A2I: Adjacency-to-Image Structural Encodings for Graph Learning

Heesoo Jung **Hogun Park***
Department of Computer Science and Engineering
Sungkyunkwan University
Suwon, South Korea 16419
{steve305, hogunpark}@skku.edu

Abstract

Graph representation learning often relies on message passing or spectral/positional encodings to summarize graph structure, but these indirect summaries can collapse structurally distinct graphs, including non-isomorphic cospectral pairs. We propose A2I, a structural encoding framework that renders BFS–degree-ordered local adjacency patterns as fixed-resolution images and embeds them with a frozen pre-trained vision encoder. The resulting structural tokens are mapped to learnable prototypes and aggregated with node embeddings via a lightweight Transformer. We provide a conditional analysis with two related findings: a stability result, showing that two orderings of the same graph yield embeddings that converge under bounded-degree assumptions, and a separability result, showing that two distinct graphs yield embeddings whose gap grows at least linearly with the difference in their BFS profiles. Empirically, A2I is competitive with recent GNN, Graph Transformer, and structural-encoding baselines on six node-classification benchmarks, with pronounced gains for GCN-based models on heterophilous graphs and under partially missing features. A2I highlights the potential of image-based structural encodings as a generalizable, structure-aware foundation for future graph models.

1 Introduction

Many real-world systems can be represented as graphs, where nodes denote entities and edges capture their relationships, spanning domains such as social networks [48], biological systems [32], and knowledge graphs [42]. Since the defining characteristic of the graph data lies in its underlying structure, as encoded by the adjacency matrix, prior work has focused on encoding structural information [12, 17]. Broadly, these approaches fall into two major families: *structure-aware architectures* [26, 12] and *explicit structural encoding methods* [43, 16].

Structure-aware architectures can be further divided into walk-based [11] and aggregation-based paradigms. Walk-based methods, e.g., DeepWalk [26], capture global connectivity through sampled random walks but can be limited by the walk statistics they preserve, causing them to conflate some non-isomorphic cospectral graphs. On the other hand, aggregation-based methods [6, 8] integrate attributes with topology but face different structural hurdles: local aggregation-driven approaches, such as GNNs [17, 19], are theoretically constrained by the 1-WL test [37], while global aggregation-driven approaches, e.g., Graph Transformers [44], often rely on global node-feature attention and can lack topological awareness unless explicit structural encoding methods are provided.

Explicit structural encoding methods [43] inject topology-dependent embeddings directly into structure-aware architectures to enrich node representations. These approaches, e.g., RWSE [28] or

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

LPE [9], attempt to address such limitations by injecting topological signals, yet they do so through statistical summaries or spectral decompositions, which may fail to distinguish cospectral graphs and incur computational overhead. Furthermore, recent work [3] suggests that simply injecting these embeddings is insufficient and may lead to feature interference.

Therefore, we ask:

> *"Can structural embeddings be derived directly from local adjacency patterns, without passing through spectral or walk-statistic summaries? And can such embeddings be integrated with node features to avoid unnecessary interference?"*

We address both questions with our approach, A2I (Adjacency to Image encoding). Rather than relying on aggregated messages or spectral summaries, A2I directly represents local structure by sorting each target node's k-hop adjacency matrix by BFS distance and degree, rendering it as a fixed-resolution image, and embedding it with a frozen pre-trained vision encoder. The pipeline directly encodes local adjacency information as images, while A2I demonstrates scalability and permutation robustness under a bounded-degree assumption. Moreover, A2I is motivated by graphon theory, where resized adjacency images offer a graphon-inspired view that can yield similar structural embeddings for graphs from related graphon families. The Transformer-based module integrates structural embeddings with node-feature embeddings through gated fusion, mitigating distortion of the feature embeddings. We evaluate A2I on six real-world node classification datasets spanning homophilous, heterophilous, and large-scale graphs. We also show that the structural embeddings of A2I transfer to unseen graphs in few-shot scenarios. Finally, A2I separates representative cospectral pairs that spectral and aggregation-based encodings collapse into identical representations.

We summarize our contributions as follows:

- We propose A2I, a structural encoding framework that renders local adjacency matrices as images and embeds them via a vision encoder. The embeddings are stable under permutation, converge to a graphon limit, and separate cospectral graphs.
- We design a Transformer with a prototype bottleneck that integrates structural and feature representations, and show that this integration rescales rather than distorts the original feature attention.
- On six node classification benchmarks, A2I is competitive with and often improves over recent baselines, with robustness under partial feature masking and few-shot transfer across graphs of different sizes drawn from the same graphon family, as well as empirical separation of representative 1-WL-indistinguishable and cospectral graph pairs.

2 Related Work

2.1 Structure-aware architecture methods

Walk-based methods. DeepWalk [26] pioneered this line of work by leveraging random walks and the Skip-Gram model to learn node representations that reflect network structure. node2vec [11] extends this framework by introducing a biased random-walk process that interpolates between BFS and DFS, thereby capturing flexible structural representations.

Aggregation-based methods. Graph Neural Networks (GNNs) [17, 12] have emerged as the dominant paradigm for graph representation learning [50, 23]. These models iteratively update node representations by aggregating information from neighboring nodes, thereby integrating both structural and attribute information. To enhance structural expressivity, subgraph-based methods extract and process multiple local subgraphs, but often incur substantial computational overhead. Recently, Graph Transformers have extended this paradigm by performing a form of global aggregation [44].

