Implicit Hypergraph Neural Networks: A Stable Framework for Higher-Order Relational Learning with Provable Guarantees

Xiaoyu Li[*]
University of New South Wales
xiaoyu.li2@student.unsw.edu.au

Guangyu Tang[*]
University of New South Wales
tang_guangyu@126.com

Jiaojiao Jiang[†]
University of New South Wales
jiaojiao.jiang@unsw.edu.au

Abstract

Many real-world interactions are group-based rather than pairwise, e.g., papers with multiple co-authors or users jointly engaging with items. Hypergraph neural networks (HGNNs) capture such higher-order relations, but fixed-depth message passing can miss long-range dependencies and destabilize training as depth grows. We introduce **Implicit Hypergraph Neural Network** (**IHGNN**), bringing the implicit equilibrium formulation to hypergraphs: instead of stacking layers, IHGNN computes representations as the solution to a nonlinear fixed-point equation, enabling stable, efficient global propagation across hyperedges without deep architectures. We develop a well-posed training scheme with provable convergence, characterize conditions for oversmoothing and the model's expressivity, and derive a transductive generalization bound on hypergraphs. Training uses an implicit-gradient method coupled with a projection-based stabilizer. On citation benchmarks, IHGNN consistently outperforms strong graph and hypergraph baselines in both accuracy and robustness, and is notably resilient to random initialization and hyperparameter variation—highlighting strong generalization and practical value for higher-order relational learning.

1 Introduction

Graph neural networks (GNNs) have emerged as a powerful paradigm for learning from graph-structured data, where nodes represent entities and edges capture their pairwise relationships [35, 63, 62]. However, many real-world scenarios involve complex, higher-order interactions that cannot be fully captured by simple pairwise connections. For example, in a coauthorship network [36], a single paper often involves more than two authors. Such relationships are more naturally represented using a hypergraph, in which each vertex corresponds to an author and each hyperedge connects all authors of the same paper. By explicitly modeling these multi-way correlations, hypergraphs can capture intricate interdependencies among multiple entities simultaneously, providing a richer and more faithful representation of complex relationships present in real-world data. Hypergraph neural networks (HGNNs) naturally generalize GNNs to learn from hypergraph-structured data [15, 67, 30, 19]. By

[*]Equal contribution.
[†]Corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: New Perspectives in Graph Machine Learning.

extending the capabilities of GNNs, HGNNs can flexibly model and analyze complex, higher-order relationships that arise in many domains [55, 24, 46, 37, 10, 4, 14].

Despite their expressive power, conventional HGNNs still rely on explicit message passing across stacked layers, which is inherently limited in capturing long-range dependencies. As depth increases, training becomes prone to vanishing or exploding gradients [25, 23], and models often suffer from computational inefficiency and instability. Moreover, oversmoothing [40, 50, 7], where node representations become indistinguishable as layers stack, can severely degrade performance, particularly in tasks requiring fine-grained discrimination. These challenges highlight the need for more effective architectures that can capture global context without sacrificing stability or efficiency.

To address the these challenges, we draw inspiration from the success of implicit models [8, 1, 20, 18], which compute feature representations by solving nonlinear fixed-point equations rather than propagating information through stacked message-passing layers. [20] applied this paradigm to GNNs, enabling the capture of long-range dependencies in graphs. However, its extension to hypergraph neural networks remains unexplored. To fill this gap, we propose the **Implicit Hypergraph Neural Network** (**IHGNN**), which enjoys both the expressive power of hypergraph modeling with the stability and depth efficiency and stability of implicit architectures. IHGNN performs global reasoning in a single step by directly solving a nonlinear fixed-point equation, effectively capturing higher-order, long-range dependencies while avoiding the instability and inefficiency of deep stacked models. We provide both theoretical analysis and empirical evaluation to demonstrate its effectiveness.

We summarize the main contributions of this work: **(i) Implicit Hypergraph Learning Framework:** We propose the first hypergraph neural architecture that integrates implicit equilibrium formulations, enabling expressive representation learning without layer-wise message-passing iterations. **(ii) Theoretical Analysis:** We establish a well-posed training scheme with provable convergence guarantees, theoretically show that IHGNN mitigates oversmoothing, and derive a generalization bound for transductive learning on hypergraphs. **(iii) Empirical Evaluation:** On Cora, Pubmed, and Citeseer citation benchmarks, IHGNN achieves state-of-the-art performance, exhibiting robust accuracy, parameter stability, and training resilience under diverse conditions.

2 Preliminaries

Notation. We use $[n]$ to denote the set $\{1, 2, \ldots, n\}$. We denote vectors and matrices by lower- and upper-case boldface letters, respectively. For a vector $\mathbf{x} \in \mathbb{R}^d$, we use $\|\mathbf{x}\|_1, \|\mathbf{x}\|_2$, and $\|\mathbf{x}\|_\infty$ to denote the ℓ_1-, ℓ_2-, and ℓ_∞-norm of $\mathbf{x}$. We use $\mathbf{0}_n$ and $\mathbf{1}_n$ to denote an n-dimensional all-zero and all-one vectors, respectively. For two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, we use $\langle \mathbf{x}, \mathbf{y} \rangle$ or $\mathbf{x}^\top \mathbf{y}$ to denote their standard inner product. Given a vector $\mathbf{x} \in \mathbb{R}^d$, let $\mathrm{Diag}(\mathbf{x})$ denote the diagonal matrix with $\mathrm{Diag}(\mathbf{x})_{i,i} = \mathbf{x}_i$ for $i \in [d]$ and zeros elsewhere. The largest eigenvalue of $\mathbf{A}$ is denoted as $\lambda_{\max}(\mathbf{A})$. For two matrices $\mathbf{A}, \mathbf{B}$ we denote their Kronecker product and Hardmard product as $\mathbf{A} \otimes \mathbf{B}$ and $\mathbf{A} \odot \mathbf{B}$, respectively. For a matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$, its vectorization is defined as $\mathrm{vec}(\mathbf{A}) \in \mathbb{R}^{nd}$. We use $\|\mathbf{A}\|_F$ to denote its Frobenius norm, $\|\mathbf{A}\|$ to denote its spectral norm, and $\|\mathbf{A}\|_\infty$ to denote its maximum-row-sum norm. We define the inner product $\langle \mathbf{A}, \mathbf{B} \rangle := \mathrm{tr}[\mathbf{A}\mathbf{B}^\top]$.

Hypergraph. A (weighted) hypergraph is defined as $G = (V, E, \mathbf{w})$, which contains a set of nodes $V = \{v_1, v_2, \ldots, v_{|V|}\}$, a set of hyperedges $E = \{e_1, \ldots, e_{|E|}\}$, and a hyperedge-weight vector $\mathbf{w} = [w_1, \ldots, w_{|E|}]^\top \in \mathbb{R}^{|E|}$. Each hyperedge e_j is a nonempty subset of nodes and is assigned with a weight w_j. We denote $n := |V|$ as the number of nodes and $m := |E|$ as the number of hyperedges. We can represent the set E as an incidence matrix $\mathbf{H} \in \{0, 1\}^{|V| \times |E|}$, where for every $i \in [n], j \in [m]$, $\mathbf{H}_{i,j} := 1$ if $v_i \in e_j$, and $\mathbf{H}_{i,j} := 0$ otherwise. The hyperedge weight matrix $\mathbf{E} := \mathrm{Diag}(\mathbf{w}) \in \mathbb{R}^{m \times m}$ is defined as a diagonal matrix with $\mathbf{E}_{j,j} := w_j$ for each $j \in [m]$. The node degree matrix $\mathbf{D} := \mathrm{Diag}(\mathbf{H}\mathbf{w}) \in \mathbb{R}^{n \times n}$ is defined as a diagonal matrix with $\mathbf{D}_{i,i} := \sum_{j=1}^{m} \mathbf{H}_{i,j} w_{i,j}$ for each $i \in [n]$. The hyperedge degree matrix $\mathbf{B} := \mathrm{Diag}(\mathbf{H}^\top \mathbf{1}_m) \in \mathbb{R}^{m \times m}$ is defined as a diagonal matrix with $\mathbf{B}_{j,j} := \sum_{i=1}^{n} \mathbf{H}_{i,j}$ for each $j \in [m]$.

Hypergraph Neural Networks. We assume that each node v_i is equipped with input node feature $\mathbf{x}_i \in \mathbb{R}^d$. We denote the input node feature matrix as $\mathbf{X} := [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n] \in \mathbb{R}^{n \times d}$ where the i-th row of $\mathbf{X}$ is node feature $\mathbf{x}_i$. In the traditional Hypergraph Neural Network (HGNN) framework [16, 2], node features are updated iteratively through explicit layer-wise propagation. Formally, the t-th layer of an HGNN is defined as $\mathbf{X}^{(t+1)} = \phi\big(\mathbf{D}^{-1/2}\mathbf{H}\mathbf{E}\mathbf{B}^{-1}\mathbf{H}^\top\mathbf{D}^{-1/2}\mathbf{X}^{(t)}\mathbf{W}\big)$,

where $\mathbf{W} \in \mathbb{R}^{d\times d}$ is the trainable weight matrix, $\mathbf{X}^{(t)}$ is the input feature matrix at the t-th layer with $\mathbf{X}^{(1)} := \mathbf{X}$, $\phi : \mathbb{R} \to \mathbb{R}$ is an entry-wise nonlinear activation function, and $\mathbf{D}^{-1/2}\mathbf{H}\mathbf{E}\mathbf{B}^{-1}\mathbf{H}^\top\mathbf{D}^{-1/2}$ is the normalized hypergraph Laplacian matrix that governs feature propagation on the hypergraph. In other words, the normalized hypergraph Laplacian matrix serves as the propagation operator in HGNN, allowing information to flow across multiple nodes connected by common hyperedges and thus enabling the learning of high-order relationships. Note that when a hypergraph degenerates to a graph, it is exactly the normalized Laplacian matrix for the graph.

3 Implicity Hypergraph Neural Networks

3.1 The Architecture of IHGNN

Traditional HGNNs rely on explicit layer-wise propagation over a fixed number of iterations to perform feature aggregation. While effective, this approach often struggles to capture long-range dependencies and can suffer from instability in deep architectures. To address these limitations, we propose the Implicit Hypergraph Neural Network (IHGNN), which incorporates the nonlinear equilibrium formulation from implicit graph neural networks [20] into the hypergraph setting.

In IHGNN, the node representations are derived from a nonlinear fixed-point equation, rather than through iterative layer-wise updates. This equilibrium-based formulation allows for the modeling of global dependencies without increasing model depth, thereby improving stability and scalability.

