Permutation Equivariant Framelet-based Hypergraph Neural Networks

Anonymous submission

Abstract

Hypergraphs provide a natural and expressive framework for modeling high-order relationships, enabling the representation of group-wise interactions beyond pairwise connections. While hypergraph neural networks (HNNs) have shown promise for learning on such structures, existing models often rely on shallow message passing and lack the ability to extract multiscale patterns. Framelet-based techniques offer a principled solution by decomposing signals into multiple frequency bands. However, most prior framelet systems, particularly Haar-type ones, are sensitive to node ordering and fail to ensure consistent representations under permutation, leading to instability in hypergraph learning. To address this, we propose Permutation Equivariant Framelet-based Hypergraph Neural Networks (PEF-HNN), a novel framework that integrates multiscale framelet analysis with permutation-consistent learning. We construct a new family of permutation equivariant Haar-type framelets specifically designed for hypergraphs, supported by theoretical analysis of their stability and decomposition properties. Built upon these framelets, PEF-HNN incorporates both low-pass and high-pass components across multiple scales into a unified neural architecture. Extensive experiments on nine benchmark datasets, including three homophilic and four heterophilic hypergraphs, as well as two real-world datasets for visual object classification, demonstrate the effectiveness of our approach, consistently outperforming existing HNN baselines and highlighting the advantages of permutation equivariant framelet design in hypergraph representation learning.

1 Introduction

Hypergraphs offer a natural way to model high-order relationships among entities by allowing hyperedges to connect arbitrary-sized subsets of nodes. This generalization of pairwise graphs makes hypergraphs especially suitable for representing group-wise interactions in real-world systems such as co-authorship networks, biological complexes, and multi-party communications (Antelmi et al. 2023). Recent works (Wang and Kleinberg 2024; Millán et al. 2025) have already demonstrated both theoretically and empirically the advantages of using hypergraphs directly, rather than simplifying them into pairwise graphs for specific problem formulations.

While hypergraph neural networks (HNNs) (Kim et al. 2024) have emerged as a promising tool for learning from such structures, most existing models rely on local or shallow message passing, which may be insufficient to capture the diverse and multiscale dependencies inherent in complex hypergraph data. To enhance representational power, recent works (Li et al. 2025a) have explored the integration of spectral techniques, such as wavelets (Hammond, Vandergheynst, and Gribonval 2011) and framelets (Dong 2017), into neural architectures. Framelets, in particular, enable multiscale decomposition of signals, allowing simultaneous extraction of smooth (low-frequency) and detailed (high-frequency) structural patterns (Zheng et al. 2021, 2022). However, a critical limitation persists: many existing framelet constructions, especially Haar-type framelets, are sensitive to node ordering. Since their bases often depend on hierarchical structures built from a specific node sequence, reordering the nodes while preserving the hypergraph structure can lead to inconsistencies in the generated framelet representations. This lack of permutation equivariance undermines the stability and reproducibility of learned features, especially in hypergraph settings where the structural complexity magnifies sensitivity to input permutations. While recent advances have proposed permutation equivariant framelets for standard graphs (Li et al. 2024), the extension to hypergraphs remains unexplored.

To address this gap, we propose a novel framework, Permutation Equivariant Framelet-based Hypergraph Neural Networks (termed PEF-HNN), which combines the strengths of multiscale framelet analysis with permutation-consistent representation learning on hypergraphs. At the core of our framework lies a new design of Haar-type framelets that are permutation equivariant and specifically tailored for hypergraph structures, constructed based on hierarchical decompositions of node groups derived from hyperedge incidence. These framelets ensure that the transformation of node signals remains consistent under arbitrary reordering. Built on this foundation, PEF-HNN incorporates multi-scale framelet transforms as input channels within a unified neural architecture, enabling effective capture of both low-pass and high-pass components across multiple structural resolutions. We evaluate PEF-HNN on hypergraph node classification tasks using nine benchmark datasets, including three homophilic and four heterophilic hypergraphs, as well as two real-world datasets for visual object classification. Extensive comparisons against existing HNN baselines demonstrate the compelling performance of our model, highlighting the benefits of constructing permu-

tation equivariant framelets for enhancing hypergraph representation learning.

In summary, our contributions are three-fold:

- **Theoretical Result**: We construct a new family of permutation equivariant Haar-type framelets on hypergraphs and provide theoretical analysis of their key properties, enabling stable and consistent multiscale decomposition under arbitrary node reorderings;
- **Model Development**: We propose PEF-HNN, a novel framework that integrates permutation equivariant framelet-based multiscale representations into hypergraph neural networks for effective hypergraph learning;
- **Experimental Study**: We validate the effectiveness of PEF-HNN through extensive experiments on nine benchmark datasets, including five homophilic and four heterophilic hypergraphs, as well as two real-world datasets for visual object classification. The results consistently show performance gains over existing HNN baselines, underscoring the benefits of incorporating permutation equivariant framelet design in hypergraph learning.

2 Related Work

Hypergraph neural networks (HNNs) have emerged as powerful tools for modeling high-order relational structures, where each hyperedge can connect an arbitrary subset of nodes. Classical models such as HGNN (Feng et al. 2019) and HyperGCN (Yadati et al. 2019) extend traditional graph neural networks (GNNs) by using incidence matrix-based propagation or clique expansion techniques. However, these methods primarily rely on shallow message passing and are constrained by their inability to effectively capture multiscale or high-frequency information inherent in complex hypergraphs. To improve learning capacity, UniGCNII (Huang and Yang 2021) introduces deep architectures with identity and residual connections to mitigate oversmoothing, while ED-HNN (Wang et al. 2023) incorporates edge-dependent transformations for more flexible and adaptive aggregation. Set-based approaches such as AllDeepSets and AllSetTransformer (Chien et al. 2022) treat each hyperedge as a set and model intra-hyperedge interactions through permutation-invariant functions, offering a principled way to handle unordered node sets and dynamic hyperedge cardinality. Nevertheless, these models largely ignore the global hierarchical structure of hypergraphs and often lack a spectral or frequency-aware perspective.

To overcome the limitations of local aggregation, recent efforts have explored spectral methods for hypergraph learning. For example, Li et al. (2025a,b) propose framelet-based HNNs that construct multiscale representations using hypergraph Laplacian eigenbases. These spectral framelets enable the decomposition of node features into low- and high-frequency components, allowing the model to capture both smooth and oscillatory patterns. However, such Laplacian-induced framelet transforms are sensitive to node orderings and are typically designed in a global and rigid fashion, without accounting for the hierarchical or local multiscale structure within the hypergraph. As a result, they lack permutation consistency, meaning that reordering the input nodes can yield inconsistent feature representations and unstable learning outcomes. PEGFAN (Li et al. 2024) introduces Haar-type framelets with explicit permutation equivariance for standard graph neural networks. By constructing hierarchical trees over graph nodes and defining framelets in a spatially localized manner, PEGFAN achieves consistent multiscale decomposition regardless of node ordering. This property is crucial for learning robust representations and ensuring stability across varying graph structures. However, the extension of permutation equivariant framelets to hypergraphs has not been systematically studied. Hypergraphs present additional challenges due to their higher-order connectivity and more complex combinatorial structures, which require novel design principles beyond those developed for pairwise graphs.