2.2 Explicit structural encoding methods

A parallel line of work injects explicit structural signals. These approaches generally fall into two categories: spectral and spatial or statistical encoding methods. Spectral methods, such as Laplacian Positional Encodings (LPE) [9, 16], leverage the eigenvectors of the graph Laplacian, enabling the model to distinguish nodes based on their spectral properties. Conversely, spatial and

statistical methods, including Random Walk Structural Encodings (RWSE) [28] and shortest-path distances [44], encode local connectivity patterns and relative structural roles. Despite their effectiveness, low-dimensional summaries of structural information can collapse to identical representations on structurally distinct graphs, especially when graphs are cospectral or share similar walk statistics. A2I departs from this paradigm by rendering local adjacency patterns as fixed-resolution images and using them as direct structural tokens. Unlike PATCHY-SAN [22], whose CNN processes structure and node attributes jointly and is thus closer to aggregation-based methods, A2I derives these tokens from adjacency alone, independent of node features.

3 Preliminaries

3.1 Notations

Let $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ be a graph, where $\mathcal{V}$ is the node set and $\mathcal{E}$ is an edge set. $\mathbf{X} \in \mathbb{R}^{N \times D_{node}}$ represents the node feature matrix. N represents the total number of nodes in the graph. $\mathbf{A} \in \{0,1\}^{N \times N}$ is an adjacency matrix where $\mathbf{A}[i,j] = 1$ when v_i and v_j are connected by an edge, and 0 otherwise. For node v_i, the k-hop subgraph centered at v_i, $\mathcal{G}_{v_i}^k = \{\mathcal{V}_{v_i}^k, \mathcal{E}_{v_i}^k\}$, is defined as follows:

$$\begin{aligned} \mathcal{V}_{v_i}^k &= \{v \in \mathcal{V} \mid \text{dist}(v, v_i) \leq k\}, \\ \mathcal{E}_{v_i}^k &= \{(v_1, v_2) \in \mathcal{E} \mid v_1, v_2 \in \mathcal{V}_{v_i}^k\}, \end{aligned} \tag{1}$$

where $\text{dist}(v, v_i)$ denotes the shortest distance between node v and v_i. A2I extracts the multi-scale subgraphs ranging from 1 to K hops, which are subsequently encoded into representations using a frozen pre-trained vision encoder f_{vision} (see Section 4.1 for details).

3.2 Graph Neural Networks

Graph Neural Networks (GNNs) [17] update the representation of each node by aggregating information from its neighboring nodes, which can be expressed as:

$$\begin{aligned} \mathbf{h}_i^{l+1} &= f_{\text{gnn}}(\mathbf{h}_i^l, \mathcal{E}_{v_i}^1) \\ &= \text{UPDATE}\big(\mathbf{h}_i^l, \text{AGGREGATE}\big(\{\mathbf{h}_j^l \mid (i,j) \in \mathcal{E}_{v_i}^1\}\big)\big), \end{aligned} \tag{2}$$

where f_{gnn} is composed of UPDATE and AGGREGATE functions. $\mathbf{h}_i^l$ is the representation of the node i after l GNN layers. The UPDATE function updates representations using both aggregated and original messages, while AGGREGATE summarizes messages from neighboring nodes. By focusing predominantly on immediate neighbors, GNNs are limited in their ability to represent the full local structural context [41, 46].

4 Methodology

Figure 1 illustrates the overall flow of A2I. A preprocessing step transforms the multi-scale local adjacency patterns from 1-hop to K-hop neighborhoods into image representations, which are then encoded using a frozen pre-trained vision encoder f_{vision}. A representation integration step combines the resulting structural tokens with aggregation-based node embeddings. In the following section, we describe the preprocessing procedure and the representation integration step in detail.

4.1 Preprocessing

We first explain how the preprocessing step works in an idealized and naïve approach. We then analyze both the effectiveness and its associated computational bottleneck. Finally, we introduce our method, which mitigates the bottleneck while retaining the intended structural information.

4.1.1 Ideal and naïve approach

For each node v_i in the input graph, the preprocessing step extracts multi-scale subgraphs ranging from 1 to K hops ($\mathcal{G}_{v_i}^1, \mathcal{G}_{v_i}^2, \ldots, \mathcal{G}_{v_i}^K$), transforms each subgraph into a fixed-size image, and applies a pre-trained vision encoder to generate the structural embedding. However, this direct transformation

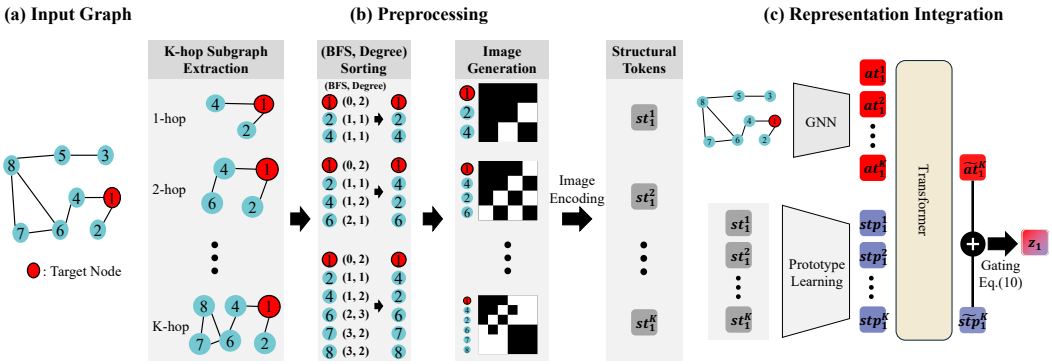


Figure 1: Overview of the A2I framework for target node representation.

is challenging for two primary reasons. The first issue is that of permutation equivariance: while the adjacency matrix is permutation-equivariant, the corresponding image representation is not. The second is the variability in subgraph size, as higher-hop subgraphs typically include more nodes.