Definition 3.1 (IHGNN)**.** The architecture of IHGNN is defined as the following mapping

$$\widehat{\mathbf{Y}} = f(\mathbf{X}; \mathbf{W}, \mathbf{\Theta}_1, \mathbf{\Theta}_2, \mathbf{b})$$

where $\mathbf{X} \in \mathbb{R}^{n\times d}$ is the input node feature matrix, $\mathbf{W} \in \mathbb{R}^{d_h\times d_h}, \mathbf{\Theta}_1 \in \mathbb{R}^{d\times d_h}, \mathbf{b} \in \mathbb{R}^{d_h}$, $\mathbf{\Theta}_2 \in \mathbb{R}^{d_h\times d'}$ are trainable weights, and f can be described with the following equations:

$$\widetilde{\mathbf{X}} = \mathbf{X}\mathbf{\Theta}_1 + \mathbf{1}_n\mathbf{b}^\top, \tag{1}$$

$$\mathbf{Z} = \phi\left(\mathbf{D}^{-1/2}\mathbf{H}\mathbf{E}\mathbf{B}^{-1}\mathbf{H}^\top\mathbf{D}^{-1/2}\mathbf{Z}\mathbf{W} + \widetilde{\mathbf{X}}\right), \tag{2}$$

$$\widehat{\mathbf{Y}} = \mathbf{Z}\mathbf{\Theta}_2, \tag{3}$$

where $\phi : \mathbb{R} \to \mathbb{R}$ is a nonlinear activation function.

Now we introduce the components of the IHGNN. Equation (1) defines an affine transformation that serves as a feature–preprocessing unit, injecting a learned skip term into the equilibrium layer, and Equation (3) produces the final prediction via a linear readout on the equilibrium state. The implicit layer Equation (2) can be viewed as solving the fixed-point equation $\mathbf{Z} = \mathcal{T}(\mathbf{Z}) := \phi(\mathbf{D}^{-1/2}\mathbf{H}\mathbf{E}\mathbf{B}^{-1}\mathbf{H}^\top\mathbf{D}^{-1/2}\mathbf{Z}\mathbf{W} + \widetilde{\mathbf{X}})$. We will show that under some mild conditions, $\mathbf{Z} = \mathcal{T}(\mathbf{Z})$ has a unique solution $\mathbf{Z}^*$, and the fixed-point iteration $\mathbf{Z}^{(t+1)} = \mathcal{T}(\mathbf{Z}^{(t)})$ converges to $\mathbf{Z}^*$ as $t \to \infty$. Hence f is a well-defined mapping in such cases.

3.2 Well-Posedness and Convergence Analysis

To ensure that IHGNN produces valid and stable node representations, it is essential to guarantee the existence and uniqueness of a solution to the implicit equilibrium equation for any given input. To simplify the notation, we denote the normalized hypergraph Laplacian matrix as $\mathbf{M} := \mathbf{D}^{-1/2}\mathbf{H}\mathbf{E}\mathbf{B}^{-1}\mathbf{H}^\top\mathbf{D}^{-1/2}$. Then the fixed-point equilibrium equation ,i.e., Equation (2), in IHGNN becomes $\mathbf{Z} = \phi(\mathbf{M}\mathbf{Z}\mathbf{W} + \widetilde{\mathbf{X}})$. We say that the fixed-point equilibrium equation is well-posed if for any $\widetilde{\mathbf{X}} \in \mathbb{R}^{d\times d}$, it has a unique solution $\mathbf{Z}^*$.

To derive a sufficient condition for well-posedness. We assume IHGNN is constructed on an admissible hypergraph, i.e., a hypergraph where each hyperedge is associated with a non-negative weight, and each node has a positive degree. Nonnegative hyperedge weights and strictly positive node degrees ensure the normalized hypergraph Laplacian $\mathbf{M} = \mathbf{D}^{-1/2}\mathbf{H}\mathbf{E}\mathbf{B}^{-1}\mathbf{H}^\top\mathbf{D}^{-1/2}$ is well defined, since $\mathbf{E} \succeq 0$ and $\mathbf{D}^{-1/2}$ and $\mathbf{B}^{-1}$ exist. Under these conditions $\mathbf{M}$ is positive semidefinite. Combined with the Lipschitz continuity of activation function ϕ and a spectral norm bound on $\mathbf{W}$, this yields a simple well-posedness condition. The proof is defered to Appendix C.

Theorem 3.2 (Sufficient condition for well-posedness)**.** Let $\mathbf{M} \in \mathbb{R}^{n\times n}$ be the normalized hypergraph Laplcaian matrix of an admissible hypergraph. Assume that $\phi : \mathbb{R} \to \mathbb{R}$ is a nonexpansive activation function, i.e., ϕ is 1-Lipschit. If the weight matrix $\mathbf{W} \in \mathbb{R}^{d_h\times d_h}$ of an IHGNN satisfies $\lambda_{\max}(|\mathbf{W}|) < 1$, then for any $\widetilde{\mathbf{X}} \in \mathbb{R}^{d\times d}$, the fixed-point equilibrium equation $\mathbf{Z} = \phi(\mathbf{M}\mathbf{Z}\mathbf{W} + \widetilde{\mathbf{X}})$ has a unique solution $\mathbf{Z}^* \in \mathbb{R}^{n\times d}$, and the fixed point iteration $\mathbf{Z}^{(t+1)} = \phi(\mathbf{M}\mathbf{Z}^{(t)}\mathbf{W}+\widetilde{\mathbf{X}})$ converges to $\mathbf{Z}^*$ as $t \to \infty$. Futhermore, if we assume that $\|\mathbf{Z}^*\|_F \le C_0$ for some $C_0 > 0$, $\lambda_{\max}(|\mathbf{W}|) = \kappa$ for some $\kappa \in [0, 1)$, and $\mathbf{Z}^{(1)} = \mathbf{0}_{n\times d}$, then for any integer $t \ge 1$, $\|\mathbf{Z}^{(t)} - \mathbf{Z}^*\|_F \le \kappa^{t-1}C_0$.

Note that in [20], they study the fixed-point layer $\mathbf{Z} = \phi(\mathbf{A}\mathbf{Z}\mathbf{W} + \text{bias})$ on *graphs* with adjacency matrix $\mathbf{A}$, and proves well-posedness under the spectral condition of $\mathbf{A}\otimes\mathbf{W}$. In contrast, Theorem 3.2 is stated for *hypergraps* and uses the normalized hypergraph operator $\mathbf{M}$, which collapses the joint constraint into the graph-agnostic requirement $\lambda_{\max}(|\mathbf{W}|) < 1$, yielding existence, uniqueness, and geometric convergence with rate $\kappa = \lambda_{\max}(|\mathbf{W}|)$. Moreover, employing a normalized Laplacian rather than the raw adjacency is standard and more realistic in practice: it controls the spectrum, mitigates degree heterogeneity, and avoids graph-dependent spectral blow-up.

3.3 Oversmoothing Analysis

Oversmoothing, i.e., the tendency of node embeddings to collapse toward an indistinguishable constant vector as depth increases—remains a central obstacle for deep (hyper)graph networks. Since IHGNN is an implicit architecture, depth is replaced by a fixed-point-solving procedure. Consequently, the classical layer-wise view of oversmoothing no longer applies directly.

Our first result is a sufficient condition for IHGNN to provably avoid the trivial constant solution. Our second result complements the first one by showing that, even under the identity activation, IHGNN remains as expressive as any K-th-order polynomial hypergraph filter. The proofs are in Appendix D.

Theorem 3.3 (Sufficient condition for nonidentical node features)**.** Let $\mathbf{M} \in \mathbb{R}^{n\times n}$ be the normalized hypergraph Laplcaian matrix of an admissible hypergraph. Let $\phi : \mathbb{R} \to \mathbb{R}$ be a strictly increasing nonexpansive activation function. Suppose that the weight matrix $\mathbf{W} \in \mathbb{R}^{d\times d}$ of an IHGNN satisfies $\lambda_{\max}(|\mathbf{W}|) < 1$, then for any $\widetilde{\mathbf{X}} \in \mathbb{R}^{d\times d}$ satisfying $\mathbf{x}_i \ne \mathbf{x}_j$ for some $i, j \in [n]$, there does not exists $\mathbf{z}_0 \in \mathbb{R}^d$, such that $\mathbf{Z}^* = \mathbf{1}_n\mathbf{z}_0^\top$.

Theorem 3.4 (Expressivity of IHGNN)**.** Let $\mathbf{M} \in \mathbb{R}^{n\times n}$ be the normalized hypergraph Laplcaian matrix of an admissible hypergraph. Let $K \in \mathcal{N}$. For every K-order polynomial filter function $p(\mathbf{X}) := (\sum_{k=0}^{K} \theta_k\mathbf{M}^k)\mathbf{X}$ with arbitrary coefficients $\{\theta_k\}_{k=0}^K$ and input feature matrix $\mathbf{x} \in \mathbb{R}^{n\times d}$, there exists an IHGNN with identity activation can express it.

3.4 Transductive Generalization Analysis

In this section, we conduct a theoretical analysis of transductive learning on hypergraphs. Let $\mathcal{X} := \mathbb{R}^d$ be the input feature space and $\mathcal{Y} := \mathbb{R}^{d'}$ be the output label space. In the transductive setting, we observe the entire hypergraph G and node features $\{\mathbf{x}_i\}_{i=1}^n$, but labels only for a subset of nodes. Let $S \subseteq [n]$ denote the labeled indices with $|S| = s$ and $U = [n]\setminus S$ the unlabeled indices with $|U| = u$ (so $s + u = n$). During training, the learner has access to $\{\mathbf{x}_i\}_{i=1}^n$ and $\{\mathbf{y}_i\}_{i\in S}$, and the goal is to predict $\{\mathbf{y}_i\}_{i\in U}$, i.e., the labels for all nodes with inidices in U. Without loss of generality, we index nodes so that $S = \{1, \dots, s\}$ and $U = \{s + 1, \dots, n\}$. For any $f \in \mathcal{H}$, we define the training and testing as $\widehat{\mathcal{L}}_s(f) := \frac{1}{s}\sum_{i=1}^{s} \ell(f(\mathbf{x}_i), \mathbf{y}_i)$ and $\mathcal{L}_u(f) := \frac{1}{u}\sum_{i=s+1}^{s+u} \ell(f(\mathbf{x}_i), \mathbf{y}_i)$, respectively, where $\ell : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \to [0, \infty)$ is a loss function.

Assumption 3.5. We assume the following conditions hold.

- **Bounded input features**: The input node feature matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$ satisfies $\|\mathbf{x}_i\|_2 \le C_X$ for each $i \in [n]$ for some $C_X \ge 0$.
- **Bounded trainable parameters**: The trainable parameters $\mathbf{\Theta}_1, \mathbf{\Theta}_2, \mathbf{b}, \mathbf{W}$ satisfies $\|\mathbf{\Theta}_1\|_F \le \rho_1, \|\mathbf{\Theta}_2\|_F \le \rho_2, \|\mathbf{b}\|_2 \le C_b$ for some $\rho_1, \rho_2, C_b \ge 0$, and $\|\mathbf{W}\| \le \kappa$ for some $\kappa \in [0, 1)$. For simplicity, we assume their dimensions satisfies $d = d_h = d'$.
- **Lipschitz loss**: The loss function ℓ is C_ℓ-Lipschitz for some $C_\ell \ge 0$.
- **Nonexpansive activation**: The activation function ϕ is nonexpansive, i.e., 1-Lipschitz.