Motivated by these observations, our work develops a new family of permutation equivariant Haar-type framelets specifically designed for hypergraphs. By leveraging hyperedge-induced hierarchical groupings, we construct localized framelet transforms that preserve equivariance under node permutations and support multiscale signal extraction. These framelets are integrated into a unified HNN framework, enabling robust and expressive hypergraph representation learning across both homophilic and heterophilic structures.

3 Proposed Method

3.1 Notation and Preliminaries

A hypergraph is represented as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, comprising a vertex set $\mathcal{V}$ of size $N = |\mathcal{V}|$, a hyperedge set $\mathcal{E}$ of size $M = |\mathcal{E}|$. Suppose that vertices and hyperedges have feature dimensions d and m, respectively, we have the representation of vertex data as $\boldsymbol{X} \in \mathbb{R}^{N\times d}$ and hyperedge data as $\boldsymbol{Y} \in \mathbb{R}^{M\times o}$.

The hypergraph structure, from a vertex perspective, is defined by an incidence matrix $\boldsymbol{H} \in \{0,1\}^{N\times M}$ where $\boldsymbol{H}(v,e) = 1$ if vertex v is contained in hyperedge e, and 0 otherwise, as represented by:

$$\boldsymbol{H}(v,e) = \begin{cases} 1, & \text{if } v \in e; \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

The degrees of vertex v and hyperedge e are denoted by diagonal matrices $\boldsymbol{D}_v \in \mathbb{R}^{N\times N}$ and $\boldsymbol{D}_e \in \mathbb{R}^{M\times M}$, calculated as $\sum_{e\in\mathcal{E}} \boldsymbol{H}(v,e)$ and $\sum_{v\in\mathcal{V}} \boldsymbol{H}(v,e)$, respectively.

3.2 Construction of Permutation Equivariant Framelets on Hypergraphs

A hypergraph signal $\boldsymbol{f} = [f_1, \ldots, f_N]^\top \in \mathbb{R}^N$ is considered as $\boldsymbol{f} : \mathcal{V} = \{1, 2, \ldots, N\} \to \mathbb{R}$ with ℓ_2 norm $\|\boldsymbol{f}\|^2 = \sum_{i=1}^N |f_i|^2 < \infty$. All such signals form a Hilbert space $L_2(\mathcal{G})$ under the usual inner product $\langle \boldsymbol{f}, \boldsymbol{g}\rangle := \boldsymbol{f}^\top \boldsymbol{g}$ for $\boldsymbol{f}, \boldsymbol{g} \in L_2(\mathcal{G})$. A collection $\{\boldsymbol{e}_i : i \in [I]\} \subset L_2(\mathcal{G})$ is a *tight frame* of $L_2(\mathcal{G})$ if $\boldsymbol{f} = \sum_{i=1}^I \langle \boldsymbol{f}, \boldsymbol{e}_i\rangle \boldsymbol{e}_i$ for all $\boldsymbol{f} \in L_2(\mathcal{G})$, where we denote $[I] := \{1, \ldots, I\}$. We denote the i-th 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 $s_{\boldsymbol{\Lambda}}$, and $\dim(\boldsymbol{\Lambda})$ is 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 i-th 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*.

Given $\mathcal{P}_K$ and any $j_0 \in [K]$, we next define a sequence of framelet systems

$$\begin{aligned}\mathcal{F}_{j_0}(\mathcal{P}_K) :=& \{\boldsymbol{\phi}_{\boldsymbol{\Lambda}} : \dim(\boldsymbol{\Lambda}) = j_0\} \\ &\cup \{\boldsymbol{\psi}_{\boldsymbol{\Lambda}} : \dim(\boldsymbol{\Lambda}) = j\}_{j=j_0+1}^{K}\end{aligned} \tag{2}$$

of scaling vectors $\boldsymbol{\phi}_{\boldsymbol{\Lambda}}$ and framelet vectors $\boldsymbol{\psi}_{(\boldsymbol{\Lambda},m)}$ in $L_2(\mathcal{G})$. For the scaling vectors $\boldsymbol{\phi}_{\boldsymbol{\Lambda}}$, they are defined iteratively from $\dim(\boldsymbol{\Lambda}) = K$ to $\dim(\boldsymbol{\Lambda}) = 1$. When $\dim(\boldsymbol{\Lambda}) = K$, each cluster (node) $s_{\boldsymbol{\Lambda}}$ contains only one vertex in hypergraph $\mathcal{G}$, and we define $\boldsymbol{\phi}_{\boldsymbol{\Lambda}} = \boldsymbol{I}_{:i}$, where $i \in s_{\boldsymbol{\Lambda}} \subset \mathcal{V}$ and $\boldsymbol{I}_{:i}$ is the i-th column of the identity matrix $\boldsymbol{I} \in \mathbb{R}^{N\times N}$. When $\dim(\boldsymbol{\Lambda}) < K$, we define

$$\boldsymbol{\phi}_{\boldsymbol{\Lambda}} := \sum_{\ell \in [L_{\boldsymbol{\Lambda}}]} p_{(\boldsymbol{\Lambda},\ell)} \boldsymbol{\phi}_{(\boldsymbol{\Lambda},\ell)}, \tag{3}$$

where $p_{\boldsymbol{\Lambda},\ell} \equiv \frac{1}{\sqrt{L_{\boldsymbol{\Lambda}}}}$. Obviously, $\boldsymbol{\phi}_{\boldsymbol{\Lambda}}$ is with support $\mathrm{supp}\boldsymbol{\phi}_{\boldsymbol{\Lambda}} = s_{\boldsymbol{\Lambda}}$ and $\|\boldsymbol{\phi}_{\boldsymbol{\Lambda}}\| = 1$. For the framelet vectors, we define $\boldsymbol{\psi}_{(\boldsymbol{\Lambda},i)}$, $i \in [I_{\boldsymbol{\Lambda}}]$ with $I_{\boldsymbol{\Lambda}} := \frac{L_{\boldsymbol{\Lambda}}(L_{\boldsymbol{\Lambda}}-1)}{2}$ by

$$\boldsymbol{\psi}_{(\boldsymbol{\Lambda},i)} := \sum_{\ell \in [L_{\boldsymbol{\Lambda}}]} (\boldsymbol{B}_{\boldsymbol{\Lambda}})_{i,\ell} \, \boldsymbol{\phi}_{(\boldsymbol{\Lambda},\ell)}, \tag{4}$$

where the matrices $\boldsymbol{B}_{\boldsymbol{\Lambda}} \in \mathbb{R}^{I_{\boldsymbol{\Lambda}} \times L_{\boldsymbol{\Lambda}}}$ are defined row-by-row with its i-th row $[\boldsymbol{B}_{\boldsymbol{\Lambda}}]_{i:} = [w_1, \ldots, w_{L_{\boldsymbol{\Lambda}}}]$ being given by

$$w_\tau = \begin{cases} \frac{1}{\sqrt{L_{\boldsymbol{\Lambda}}}} & \tau = \ell_1; \\ \frac{-1}{\sqrt{L_{\boldsymbol{\Lambda}}}} & \tau = \ell_2; \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

Here $1 \le \ell_1 < \ell_2 \le L_{\boldsymbol{\Lambda}}$ is uniquely determined by $i = i(\ell_1, \ell_2, L_{\boldsymbol{\Lambda}}) = \frac{(2L_{\boldsymbol{\Lambda}} - \ell_1)(\ell_1 - 1)}{2} + \ell_2 - \ell_1$. In short, the row of $\boldsymbol{B}_{\boldsymbol{\Lambda}}$ is obtained by permutating the row vector $\frac{1}{\sqrt{L_{\boldsymbol{\Lambda}}}}[1, -1, 0, \ldots, 0]$.