The first challenge arises from the fact that multiple node orderings of a graph may induce different adjacency matrix representations. While this is harmless for permutation-equivariant graph methods, it becomes a problem once the matrix is rendered as an image representation. We therefore apply a sorting method to every adjacency matrix before rendering, so that arbitrary initial orderings of the same graph are mapped to a nearly common representation. We first introduce *admissible canonical orderings*: orderings under which two graphs share the same ordered adjacency matrix if and only if they are isomorphic. Further details on Definition 1 are provided in Appendix A.2.

Definition 1 (Admissible Canonical Ordering)**.** *Let $\mathbb{G}$ be the space of all graphs. For $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ with $|\mathcal{V}| = N$ and adjacency matrix $\mathbf{A} \in \{0,1\}^{N\times N}$, an* ordering function $\sigma : \mathbb{G} \to \mathcal{S}_N$ *(where $\mathcal{S}_N$ denotes the symmetric group of permutations of N elements) produces an ordered adjacency matrix*

$$\mathbf{A}^{\sigma}_{\mathcal{G}} = \mathbf{P}_{\sigma}\, \mathbf{A}\, \mathbf{P}^{T}_{\sigma}, \tag{3}$$

where $\mathbf{P}_{\sigma}$ is the permutation matrix induced by $\sigma(\mathcal{G})$. σ is canonical *if it is invariant under relabeling: for all N, all $\mathcal{G} \in \mathbb{G}$ and all $\beta \in \mathcal{S}_N$,*

$$\mathbf{A}^{\sigma}_{\beta\cdot\mathcal{G}} \;=\; \mathbf{A}^{\sigma}_{\mathcal{G}}. \tag{4}$$

We denote the (non-empty) class of such orderings by Σ^*, and write $\mathbf{A}^{*}_{\mathcal{G}} := \mathbf{A}^{\sigma}_{\mathcal{G}}$ for an arbitrary $\sigma \in \Sigma^*$ when the choice is immaterial.

For the second challenge, we resize each (variable-size) sorted adjacency matrix to a fixed $M \times M$ representation by block-averaging:

$$\mathbf{A}_{\text{resized}}[i,j] = \frac{1}{|\mathcal{B}_{i,j}|} \sum_{(u,v)\in\mathcal{B}_{i,j}} \mathbf{A}^{*}[u,v], \tag{5}$$

where $\{\mathcal{B}_{i,j}\}$ denotes the collection of index blocks induced by the resizing operator.

We then treat each resized matrix as an image-like input and feed it to a pre-trained vision encoder f_{vision}, yielding K structural tokens per node v_i:

$$\mathbf{st}^{k}_{v_i} = f_{\text{vision}}\big(\text{RGB}(\mathcal{T}(\mathbf{A}^{*}_{\mathcal{G}^{k}_{v_i}}))\big), \quad k = 1, \ldots, K, \tag{6}$$

where $\mathbf{st}^{k}_{v_i} \in \mathbb{R}^{F}$ encodes the k-hop subgraph $\mathcal{G}^{k}_{v_i}$ and F is the token dimension, and $\mathcal{T}$ denotes the resizing operator. $\text{RGB}(\cdot)$ replicates the single-channel adjacency map across RGB channels.

Limitations. The ideal yet naïve approach distinguishes graphs by transforming multi-scale subgraphs into canonically sorted representations. However, **1)** designing an admissible canonical sorting at scale remains challenging, and **2)** the associated preprocessing can be computationally expensive. The naïve pipeline consists of: *extracting k-hop subgraphs $\to$ densifying into adjacency matrices $\to$ generating fixed-size images*, followed by a pre-trained vision encoder. The corresponding

computational complexities for these preprocessing stages are:

$$\mathcal{O}\left(\sum_{v_i} |\mathcal{E}^k_{v_i}|\right) \to \mathcal{O}\left(\sum_{v_i} \left((|\mathcal{V}^k_{v_i}|)^2 + |\mathcal{E}^k_{v_i}|\right)\right) \to \mathcal{O}(NM^2). \tag{7}$$

Accordingly, the preprocessing strategy of A2I aims to approximate admissible canonical sorting while mitigating the computational bottleneck on *densifying into adjacency matrices*.

4.1.2 A scalable approach for preprocessing

An admissible canonical ordering is, in general, as hard to compute as graph isomorphism itself, and no polynomial-time construction is known. We therefore do not aim to realize the ordering function σ exactly; rather, we use Definition 1 as a reference point and design a tractable sorting method with bounded instability under different initial node orderings, as made precise by Theorem 1 and Proposition 1.

