O(nh)$. □

C. Experiment Details on the Synthetic Dataset

A theoretical characterization of graphs, given in [52], explains when GCN fails to produce acceptable performance. We follow and modify in [52, Algorithm 2] to generate synthetic data.

The key idea of the algorithm is to generate edges of nodes in a graph such that the intraclass and interclass similarities are properly controlled. The intraclass and interclass similarity are defined by Definition 1. Cross-class neighborhood similarity (CCNS) measures how close the patterns of connections of nodes between two classes are. In our experiment, we generate graphs with 3000 nodes for which assign labels from $\mathcal{C} = \{0, 1, 2, 3\}$ randomly. It means that we have $n_c = 4$ classes. We generate edges according to Algorithm 4. The distributions that control CCNS are designed based on the uniform distribution and a prescribed distribution $\{\mathcal{D}_c : c \in \mathcal{C}\}$. The distributions $\{\mathcal{D}_c : c \in \mathcal{C}\}$ can be found in Table V.

TABLE V
DISTRIBUTION D_4

$\{\mathcal{D}_c : c \in \mathcal{C}\}$	class 0	class 1	class 2	class 3
$\mathcal{D}_0$	0.1	0.4	0	0.5
$\mathcal{D}_1$	0.5	0	0.5	0
$\mathcal{D}_2$	0.2	0	0.5	0.3
$\mathcal{D}_3$	0.25	0.25	0.25	0.25

Integer N is set to be 45 000. The hyperparameter $\gamma \in \{0, 0.2, 0.4, 0.6, 0.8, 1\}$ indicates the probability of sampling neighbors from uniform distributions other than the predefined distributions that are much more distinguishable for different classes. When γ is small, vertices of the neighborhood are more likely to be sampled according to $\{\mathcal{D}_c : c \in \mathcal{C}\}$, and when γ is large, it is more likely to sample from the indistinguishable uniform distribution. As a result, when γ becomes larger, the uniform distribution has more impact on CCNS, and thus, the metric becomes more similar between classes. We evaluate CCNSs on the generated graphs and show heatmaps in Fig. 6. It is clear that, when $\gamma = 0$, CCNS is dominated by $\{\mathcal{D}_c : c \in \mathcal{C}\}$, and since $\mathcal{D}_0$ is more similar to $\mathcal{D}_2$ than $\mathcal{D}_1$, we get $s(0, 1) = 0.367 \le 0.686 = s(0, 2)$. Finally, when $\gamma = 1$, all CCNSs are almost 0.91. Notice that in Algorithm 4, we slightly modify the algorithm in [52]. Since we generate graphs with only nodes initialized, when $r \le \gamma$, we sample label c from all labels $\mathcal{C}$, instead of $\mathcal{C} - \{y_i\}$ used in [52]. Table VI shows the hyperparameter search range for the experiments of FSGNN and PEGFAN on the synthetic data, where $\{\mathrm{WD}_{\mathrm{sca}}, \mathrm{LR}_{\mathrm{sca}}, \mathrm{WD}_{\mathrm{fc1}}, \mathrm{WD}_{\mathrm{fc2}}, \mathrm{LR}_{\mathrm{fc}}\}$ are the weight decay coefficient of attention weights, the learning rate of attention weights, the weight decay coefficient of the first fully connected layer, weight decay coefficient of the second fully connected layer, and the learning rate of the fully connected part, respectively.

Definition 1 (Cross-Class Neighborhood Similarity [52]): Given graph $\mathcal{G}$ and node labels $y_i \in \{0, 1, \ldots, n_c - 1\}$ for

TABLE VI
HYPERPARAMETER SEARCH RANGE

Hyperparameters	Values
WD_{sca}	$0.0, 0.001, 0.1$
LR_{sca}	$0.01, 0.04$
WD_{fc1}	$0.0, 0.0001, 0.001$
WD_{fc2}	$0.0, 0.0001, 0.001$
LR_{fc}	$0.005, 0.01$

$i \in \mathcal{V}$, the metric between classes c and c' is $s(c, c') = (1/|\mathcal{V}_c||\mathcal{V}_{c'}|) \sum_{i\in\mathcal{V}_c, j\in\mathcal{V}_{c'}} \cos\langle d(i), d(j)\rangle$, where $\mathcal{V}_c := \{i \in \mathcal{V} : y_i = c\}$ and $d(i) \in \mathbb{R}^{n_c}$ is a vector with elements defined by $\#\{j : (i, j) \in \mathcal{E}, y_j = c\}$ for any $c \in \{0, 1, \ldots, n_c - 1\}$.