These assumptions are standard and easy to meet. Feature vectors are routinely normalized in practice, e.g., ℓ_2-normalization, and many benchmark node features are already bounded. During the training, weight decay/regularization directly impose norm constraints on $\mathbf{\Theta}_1, \mathbf{\Theta}_2, \mathbf{b}$. The spectral bound $\|\mathbf{W}\| \leq \kappa < 1$ directly enforces the contraction needed for a unique equilibrium when combined with the normalized operator so the fixed point exists and is reached geometrically. Lipschitz loss and nonexpansive activations are very common and easy to meet this on bounded domains, e.g., squared/hinge losses, cross-entropy are Lipschitz in the probability simplex or when logits are bounded, and ReLU ≤ 1, $\tanh$, sigmoid activations are 1-Lipschitz.

Next, we state our main results of generalization bounds. The proofs are deferred in Appendix E.

Theorem 3.6 (Transductive generalization bound of IHGNN)**.** Suppose Assumption 3.5 is satisfied. Let $\mathcal{H}$ be the hypothesis class of IHGNN models defined on the any admissble hypergraph. Let $P := \frac{1}{s} + \frac{1}{u}$, and $Q := \frac{s+u}{(s+u-1/2)(1-1/(2\max\{s,u\}))}$. Then, for any $\delta > 0$, with probability at least $1-\delta$ over the choice of the training set $\{\mathbf{x}_i\}_{i=1}^{s+u} \cup \{y_i\}_{i=1}^{s}$, for all $f \in \mathcal{H}$, we have

$$\mathcal{L}_u(f) \leq \widehat{\mathcal{L}}_s(f) + \frac{\sqrt{2}\rho_2 C_\ell(\rho_1 C_x + \sqrt{d} C_b)}{(1-\kappa)\sqrt{s+u}} + \sqrt{\frac{32\log(4e)}{3}} P\sqrt{\min\{s,u\}} + \sqrt{\frac{PQ}{2}\log\frac{1}{\delta}}.$$

In the asymptotic regime, we further simplify the generalization bound.

Corollary 3.7 (Asymptotic transductive generalization bound of IHGNN)**.** Under the same conditions in Theorem 3.6, for sufficiently large training-set size s and testing-set size u, for any $\delta > 0$, with probability at least $1-\delta$ over the choice of the training set, for all $f \in \mathcal{H}$, we have

$$\mathcal{L}_u(f) \leq \widehat{\mathcal{L}}_s(f) + O\left(\frac{d}{s+u}\right)^{\frac{1}{2}} + O\left(\frac{\log(1/\delta)}{\min\{s,u\}}\right)^{\frac{1}{2}}.$$

Note that the second term $O(\frac{d}{s+u})^{1/2}$ decays as either the training-set size s or the testing-set size u increases. However, the last term $O(\frac{\log(1/\delta)}{\min\{s,u\}})^{1/2}$ converges slowly whenever $s \ll u$ or $u \ll s$. The regime $s \ll u$ corresponds to an under-fitted model, whereas when $u \ll s$, the sample mean computed from the u test nodes, drawn out of the $s+u$ available nodes, has high variance.

3.5 Training of IHGNN

Directly enforcing the constraint $\lambda_{\max}(|\mathbf{W}|) < 1$ is computationally difficult due to its non-convexity with respect to $\mathbf{W}$. To address this, we introduce a more tractable surrogate constraint. By assuming the activation is positively homogeneous, meaning $\sigma(\alpha x) = \alpha\sigma(x)$ for all $\alpha \geq 0$. We can derive the following condition and its proof is in Appendix F.

Theorem 3.8 (Scaled Well-Posedness of IHGNN)**.** Suppose that the activation function $\phi : \mathbb{R} \to \mathbb{R}$ is positively homogeneous and nonexpansive. If an IHGNN model with weights $\mathbf{W}, \mathbf{\Theta}_1, \mathbf{\Theta}_2, \mathbf{b}$ satisfies $\lambda_{\max}(|\mathbf{W}|) < 1$, then there exists an equivalent IHGNN model with weights $\widetilde{\mathbf{W}}, \widetilde{\mathbf{\Theta}}_1, \widetilde{\mathbf{\Theta}}_2, \widetilde{\mathbf{b}}$ such that $\|\widetilde{\mathbf{W}}\|_\infty < 1$, and both models produce identical outputs for all same inputs.

Although this constraint is stricter, it remains valid for many activation functions (e.g., ReLU, identity) that are positively homogeneous. To maintain this condition during training, we apply a projection step after each gradient update: $\Pi_C(W) := \arg\min_{\|\mathbf{W}'\|_\infty \leq \kappa} \|\mathbf{W}' - \mathbf{W}\|_F^2$, where $\kappa \in [0, 1)$ is a relaxation parameter. This operation ensures that the updated weights lie within the feasible region defined by the well-posedness constraint.

Using the chain rule, we compute gradients with respect to parameters Ω and the hidden state $\mathbf{Z}$, followed by gradients for parameters $q \in \{\mathbf{W}, \mathbf{\Theta}_1, \mathbf{b}\}$. The gradient of the loss with respect to q is expressed as: $\nabla_q \mathcal{L} = \langle \frac{\partial}{\partial} q(\mathbf{MZW} + \mathbf{X\Theta}_1 + \mathbf{1}_n \mathbf{b}^\top), \nabla_{\mathbf{Z}} L \rangle$, where $\mathbf{X}$ is treated as fixed during this step. The quantity $\nabla_{\mathbf{Z}}\mathcal{L}$ satisfies the recursive equation: $\nabla_{\mathbf{Z}}\mathcal{L} = \phi'(\mathbf{Z}) \odot \left(\mathbf{M}^\top \nabla_{\mathbf{Z}} L \mathbf{W}^\top + \nabla_{\mathbf{\Theta}_1}\mathcal{L} + \nabla_{\mathbf{b}}\mathcal{L}\right)$. Note that our previous analysis can be adapted to show that this equation can be efficiently approximately solved via fixed-point iteration.

4 Experiment

4.1 Experimental Setup

We implement IHGNN in PyTorch (Python 3.12). Experiments run on a workstation with an Intel i5-12600KF CPU and an NVIDIA RTX 4070 GPU (Windows 11). For evaluation, we use the standard citation benchmarks Cora [56], Citeseer [56], and PubMed [49] for node classification and generalization. Publications are nodes. To construct hypergraphs, we adopt a bibliographic-coupling scheme: for each cited paper, we create a hyperedge connecting all papers that cite it. This captures community-level citation interactions and higher-order structure beyond pairwise links.

4.2 Performance Analysis

Table 1 shows the performance of IHGNN against the various baselines. IHGNN ranks first on all three datasets (Cora/Pubmed/Citeseer: 85.9/83.8/75.1), exceeding the best baseline by +1.5, +3.5, and +1.5 points, respectively (avg. +2.2). Classical embeddings (Semi-SE, DeepWalk) underperform, while GCN/GAT improve but are limited to pairwise edges. HGNN captures higher-order structure yet degrades with depth; IGNN stabilizes deep propagation on pairwise graphs. By combining an implicit equilibrium layer with hypergraph modeling, IHGNN captures higher-order *and* long-range dependencies, yielding consistent gains across diverse citation networks.

Table 1: Accuracy (%) on citation benchmarks.

Model	Cora	Pubmed	Citeseer
Semi-SE [61]	59.0	70.7	60.1
DeepWalk [51]	67.2	65.3	43.2
Planetoid [70]	75.7	77.2	64.7
GCN [35]	81.5	79.0	70.3
GAT [60]	83.0	79.0	72.5
HGNN [15]	81.6	80.2	69.2
IGNN [20]	84.4	80.3	73.6
IHGNN	**85.9**	**83.8**	**75.1**

4.3 Stability and Robustness Analysis

We evaluate stability under two perturbations—50 random seeds and hyperparameter sweeps (hidden size, learning rate, dropout)—and observe consistently small variance: on Cora, F1/Acc $\approx$ 0.862/0.862 with std 0.0066; on Pubmed, $\approx$ 0.835/0.836 with std 0.0028; on Citeseer, $\approx$ 0.753/0.751 with std 0.0069/0.0068, with 95% CI half-widths ≤ 0.002 across datasets. Hyperparameter changes shift scores by only $\pm$0.01–0.04, indicating weak sensitivity and a low tuning burden. This is due to the implicit equilibrium layer with non-expansive activation and constrained weight norm, together with hyperedge aggregation.

Table 2: Fifty-seed stability: mean, standard deviation, and 95% CI for F1/Accuracy.

Dataset	Metric	Mean $\pm$ Std	95% CI
Cora	F1	0.8621 ± 0.0066	[0.8603, 0.8639]
	Acc	0.8619 ± 0.0066	[0.8601, 0.8637]
Pubmed	F1	0.8352 ± 0.0028	[0.8345, 0.8360]
	Acc	0.8358 ± 0.0028	[0.8350, 0.8366]
Citeseer	F1	0.7529 ± 0.0069	[0.7510, 0.7548]
	Acc	0.7508 ± 0.0068	[0.7489, 0.7527]

4.4 Convergence and Oversmoothing Anlaysis

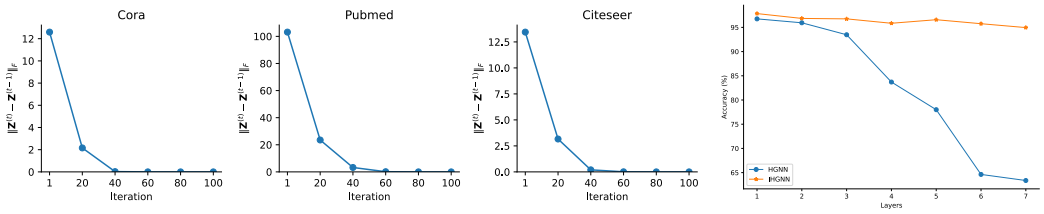

Figure 1: (a) Fixed-point convergence on Cora, Pubmed, and Citeseer measured by the residual $\Delta_t = \|\mathbf{Z}^{(t)} - \mathbf{Z}^{(t-1)}\|_F$. (b) Test accuracy vs. depth on ModelNet40 for HGNN and IHGNN.

We track the fixed-point residual $\Delta_t = \|\mathbf{Z}^{(t)} - \mathbf{Z}^{(t-1)}\|_F$ on benchmarks; Figure 1(a) shows rapid decay of Δ_t toward zero across datasets, confirming the expected contraction behavior induced by non-expansive activations and norm-constrained weights. To probe depth effects, we compare with

the HGNN on ModelNet40 and vary propagation depth: as shown in Figure 1(b), HGNN accuracy degrades with increasing depth (over-smoothing), whereas IHGNN remains stable, indicating that the implicit hypergraph formulation preserves discriminative signals even at large effective depth.

5 Conclusion

In this work, we introduce IHGNN for stable higher-order relational learning. We prove well-posedness, convergence, and a transductive generalization bound. On citation benchmarks, IHGNN consistently improves accuracy, robustness, and training stability over strong GNN/HGNN baselines. These results show IHGNN captures long-range higher-order dependencies and establishes implicit methods as a practical basis for learning on non-pairwise structures.

References

[1] S. Bai, J. Z. Kolter, and V. Koltun. Deep equilibrium models. *Advances in neural information processing systems*, 32, 2019.

[2] S. Bai, F. Zhang, and P. H. Torr. Hypergraph convolution and hypergraph attention. *Pattern Recognition*, 110:107637, 2021.