3.3 Theoretical Properties

Theorem 1 shows that $\mathcal{F}_{j_0}(\mathcal{P}_K)$ is a tight frame.

Theorem 1. *Let $\mathcal{F}_{j_0}(\mathcal{P}_K)$ be defined as by* (2)*. Then $\mathcal{F}_{j_0}(\mathcal{P}_K)$ is a tight frame of $L_2(\mathcal{G})$ for any $j_0 \in [K]$.*

Proof. We prove the result by induction on j_0. Obviously, for $j_0 = K$, $\mathcal{F}_K(\mathcal{P}_K) = \{\boldsymbol{I}_{:i}\}_{i=1}^N$ is simply the orthonormal basis and thus a tight frame. Suppose for $j_0 = k+1$, $\mathcal{F}_{k+1}(\mathcal{P}_K)$ is tight. We need to show that $\mathcal{F}_k(\mathcal{P}_K)$ is tight. That is, for all $\boldsymbol{f} \in L_2(\mathcal{G})$, we have

$$\boldsymbol{f} = \sum_{\dim(\boldsymbol{\Lambda})=k} \langle \boldsymbol{f}, \boldsymbol{\phi}_{\boldsymbol{\Lambda}} \rangle \boldsymbol{\phi}_{\boldsymbol{\Lambda}} + \sum_{j=k+1}^{K} \sum_{\dim(\boldsymbol{\Lambda})=j} \langle \boldsymbol{f}, \boldsymbol{\psi}_{\boldsymbol{\Lambda}} \rangle \boldsymbol{\psi}_{\boldsymbol{\Lambda}}.$$

By the induction hypothesis, we only need to show

$$\begin{aligned}&\sum_{\dim(\boldsymbol{\Lambda})=k} \langle \boldsymbol{f}, \boldsymbol{\phi}_{\boldsymbol{\Lambda}} \rangle \boldsymbol{\phi}_{\boldsymbol{\Lambda}} + \sum_{\dim((\boldsymbol{\Lambda},i))=k+1} \langle \boldsymbol{f}, \boldsymbol{\psi}_{(\boldsymbol{\Lambda},i)} \rangle \boldsymbol{\psi}_{(\boldsymbol{\Lambda},i)} \\ &= \sum_{\dim(\boldsymbol{\Lambda})=k+1} \langle \boldsymbol{f}, \boldsymbol{\phi}_{\boldsymbol{\Lambda}} \rangle \boldsymbol{\phi}_{\boldsymbol{\Lambda}}.\end{aligned}$$

By definition of $\boldsymbol{\phi}_{\boldsymbol{\Lambda}}$ and $\boldsymbol{\psi}_{(\boldsymbol{\Lambda},i)}$ in (3) and (4), it is equivalent to that for each $\boldsymbol{\Lambda}$ with $\dim(\boldsymbol{\Lambda}) = k$, it holds

$$\boldsymbol{p}_{\boldsymbol{\Lambda}} \boldsymbol{p}_{\boldsymbol{\Lambda}}^\top + \boldsymbol{B}_{\boldsymbol{\Lambda}}^\top \boldsymbol{B}_{\boldsymbol{\Lambda}} = \boldsymbol{I},$$

where $\boldsymbol{p}_{\boldsymbol{\Lambda}} := [p_{(\boldsymbol{\Lambda},1)}, \ldots, p_{(\boldsymbol{\Lambda},L_{\boldsymbol{\Lambda}})}]^\top \equiv \frac{1}{\sqrt{L_{\boldsymbol{\Lambda}}}}\mathbf{1}$ is the constant vector and $\boldsymbol{B}_{\boldsymbol{\Lambda}}$ is the matrix defined in (5), which is true thanks to the structures of $\boldsymbol{p}_{\boldsymbol{\Lambda}}$ and $\boldsymbol{B}_{\boldsymbol{\Lambda}}$. Therefore, $\mathcal{F}_k(\mathcal{P}_K)$ is tight. □

We have the following remarks concerning the properties of $\mathcal{F}_{j_0}(\mathcal{P}_K)$.

Remark 1. *Let $\mathcal{F}_{j_0}(\mathcal{P}_K) := \{\boldsymbol{\phi}_{\boldsymbol{\Lambda}}, \dim(\boldsymbol{\Lambda}) = j_0\} \cup \{\boldsymbol{\psi}_{\boldsymbol{\Lambda}}, \dim(\boldsymbol{\Lambda}) = j\}_{j=j_0+1}^K =: \{\boldsymbol{u}_i\}_{i=1}^{I_{\mathcal{G}}}$ with $I_{\mathcal{G}}$ the total number of vectors in $\mathcal{F}_{j_0}(\mathcal{P}_K)$. We denote $\mathcal{F} := [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{I_{\mathcal{G}}}]^\top \in \mathbb{R}^{I_{\mathcal{G}} \times N}$ to be its matrix representation. Then, by the tight frame property, we have $\mathcal{F}^\top \mathcal{F} = \boldsymbol{I} \in \mathbb{R}^{N\times N}$.*

Remark 2. *If for all $\boldsymbol{\Lambda}$, we have $L_{\boldsymbol{\Lambda}} \le h$ for some integer $h \ge 2$, then $K = O(\log_h N)$. One can show that the total number $I_{\mathcal{G}}$ is of order $I_{\mathcal{G}} = O(Nh)$ and hence the total number $\mathrm{nnz}(\mathcal{F})$ of nonzero entries of $\mathcal{F}$ is of order $\mathrm{nnz}(\mathcal{F}) = O(Nh \log_h N)$. In practice, h is usually small (e.g., 2, 4, or 8), and hence $\mathcal{F}$ is a sparse matrix. Thus, $\mathcal{F}$ can be stored as a sparse matrix and the framelet coefficient vectors $\hat{\boldsymbol{f}} := \mathcal{F}\boldsymbol{f}$ can be computed efficiently with the computational complexity of order $O(Nh \log_h N)$.*

Let $\pi : \mathcal{V} \to \mathcal{V}$ be a reordering (relabeling, bijection) of $\mathcal{V} = \{1, 2, \ldots, N\}$, i.e., π is w.r.t. a *permutation* on $[N]$ with $\pi(V) = \{\pi(1), \ldots, \pi(N)\}$. We denote $\pi(\mathcal{G}) = (\pi(V), \pi(\mathcal{E}))$ with $\pi(\mathcal{E}) := \{\pi(e) : e \in \mathcal{E}\}$ being given by $\pi(e) := \{\pi(v) : v \in e\}$. The corresponding signal $\boldsymbol{f}$ on $\mathcal{G}$ is reordered to be $\pi(\boldsymbol{f})$ under the newly ordered $\pi(\mathcal{G})$. In other words, given a π, there exists a permutation matrix $\boldsymbol{P}_\pi$ of size $N \times N$ such that $\pi(\boldsymbol{f}) = \boldsymbol{P}_\pi \boldsymbol{f}$. Fix $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and $\mathcal{P}_K$. Denote our construction of $\mathcal{F} = \mathcal{F}_{j_0}(\mathcal{P}_K)$ by

$$\mathcal{A}(\mathcal{G}, \mathcal{P}_K) = \mathcal{F}.$$

For each permutation π, the construction $\mathcal{A}$ is called *permutation equivariant* if

$$\mathcal{A}\left(\pi(\mathcal{G}), \mathcal{P}_K\right) = \pi\left(\mathcal{A}(\mathcal{G}, \mathcal{P}_K)\right),$$

where $\pi(\mathcal{F}) := \mathcal{F}\boldsymbol{P}_\pi$. We have the following result concerning the permutation equivariant property of our construction of $\mathcal{F}_{j_0}(\mathcal{P}_K)$ with respect to the reordering of $\mathcal{V}$.