To ensure a consistent structural representation, we establish an ordering based on the node's proximity to the target node v_i and its structural importance using **BFS distance** and **degree**. We compute the degree of node i as d_i and the BFS distance as b_i to the target node. We show that the embedding discrepancy between two orderings of the same subgraph is bounded and decreases as the subgraph grows.

Theorem 1 (Embedding Error Bound)**.** *Let $\mathbf{A}_1, \mathbf{A}_2 \in \{0,1\}^{N_k \times N_k}$ be the same k-hop subgraph with N_k nodes with two different initial orderings, drawn from a graph family of average degree $\bar{d}$ bounded independently of N_k. Let $\mathcal{T} : \mathbb{R}^{N_k \times N_k} \to \mathbb{R}^{M \times M}$ be the block-average resizing operator at fixed resolution M, and let $\Phi : \mathbb{R}^{M \times M} \to \mathbb{R}^F$ be L_Φ-Lipschitz. Then,*

$$\left\|\Phi(\mathcal{T}(\mathbf{A}_1^{\mathrm{BFS}})) - \Phi(\mathcal{T}(\mathbf{A}_2^{\mathrm{BFS}}))\right\|_2 \;\le\; 2L_\Phi\, M\sqrt{2\bar{d}} \cdot \frac{1}{\sqrt{N_k}}. \tag{8}$$

This indicates that the embedding discrepancy between two initial orderings of the same subgraph is bounded and decreases with graph size; the bound is informative mainly for $N_k \geq M$, since $\mathcal{T}$ is expansive for $N_k < M$ (Remark 1, Appendix A.3). Theorem 1 holds for any pair of orderings; the sense in which BFS+degree sorting ais a reasonable surrogate for an admissible canonical ordering follows from a sorting-specific bound (Proposition 1, Appendix A.3). The proof of Theorem 1 is provided in Appendix A.3.

To address the second computational limitation, we simplify the pipeline to: *extracting k-hop subgraphs → generating fixed-size images*, followed by a pre-trained vision encoder. Removing the *densifying into adjacency matrices* step substantially reduces preprocessing cost. Specifically, from Eq. (5), the correspondence between sorted subgraph node indices and image pixels (i, j) can be computed directly. This direct mapping enables the construction of fixed-size image representations without explicitly forming dense adjacency matrices. Figure 2 compares the preprocessing time of the naïve approach with that of A2I, showing a substantial reduction in preprocessing cost across all datasets.

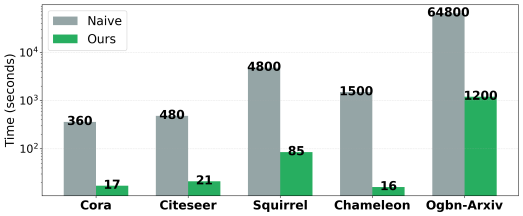


Figure 2: Time comparison of the pre-processing step in log-scale between ours and the naive approach.

4.2 Representation integration

We integrate the structural tokens with node-feature embeddings via a Transformer with a prototype-based bottleneck on the structural side, which improves generalization.

Structural prototype tokens. We maintain P learnable prototypes $\{\mathbf{pro}_1, \ldots, \mathbf{pro}_P\}$ that capture recurring topological patterns. For each k-hop structural token $\mathbf{st}^k_{v_i}$, we compute soft assignments $\boldsymbol{\alpha}^k_i \in \Delta^{P-1}$ over prototypes and form the *structural prototype token*

$$\mathbf{stp}^k_i = \sum_{j=1}^{P} \alpha^k_{ij}\, \mathbf{pro}_j, \qquad \boldsymbol{\alpha}^k_i = \mathrm{softmax}\big(\mathbf{st}^k_{v_i}\, \mathbf{W}^k\big), \tag{9}$$

with $\mathbf{W}^k \in \mathbb{R}^{F\times P}$. Since *structural prototype tokens* are derived from pure topological patterns, these tokens can be reused to transfer structural information between disparate graphs.

Aggregation tokens. In parallel, we define $K+1$ *aggregation tokens* $\mathbf{at}_i^k := \mathbf{h}_i^k$ from a standard message-passing backbone (Eq. (2)) [38, 17, 35, 12, 6], where $\mathbf{at}_i^0$ is the initial feature.

Token fusion. The two token streams are concatenated and passed through a Transformer, and the final representation $\mathbf{z}_i$ is obtained by gated fusion of the K-th output from each stream:

$$\mathbf{z}_i = (1-\mathbf{g}_i)\odot \tilde{\mathbf{stp}}_i^K + \mathbf{g}_i \odot \tilde{\mathbf{at}}_i^K, \tag{10}$$

where $\tilde{\mathbf{stp}}_i^k, \tilde{\mathbf{at}}_i^k$ are the Transformer outputs and $\mathbf{g}_i \in [0,1]^F$ is a learned gate. Full Transformer and gating details are given in Appendix A.1; extended comparisons with graph Transformer and prototype-based methods are discussed in Appendix A.8.