[3] J. Baker, Q. Wang, C. D. Hauck, and B. Wang. Implicit graph neural networks: A monotone operator viewpoint. In *International Conference on Machine Learning*, pages 1521–1548. PMLR, 2023.

[4] A. Bazaga, P. Lio, and G. Micklem. Hyperbert: Mixing hypergraph-aware layers with language models for node classification on text-attributed hypergraphs. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 9181–9193, 2024.

[5] C. Bodnar, F. Frasca, and M. M. Bronstein. Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in graph neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2023.

[6] D. Cai, M. Song, C. Sun, B. Zhang, S. Hong, and H. Li. Hypergraph structure learning for hypergraph neural networks. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pages 1923–1929, 2022.

[7] G. Chen, J. Zhang, X. Xiao, and Y. Li. Preventing over-smoothing for hypergraph neural networks. *arXiv preprint arXiv:2203.17159*, 2022.

[8] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems (NeurIPS)*, 31, 2018.

[9] K. Ding, A. J. Liang, B. Perozzi, T. Chen, R. Wang, L. Hong, E. H. Chi, H. Liu, and D. Z. Cheng. Hyperformer: Learning expressive sparse feature representations via hypergraph transformer. In *Proceedings of the 46th international ACM SIGIR conference on research and development in information retrieval*, pages 2062–2066, 2023.

[10] K. Ding, J. Wang, J. Li, D. Li, and H. Liu. Be more with less: Hypergraph attention networks for inductive text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4927–4936, 2020.

[11] Y. Dong, W. Sawin, and Y. Bengio. Hnhn: Hypergraph networks with hyperedge neurons. *arXiv preprint arXiv:2006.12278*, 2020.

[12] E. Dupont, A. Doucet, and Y. W. Teh. Augmented neural odes. *Advances in neural information processing systems (NeurIPS)*, 32, 2019.

[13] R. El-Yaniv and D. Pechyony. Transductive rademacher complexity and its applications. *Journal of Artificial Intelligence Research*, 35:193–234, 2009.

[14] Y. Feng, C. Yang, X. Hou, S. Du, S. Ying, Z. Wu, and Y. Gao. Beyond graphs: Can large language models comprehend hypergraphs? In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025.

[15] Y. Feng, H. You, Z. Zhang, R. Ji, and Y. Gao. Hypergraph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3558–3565, 2019.

[16] Y. Feng, H. You, Z. Zhang, R. Ji, and Y. Gao. Hypergraph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3558–3565, 2019.

[17] Y. Feng, Y. Zhang, S. Ying, S. Du, and Y. Gao. Kernelized hypergraph neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

[18] G. Fu, M. H. Dupty, Y. Dong, and L. W. Sun. Implicit graph neural diffusion networks: Convergence, generalization, and over-smoothing. *arXiv preprint arXiv:2308.03306*, 2023.

[19] Y. Gao, Y. Feng, S. Ji, and R. Ji. Hgnn+: General hypergraph neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3181–3199, 2022.

[20] F. Gu, H. Chang, W. Zhu, S. Sojoudi, and L. El Ghaoui. Implicit graph neural networks. *Advances in neural information processing systems (NeurIPS)*, 33:11984–11995, 2020.

[21] W. L. Hamilton, R. Ying, and J. Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1024–1034, 2017.

[22] A. Han, D. Shi, L. Lin, and J. Gao. From continuous dynamics to graph neural networks: Neural diffusion and beyond. *Transactions on Machine Learning Research*, 2024.

[23] J. Han, Y. Li, T. Gao, and C.-T. Wang. Resgcn: Residual graph convolutional networks for graph classification. *IEEE Transactions on Neural Networks and Learning Systems*, 32(10):4514–4526, 2021.

[24] Y. Han, P. Wang, S. Kundu, Y. Ding, and Z. Wang. Vision hgnn: An image is more than a graph of nodes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19878–19888, 2023.

[25] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CVPR*, 2016.

[26] R. A. Horn and C. R. Johnson. *Topics in matrix analysis*. Cambridge university press, 1994.

[27] J. Huang, Y. Pu, D. Zhou, J. Cao, J. Gu, Z. Zhao, and D. Xu. Dynamic hypergraph convolutional network for multimodal sentiment analysis. *Neurocomputing*, 565:126992, 2024.

[28] J. Huang and J. Yang. Unignn: a unified framework for graph and hypergraph neural networks. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pages 2563–2569, 2021.

[29] S. Ji, Y. Feng, D. Di, S. Ying, and Y. Gao. Mode hypergraph neural network. *IEEE Transactions on Neural Networks and Learning Systems*, 2025.

[30] J. Jiang, Y. Wei, Y. Feng, J. Cao, and Y. Gao. Dynamic hypergraph neural networks. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 2635–2641, 2019.

[31] J. Jiang, Y. Wei, Y. Feng, J. Cao, and Y. Gao. Dynamic hypergraph neural networks. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2635–2641, 2019.

[32] M. Jin, Y. Zheng, Y.-F. Li, S. Chen, B. Yang, and S. Pan. Multivariate time series forecasting with dynamic graph neural odes. *IEEE Transactions on Knowledge and Data Engineering*, 35(9):9168–9180, 2022.

[33] B. Khan, J. Wu, J. Yang, and X. Ma. Heterogeneous hypergraph neural network for social recommendation using attention network. *ACM Transactions on Recommender Systems*, 3(3):1–22, 2025.

[34] E.-S. Kim, W. Y. Kang, K.-W. On, Y.-J. Heo, and B.-T. Zhang. Hypergraph attention networks for multimodal learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14581–14590, 2020.

[35] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

[36] S. Kumar. Co-authorship networks: a review of the literature. *Aslib Journal of Information Management*, 67(1):55–73, 2015.

[37] M. Lei, Y. Wu, S. Li, X. Zheng, J. Wang, Y. Gao, and S. Du. Softhgnn: Soft hypergraph neural networks for general visual recognition. *arXiv preprint arXiv:2505.15325*, 2025.

[38] G. Li, M. Müller, A. Thabet, B. Ghanem, V. Koltun, and L. J. Guibas. DeeperGCN: All you need to train deeper GCNs. *arXiv preprint arXiv:2006.07739*, 2020.

[39] M. Li, Y. Fang, Y. Wang, H. Feng, Y. Gu, L. Bai, and P. Lio. Deep hypergraph neural networks with tight framelets. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 18385–18392, 2025.

[40] Q. Li, Z. Han, and X.-M. Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence (AAAI)*, volume 32, 2018.

[41] X. Li, Y. Liu, and M. Wang. HyperRec: Hypergraph neural recommendation. In *Proceedings of the 45th International ACM SIGIR Conference (SIGIR)*, pages 373–383, 2022.

[42] J. Lin, Z. Ling, Z. Feng, J. Xu, M. Liao, F. Zhou, T. Hou, Z. Liao, and R. C. Qiu. Ignn-solver: A graph neural solver for implicit graph neural networks. *arXiv preprint arXiv:2410.08524*, 2024.

[43] J. Liu, B. Hooi, K. Kawaguchi, Y. Wang, C. Dong, and X. Xiao. Scalable and effective implicit graph neural networks on large graphs. In *The Twelfth International Conference on Learning Representations*, 2024.

[44] J. Liu, B. Hooi, K. Kawaguchi, and X. Xiao. Mgnni: Multiscale graph neural networks with implicit layers. *Advances in Neural Information Processing Systems*, 35:21358–21370, 2022.

[45] Z. Liu, X. Wang, B. Wang, Z. Huang, C. Yang, and W. Jin. Graph odes and beyond: A comprehensive survey on integrating differential equations with graph neural networks. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 6118–6128, 2025.

[46] N. Ma, Z. Wu, Y. Feng, C. Wang, and Y. Gao. Multi-view time-series hypergraph neural network for action recognition. *IEEE Transactions on Image Processing*, 33:3301–3313, 2024.

[47] A. Maurer. A vector-contraction inequality for rademacher complexities. In *International Conference on Algorithmic Learning Theory*, pages 3–17. Springer, 2016.

[48] M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of machine learning*. MIT press, 2018.

[49] G. Namata, B. London, and L. Getoor. Query-driven active surveying for collective classification. In *10th International Workshop on Mining and Learning with Graphs (MLG)*, pages 1–8, 2012.

[50] K. Oono and T. Suzuki. Graph neural networks exponentially lose expressive power for node classification. In *International Conference on Learning Representations (ICLR)*, 2020.

[51] B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, 2014.

[52] K. B. Petersen, M. S. Pedersen, et al. The matrix cookbook. *Technical University of Denmark*, 7(15):510, 2008.

[53] M. Poli, S. Massaroli, J. Park, A. Yamashita, H. Asama, and J. Park. Graph neural ordinary differential equations. *arXiv preprint arXiv:1911.07532*, 2019.

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

[55] N. Sasikaladevi and A. Revathi. Hypergraph convolutional neural network for fast and accurate diagnosis (fat) of covid from x-ray images. *International Journal of Pattern Recognition and Artificial Intelligence*, 36(10):2257005, 2022.

[56] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad. Collective classification in network data. In *AI magazine*, volume 29, pages 93–93. Association for the Advancement of Artificial Intelligence, 2008.

[57] Z. Shao, D. Shi, A. Han, Y. Guo, Q. Zhao, and J. Gao. Unifying over-smoothing and over-squashing in graph neural networks: A physics informed approach and beyond. *arXiv preprint arXiv:2309.02769*, 2023.

[58] D. Shi, A. Han, L. Lin, Y. Guo, and J. Gao. Exposition on over-squashing problem on gnns: Current methods, benchmarks and challenges. *arXiv preprint arXiv:2311.07073*, 2023.

[59] M. Taheri, A. Scott, and J. Jones. Hyper–SocialNet: Hypergraph neural networks for social network analysis. In *Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 1–8, 2021.

[60] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[61] J. Weston, F. Ratle, and R. Collobert. Deep learning via semi-supervised embedding. In *Proceedings of the 25th international conference on Machine learning*, pages 1168–1175, 2008.

[62] S. Wu, F. Sun, W. Zhang, X. Xie, and B. Cui. Graph neural networks in recommender systems: a survey. *ACM Computing Surveys*, 55(5):1–37, 2022.

[63] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24, 2020.

[64] L.-P. Xhonneux, M. Qu, and J. Tang. Continuous graph neural networks. In *International conference on machine learning*, pages 10432–10441. PMLR, 2020.

[65] L. Xie, S. Gao, J. Liu, M. Yin, and T. Jin. K-hop hypergraph neural network: A comprehensive aggregation approach. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 21679–21687, 2025.

[66] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How powerful are graph neural networks? In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[67] N. Yadati, M. Nimishakavi, P. Yadav, V. Nitin, A. Louis, and P. Talukdar. Hypergcn: A new method for training graph convolutional networks on hypergraphs. *Advances in neural information processing systems (NeurIPS)*, 32, 2019.

[68] M. Yang and X.-J. Xu. Recent advances in hypergraph neural networks. *arXiv preprint arXiv:2503.07959*, 2025.

[69] Y. Yang, T. Liu, Y. Wang, Z. Huang, and D. Wipf. Implicit vs unfolded graph neural networks. *Journal of Machine Learning Research*, 26(82):1–46, 2025.

[70] Z. Yang, W. W. Cohen, and R. Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. In *ICML*, pages 40–48, 2016.