Theorem 2. *For any permutation π, we have $\mathcal{A}(\pi(\mathcal{G}), \mathcal{P}_K) = \pi(\mathcal{A}(\mathcal{G}, \mathcal{P}_K))$.*

Proof. Let $\boldsymbol{\Phi}_{\boldsymbol{\Lambda}} := [\boldsymbol{\phi}_{(\boldsymbol{\Lambda},1)}, \ldots, \boldsymbol{\phi}_{(\boldsymbol{\Lambda},L_{\boldsymbol{\Lambda}})}]^\top$ and $\boldsymbol{\Psi}_{\boldsymbol{\Lambda}} := [\boldsymbol{\psi}_{(\boldsymbol{\Lambda},1)}, \ldots, \boldsymbol{\psi}_{(\boldsymbol{\Lambda},I_{\boldsymbol{\Lambda}})}]^\top$. 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 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 equivariant properties for each $\boldsymbol{\Lambda}$. Note that $\boldsymbol{\phi}_{(\boldsymbol{\Lambda},\ell)} : \mathcal{V} \to \mathbb{R}$ only depends on $\mathcal{G}$, $\mathcal{P}_K$, and $\boldsymbol{p}_{\boldsymbol{\Lambda}} = \frac{1}{\sqrt{L_{\boldsymbol{\Lambda}}}}\mathbf{1}$. For any permutation π, the $\mathcal{P}_K$ is determined by the index vectors $\boldsymbol{\Lambda}$ according to a tree structure and is independent of the permutation π. Moreover, the vectors $\boldsymbol{p}_{\boldsymbol{\Lambda}}$ are fixed constants. Hence, iteratively, after the permutation π acting on $\mathcal{G}$, 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 hypergraph $\pi(\mathcal{G})$ 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. □

Remark 3. *Theorem 2 shows that framelet system $\mathcal{F} = \mathcal{F}_{j_0}(\mathcal{P}_K)$ is permutation equivariant when reordering node indices. We call $\mathcal{F}$ a* Permutation Equivariant Framelet (PEF) *system.*

3.4 Hypergraph Neural Networks with PEF

After constructing the permutation equivariant framelet (PEF) system (see the above section), we can design Hypergraph Neural Networks with PEF, termed as PEF-HNN, as follows:

$$\begin{aligned} \boldsymbol{X}^{(\ell+1)} =& \sigma\Bigg(\Big((1-\alpha_\ell)\mathcal{F}^\top \mathrm{diag}(\theta)\mathcal{F}\boldsymbol{X}^{(\ell)} + \alpha_\ell \boldsymbol{X}^{(0)}\Big) \\ & \cdot \Big((1-\beta_\ell)\boldsymbol{I} + \beta_\ell \boldsymbol{\Theta}^{(\ell)}\Big)\Bigg), \end{aligned} \tag{6}$$

where $\theta \in \mathbb{R}^{I_{\mathcal{G}}}$ is the learnable filter, $\mathcal{F} \in \mathbb{R}^{I_{\mathcal{G}} \times N}$ denotes framelet matrix representation of our PEF system.

Theorem 3. *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a hypergraph with feature matrix $\boldsymbol{X}^{(0)}$ and a K-hierarchical partition $\mathcal{P}_K$. Let $\boldsymbol{P}$ be a permutation matrix w.r.t. to a permutation π on $\mathcal{V}$. If the permuted feature matrix $\boldsymbol{P}\boldsymbol{X}^{(0)}$ and framelet system $\pi(\mathcal{A}(\mathcal{G}, \mathcal{P}_K))$ are used in the PEF-HNN network* (6)*, then the new output $\boldsymbol{X}_{\boldsymbol{P}}^{(\ell+1)}$ of each layer differs from the original one by a permutation matrix, i.e. $\boldsymbol{X}_{\boldsymbol{P}}^{(\ell+1)} = \boldsymbol{P}\boldsymbol{X}^{(\ell+1)}$.*

Proof. Let $\mathcal{F} := \mathcal{A}(\mathcal{G}, \mathcal{P}_K)$. Then we have $\mathcal{F}_\pi := \pi(\mathcal{A}(\mathcal{G}, \mathcal{P}_K)) = \mathcal{F}\boldsymbol{P}$. Thus, (by induction on ℓ), we have

$$\begin{aligned} \boldsymbol{X}_{\boldsymbol{P}}^{(\ell+1)} &= \sigma\Bigg(\Big((1-\alpha_\ell)\mathcal{F}_\pi^\top \mathrm{diag}(\theta)\mathcal{F}_\pi \boldsymbol{P}\boldsymbol{X}^{(\ell)} + \alpha_\ell \boldsymbol{P}\boldsymbol{X}^{(0)}\Big) \\ &\quad \cdot \Big((1-\beta_\ell)\boldsymbol{I} + \beta_\ell \boldsymbol{\Theta}^{(\ell)}\Big)\Bigg) \\ &= \sigma\Bigg(\Big((1-\alpha_\ell)\boldsymbol{P}\mathcal{F}^\top \mathrm{diag}(\theta)\mathcal{F}\boldsymbol{X}^{(\ell)} + \alpha_\ell \boldsymbol{P}\boldsymbol{X}^{(0)}\Big) \\ &\quad \cdot \Big((1-\beta_\ell)\boldsymbol{I} + \beta_\ell \boldsymbol{\Theta}^{(\ell)}\Big)\Bigg) \\ &= \boldsymbol{P}\mathbf{X}^{(\ell+1)}. \end{aligned}$$

4 Experiments

4.1 Datasets

We evaluate the performance of PEF-HNN across a diverse collection of hypergraph datasets spanning various domains. Specifically, we use seven publicly available real-world hypergraph datasets: Cora, Citeseer, and Cora-CA (covering both cocitation and coauthorship networks)(Yadati et al. 2019), Actor and Twitch(Li et al. 2025b), as well as Senate and House (Fowler 2006; Chodrow, Veldt, and Benson 2021). These datasets are categorized based on the hyperedge homophily ratio $\mathcal{H}_{\text{edge}}$(Li et al. 2025b), where datasets with $\mathcal{H}_{\text{edge}} > 0.5$ are labeled homophilic, and those with $\mathcal{H}_{\text{edge}} \leq 0.5$ are labeled heterophilic. In addition, we include two real-world datasets for visual object classification: the 3D NTU2012 dataset(Chen et al. 2003) and the Princeton ModelNet40 dataset (Wu et al. 2015), which evaluate the model's capability in geometric understanding and 3D shape classification. Full descriptions of all datasets can be found in the **Appendix**.