Training. Given labeled nodes $\mathcal{V}_L$ with C classes, we minimize the standard cross-entropy

$$\mathcal{L}_{\text{cls}} = -\frac{1}{|\mathcal{V}_L|}\sum_{v_i\in\mathcal{V}_L}\sum_{c=1}^{C} y_{ic}\log\big[\text{softmax}(\mathbf{W}_{cls}\mathbf{z}_i + \mathbf{b}_{cls})\big]_c. \tag{11}$$

5 Theoretical Analysis

Theorem 1 establishes that A2I is robust to input-order perturbations under bounded-degree assumptions. Robustness alone, however, is not sufficient: a useful structural embedding should also (i) assign distinct representations to non-isomorphic graphs, and (ii) preserve relative similarity, placing similar graphs closer in embedding space than dissimilar ones. We show next that A2I satisfies both properties. We also show that the Transformer-based integration does not distort the original feature representation.

Corollary 1 (Discrimination of distinct graphs)**.** *Under the assumptions of Theorem 1 and the alignment condition of Lemma 1 (Appendix A.4), suppose further that Φ is bi-Lipschitz on $\mathcal{T}(\mathcal{A})$ with lower constant $\ell_\Phi > 0$, where $\mathcal{A}$ is the set of BFS+degree-ordered adjacency matrices in the graph family. Let $\pi(\mathcal{G}) \in \mathbb{R}^{D\times D}$ denote the BFS profile of $\mathcal{G}$ (Definition 2 in Appendix A.4). Then for any two graphs $\mathcal{G}_1, \mathcal{G}_2$ with BFS depth at most M,*

$$\big\|\Phi(\mathcal{T}(\text{BFS}(\mathcal{G}_1))) - \Phi(\mathcal{T}(\text{BFS}(\mathcal{G}_2)))\big\|_2 \;\geq\; \ell_\Phi\, c\, \big\|\pi(\mathcal{G}_1) - \pi(\mathcal{G}_2)\big\|_F, \tag{12}$$

for an explicit constant $c > 0$ depending on BFS-level sizes. In particular, if $\mathcal{G}_1, \mathcal{G}_2$ are cospectral but $\pi(\mathcal{G}_1) \neq \pi(\mathcal{G}_2)$, their embeddings are strictly separated, whereas purely spectrum-based embeddings that depend only on eigenvalues assign them identical representations.

Corollary 1 shows that A2I distinguishes graph pairs whose BFS profiles differ. Details of the Corollary are discussed in Appendix A.4.

Theorem 2 (Adjacency Matrix Interpolation as Graphon Approximation)**.** *Let $\mathbf{A}_{resized} \in [0,1]^{M\times M}$ denote the resized matrix of a k-hop subgraph. Based on Eq.* (5)*, define a mapping*

$$f_M : [0,1]^2 \to [0,1], \quad f_M\Big(\frac{u}{M}, \frac{v}{M}\Big) = \mathbf{A}_{resized}[\lfloor u\rfloor, \lfloor v\rfloor], \tag{13}$$

where $u, v \in (0, M)$. Then, in the limit as $M \to \infty$, f_M converges (almost everywhere) to a measurable function

$$W(x,y) = \lim_{M\to\infty} f_M(x,y), \quad x,y \in [0,1], \tag{14}$$

which is a graphon*, representing the continuous structural pattern of the subgraph [18].*

As suggested by Theorem 2, transforming adjacency matrices into images is theoretically linked to graphon theory, with a proof sketch provided in Appendix A.6. We discuss the related work with graphons in Appendix A.8.1. Motivated by Theorem 2, which suggests that graphs from the same graphon family yield similar values, we empirically examine whether increasing M produces similar representations for cycle graphs of different sizes. Figure 3 demonstrates that as M gets larger, the difference between the two cycle graphs becomes marginal, which is consistent with the proposed graphon-motivated interpretation.

Effectiveness of representation integration. A2I utilizes the Transformer-based module to integrate *structural prototype tokens* and *aggregation tokens*. The main goal of applying the Transformer is to ensure that feature relationships are not unnecessarily distorted when the structural tokens are incorporated. The attention weight, with and without structural token integration, is formulated as:

$$with_att_i^{xy} = att_i^{xy}(1 - \sum_{k=1}^{K} with_att_i^{x(\mathbf{stp}_i^k)}), \tag{15}$$

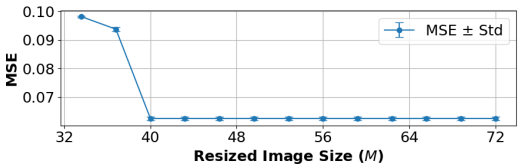


Figure 3: MSE between graphs from the same graphon family showing the preprocessing of A2I aligns with Theorem 2.

where $with_att_i^{xy}$ is the attention weight between x and y *aggregation tokens* with the addition of *structural prototype tokens*, att_i^{xy} is the attention weight without the addition of *structural prototype tokens*. The derivation of this equation is provided in Appendix A.7. From the formulation, two key observations can be made. First, the term $(1 - \sum_{k=1}^{K} with_att_i^{x(\mathbf{stp}_i^k)})$ is a scalar factor, indicating that the incorporation of structural prototype tokens does not substantially distort the original attention weight att_i^{xy}. Second, the incorporation of structural information leads to a redistribution of attention weights, thereby modulating the relative importance of different feature interactions. Taken together, these observations suggest that a Transformer architecture effectively integrates structural prototype tokens with minimal interference to their aggregation tokens.