Features on graph nodes are from $\mathbb{R}^{700}$, with each element randomly generated according to Gaussian distribution $6(-0.75+0.5c)+\xi$ (Table I) or $(-0.75+0.5c)+\xi$ (Table II) independently, where $\xi \sim N(0, 1)$ and c is the label.

Algorithm 4 See [52]

Input: Nodes $\mathcal{V}$, Integer N, Distribution matrix $\{\mathcal{D}_c : c \in \mathcal{C}\}$, labels $\mathcal{C} = \{c\}_{i=0}^{n_c-1}$, γ
initialize $\mathcal{E} = \emptyset$ and $k = 0$;
while $k \leq N$ **do**
  Sample $i \in \mathcal{V}$ and $r \in [0, 1]$ uniformly
  Obtain the label $y_i \in \mathcal{C}$ of node i
  if $r \leq \gamma$ **then**
    Sample a label c from $\mathcal{C}$ uniformly
  else
    Sample a label c from $\mathcal{C}$ according to distribution $\mathcal{D}_{y_i}$
  Sample node j from class c uniformly
  if $(i, j) \notin \mathcal{E}$ **then**
    update $\mathcal{E} \leftarrow \mathcal{E} \cup (i, j)$
    update $k \leftarrow k + 1$
Output: $\mathcal{G} = (\mathcal{V}, \mathcal{E})$

REFERENCES

[1] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. ICLR*, 2017.

[2] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *Proc. KDD*, 2018, pp. 974–983.

[3] M. Zhang, Z. Cui, M. Neumann, and Y. Chen, "An end-to-end deep learning architecture for graph classification," in *Proc. AAAI*, 2018, pp. 4438–4445.

[4] Y. Zhou, H. Zheng, X. Huang, S. Hao, D. Li, and J. Zhao, "Graph neural networks: Taxonomy, advances, and trends," *ACM Trans. Intell. Syst. Technol.*, vol. 13, no. 1, pp. 1–54, Feb. 2022.

[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] J. Zhou et al., "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, Jan. 2020.

[7] 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.

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

[9] X. Zheng et al., "Graph neural networks for graphs with heterophily: A survey," 2022, *arXiv:2202.07082*.

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

[11] N. T. Hoang and T. Maehara, "Revisiting graph neural networks: All we have is low-pass filters," 2019, *arXiv:1905.09550*.

[12] J. Zhu, Y. Yan, L. Zhao, M. Heimann, L. Akoglu, and D. Koutra, "Beyond homophily in graph neural networks: Current limitations and effective designs," in *Proc. NeurIPS*, 2020, pp. 7793–7804.

[13] T. Konstantin Rusch, M. M. Bronstein, and S. Mishra, "A survey on oversmoothing in graph neural networks," 2023, *arXiv:2303.10993*.

[14] I. Daubechies, *Ten Lectures on Wavelets*. Philadelphia, PA, USA: SIAM, 1992.

[15] B. Han, *Framelets and Wavelets: Algorithms, Analysis, and Applications*. Cham, Switzerland: Springer, 2017.

[16] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Appl. Comput. Harmon. Anal.*, vol. 30, no. 2, pp. 129–150, Mar. 2011.

[17] C. K. Chui, F. Filbir, and H. N. Mhaskar, "Representation of functions on big data: Graphs and trees," *Appl. Comput. Harmon. Anal.*, vol. 38, no. 3, pp. 489–509, May 2015.

[18] B. Dong, "Sparse representation on graphs by tight wavelet frames and applications," *Appl. Comput. Harmon. Anal.*, vol. 42, no. 3, pp. 452–479, May 2017.

[19] J. Li, H. Feng, and X. Zhuang, "Convolutional neural networks for spherical signal processing via area-regular spherical Haar tight framelets," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, 2022, doi: 10.1109/TNNLS.2022.3160169.

[20] M. Li, Z. Ma, Y. G. Wang, and X. Zhuang, "Fast Haar transforms for graph neural networks," *Neural Netw.*, vol. 128, pp. 188–198, Aug. 2020.

[21] Y. G. Wang and X. Zhuang, "Tight framelets and fast framelet filter bank transforms on manifolds," *Appl. Comput. Harmon. Anal.*, vol. 48, no. 1, pp. 64–95, Jan. 2020.

[22] X. Zheng, B. Zhou, Y. G. Wang, and X. Zhuang, "Decimated framelet system on graphs and fast G-framelet transforms," *J. Mach. Learn. Res.*, vol. 23, no. 18, pp. 1–68, 2022.

[23] M. Gavish, B. Nadler, and R. R. Coifman, "Multiscale wavelets on trees, graphs and high dimensional data: Theory and applications to semi supervised learning," in *Proc. ICML*, 2010, pp. 367–374.