4.2 Baselines and Implementation Details.

We adopt different data splits for training, validation, and testing according to standard practices. For Actor and Twitch, we follow the 40%/20%/40% split introduced in (Wang et al. 2023), while the remaining datasets use a 50%/25%/25% split following (Li et al. 2025b). PEF-HNN is evaluated against two groups of baseline methods: (i) general-purpose hypergraph neural networks, including HGNN (Feng et al. 2019), HyperGCN (Yadati et al. 2019), UniGCNII (Huang and Yang 2021), AllDeepSets, and AllSetTransformer (Chien et al. 2022); and (ii) heterophily-aware HNNs, specifically ED-HNN (Wang et al. 2023) and HyperUFG (Li et al. 2025b). Additional experimental settings as well as the code link are provided in the **Appendix**.

All experiments are implemented in PyTorch and conducted on a single NVIDIA RTX A6000 GPU with 48GB memory. We use the Adam optimizer (Kingma and Ba 2014) and perform grid search to tune hyperparameters. The learning rate is selected from $\{5\text{e}{-3}, 3\text{e}{-3}, 2\text{e}{-3}, 1\text{e}{-3}, 5\text{e}{-2}, 3\text{e}{-2}, 2\text{e}{-2}, 1\text{e}{-2}\}$; weight decay from $\{5\text{e}{-5}, 1\text{e}{-5}, 5\text{e}{-4}, 1\text{e}{-4}, 5\text{e}{-3}, 1\text{e}{-3}\}$; hidden dimensions from $\{32, 64, 128, 256, 512, 1024\}$; and the number of layers from the range $[1, 128]$. For baseline methods, we report published results when available. Otherwise, we reproduce results using the original implementations provided by the authors.

4.3 Results and Discussion

We conduct a thorough evaluation of PEF-HNN on a diverse set of hypergraphs that vary in their degree of homophily. The analysis includes comparisons with both conventional hypergraph neural networks (HNNs) and recent models specifically designed for heterophilic structures. Table 1 reports the classification accuracy (%) on three representative homophilic datasets: Cora, Citeseer, and Cora-CA, and four heterophilic datasets: Actor, Twitch, Senate, and House.

Table 1: Performance comparison between PEF-HNN and baselines on classification accuracy (%) across three homophilic and four heterophilic hypergraphs. The best results are shown in **bold**, and the second-best are underlined.

Methods **Hom. ratio,** $\mathcal{H}_{edge}$	**Cora** 0.7462	**Citeseer** 0.6814	**Cora-CA** 0.7797	**Actor** 0.4675	**Twitch** 0.4857	**Senate** 0.4642	**House** 0.4851
HGNN	79.39 ± 1.36	72.45 ± 1.16	82.64 ± 1.65	74.47 ± 0.32	51.88 ± 0.26	48.59 ± 4.52	61.39 ± 2.96
HyperGCN	78.45 ± 1.26	71.28 ± 0.82	79.48 ± 2.08	68.67 ± 4.38	51.32 ± 1.02	42.45 ± 3.67	48.32 ± 2.93
UniGCNII	78.81 ± 1.05	73.05 ± 2.21	83.60 ± 1.14	80.48 ± 1.13	50.84 ± 0.76	49.30 ± 4.25	67.25 ± 2.57
AllDeepSets	76.88 ± 1.80	70.83 ± 1.63	81.97 ± 1.50	82.00 ± 2.33	50.72 ± 0.96	52.82 ± 3.20	51.70 ± 3.37
AllSetTransformer	78.58 ± 1.47	73.08 ± 1.20	83.63 ± 1.47	83.39 ± 1.73	50.45 ± 0.76	51.83 ± 5.22	69.33 ± 2.20
ED-HNN	$\underline{80.31 \pm 1.35}$	73.70 ± 1.38	83.97 ± 1.55	$\mathbf{91.86 \pm 0.43}$	50.86 ± 0.88	64.79 ± 5.14	72.45 ± 2.28
HyperUFG	$\mathbf{81.51 \pm 0.99}$	$\underline{74.72 \pm 2.10}$	$\underline{85.18 \pm 0.69}$	89.32 ± 0.75	52.35 ± 0.04	67.61 ± 7.00	72.82 ± 2.22
PEF-HNN (Ours)	$\mathbf{81.51 \pm 0.98}$	$\mathbf{74.96 \pm 1.77}$	$\mathbf{86.00 \pm 0.71}$	$\underline{90.27 \pm 2.13}$	$\mathbf{53.18 \pm 0.37}$	$\mathbf{68.45 \pm 6.84}$	$\mathbf{73.25 \pm 1.55}$

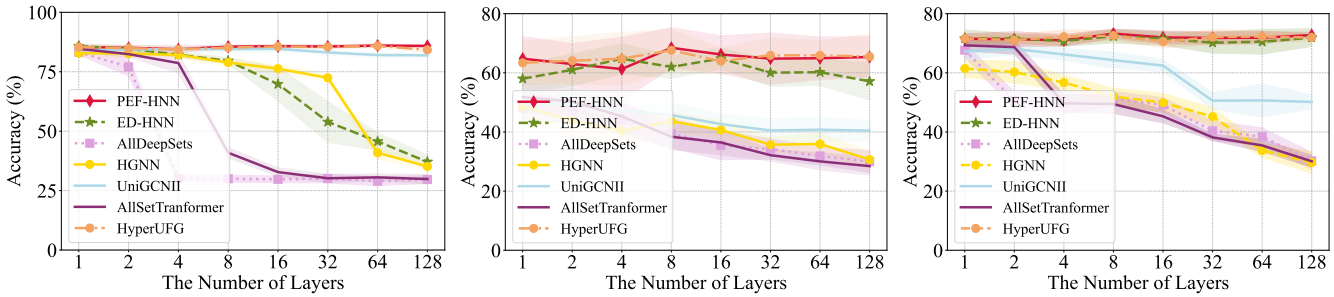

Figure 1: Comparison of the models' ability to alleviate oversmoothing: Cora-CA (left), Senate (middle), House (right).

From the results, we observe that PEF-HNN consistently achieves competitive or superior performance across all datasets. On the homophilic datasets, PEF-HNN either matches or outperforms the best baselines. Notably, it achieves the highest accuracy on Cora-CA (86.00%) and matches the top-performing model on Cora (81.51%), demonstrating its ability to capture low-frequency patterns typical of homophilic structures. On the heterophilic datasets, where high-frequency components are more prominent, PEF-HNN shows strong performance, achieving the best results on Twitch (53.18%), Senate (68.45%), and House (73.25%), while remaining competitive on Actor.

Compared to general-purpose HNNs such as HGNN, HyperGCN, and UniGCNII, PEF-HNN demonstrates more stable performance across both homophilic and heterophilic regimes. Furthermore, while ED-HNN and HyperUFG are tailored for heterophilic hypergraphs, PEF-HNN surpasses them on three of the four heterophilic datasets. These results highlight the advantage of integrating permutation equivariant framelets, which allow PEF-HNN to flexibly capture both coarse and fine-grained structural signals without sensitivity to node ordering. Overall, these findings confirm the effectiveness and generalizability of the proposed PEF-HNN framework, which demonstrates robust performance across varying hypergraph structures and homophily levels.