6 Experiments

We evaluate A2I on node classification tasks and a synthetic graph classification task with respect to answering the following research questions:

- **Effectiveness (RQ1):** How does the proposed method perform on real-world node classification benchmarks, and can it maintain this effectiveness via structural compensation when node features are partially missing?
- **Transferability (RQ2):** Since A2I disentangles structural and feature representations, can it transfer structural knowledge across graphs?
- **Expressiveness (RQ3):** Can A2I distinguish graph structures that are indistinguishable by the 1-WL test, such as regular graphs (CSL) and cospectral graph pairs?
- **Ablation & Sensitivity (RQ4):** How does each component of the proposed architecture contribute to the overall performance? (Discussed in Appendix A.11.)

6.1 Experimental setup

We conduct experiments on well-known benchmark datasets for node classification, including homophilous network datasets (Cora, Citeseer) [30], heterophilous datasets (Squirrel, Chameleon, Roman-Empire) [25, 27], and one large-scale dataset (Ogbn-Arxiv) [15]. Additionally, we use PubMed [30] and the Amazon purchase networks (Photo, Computers) [31] for cross-dataset transfer.

Thirteen baselines are compared with A2I for node classification, which are: GCN [17], GAT [35], GraphSAGE [12], GPRGNN [5], Polynormer [6], SGFormer [39], PaCEr [43], CoED [23], GPM [36], and subgraph-based baselines: ID-GNN [45], Nested GNN [47], GNN-AK [51], and ESAN [2]. Additionally, we integrate A2I into two representative backbone architectures, GCN for GNN-based models, and Polynormer for graph transformer-based approaches, to evaluate its effectiveness across distinct modeling paradigms. For these backbones, we compare A2I with representative positional and structural encoding methods, such as LPE [9], RWSE [28], and LLPE [16]. For LLPE, we applied 16 eigenvectors with the largest eigenvalues and 16 eigenvectors with the smallest eigenvalues. We employ ResNet-50 [13] as f_{vision} and keep its parameters fixed during training. Further details on the baselines, hyperparameters, and datasets are provided in Appendices A.9 and A.10. The implementation of our method is available at the following URL[2].

[2] `https://github.com/steve30572/adj2img_NeurIPS2026`

Table 1: Overall node classification performance (%) comparison against (1) GNN/Transformer baselines and (2) structural encoding variants. **Bold** and underlined denote the best and second-best results. OOM: Out of Memory. Table 9 in the Appendix provides results with standard deviations.

Method	Cora		Citeseer		Chameleon		Squirrel		Roman-Empire		Ogbn-Arxiv	
	Full	Miss.	Full	Miss.	Full	Miss.	Full	Miss.	Full	Miss.	Full	Miss.
General Baselines												
GAT	78.6	76.5	69.8	66.7	70.9	71.7	56.1	54.6	59.0	56.1	68.1	67.0
GraphSAGE	81.1	79.5	71.5	68.7	68.5	65.4	46.6	42.9	79.5	76.2	69.2	67.0
GPRGNN	83.2	77.5	71.3	67.2	66.3	58.4	53.4	43.6	71.0	64.5	67.6	64.1
SGFormer	81.2	78.5	69.6	66.3	64.6	63.0	40.3	36.5	65.7	63.1	72.6	71.5
PaCEr	79.3	78.0	69.4	67.0	70.0	70.0	50.4	51.5	64.9	62.8	64.4	63.0
CoED	82.2	78.5	66.6	65.3	78.7	78.1	**65.3**	63.3	81.5	79.1	OOM	OOM
GPM	78.8	78.0	64.4	63.2	55.2	55.0	37.8	36.7	48.3	48.0	71.9	71.5
Subgraph-based Baselines												
ID-GNN-Fast	81.6	80.7	68.0	67.6	70.2	68.8	48.3	46.5	32.8	32.2	OOM	OOM
Nested GNN	77.9	76.1	66.6	66.4	67.9	63.5	OOM	OOM	57.8	52.5	OOM	OOM
GNN-AK	66.6	63.1	60.4	57.8	61.4	59.5	OOM	OOM	80.2	81.1	OOM	OOM
ESAN	50.5	51.6	50.3	45.2	OOM	OOM	OOM	OOM	63.0	61.6	OOM	OOM
Positional/Structural Encoding (Polynormer Backbone)												
Polynormer-Base	80.8	78.6	68.1	65.9	71.6	68.9	61.2	59.8	**91.9**	90.8	72.5	71.7
Polynormer+RWSE	81.0	78.8	67.3	66.0	70.4	67.6	60.7	58.8	91.8	90.9	OOM	OOM
Polynormer+LPE	81.1	79.2	68.9	66.2	70.8	67.3	60.5	58.8	**91.9**	90.9	71.9	71.4
Polynormer+LLPE	80.4	78.7	67.7	65.7	72.1	67.4	59.8	59.5	90.9	90.6	OOM	OOM
Polynormer+A2I	82.4	81.0	70.5	67.6	72.4	75.1	61.5	61.5	91.8	**91.0**	**72.8**	**72.5**
Positional/Structural Encoding (GCN Backbone)												
GCN-Base	80.5	79.7	68.5	66.9	63.7	64.5	51.2	43.5	51.2	46.8	70.3	66.8
GCN+RWSE	81.8	80.2	70.1	69.1	71.3	71.1	51.2	48.5	51.4	47.2	OOM	OOM
GCN+LPE	81.6	80.3	69.2	69.0	71.5	70.5	50.9	48.6	51.1	46.7	67.8	66.7
GCN+LLPE	81.4	79.3	67.0	68.6	73.3	68.6	55.1	46.3	60.3	59.9	OOM	OOM
GCN+A2I	**83.3**	**82.7**	**71.6**	**71.2**	**79.4**	**78.7**	64.7	**63.4**	76.8	75.1	72.7	72.2