[24] C. K. Chui, H. Mhaskar, and X. Zhuang, "Representation of functions on big data associated with directed graphs," *Appl. Comput. Harmon. Anal.*, vol. 44, no. 1, pp. 165–188, 2018.

[25] Y. Xiao and X. Zhuang, "Adaptive directional Haar tight framelets on bounded domains for digraph signal representations," *J. Fourier Anal. Appl.*, vol. 27, no. 2, pp. 1–26, Apr. 2021.

[26] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. NIPS*, 2016.

[27] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. NIPS*, 2017.

[28] H. Pei, B. Wei, K. C. Chang, Y. Lei, and B. Yang, "Geom-GCN: Geometric graph convolutional networks," in *Proc. ICLR*, 2020.

[29] G. Zhao, T. Wang, Y. Li, Y. Jin, C. Lang, and S. Feng, "Neighborhood pattern is crucial for graph convolutional networks performing node classification," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, 2022, doi: 10.1109/TNNLS.2022.3229721.

[30] J. Chen, S. Chen, J. Gao, Z. Huang, J. Zhang, and J. Pu, "Exploiting neighbor effect: Conv-agnostic GNN framework for graphs with heterophily," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, 2023, doi: 10.1109/TNNLS.2023.3267902.

[31] L. Wu et al., "Beyond homophily and homogeneity assumption: Relation-based frequency adaptive graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, 2023, doi: 10.1109/TNNLS.2022.3230417.

[32] S. Suresh, V. Budde, J. Neville, P. Li, and J. Ma, "Breaking the limit of graph neural networks by improving the assortativity of graphs with local mixing patterns," in *Proc. KDD*, 2021, pp. 1541–1551.

[33] E. Chien, J. Peng, P. Li, and O. Milenkovic, "Adaptive universal generalized pagerank graph neural network," in *Proc. ICLR*, 2021.

[34] A. Arnaiz-Rodríguez, A. Begga, F. Escolano, and N. M. Oliver, "DiffWire: Inductive graph rewiring via the Lovász bound," in *Proc. 1st Learn. Graphs Conf.*, vol. 198, 2022, p. 15.

[35] J. Guo et al., "Homophily-oriented heterogeneous graph rewiring," in *Proc. WWW*, 2023, pp. 511–522.

[36] I. Spinelli, S. Scardapane, A. Hussain, and A. Uncini, "FairDrop: Biased edge dropout for enhancing fairness in graph representation learning," *IEEE Trans. Artif. Intell.*, vol. 3, no. 3, pp. 344–354, Jun. 2021.
[37] K. Xu, C. Li, Y. Tian, T. Sonobe, K. Kawarabayashi, and S. Jegelka, "Representation learning on graphs with jumping knowledge networks," in *Proc. ICML*, 2018, pp. 5453–5462.
[38] S. Abu-El-Haija et al., "MixHop: Higher-order graph convolutional architectures via sparsified neighborhood mixing," in *Proc. ICML*, 2019, pp. 21–29.
[39] S. K. Maurya, X. Liu, and T. Murata, "Simplifying approach to node classification in graph neural networks," *J. Comput. Sci.*, vol. 62, Jul. 2022, Art. no. 101695.
[40] D. Lim et al., "Large scale learning on non-homophilous graphs: New benchmarks and strong simple methods," in *Proc. NeurIPS*, 2021, pp. 20887–20902.
[41] M. Crovella and E. Kolaczyk, "Graph wavelets for spatial traffic analysis," in *Proc. 22nd Annu. Joint Conf. IEEE Comput. Commun. Societies*, Mar. 2003, pp. 1848–1857.
[42] B. Han, T. Li, and X. Zhuang, "Directional compactly supported box spline tight framelets with simple geometric structure," *Appl. Math. Lett.*, vol. 91, pp. 213–219, May 2019.
[43] B. Han and X. Zhuang, "Matrix extension with symmetry and its application to symmetric orthonormal multiwavelets," *SIAM J. Math. Anal.*, vol. 42, no. 5, pp. 2297–2317, 2010.
[44] X. Zhuang, "Matrix extension with symmetry and construction of biorthogonal multiwavelets with any integer dilation," *Appl. Comput. Harmon. Anal.*, vol. 33, no. 2, pp. 159–181, 2012.
[45] Q. Mo and X. Zhuang, "Matrix splitting with symmetry and dyadic framelet filter banks over algebraic number fields," *Linear Algebra Appl.*, vol. 437, no. 10, pp. 2650–2679, 2012.
[46] B. Han and X. Zhuang, "Algorithms for matrix extension and orthogonal wavelet filter banks over algebraic number fields," *Math. Comput.*, vol. 82, no. 281, pp. 459–490, 2013.
[47] P. Rodríguez, M. A. Bautista, J. González, and S. Escalera, "Beyond one-hot encoding: Lower dimensional target embedding," *Image Vis. Comput.*, vol. 75, pp. 21–31, Jul. 2018.
[48] H. Maron, H. Ben-Hamu, N. Shamir, and Y. Lipman, "Invariant and equivariant graph networks," in *Proc. ICLR*, 2019.
[49] C. Morris, G. Rattan, S. Kiefer, and S. Ravanbakhsh, "SpeqNets: Sparsity-aware permutation-equivariant graph networks," in *Proc. ICML*, 2022, pp. 16017–16042.
[50] R. Sato, "A survey on the expressive power of graph neural networks," 2020, *arXiv:2003.04078*.
[51] B. Zhang et al., "The expressive power of graph neural networks: A survey," 2023, *arXiv:2308.08235*.
[52] Y. Ma, X. Liu, N. Shah, and J. Tang, "Is homophily a necessity for graph neural networks?" in *Proc. ICLR*, 2022.
[53] M. Chen, Z. Wei, Z. Huang, B. Ding, and Y. Li, "Simple and deep graph convolutional networks," in *Proc. ICML*, 2020, pp. 1725–1735.