4.4 Ability to Alleviate Oversmoothing

We also study empirically the ability of PEF-HNN to alleviate oversmoothing as the number of layers increases. Figure 1 presents accuracy trends on three representative datasets: Cora-CA (left), Senate (middle), and House (right) as the number of layers increases from 1 to 128. We compare PEF-HNN with HGNN, UniGCNII, AllDeepSets, AllSetTransformer, ED-HNN, and HyperUFG.

Among the baselines, UniGCNII benefits from residual connections and maintains moderate performance at deeper layers on Cora-CA. However, its accuracy drops significantly beyond two layers on the heterophilic Senate and House datasets. ED-HNN performs competitively on Senate and House at shallower depths but exhibits sharp degradation and higher variance after 8 layers on Cora-CA. HyperUFG remains relatively stable across all three datasets, yet its performance does not consistently improve with depth. In contrast, PEF-HNN demonstrates strong resistance to oversmoothing and consistently benefits from increased depth. On all three datasets, its accuracy either improves or remains steady as the number of layers increases, with notably lower variance compared to other methods. This indicates that the integration of permutation equivariant framelet transforms enables effective multi-scale representation learning while preserving discriminative features even in very deep architectures. These results demonstrate that PEF-HNN effectively avoids oversmoothing on both homophilic and heterophilic hypergraphs, even at large depths.

4.5 Parameter Sensitivity Analysis

We investigate the sensitivity of PEF-HNN to two key hyperparameters, α and β, which influence the architectural dynamics of the model. As illustrated in Figure 2, we conduct univariate experiments by varying each parameter independently in the range $\{0.1, 0.2, \ldots, 0.9\}$ on three representative datasets: Cora, Senate, and House. The parameter

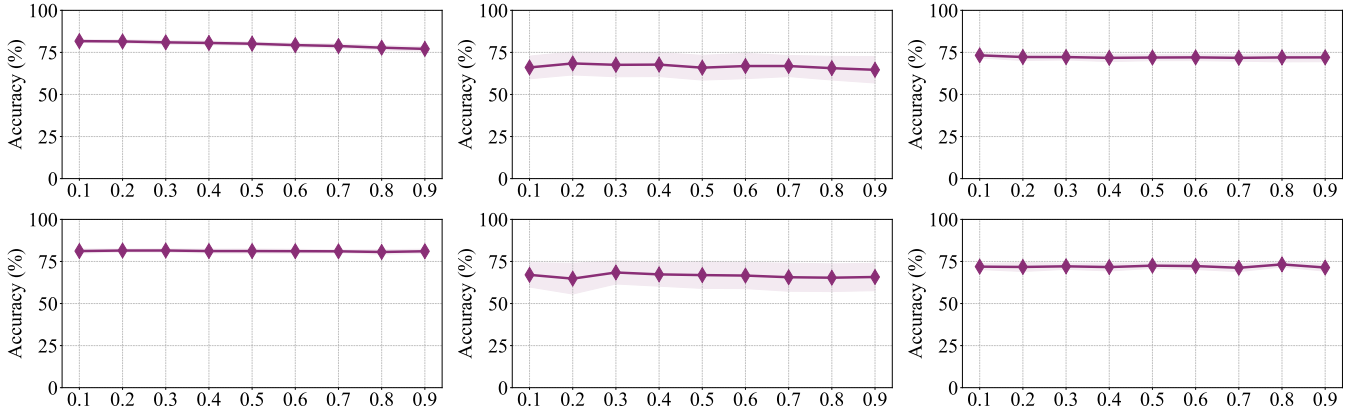

Figure 2: Sensitivity analysis of α (top row) and β (bottom row) on Cora (left), Senate (middle), and House (right) datasets.

Table 2: Ablation study evaluating the impact of low-pass and high-pass components in PEF-HNN.

	Cora	**Citeseer**	**Cora-CA**	**Actor**	**Twitch**	**Senate**	**House**
Full model	81.51 ± 0.98	74.96 ± 1.77	86.00 ± 0.71	90.27 ± 2.13	53.18 ± 0.37	68.45 ± 6.84	73.25 ± 1.55
w/o high pass	81.39 ± 1.17	75.06 ± 2.01	85.58 ± 1.00	88.71 ± 2.76	51.32 ± 0.89	67.32 ± 6.53	71.98 ± 1.76
w/o low pass	81.45 ± 0.99	74.81 ± 2.00	85.44 ± 0.72	89.12 ± 2.55	52.63 ± 1.04	66.34 ± 7.38	72.26 ± 1.83

α controls the trade-off between the current node features and their initial representations, whereas β modulates the relative importance of the initial features in each layer via a decaying schedule. Experimental results show that PEF-HNN maintains stable performance across a broad range of values for both parameters. In particular, model accuracy is moderately affected by α, indicating its importance in preserving useful initial information. In contrast, the model is less sensitive to β, likely due to its diminishing influence over deeper layers. Overall, these results suggest that PEF-HNN is robust to the choice of α and β, and does not require fine-grained tuning to achieve strong performance.

4.6 Ablation Study

We conduct an ablation study to evaluate the individual contributions of the low-pass and high-pass components in PEF-HNN. As shown in Table 2, removing either component leads to performance degradation across most datasets, confirming the effectiveness of the full multi-scale design. On heterophilic datasets (Actor, Twitch, Senate, and House), the absence of high-pass signals results in notable accuracy drops, underscoring their role in capturing fine-grained, discriminative features. On homophilic datasets (Cora, Citeseer, and Cora-CA), low-pass filtering is essential, while high-pass components still provide complementary benefits by capturing local variations. Overall, both low-pass and high-pass components are critical for achieving robust performance across hypergraphs with varying degrees of homophily.

4.7 Visualization

To qualitatively assess the representation quality, we visualize the original input features and the aggregated deep representations learned by various HNNs on a homophilic dataset (Citeseer) and a heterophilic dataset (House), as shown in Figure 3. The original features exhibit unclear or overlapping class distributions. On Citeseer, most methods produce reasonably separable clusters, though HGNN and UniGCNII show evident class mixing. ED-HNN and PEF-HNN yield cleaner separation, with PEF-HNN producing the most compact and distinct clusters. On the heterophilic House dataset, HGNN fails to distinguish class boundaries, and UniGCNII and ED-HNN provide partial improvements. In contrast, PEF-HNN forms clearly separated and well-structured clusters, demonstrating its superior ability to capture discriminative features in both homophilic and heterophilic settings.

4.8 Visual Object Classification

To further assess the generalization capability of PEF-HNN beyond graph benchmarks, we apply it to the task of 3D visual object classification. Specifically, we conduct experiments on two publicly available real-world datasets: ModelNet40 (Wu et al. 2015), a large-scale CAD model benchmark widely used in shape classification, and NTU2012 (Chen et al. 2003), a smaller but challenging dataset consisting of geometric 3D models with high intra-class variability.