Appendix

A Related Work

A.1 Graph and Hypergraph Neural Networks

Graph neural networks (GNNs) have become a foundational approach for learning over graph-structured data. Classical GNN models introduced neighborhood aggregation and attention mechanisms that achieved strong performance on node-level tasks [35, 21, 60, 66, 38]. Despite their successes, standard message-passing graph neural networks are fundamentally restricted to modeling pairwise relationships between nodes. As network depth increases, they often suffer from over-smoothing, i.e., node representations become indistinguishable, and limited receptive fields, which hinder their ability to capture long-range dependencies [50, 5, 58, 57, 54].

To capture higher-order structure information beyond pairwise edges, [16] developed HGNN which propagates signals along node–hyperedge–node paths using an incidence-based Laplacian. [2] studied hypergraph convolution and hypergraph attention networks. [11] introduced HNHN that leverages explicit hyperedge neurons with nonlinearities and degree/cardinality-aware normalization. [10] proposed HyperGAT, a hypergraph attention networks for inductive text classification. HyperFormer [9] is a hyper-relational knowledge graph completer with mixture-of-experts layers for better accuracy at lower cost. In addition, [31] extend the idea to dynamic hypergraphs by alternating topology construction with hypergraph convolution so the structure adapts during learning, and [6] introduced the hypergraph structure learning for hypergraph neural networks. [28] established a unified framework for graph neural networks and hypergraph neural networks.

These models have been adapted to a variety of domains including recommendation [41, 33], multi-modal learning [34, 27], and social network analysis [59]. More recent works on HGNNs include [39, 68, 65, 17, 29]. However, most HGNNs still rely on explicit iterative message passing, which becomes computationally expensive and can be unstable as depth grows.

A.2 Implicit Graph Models and Graph Neural ODEs

Implicit models originate from the idea of replacing explicit layer stacking with the solution of an equilibrium or continuous-depth system. Deep equilibrium models recast an infinitely deep network as a root-finding problem whose fixed point defines the representation; they train end-to-end via implicit differentiation, yielding constant memory (up to solver tolerance) and facilitating long-range dependency modeling [1]. In parallel, Neural ODEs view depth as continuous time and use ODE solvers for forward inference together with adjoint-based or implicit differentiation for training, mitigating optimization issues associated with very deep stacks [8, 12]. Both lines share a common premise: use a solver instead of layers to achieve global information propagation, better memory efficiency, and improved stability.

Building on these foundations, implicit graph models transfer the equilibrium/continuous-depth perspective to graph-structured data. [20] introduced IGNN by formulating graph inference as solving a nonlinear fixed-point equation so that information couples across the entire graph without deep propagation. Later, [3] leverage the monotone operator theory and enhance the performance of IGNN in learning long range dependencies. [44] developed multiscale graph neural networks with implicit layers. [18] introduced a framework for designing implicit graph diffusion layers using parameterized graph Laplacians. [43] showed how to efficiently train implicit GNNs to provide effective predictions on large graphs. [42] introduced a learnable neural solve for IGNNs. [22] identifies the core building blocks for adapting continuous dynamics to GNNs and proposes a general framework for designing graph neural dynamics. The recent work of [69] systematically compared implicit and unfolded GNNs from both empirical and theoretical perspectives.

From the continuous-depth side, Graph Neural ODEs model the evolution of node states along a learned vector field on the graph, often yielding smoother gradients and robust training on large or sparse graphs [53, 64, 32, 45].

B Additional Preliminaries

B.1 Kronecker Product and Vectorization

We first recall two linear-algebraic operators that we use repeatedly to manipulate block structures and linearize bilinear expressions.

Definition B.1 (Kronecker product)**.** Let $A \in \mathbb{R}^{n_1 \times d_1}$ and $B \in \mathbb{R}^{n_2 \times d_2}$ be two matrices. The Kronecker product of A and B is defined as

$$\mathbf{A} \otimes \mathbf{B} := \begin{bmatrix} \mathbf{A}_{1,1}\mathbf{B} & \mathbf{A}_{1,2}\mathbf{B} & \cdots & \mathbf{A}_{1,d_1}\mathbf{B} \\ \mathbf{A}_{2,1}\mathbf{B} & \mathbf{A}_{2,2}\mathbf{B} & \cdots & \mathbf{A}_{2,d_1}\mathbf{B} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{n_1,1}\mathbf{B} & \mathbf{A}_{n_1,2}\mathbf{B} & \cdots & \mathbf{A}_{n_1,d_1}\mathbf{B} \end{bmatrix} \in \mathbb{R}^{n_1 n_2 \times d_1 d_2},$$

where $(\mathbf{A} \otimes \mathbf{B})_{(i_1-1)n_2+i_2,(j_1-1)d_2+j_2} = \mathbf{A}_{i_1,j_1}\mathbf{B}_{i_2,j_2}$ for $i_1 \in [n_1], j_1 \in [d_1], i_2 \in [n_2], j_2 \in [d_2]$.

Definition B.2 (Vectorization)**.** Let $A \in \mathbb{R}^{n \times d}$ be a matrix. The vectorization of A is defined as

$$\mathrm{vec}(\mathbf{A}) = \begin{bmatrix} \mathbf{A}_{:,1} \\ \mathbf{A}_{:,1} \\ \vdots \\ \mathbf{A}_{:,d} \end{bmatrix} \in \mathbb{R}^{n \times d}$$

where $\mathbf{A}_{:,j}$ is the j-th column of $\mathbf{A}$, and $\mathrm{vec}(\mathbf{A})_{(i-1)d+j} = \mathbf{A}_{i,j}$ for $i \in [n], j \in [d]$.

The next identity, often called the *tensor trick*, links matrix multiplication with Kronecker products and vectorization and will be used to streamline several derivations.

Lemma B.3 (Tensor trick, Section 10.2.2 in [52])**.** Given three matrices $\mathbf{A} \in \mathbb{R}^{m \times n}, \mathbf{X} \in \mathbb{R}^{n \times p}, \mathbf{B} \in \mathbb{R}^{p \times q}$, we have

$$\mathrm{vec}(\mathbf{AXB}) = (\mathbf{B}^\top \otimes \mathbf{A})\,\mathrm{vec}(\mathbf{X}).$$

We also record a standard spectral property of Kronecker products, which allows us to relate eigenvalues and eigenvectors of factors to those of the product.

Lemma B.4 (Spectrum of the Kroneck product, Theorem 4.2.12 in [26])**.** Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{B} \in \mathbb{R}^{p \times q}$ be two matrices. Let $(\lambda, \mathbf{x})$ and $(\lambda, \mathbf{x})$ be two pairs of eigenvalue and eigenvector of $\mathbf{A}$ and $\mathbf{B}$, respectively. Then $\lambda\mu$ is an eigenvalue of $\mathbf{A} \otimes \mathbf{B}$ with corresponding eigenvector $\mathbf{x} \otimes \mathbf{y}$. Moreover, any eigenvalue of $\mathbf{A} \otimes \mathbf{B}$ is a product of eigenvalues of $\mathbf{A}$ and $\mathbf{B}$.

B.2 Rademacher Complexity

We next recall capacity measures used in our generalization analysis, beginning with the classical (scalar-valued) Rademacher complexity.

Definition B.5 (Rademacher complexity, Definition 3.1 in [48])**.** Let $\mathcal{G}$ be a family of functions mapping from $\mathcal{Z}$ to $\mathbb{R}$ and $S = \{z_i\}_{i=1}^n \subseteq \mathcal{Z}$ a set of samples with elements in $\mathcal{Z}$. Then, the (empirical) Rademacher complexity of $\mathcal{G}$ with respect to the sample set S is defined as

$$\mathcal{R}_n(\mathcal{G}) := \mathop{\mathbb{E}}_{\boldsymbol{\varepsilon}} \left[\sup_{g \in \mathcal{G}} \frac{1}{n} \sum_{i=1}^n \varepsilon_i g(z_i) \right],$$

where each $\boldsymbol{\varepsilon} = [\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n]^\top$, and for each $i \in [n]$, ε_i is an independent Rademacher random variable, i.e., $\varepsilon_i \sim \mathsf{Uniform}(\{-1, 1\})$.

Because our hypotheses are vector-valued, we also use a coordinate-wise variant that aggregates fluctuations across output dimensions.

Definition B.6 (Coordinate-wise Rademacher complexity [47])**.** Let $\mathcal{F}$ be a family of functions mapping from $\mathcal{Z}$ to $\mathbb{R}^d$ and $S = \{z_i\}_{i=1}^n \subseteq \mathcal{Z}$ a set of samples with elements in $\mathcal{Z}$. Let $f(\cdot)_j$ denote the j-th entry of the output of the function $f(\cdot)$. Then, the (empirical) coordinate-wise Rademacher complexity of $\mathcal{F}$ with respect to the sample set S is defined as

$$\mathcal{R}_n^{\text{coord}}(\mathcal{F}) := \mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}} \left[\sup_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} \boldsymbol{\mathcal{E}}_{i,j} f(z_i)_j \right],$$

where each $\boldsymbol{\mathcal{E}}$ is an $n \times d$ random matrix, and for each $i \in [n], j \in [d]$, $\boldsymbol{\mathcal{E}}_{i,j}$ is an independent Rademacher random variable, i.e., $\boldsymbol{\mathcal{E}}_{i,j} \sim \mathsf{Uniform}(\{-1, 1\})$.

The following contraction inequality will be crucial to control the effect of Lipschitz losses applied to vector-valued predictors.

Lemma B.7 (Contraction inequality for vector-valued function class [47])**.** Let $\mathcal{F}$ be a family of functions mapping from $\mathcal{Z}$ to $\mathbb{R}^d$ and $S = \{z_i\}_{i=1}^n \subseteq \mathcal{Z}$ a set of samples with elements in $\mathcal{Z}$. Let $\rho : \mathbb{R}^d \to \mathbb{R}$ be a K-Lipschitz function for some $K \geq 0$. Then we have

$$\mathcal{R}_n(\rho \circ \mathcal{F}) \leq \sqrt{2} K \cdot \mathcal{R}_n^{\text{coord}}(\mathcal{F}).$$

B.3 Transductive Rademacher Complexity

For our transductive setting, we use the transductive Rademacher complexity of [13] to quantify capacity when both labeled and unlabeled points are fixed in advance.

Definition B.8 (Transductive Rademacher Complexity, Definition 1 in [13])**.** Let $\mathcal{G}$ be a family of functions mapping from $\mathcal{Z}$ to $\mathbb{R}$ and $S = \{z_i\}_{i=1}^{s+u} \subseteq \mathcal{Z}$ a set of samples with elements in $\mathcal{Z}$. Let $\boldsymbol{\sigma} = [\sigma_1, \sigma_2, \ldots, \sigma_n]^\top$, and for each $i \in [n]$, σ_i is an independent random variable defined as

$$\sigma_i := \begin{cases} 1, & \text{with probability } p, \\ -1, & \text{with probability } p, \\ 0, & \text{with probability } 1 - 2p. \end{cases}$$

Then, the transductive Rademacher complexity of $\mathcal{G}$ with respect to the sample set S is defined as

$$\widetilde{\mathcal{R}}_{s+u}(\mathcal{G}, p) := \mathop{\mathbb{E}}_{\boldsymbol{\sigma}} \left[\sup_{g \in \mathcal{G}} \left(\frac{1}{s} + \frac{1}{u} \right) \sum_{i=1}^{s+u} \sigma_i g(z_i) \right].$$

Moreover, we define $\widetilde{\mathcal{R}}_{s+u}(\mathcal{G}) := \widetilde{\mathcal{R}}_{s+u}(\mathcal{G}, p_0)$ with $p_0 = \frac{su}{(s+u)^2}$.