6.2 Performance comparison (RQ1)

Table 1 reports classification accuracy under two settings: (i) full features and (ii) a missing-feature regime, where we randomly mask 30% of node features on homophilous datasets (Cora, Citeseer, and Ogbn-Arxiv) and 60% on heterophilous ones (Chameleon, Squirrel, and Roman-Empire). The latter setting reflects realistic attribute incompleteness and highlights the contribution of the proposed *structural prototype tokens*. Across the six benchmarks, aggregation-based methods exhibit superior performance on homophilous graphs. CoED [23] remains broadly competitive but does not scale to Ogbn-Arxiv. As discussed in the related work, the substantial computational overhead of subgraph-based methods causes most such baselines to encounter out-of-memory (OOM) on large graphs. A2I attains strong accuracy throughout, with gains most pronounced on heterophilous graphs where direct neighbors share limited information and structural cues become correspondingly more important. On Roman-Empire, Polynormer variants remain within a narrow 1-point range (90.9–91.9), whereas under GCN, A2I achieves a 25.6-point gain over GCN-Base (51.2 $\rightarrow$ 76.8), while RWSE and LPE offer negligible gains, highlighting its structural encoding effectiveness. Under the feature-missing regime, most baselines degrade[3] as missing node attributes weaken the signals propagated through neighborhood aggregation, whereas A2I remains comparatively stable, suggesting that the structural information compensates for the missing information.

6.3 Generalization and transferability in few-shot scenarios (RQ2)

6.3.1 Transferability to synthetic graphs

Previous studies [29] show that GNNs trained on graphs from the same graphon family transfer to one another. We investigate whether A2I can transfer between two synthetic graphs that share underlying structural patterns but differ in size. To this end, we experiment on graphs generated from

[3]On Chameleon and Squirrel, several methods, including ours, show comparable or higher accuracy under feature missingness than with full features. This is consistent with prior observations [27] that node features in these benchmarks are noisy under the standard heterophilic split, where masking can act as denoising.

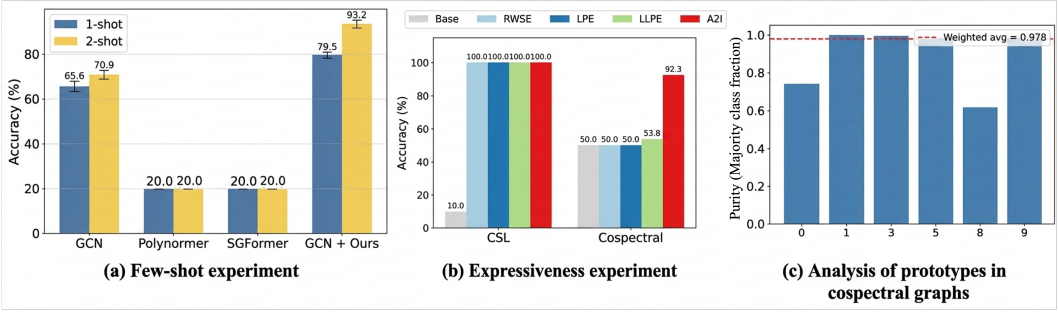


Figure 4: Experimental results on (a) Few-shot experiment, (b) Synthetic dataset to validate the expressiveness, and (c) analysis of prototypes on cospectral graphs.

the Contextual Stochastic Block Model (CSBM) [7], where each graph follows the same homophilous pattern. We use slightly different within- and between-community probabilities for each community so that they exhibit distinguishable local structures while sharing the same overall generative pattern. The number of classes is set to 5 for both graphs to allow evaluation of few-shot transferability.

Figure 4-(a) illustrates the few-shot performance of three baseline models (GCN, Polynormer, SGFormer) in comparison to GCN+A2I. A notable limitation of different baselines is that they require explicit knowledge of the graph's node count during training or inference, limiting their evaluation across graphs of different sizes. The results show that A2I exhibits consistently superior performance in the few-shot setting, effectively leveraging structural information to generalize across graphs with varying numbers of nodes. The failure of graph transformer baselines highlights their sensitivity to the number of nodes and label space of the training graph. This coupling makes it challenging for such models to generalize to new graphs under few-shot constraints [34]. This result suggests that A2I is not only capable of adapting to new graph instances but also holds promise as a foundation for graph-level models, potentially enabling transfer learning and few-shot generalization in more complex, real-world graph applications.

6.3.2 Transferability to real-world graphs

We test whether the structural representations learned by A2I transfer beyond synthetic CSBMs in two real-world settings (details of the experimental setting are in Appendix A.11.3). **(i) Cross-dataset fine-tuning:** a model pre-trained on a source graph has only its input and output layers re-initialized to fit the target's feature dimension and label set, and is then fine-tuned with 1 or 5 shots per class. **(ii) Temporal shift:** on Ogbn-Arxiv, models trained on papers published before 2016 are evaluated, without any fine-tuning, on papers published after 2017.