Hypergraph Construction. We construct both single-view and multi-view hypergraphs based on shape features extracted from two well-established deep architectures: Multi-view Convolutional Neural Network (MVCNN)(Su et al. 2015) and Group-View Convolutional Neural Network (GVCNN)(Feng et al. 2018). In the single-view setting, each 3D object is embedded into either the MVCNN or GVCNN feature space, and a hypergraph is constructed using the k-nearest neighbor (k-NN) approach: each object is treated as a centroid and connected to its k nearest neighbors to form one hyperedge. This procedure results in N hyperedges for N objects, where the structure reflects local similarities in the selected feature space. For the multi-view set-

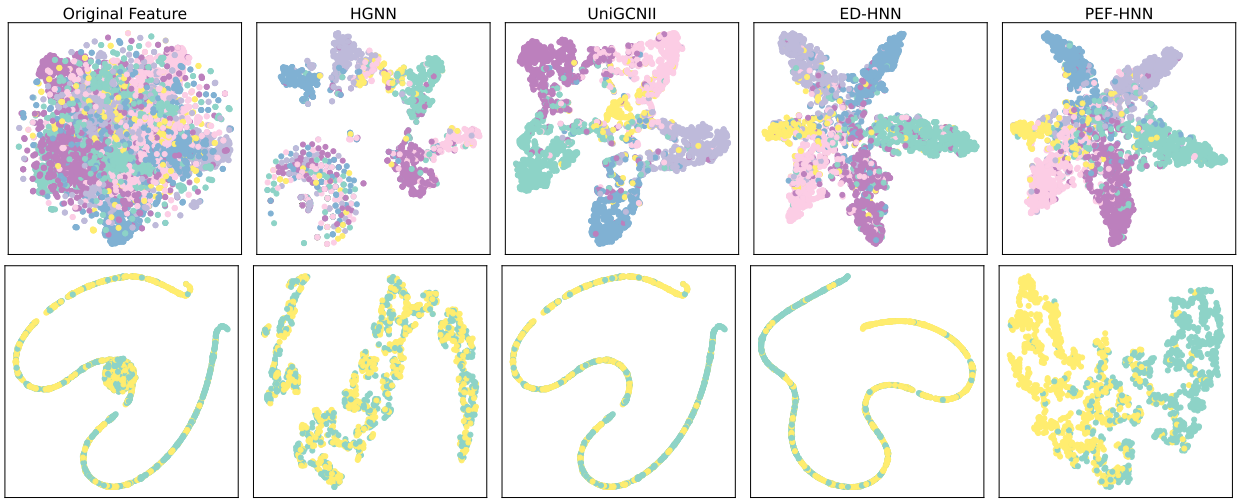

Figure 3: Visualization comparison of raw input features and learned representations on Citeseer (top) and House (bottom).

Table 3: Comparison between GCN, HGNN and PEF-HNN on ModelNet40 (top) and NTU2012 (bottom).

Feature	GCN	HGNN	PEF-HNN
GVCNN	91.8%	92.6%	97.2%
MVCNN	86.7%	91.0%	92.1%
GVCNN&MVCNN	94.4%	96.7%	**98.4%**
Feature	**GCN**	**HGNN**	**PEF-HNN**
GVCNN	78.8%	82.5%	**93.3%**
MVCNN	71.3%	75.6%	89.1%
GVCNN&MVCNN	76.1%	84.2%	91.4%

Table 4: Performance comparison of state-of-the-art classification methods on ModelNet40.

Methods	Accuracy
PointNet (Qi et al. 2017a)	89.2%
PointNet++ (Qi et al. 2017b)	90.7%
PointCNN (Li et al. 2018)	91.8%
SO-Net (Li, Chen, and Lee 2018)	93.4%
Point-UMAE (Zeng et al. 2025)	94.2%
HGNN (Feng et al. 2019)	96.7%
ED-HNN (Wang et al. 2023)	97.8%
PEF-HNN	**98.4%**

ting, we independently perform k-NN hyperedge construction in both feature spaces and then concatenate the resulting hyperedge incidence matrices. This fused representation integrates complementary information from both MVCNN and GVCNN, thereby enriching the relational structure used in PEF-HNN. Full implementation details and hyperparameter choices are provided in the **Appendix**.

Performance Comparison. As shown in Tables 3 and 4, PEF-HNN consistently achieves superior performance across both ModelNet40 and NTU2012 benchmarks. On ModelNet40, it attains the highest classification accuracy of 98.4%, outperforming strong point-based models such as PointCNN (91.8%) and Point-UMAE (94.2%), as well as existing hypergraph methods like HGNN and ED-HNN. This highlights the effectiveness of our permutation equivariant framelet design in capturing structured geometric information. Furthermore, PEF-HNN demonstrates robust performance across different feature extraction settings: when using single-view embeddings (MVCNN or GVCNN), it consistently surpasses both GCN and HGNN on both datasets. Notably, the multi-view fusion setting (combining GVCNN and MVCNN) further boosts accuracy, allowing PEF-HNN to leverage complementary shape features for more discriminative representations. These results collectively confirm that PEF-HNN is well-suited for visual object classification and generalizes effectively to complex, real-world hypergraph structures.

5 Conclusion

This paper presents PEF-HNN, a novel hypergraph neural network framework that leverages permutation equivariant framelets for multiscale hypergraph representation learning. By constructing a new class of Haar-type framelets specifically tailored for hypergraph structures and ensuring consistency under arbitrary node reorderings, our method addresses a critical limitation of existing framelet-based approaches. The integration of these framelets into a unified architecture enables the model to capture both global and local structural patterns, enhancing its adaptability to diverse hypergraph characteristics. Experimental results across seven benchmark datasets and two real-world visual object classification datasets confirm the effectiveness of PEF-HNN in both homophilic and heterophilic hypergraph settings. For the future work, it is promising to explore extensions of this framework to dynamic or temporal hypergraphs, where structural evolution adds another layer of complexity. Moreover, combining permutation equivariant framelets with other types of high-order signal processing techniques may further improve model generalization and interpretability in broader learning tasks.

References

Antelmi, A.; Cordasco, G.; Polato, M.; Scarano, V.; Spagnuolo, C.; and Yang, D. 2023. A survey on hypergraph representation learning. *ACM Computing Surveys*, 56(1): 1–38.

Chen, D.-Y.; Tian, X.-P.; Shen, Y.-T.; and Ouhyoung, M. 2003. On visual similarity based 3D model retrieval. *Computer Graphics Forum*, 22(3): 223–232.

Chien, E.; Pan, C.; Peng, J.; and Milenkovic, O. 2022. You are AllSet: A multiset function framework for hypergraph neural networks. In *ICLR*.

Chodrow, P. S.; Veldt, N.; and Benson, A. R. 2021. Generative hypergraph clustering: From blockmodels to modularity. *Science Advances*, 7(28): eabh1303.

Dong, B. 2017. Sparse representation on graphs by tight wavelet frames and applications. *Applied and Computational Harmonic Analysis*, 42(3): 452–479.

Feng, Y.; You, H.; Zhang, Z.; Ji, R.; and Gao, Y. 2019. Hypergraph neural networks. In *AAAI*, 3558–3565.

Feng, Y.; Zhang, Z.; Zhao, X.; Ji, R.; and Gao, Y. 2018. GVCNN: Group-view convolutional neural networks for 3D shape recognition. In *CVPR*, 264–272.

Fowler, J. H. 2006. Legislative cosponsorship networks in the US House and Senate. *Social Networks*, 28(4): 454–465.

Hammond, D. K.; Vandergheynst, P.; and Gribonval, R. 2011. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2): 129–150.