The next result provides a transductive generalization bound that we will invoke to control test error on the unlabeled subset.

Lemma B.9 (Corollary 1 in [13])**.** Let $\mathcal{H}$ be a hypothesis class of functions from $\mathcal{X} \to \mathcal{Y}$, and $\ell : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \to [0, \infty)$ be a loss function. and $S = \{(z_i)\}_{i=1}^{s+u} \subseteq \mathcal{Z}$ a set of samples with elements in $\mathcal{Z} := \mathcal{X} \times \mathcal{Y}$. Let $c_0 := \sqrt{\frac{32 \log(4e)}{3}} < 5.05$. Let $P := \frac{1}{s} + \frac{1}{u}$, and $Q := \frac{s+u}{(s+u-1/2)(1-1/(2\max\{s,u\}))}$. Then, for any $\delta \in (0, 1)$, with probability at least $1 - \delta$ over the random choice of s training samples from S, for any $h \in \mathcal{H}$, it holds that

$$\mathcal{L}_s(f) \leq \widetilde{\mathcal{L}}_u(f) + \widetilde{R}_{s+u}(\ell \circ \mathcal{H}) + c_0 P \sqrt{\min\{s, u\}} + \sqrt{\frac{PQ}{2} \log \frac{1}{\delta}}.$$

Finally, we relate the transductive complexity to its standard inductive counterpart, which will let us reuse familiar bounds.

Lemma B.10 (Lemma B.8 in [18])**.** Let $\mathcal{G}$ be a family of functions mapping from $\mathcal{Z}$ to $\mathbb{R}$ and $S = \{z_i\}_{i=1}^{s+u} \subseteq \mathcal{Z}$ a set of samples with elements in $\mathcal{Z}$. Let $n := s + u$. Then we have

$$\widetilde{\mathcal{R}}_{s+u}(\mathcal{G}) \leq \mathcal{R}_n(\mathcal{G}).$$

C Well-posedness and Convergence Analaysis

Definition C.1 (Admissible hypergraph)**.** We say that a hypergraph is admissible if each hyperedge is associated with a non-negative weight, and each node has a positive degree.

Lemma C.2. Let $\mathbf{M}$ be a hypergraph Laplacian matrix of an admissible hypergraph. Then each eigenvalue λ_i of $\mathbf{M}$ satisfies $|\lambda_i| \leq 1$. Moreover, we have $\lambda_{\max}(\mathbf{M}) = 1$.

Proof. We define the matrix

$$\mathbf{P} := \mathbf{D}^{-1}\mathbf{H}\mathbf{E}\mathbf{B}^{-1}\mathbf{H}^\top.$$

Then we can write the normalized hypergraph Laplacian matrix as

$$\mathbf{M} = \mathbf{D}^{1/2}\mathbf{P}\mathbf{D}^{-1/2}.$$

For every $i \in [n]$, by simple calculation, we have

$$\begin{aligned}
\sum_{k=1}^{n} \mathbf{P}_{ik} &= \frac{1}{\mathbf{D}_{i,i}} \sum_{j=1}^{m} \mathbf{H}_{i,j} w_j \cdot \frac{1}{\mathbf{B}_{j,j}} \sum_{k=1}^{m} \mathbf{H}^\top_{j,k} && \text{(By definitions of } \mathbf{D}, \mathbf{H}, \mathbf{B}, \mathbf{E}) \\
&= \frac{1}{\mathbf{D}_{i,i}} \sum_{j=1}^{m} \mathbf{H}_{i,j} w_j \cdot \frac{1}{\mathbf{B}_{j,j}} \sum_{k=1}^{m} \mathbf{H}_{k,j} && (\mathbf{H}^\top_{j,k} = \mathbf{H}_{k,j}) \\
&= \frac{1}{\mathbf{D}_{i,i}} \sum_{j=1}^{m} \mathbf{H}_{i,j} w_j && (\mathbf{B}_{j,j} = \textstyle\sum_{k=1}^{m} \mathbf{H}_{k,j}) \\
&= 1. && (\mathbf{D}_{i,i} = \textstyle\sum_{j=1}^{m} \mathbf{H}_{i,j} w_j)
\end{aligned}$$

Hence every row of $\mathbf{P}$ sums to 1 and every entry of $\mathbf{P}$ is nonnegative because the hypergraph is admissible. Since $\mathbf{M}$ and $\mathbf{P}$ have the same spectrum, it suffices to show the eigenvalues of $\mathbf{P}$ satisfies the desired properties.

We first show that every eigenvalue λ_i of $\mathbf{P}$ satisfies $|\lambda_i| \leq 1$. Note that for any $\mathbf{x} \in \mathbb{R}^n$,

$$\|\mathbf{P}\mathbf{x}\|_1 \leq \|\mathbf{P}\|_\infty \|\mathbf{x}\|_1 \leq \|\mathbf{x}\|_1. \tag{4}$$

Let $\mathbf{v}_i$ be an eigenvector associated with the eigenvalue λ_i. Note that

$$\|\mathbf{P}\mathbf{v}_i\|_1 = \|\lambda_i \mathbf{v}_i\|_1 = |\lambda_i| \|\mathbf{v}_i\|_1.$$

Then by Eq. (4), we have $|\lambda_i| \leq 1$.

To show that there exists some $\lambda_i = 1$, we see that

$$\mathbf{P}\mathbf{1}_n = \begin{bmatrix} \sum_{j=1}^{n} \mathbf{P}_{1,j} \\ \sum_{j=1}^{n} \mathbf{P}_{2,j} \\ \vdots \\ \sum_{j=1}^{n} \mathbf{P}_{m,j} \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} = \mathbf{1}_n.$$

Thus can conclude that $\lambda_{\max}(\mathbf{P}) = 1$. □

Theorem C.3 (Sufficient condition for well-posedness, restatement of Theorem 3.2)**.** Let $\mathbf{M} \in \mathbb{R}^{n \times n}$ be the normalized hypergraph Laplcaian matrix of an admissible hypergraph. Assume that $\phi : \mathbb{R} \to \mathbb{R}$ is an contractive activation function, i.e., ϕ is 1-Lipschitz. If the weight matrix $\mathbf{W} \in \mathbb{R}^{d \times d}$ satisfies $\lambda_{\max}(|\mathbf{W}|) := \kappa \in [0, 1)$ then for any $\widetilde{\mathbf{X}} \in \mathbb{R}^{d \times d}$, the fixed-point equilibrium equation

$$\mathbf{Z} = \phi\left(\mathbf{M}\mathbf{Z}\mathbf{W} + \widetilde{\mathbf{X}}\right)$$

has a unique solution $\mathbf{Z}^* \in \mathbb{R}^{n \times d}$, and the fixed point iteration

$$\mathbf{Z}^{(t+1)} = \phi\left(\mathbf{M}\mathbf{Z}^{(t)}\mathbf{W} + \widetilde{\mathbf{X}}\right)$$

converges to $\mathbf{Z}^*$ as $t \to \infty$. Futhermore, if we assume that $\|\mathbf{Z}^*\|_F \leq C_0$ for some $C_0 \geq 0$, and $\mathbf{Z}^{(1)} = \mathbf{0}_{n\times d}$, then for any integer $t \geq 1$,

$$\|\mathbf{Z}^{(t)} - \mathbf{Z}^*\|_F \leq \kappa^{t-1} C_0.$$

Proof. By Lemma B.3, we can deduce that

$$\operatorname{vec}(\mathbf{MZW}) = (\mathbf{W}^\top \otimes \mathbf{M}) \operatorname{vec}(\mathbf{Z}). \tag{5}$$

Note that we assume $\lambda_{\max}(|\mathbf{W}|) < 1$, and Lemma C.2 implies that $\lambda_{\max}(\mathbf{M}) := \kappa < 1$. By Lemma B.4, we can conclude that

$$\begin{aligned}\lambda_{\max}(|\mathbf{W}^\top \otimes \mathbf{M}|) &= \lambda_{\max}(|\mathbf{W}^\top|)\lambda_{\max}(|\mathbf{M}|) \\ &= \lambda_{\max}(|\mathbf{W}|)\lambda_{\max}(|\mathbf{M}|) \\ &= \kappa < 1.\end{aligned}$$

Let $g(\mathbf{z}) := (\mathbf{M} \otimes \mathbf{W}^\top)\mathbf{z} + \widetilde{\mathbf{X}}$. Note that $\phi : \mathbb{R} \to \mathbb{R}$ is 1-Lipschitz and g is an affine mapping. Since $\lambda_{\max}(|M \otimes \mathbf{W}^\top|) < 1$, the mapping g is contractive, i.e., for any $\mathbf{z}_1, \mathbf{z}_2 \in \mathbb{R}^{nd}$,

$$\|\phi(g(\mathbf{z}_1)) - \phi(g(\mathbf{z}_2))\|_2 \leq \kappa \|\mathbf{z}_1 - \mathbf{z}_2\|_2, \tag{6}$$

then by the well-known Banach fixed point theorem, the equation

$$\operatorname{vec}(\mathbf{Z}) = \phi\left((\mathbf{M} \otimes \mathbf{W}^\top) \operatorname{vec}(\mathbf{Z}) + \widetilde{\mathbf{X}}\right)$$

has a unique solution $\mathbf{Z}^*$, and the fixed point iteration can converges to it. By Eq. (5), we prove the first result.

Note that, by tensor trick, we have $\operatorname{vec}(\mathbf{Z}^{(t)}) = \phi(g(\operatorname{vec}(\mathbf{Z}^{(t-1)})))$ and $\operatorname{vec}(\mathbf{Z}^*) = \phi(g(\operatorname{vec}(\mathbf{Z}^*)))$. Next, by Eq. (6) and $\|\operatorname{vec}(\cdot)\|_2 = \|\cdot\|_F$, we have

$$\begin{aligned}\|\mathbf{Z}^{(t)} - \mathbf{Z}^*\|_F &= \|\operatorname{vec}(\mathbf{Z}^{(t)}) - \operatorname{vec}(\mathbf{Z}^*)\|_2 \\ &= \|\phi(g(\operatorname{vec}(\mathbf{Z}^{(t-1)}))) - \phi(g(\operatorname{vec}(\mathbf{Z}^*)))\|_2 && \text{(Simple algebra)} \\ &\leq \kappa \|\operatorname{vec}(\mathbf{Z}^{(t-1)}) \operatorname{vec}(\mathbf{Z}^*)\|_2 && \text{(Contraction of } \phi(g(\cdot))) \\ &\leq \cdots \\ &\leq \kappa^{t-1} \|\operatorname{vec}(\mathbf{Z}^{(1)}) - \operatorname{vec}(\mathbf{Z}^*)\|_2 \\ &= \kappa^{t-1} \|\operatorname{vec}(\mathbf{Z}^*)\|_2 && (\mathbf{Z}^{(1)} = \mathbf{0}_{n\times d}) \\ &= \kappa^{t-1} \|\mathbf{Z}^*\|_F && (\|\operatorname{vec}(\cdot)\|_2 = \|\cdot\|_F) \\ &\leq \kappa^{t-1} C_0. && (\|\mathbf{Z}^*\|_F \leq C_0)\end{aligned}$$

Thus we complete the proof.