Jianfei Li received the B.S. degree in mathematics from the Ocean University of China, Qingdao, China, in 2017, and the M.S. degree in mathematics from Sun Yat-sen University, Guangzhou, China, in 2020. He is currently pursuing the Ph.D. degree with the Department of Mathematics, City University of Hong Kong, Hong Kong.

His research interests include approximation theory, signal processing, and deep neural networks.

Ruigang Zheng received the B.S. degree from the College of Mathematics and Statistics, Shenzhen University, Shenzhen, China, in 2016, and the M.S. degree in computational mathematics from the School of Mathematics, Sun Yat-sen University, Guangzhou, China, in 2020. He is currently pursuing the Ph.D. degree with the Department of Mathematics, City University of Hong Kong, Hong Kong.

His current research interests include graph-relevant machine learning problems and applied harmonic analysis.

Han Feng received the B.S. degree in mathematics from Beijing Normal University, Beijing, China, in 2011, and the M.Sc. and Ph.D. degrees in mathematics from the University of Alberta, Edmonton, AB, Canada, in 2013 and 2016, respectively.

She is currently an Assistant Professor at the Department of Mathematics, City University of Hong Kong, Hong Kong. Her research interests include spherical approximation theory, learning theory, and deep neural networks.

Ming Li (Member, IEEE) received the Ph.D. degree from the Department of Computer Science and Information Technology, La Trobe University, Bundoora, VIC, Australia, in 2017.

After the Ph.D. degree, he completed two post-doctoral fellowship positions at the Department of Mathematics and Statistics, La Trobe University, and the Department of Information Technology in Education, South China Normal University, Guangzhou, China. He is currently a "Shuang Long Scholar" Distinguished Professor with the Key Laboratory of Intelligent Education Technology and Application, Zhejiang Normal University, Jinhua, China. He has authored or coauthored the top-tier journals and conferences, including IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, *Artificial Intelligence*, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS (TNNLS), IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS ON ARTIFICIAL INTELLIGENCE, IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, IEEE TRANSACTIONS ON BIG DATA, *Information Fusion*, *Pattern Recognition*, *Neural Networks*, *Computers & Education*, NeurIPS, ICML, and IJCAI. His research interests include graph neural networks, graph learning, geometric deep learning, artificial intelligence (AI) for education, and educational data mining.

Dr. Li, as a leading Guest Editor, organized the Special Issue on Deep Neural Networks for Graphs: Theory, Models, Algorithms and Applications of IEEE TNNLS. He is an Associate Editor of *Neural Networks*, *Applied Intelligence*, *Alexandria Engineering Journal*, *Soft Computing*, and *Neural Processing Letters*.

Xiaosheng Zhuang received the bachelor's and master's degrees in mathematics from Sun Yat-sen University, Guangzhou, China, in 2003 and 2005, respectively, and the Ph.D. degree in applied mathematics from the University of Alberta, Edmonton, AB, Canada, in 2010.

He was a Post-Doctoral Fellow with Universität Osnabrück, Osnabrück, Germany, in 2011, and Technische Universität Berlin, Berlin, Germany, in 2012. He is currently an Associate Professor at the Department of Mathematics, City University of Hong Kong, Hong Kong. His research interests include applied and computational harmonic analysis, sparse approximation and directional multiscale representation systems, deep and machine learning, and signal/image processing.