Huang, J.; and Yang, J. 2021. UniGNN: A unified framework for graph and hypergraph neural networks. In *IJCAI*, 2563–2569.

Kim, S.; Lee, S. Y.; Gao, Y.; Antelmi, A.; Polato, M.; and Shin, K. 2024. A survey on hypergraph neural networks: An in-depth and step-by-step guide. In *KDD*, 6534–6544.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Li, J.; Chen, B. M.; and Lee, G. H. 2018. SO-Net: Self-organizing network for point cloud analysis. In *CVPR*, 9397–9406.

Li, J.; Zheng, R.; Feng, H.; Li, M.; and Zhuang, X. 2024. Permutation equivariant graph framelets for heterophilous graph learning. *IEEE Transactions on Neural Networks and Learning Systems*, 35(9): 11634–11648.

Li, M.; Fang, Y.; Wang, Y.; Feng, H.; Gu, Y.; Bai, L.; and Lio, P. 2025a. Deep hypergraph neural networks with tight framelets. In *AAAI*, 18385–18392.

Li, M.; Gu, Y.; Wang, Y.; Fang, Y.; Bai, L.; Zhuang, X.; and Lio, P. 2025b. When hypergraph meets heterophily: New benchmark datasets and baseline. In *AAAI*, 18377–18384.

Li, Y.; Bu, R.; Sun, M.; Wu, W.; Di, X.; and Chen, B. 2018. PointCNN: Convolution on X-transformed points. In *NeurIPS*, 652–660.

Millán, A. P.; Sun, H.; Giambagli, L.; Muolo, R.; Carletti, T.; Torres, J. J.; Radicchi, F.; Kurths, J.; and Bianconi, G. 2025. Topology shapes dynamics of higher-order networks. *Nature Physics*, 21: 353—361.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017a. PointNet: Deep learning on point sets for 3D classification and segmentation. In *CVPR*.

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J. 2017b. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In *NIPS*, 5105–5114.

Su, H.; Maji, S.; Kalogerakis, E.; and Learned-Miller, E. 2015. Multi-view convolutional neural networks for 3D shape recognition. In *ICCV*, 945–953.

Wang, P.; Yang, S.; Liu, Y.; Wang, Z.; and Li, P. 2023. Equivariant hypergraph diffusion neural operators. In *ICLR*.

Wang, Y.; and Kleinberg, J. 2024. From Graphs to Hypergraphs: Hypergraph Projection and its Reconstruction. In *ICLR*.

Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; and Xiao, J. 2015. 3D ShapeNets: A deep representation for volumetric shapes. In *CVPR*, 1912–1920.

Yadati, N.; Nimishakavi, M.; Yadav, P.; Nitin, V.; Louis, A.; and Talukdar, P. 2019. HyperGCN: A new method for training graph convolutional networks on hypergraphs. In *NeurIPS*, 1511–1522.

Zeng, H.; Zhang, P.; Li, F.; Ye, T.; Wang, J.; and Yang, X. 2025. Point-UMAE: Unet-like masked autoencoders for point cloud self-supervised learning. In *ICASSP*.

Zheng, X.; Zhou, B.; Gao, J.; Wang, Y. G.; Lió, P.; Li, M.; and Montúfar, G. 2021. How framelets enhance graph neural networks. In *ICML*, 12761–12771.

Zheng, X.; Zhou, B.; Wang, Y. G.; and Zhuang, X. 2022. Decimated framelet system on graphs and fast G-framelet transforms. *Journal of Machine Learning Research*, 23(18): 1–68.

Reproducibility Checklist

1. General Paper Structure

1.1. Includes a conceptual outline and/or pseudocode description of AI methods introduced (yes/partial/no/NA) yes

1.2. Clearly delineates statements that are opinions, hypothesis, and speculation from objective facts and results (yes/no) yes

1.3. Provides well-marked pedagogical references for less-familiar readers to gain background necessary to replicate the paper (yes/no) yes

2. Theoretical Contributions

2.1. Does this paper make theoretical contributions? (yes/no) yes

If yes, please address the following points:

2.2. All assumptions and restrictions are stated clearly and formally (yes/partial/no) yes

2.3. All novel claims are stated formally (e.g., in theorem statements) (yes/partial/no) yes

2.4. Proofs of all novel claims are included (yes/partial/no) yes

2.5. Proof sketches or intuitions are given for complex and/or novel results (yes/partial/no) yes

2.6. Appropriate citations to theoretical tools used are given (yes/partial/no) yes

2.7. All theoretical claims are demonstrated empirically to hold (yes/partial/no/NA) yes

2.8. All experimental code used to eliminate or disprove claims is included (yes/no/NA) yes

3. Dataset Usage

3.1. Does this paper rely on one or more datasets? (yes/no) yes

If yes, please address the following points:

3.2. A motivation is given for why the experiments are conducted on the selected datasets (yes/partial/no/NA) yes

3.3. All novel datasets introduced in this paper are included in a data appendix (yes/partial/no/NA) yes

3.4. All novel datasets introduced in this paper will be made publicly available upon publication of the paper with a license that allows free usage for research purposes (yes/partial/no/NA) yes

3.5. All datasets drawn from the existing literature (potentially including authors' own previously published work) are accompanied by appropriate citations (yes/no/NA) yes

3.6. All datasets drawn from the existing literature (potentially including authors' own previously published work) are publicly available (yes/partial/no/NA) yes

3.7. All datasets that are not publicly available are described in detail, with explanation why publicly available alternatives are not scientifically satisficing (yes/partial/no/NA) yes

4. Computational Experiments

4.1. Does this paper include computational experiments? (yes/no) yes

If yes, please address the following points:

4.2. This paper states the number and range of values tried per (hyper-) parameter during development of the paper, along with the criterion used for selecting the final parameter setting (yes/partial/no/NA) partial

4.3. Any code required for pre-processing data is included in the appendix (yes/partial/no) yes

4.4. All source code required for conducting and analyzing the experiments is included in a code appendix (yes/partial/no) partial

4.5. All source code required for conducting and analyzing the experiments will be made publicly available upon publication of the paper with a license that allows free usage for research purposes (yes/partial/no) yes

4.6. All source code implementing new methods have comments detailing the implementation, with references to the paper where each step comes from (yes/partial/no) yes

4.7. If an algorithm depends on randomness, then the method used for setting seeds is described in a way sufficient to allow replication of results (yes/partial/no/NA) yes

4.8. This paper specifies the computing infrastructure used for running experiments (hardware and software), including GPU/CPU models; amount of memory; operating system; names and versions of relevant software libraries and frameworks (yes/partial/no) yes

4.9. This paper formally describes evaluation metrics used and explains the motivation for choosing these metrics (yes/partial/no) yes

4.10. This paper states the number of algorithm runs used to compute each reported result (yes/no) yes

4.11. Analysis of experiments goes beyond single-dimensional summaries of performance (e.g., average; median) to include measures of variation, confidence, or other distributional information (yes/no) yes

4.12. The significance of any improvement or decrease in performance is judged using appropriate statistical tests (e.g., Wilcoxon signed-rank) (yes/partial/no) partial

4.13. This paper lists all final (hyper-)parameters used for each model/algorithm in the paper's experiments (yes/partial/no/NA) NA