□

D Oversmoothing Analysis

Theorem D.1 (Sufficient condition for nonidentical node features, restatement of Theorem 3.3)**.** Let $\mathbf{M} \in \mathbb{R}^{n\times n}$ be the normalized hypergraph Laplcaian matrix of an admissible hypergraph. Let $\phi : \mathbb{R} \to \mathbb{R}$ be a strictly increasing nonexpansive activation function. Suppose that the weight matrix $\mathbf{W} \in \mathbb{R}^{d\times d}$ of an IHGNN satisfies $\lambda_{\max}(|\mathbf{W}|) < 1$, then for any $\widetilde{\mathbf{X}} \in \mathbb{R}^{d\times d}$ satisfying $\mathbf{x}_i \neq \mathbf{x}_j$ for some $i, j \in [n]$, there does not exists $\mathbf{z}_0 \in \mathbb{R}^d$, such that $\mathbf{Z}^* = \mathbf{1}_n \mathbf{z}_0^\top$.

Proof. Assume for contradiction that there exists a vector $\mathbf{z}_0 \in \mathbb{R}^d$ such that the constant-row matrix $\mathbf{Z}^* = \mathbf{1}_n \mathbf{z}_0^\top$ is a fixed point of the implicit layer, that is

$$\mathbf{Z}^* = \phi\left(\mathbf{M}\mathbf{Z}^*\mathbf{W} + \widetilde{\mathbf{X}}\right).$$

Because $\mathbf{M}$ is the normalized hypergraph Laplacian matrix and it is row-stochastic, $\mathbf{M}\mathbf{1}_n = \mathbf{0}_n$. Hence

$$\mathbf{M}\mathbf{Z}^* = \mathbf{M}\left(\mathbf{1}_n\mathbf{z}_0^\top\right) = (\mathbf{M}\mathbf{1}_n)\mathbf{z}_0^\top = \mathbf{0}_{n\times d}.$$

Substituting this into the fixed-point equation gives, for every $i \in [n]$, $\mathbf{z}_0 = \phi\left(\widetilde{\mathbf{x}}_i\right)$, where $\widetilde{\mathbf{x}}_i^\top$ is the i-th row of $\widetilde{\mathbf{X}}$. Since ϕ is strictly increasing, it is injective, so the above equalities imply

$$\widetilde{\mathbf{x}}_i = \widetilde{\mathbf{x}}_j \quad \forall i, j \in [n].$$

This contradicts the hypothesis that there exist indices $i \neq j$ with $\widetilde{\mathbf{x}}_i \neq \widetilde{\mathbf{x}}_j$. Therefore no constant-row fixed point of the form $\mathbf{Z}^* = \mathbf{1}_n\mathbf{z}_0^\top$ can exist. □

Theorem D.2 (Expressivity of IHGNN, restatement of Theorem 3.4)**.** Let $\mathbf{M} \in \mathbb{R}^{n\times n}$ be the normalized hypergraph Laplcaian matrix of an admissible hypergraph. Let $K \in \mathcal{N}$. For every K-order polynomial filter function $p(\mathbf{X}) := (\sum_{k=0}^{K}\theta_k\mathbf{M}^k)\mathbf{X}$ with arbitrary coefficients $\{\theta_k\}_{k=0}^{K}$ and input feature matrix $\mathbf{x} \in \mathbb{R}^{n\times d}$, there exists an IHGNN with identity activation can express it.

Proof. Fix an input feature matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$ and denote $p(\mathbf{X}) = (\sum_{k=1}^{K}\theta_k\mathbf{M}^k)\mathbf{X}$. We construct an IHGNN with identity activation that produces exactly this mapping. First, we set the hidden state dimension $d_h := (K+1)d$, and we write the hidden state

$$\mathbf{Z} = \begin{bmatrix}\mathbf{Z}^{(0)} & \mathbf{Z}^{(1)} & \cdots & \mathbf{Z}^{(K)}\end{bmatrix}$$

where each $\mathbf{Z}^{(k)}$ is a block matrix of size $n \times d$. Note that here the supscript (k) denotes k-th block matrix, not the iteration number of the fixed point iteration. However, we will show that they coincides with each other. We define $\mathbf{\Theta}_1 \in \mathbb{R}^{d\times d_h}$ as

$$\mathbf{\Theta}_1 = \begin{bmatrix}\mathbf{I}_d & \mathbf{0}_{d\times d} & \cdots & \mathbf{0}_{d\times d}\end{bmatrix}$$

and $\mathbf{b} = \mathbf{0}_{d_h}$. We define the weight matrix $\mathbf{W} \in \mathbb{R}^{d_h\times d_h}$ as

$$\mathbf{W} := \begin{bmatrix}\mathbf{0}_{d\times d} & \mathbf{I}_d & \mathbf{0}_{d\times d} & \mathbf{0}_{d\times d} & \cdots & \mathbf{0}_{d\times d}\\ \mathbf{0}_{d\times d} & \mathbf{0}_{d\times d} & \mathbf{I}_d & \mathbf{0}_{d\times d} & \cdots & \mathbf{0}_{d\times d}\\ \mathbf{0}_{d\times d} & \mathbf{0}_{d\times d} & \mathbf{0}_{d\times d} & \mathbf{I}_d & \cdots & \mathbf{0}_{d\times d}\\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots\\ \mathbf{0}_{d\times d} & \mathbf{0}_{d\times d} & \mathbf{0}_{d\times d} & \mathbf{0}_{d\times d} & \cdots & \mathbf{I}_d\\ \mathbf{0}_{d\times d} & \mathbf{0}_{d\times d} & \mathbf{0}_{d\times d} & \mathbf{0}_{d\times d} & \cdots & \mathbf{0}_{d\times d}\end{bmatrix}.$$

Next, the fixed-point equilibrium equation

$$\mathbf{Z} = \phi\left(\mathbf{M}\mathbf{Z}\mathbf{W} + \mathbf{X}\mathbf{\Theta}_1 + \mathbf{b}\right)$$

can be blockwisely written as

$$\begin{cases}\mathbf{Z}^{(0)} = \mathbf{X},\\ \mathbf{Z}^{(1)} = \mathbf{M}\mathbf{Z}^{(0)} = \mathbf{M}\mathbf{X},\\ \mathbf{Z}^{(2)} = \mathbf{M}\mathbf{Z}^{(1)} = \mathbf{M}^2\mathbf{X},\\ \quad\vdots\\ \mathbf{Z}^{(K)} = \mathbf{M}\mathbf{Z}^{(K-1)} = \mathbf{M}^K\mathbf{X}.\end{cases}$$

We define $\mathbf{\Theta}_2 \in \mathbb{R}^{d_h\times d}$ as

$$\mathbf{\Theta}_2 = \begin{bmatrix}\theta_0\mathbf{I}_d & \theta_1\mathbf{I}_d & \cdots & \theta_K\mathbf{I}_d\end{bmatrix}^\top.$$

Then we can conclude that

$$\mathbf{\Theta}_2\mathbf{Z} = (\sum_{k=0}^{K}\theta_k\mathbf{M}^k)\mathbf{X}.$$

Thus we complete the proof. □

E Transductive Generalization Bound

Assumption E.1. We assume the following conditions hold.

- **Bounded input features**: The input node matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$ satisfies, for each $i \in [n]$, $\|\mathbf{x}_i\|_2 \le C_X$ for some $C_X > 0$.
- **Bounded trainable parameters**: The trainable parameters satisfies $\|\mathbf{\Theta}_1\|_F \le \rho_1, \|\mathbf{\Theta}_2\|_F \le \rho_2, \|\mathbf{b}\|_2 \le C_b$ for some $\rho_1, \rho_2, C_b > 0$, and $\|\mathbf{W}\| \le \kappa$ for some $\kappa \in [0, 1)$, and their dimensions satisfies $d = d_h = d'$.
- **Lipschitz loss**: The loss function $\ell : \mathbb{R} \times \mathbb{R} \to [0, \infty)$ is C_ℓ-Lipschitz.
- **Lipschitz activation**: The activation function $\phi : \mathbb{R} \to \mathbb{R}$ is 1-Lipschitz.

Lemma E.2. Assume that Assumption 3.5 hold. We have

$$\mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\langle \boldsymbol{\mathcal{E}}, \widetilde{\mathbf{X}}\rangle\right] \le \sqrt{n}\rho_1 C_X + \sqrt{nd}C_b.$$

where $\boldsymbol{\mathcal{E}} \in \mathbb{R}^{n\times d}$ is a random matrix, each entry of which is a Rademacher random variable.

Proof. We can show that

$$\begin{aligned}\mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\langle \boldsymbol{\mathcal{E}}, \widetilde{\mathbf{X}}\rangle\right] &= \mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\langle \boldsymbol{\mathcal{E}}, \mathbf{X}\mathbf{\Theta}_1 + \mathbf{1}_n\mathbf{b}^\top\rangle\right] \\ &= \mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\langle \boldsymbol{\mathcal{E}}, \mathbf{X}\mathbf{\Theta}_1\rangle\right] + \mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\langle \boldsymbol{\mathcal{E}}, \mathbf{1}_n\mathbf{b}^\top\rangle\right].\end{aligned}$$

We bound the two terms on the right hand side separately.

$$\begin{aligned}\mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\langle \boldsymbol{\mathcal{E}}\mathbf{X}^\top, \mathbf{\Theta}_1\rangle\right] &\le \mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\|\boldsymbol{\mathcal{E}}\mathbf{X}^\top\|_F\|\mathbf{\Theta}_1\|_F\right] \\ &= \|\mathbf{\Theta}_1\|_F \cdot \mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\|\boldsymbol{\mathcal{E}}\mathbf{X}^\top\|_F\right] \\ &\le \rho_1 \cdot \mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\|\boldsymbol{\mathcal{E}}\mathbf{X}^\top\|_F\right] \\ &\le \rho_1 \cdot \left(\mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\sum_{i=1}^n \|\mathbf{x}_i\|_2^2\right]\right)^{1/2} \\ &\le \rho_1\sqrt{n}C_X.\end{aligned}$$

Similary, we can show that

$$\begin{aligned}\mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\langle \boldsymbol{\mathcal{E}}, \mathbf{1}_n\mathbf{b}^\top\rangle\right] &= \mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\langle \mathbf{b}, \boldsymbol{\mathcal{E}}^\top\mathbf{1}_n\rangle\right] \\ &\le \|\mathbf{b}\|_2 \mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\|\boldsymbol{\mathcal{E}}^\top\mathbf{1}_n\|_2\right] \\ &\le C_b \mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\|\boldsymbol{\mathcal{E}}^\top\mathbf{1}_n\|_2\right] \\ &\le C_b \left(\mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\sum_{i=1}^n\sum_{j=1}^d |\boldsymbol{\mathcal{E}}_{i,j}|^2\right]\right)^{1/2} \\ &= C_b \cdot \sqrt{nd}.\end{aligned}$$

Hence we complete the proof. □

Lemma E.3 (Rademacher complexity of the implicit layer)**.** We define

$$\mathcal{F}_T := \{f(\cdot) : f(\widetilde{\mathbf{X}}) = \mathbf{Z}^{(t+1)} := \phi(\mathbf{M}\mathbf{Z}^{(t)}\mathbf{W} + \widetilde{\mathbf{X}}), t \in [T]\}.$$

Assume that Assumption 3.5 hold. Then we have

$$\mathcal{R}_n^{\text{coord}}(\mathcal{F}_T) \le \frac{\rho_1 C_x + \sqrt{d}C_b}{\sqrt{n}(1-\kappa)}.$$

Proof. We can show that

$$
\begin{aligned}
n\mathcal{R}_n^{\text{coord}}(\mathcal{F}_T) &= \mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\sup_{f\in\mathcal{F}_T}\langle \boldsymbol{\mathcal{E}}, f(\widetilde{\mathbf{X}})\rangle\right] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\sup_{f\in\mathcal{F}_{T-1}}\langle \boldsymbol{\mathcal{E}}, \mathbf{M}f(\widetilde{\mathbf{X}})\mathbf{W}+\widetilde{\mathbf{X}}\rangle\right] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\sup_{f\in\mathcal{F}_{T-1}}\langle \boldsymbol{\mathcal{E}}, \mathbf{M}f(\widetilde{\mathbf{X}})\mathbf{W}\rangle\right] + \mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\langle \boldsymbol{\mathcal{E}}, \widetilde{\mathbf{X}}\rangle\right] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\sup_{f\in\mathcal{F}_{T-2}}\langle \boldsymbol{\mathcal{E}}, \mathbf{M}(\mathbf{M}f(\widetilde{\mathbf{X}})\mathbf{W}+\widetilde{\mathbf{X}})\mathbf{W}\rangle\right] + \mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\langle \boldsymbol{\mathcal{E}}, \widetilde{\mathbf{X}}\rangle\right] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\sup_{f\in\mathcal{F}_{T-2}}\langle \boldsymbol{\mathcal{E}}, \mathbf{M}^2f(\widetilde{\mathbf{X}})\mathbf{W}^2\rangle\right] + \mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\langle \boldsymbol{\mathcal{E}}, \mathbf{M}\widetilde{\mathbf{X}}\mathbf{W}\rangle\right] + \mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\langle \boldsymbol{\mathcal{E}}, \widetilde{\mathbf{X}}\rangle\right] \\
&= \cdots \\
&= \mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\langle \boldsymbol{\mathcal{E}}, \mathbf{M}^{T-1}f(\widetilde{\mathbf{X}})\mathbf{W}^{T-1}\rangle\right] + \cdots + \mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\langle \boldsymbol{\mathcal{E}}, \mathbf{M}\widetilde{\mathbf{X}}\mathbf{W}\rangle\right] + \mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\langle \boldsymbol{\mathcal{E}}, \widetilde{\mathbf{X}}\rangle\right] \\
&= \sum_{t=1}^{T}\mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\langle \boldsymbol{\mathcal{E}}, \mathbf{M}^{t-1}\widetilde{\mathbf{X}}\mathbf{W}^{t-1}\rangle\right] \\
&\le \sum_{t=1}^{T}\kappa^{t-1}\mathop{\mathbb{E}}_{\boldsymbol{\mathcal{E}}}\left[\langle \boldsymbol{\mathcal{E}}, \widetilde{\mathbf{X}}\rangle\right] \\
&= \frac{1-\kappa^T}{1-\kappa}(\sqrt{n}\rho_1 C_x + \sqrt{nd}C_b) \\
&\le \frac{1}{1-\kappa}(\sqrt{n}\rho_1 C_x + \sqrt{nd}C_b).
\end{aligned}
$$

Dividing n on both sides gives

$$
\mathcal{R}_n^{\text{coord}}(\mathcal{F}_T) \le \frac{\rho_1 C_x + \sqrt{d}C_b}{\sqrt{n}(1-\kappa)}.
$$

Thus the proof is complete. □

Theorem E.4 (Transductive generalization bound of IHGNN, restatement of Theorem 3.6)**.** We assume that the hypergraph is admissible and Assumption E.1 are satisfied. Let $\mathcal{H}$ be the hypothesis class of IHGNN models defined on the given hypergraph. Let $c_0 := \sqrt{\frac{32\log(4e)}{3}} < 5.05$. Let $P := \frac{1}{s}+\frac{1}{u}$, $Q := \frac{s+u}{(s+u-1/2)(1-1/(2\max\{s,u\}))}$. Then, for any $\delta > 0$, with probability at least $1-\delta$ over the choice of the training set $\{\mathbf{x}_i\}_{i=1}^{s+u}\cup\{y_i\}_{i=1}^{u}$, for all $f\in\mathcal{H}$, we have

$$
\mathcal{L}_u(f) \le \widehat{\mathcal{L}}_s(f) + \frac{\sqrt{2}\rho_2 C_\ell(\rho_1 C_x + \sqrt{d}C_b)}{(1-\kappa)\sqrt{s+u}} + c_0 P\sqrt{\min\{s,u\}} + \sqrt{\frac{PQ}{2}\log\frac{1}{\delta}}.
$$

Proof. By Lemma E.3, Lemma B.7, Lemma B.9, and Lemma B.10, we can show that

$$
\begin{aligned}
\mathcal{L}_s(f) &\le \widetilde{\mathcal{L}}_u(f) + \widetilde{R}_{s+u}(\ell\circ\mathcal{H}) + c_0 P\sqrt{\min\{s,u\}} + \sqrt{\frac{PQ}{2}\log\frac{1}{\delta}} \\
&\le \widetilde{\mathcal{L}}_u(f) + R_{s+u}(\ell\circ\mathcal{H}) + c_0 P\sqrt{\min\{s,u\}} + \sqrt{\frac{PQ}{2}\log\frac{1}{\delta}} \\
&= \widetilde{\mathcal{L}}_u(f) + R_{s+u}(\ell\circ\phi\circ\mathcal{F}_T) + c_0 P\sqrt{\min\{s,u\}} + \sqrt{\frac{PQ}{2}\log\frac{1}{\delta}}
\end{aligned}
$$

$$\leq \widetilde{\mathcal{L}}_u(f) + \sqrt{2}\rho_2 C_\ell \cdot R^{\text{coord}}_{s+u}(\mathcal{F}_T) + c_0 P\sqrt{\min\{s,u\}} + \sqrt{\frac{PQ}{2}\log\frac{1}{\delta}}$$
$$\leq \widetilde{\mathcal{L}}_u(f) + \frac{\sqrt{2}\rho_2 C_\ell(\rho_1 C_x + \sqrt{d}C_b)}{\sqrt{s+u}(1-\kappa)} + c_0 P\sqrt{\min\{s,u\}} + \sqrt{\frac{PQ}{2}\log\frac{1}{\delta}}.$$

Thus we complete the proof. □

Corollary E.5 (Asymptotic transductive generalization bound of IHGNN, restatement of Corollary 3.7)**.** Under the same conditions in Theorem E.4. For sufficiently large training-set size s and testing-set size u, for any $\delta > 0$, with probability at least $1-\delta$ over the choice of the training set, for all $f \in \mathcal{H}$, we have

$$\mathcal{L}_u(f) \leq \widehat{\mathcal{L}}_s(f) + O\left(\frac{d}{s+u}\right)^{\frac{1}{2}} + O\left(\frac{\log(1/\delta)}{\min\{s,u\}}\right)^{\frac{1}{2}}.$$

Proof. It is not hard to see when s and u are sufficiently large, we have $Q = O(1)$.

First, it is not hard to see that

$$\frac{\sqrt{2}\rho_2 C_\ell(\rho_1 C_x + \sqrt{d}C_b)}{(1-\kappa)\sqrt{s+u}} = O\left(\sqrt{\frac{d}{s+u}}\right).$$

Next, we can show that

$$\begin{aligned} c_0 P\sqrt{\min\{s,u\}} + \sqrt{\frac{PQ}{2}\log\frac{1}{\delta}} &= c_0\left(\frac{1}{s}+\frac{1}{u}\right)\sqrt{\min\{s,u\}} + O\left(\sqrt{\left(\frac{1}{s}+\frac{1}{u}\right)\log\frac{1}{\delta}}\right) \\ &= O\left(\frac{1}{\sqrt{s}}+\frac{1}{\sqrt{u}}\right) + O\left(\sqrt{\left(\frac{1}{s}+\frac{1}{u}\right)\log\frac{1}{\delta}}\right) \\ &= O\left(\sqrt{\left(\frac{1}{s}+\frac{1}{u}\right)\log\frac{1}{\delta}}\right). \end{aligned}$$

Thus, by Theorem E.4, we complete the proof. □

F Training of IHGNN

Theorem F.1 (Scaled well-posedness of IHGNN, restatement of Theorem 3.8)**.** Suppose that the activation function $\phi : \mathbb{R} \to \mathbb{R}$ is positively homogeneous and nonexpansive. If an IHGNN model with weights $\mathbf{W}, \mathbf{\Theta}_1, \mathbf{\Theta}_2, \mathbf{b}$ satisfies $\lambda_{\max}(|\mathbf{W}|) < 1$, then there exists an equivalent IHGNN model with weights $\widetilde{\mathbf{W}}, \widetilde{\mathbf{\Theta}}_1, \widetilde{\mathbf{\Theta}}_2, \widetilde{\mathbf{b}}$ such that $\|\widetilde{\mathbf{W}}\|_\infty < 1$, and both models produce identical outputs for all same inputs.

Proof. Let $\alpha \in (0,1)$ be a scaling factor such that $\widetilde{\mathbf{W}} := \alpha\mathbf{W}$ satisfies

$$\|\widetilde{\mathbf{W}}\|_\infty = \alpha\|\mathbf{W}\|_\infty < \alpha.$$

We define $\widetilde{\mathbf{\Theta}_1} := \alpha\mathbf{\Theta}_1$, and $\widetilde{\mathbf{b}} := \alpha\mathbf{b}$. Then it is not hard to see that for any $\mathbf{X} \in \mathbb{R}^{n\times d}$, the unique solution $\widetilde{\mathbf{Z}}^*$ of the equation

$$\widetilde{\mathbf{Z}} = \phi\left(\mathbf{M}\widetilde{\mathbf{Z}}\widetilde{\mathbf{W}} + \mathbf{X}\widetilde{\mathbf{\Theta}}_1 + \widetilde{\mathbf{b}}\right)$$

satisfies $\widetilde{\mathbf{Z}}^* = \alpha\mathbf{Z}^*$, where $\mathbf{Z}^*$ is the fixed point solution of $\mathbf{Z} = \phi\left(\mathbf{M}\mathbf{Z}\mathbf{W} + \mathbf{X}\mathbf{\Theta}_1 + \mathbf{b}\right)$. Hence, by defining $\widetilde{\mathbf{\Theta}}_2 := \alpha^{-1}\mathbf{\Theta}_2$, we complete the proof. □