Table 2: **Cross-dataset transfer.** Accuracy gain (pp) over training from scratch, averaged across 1- and 5-shot fine-tuning. C: Cora, CS: Citeseer, P: PubMed, Ph: Photo, Co: Computers.

	C→P	CS→C	Ph→Co	C→Ph	C→CS	Mean
MLP	+1.07	−0.25	+2.06	−0.73	**+2.81**	+0.99
GCN	+1.31	+0.24	+0.26	+0.89	−0.39	+0.46
GraphSAGE	+0.61	+0.17	+2.78	−0.81	−0.05	+0.54
A2I	**+9.95**	**+7.45**	**+6.70**	**+4.81**	−0.10	**+5.76**

Table 3: **Zero-shot temporal transfer on Ogbn-Arxiv.** Models are pre-trained on pre-2016 nodes and evaluated zero-shot on post-2017 nodes.

Method	Zero-Shot Accuracy (%)
MLP (Feature-only)	60.65 ± 0.12
GCN	70.80 ± 0.09
GraphSAGE	70.93 ± 0.18
A2I (Ours)	$\mathbf{71.62 \pm 0.62}$

In the cross-dataset setting (Table 2), pre-training gives MLP, GCN, and GraphSAGE less than 1 pp on average. In contrast, A2I gains 5.76 pp and is best on four of five pairs, including the cross-domain pair Cora→Photo. The exception is Cora→Citeseer, where no graph-based model benefits. This suggests that A2I learns structural priors that remain useful even after its input and output layers are replaced. Under temporal shift, A2I reaches 71.62 ± 0.62, slightly above GraphSAGE (70.93 ± 0.18) and GCN (70.80 ± 0.09). All three graph models exceed a feature-only MLP (60.65) by about 10 pp (Table 3). This shows that the image-based structural encoding of A2I is at least as robust to temporal shift as message passing. Moreover, A2I achieves strong transferability using a frozen vision encoder, a design choice validated against fine-tuning alternatives in Figure 8 in the Appendix.

6.4 Expressiveness of A2I (RQ3)

Having established the theoretical capacity of A2I as a structural embedding through Corollary 1 and Theorem 3 in Appendix A.5, we now empirically verify its discriminative power on graphs. We consider two such families: graphs indistinguishable by the 1-WL test and cospectral graph pairs. For the former, we use the CSL benchmark [21], which comprises 150 4-regular graphs that 1-WL collapses. For the latter, we construct 600 canonical cospectral pairs, the 4×4 Rook's graph and the Shrikhande graph, which share identical spectral properties but are non-isomorphic.

Figure 4-(b) compares A2I against representative positional and structural encoding methods, using Polynormer as the backbone. The same experiment performed with the GCN backbone is included in the Appendix. While most encodings successfully separate the CSL graphs, they fail on the cospectral pair. A2I, in contrast, separates both families. The basis for distinguishability is both demonstrated in Theorem 3 and visualized in Figure 5 (Appendix A.5), where the sorted image representations are visibly distinct, confirming that A2I captures structural information beyond what spectral encodings can express.

Since classification on the cospectral dataset relies entirely on structural information, it provides a clean setting to examine what A2I learns. We therefore analyze the resulting prototype assignments. As shown in Figure 4-(c), the assignments are strongly class-aligned: several prototypes (e.g., 1, 3, 5, and 9) are activated almost exclusively for a single class, indicating that A2I routes graphs to a small, distinct set of prototypes based on structure alone. The average purity rate reaches 97.8%, suggesting that the prototypes capture sufficient structural information to support classification independently.

7 Limitations

While A2I's preprocessing is substantially cheaper than the naïve pipeline, it still depends on K-hop subgraph extraction and may become impractical on graphs much larger than Ogbn-Arxiv or those with heavy-tailed degree distributions, where hop-neighborhoods around hub nodes grow rapidly. Theorem 1 likewise assumes bounded average degree and is most informative in this regime. Finally, our evaluation centers on node classification, with graph-level expressiveness verified on synthetic constructions (CSL, cospectral pairs); extending A2I to large-scale graph classification, link prediction, or other graph tasks remains future work.

8 Conclusion

We propose A2I, a framework that incorporates graph structural information by rendering local adjacency patterns as images, motivated by a graphon-inspired view of resized adjacency matrices. We further introduce a Transformer-based module that integrates structural prototype tokens with aggregation tokens through a prototype bottleneck and gated fusion. We show theoretically that A2I separates graphs with distinct BFS profiles, and empirically that it distinguishes representative 1-WL-indistinguishable and cospectral pairs. Experiments on six node classification benchmarks show that A2I is competitive with and often improves over recent baselines, with pronounced gains on heterophilous graphs and under partial feature masking. Combined with its few-shot transferability across graphs of different sizes, A2I offers a promising direction for extending structural encoding to graph-level tasks.

Acknowledgements

This work was supported by the Institute of Information and Communications Technology Planning and Evaluation (IITP) through grants funded by the Korean government (IITP-2026-RS-2026-25549037, RS-2025-02218768, RS-2019-II190421).

References

[1] Ralph Abboud, Ismail Ilkan Ceylan, Martin Grohe, and Thomas Lukasiewicz. The surprising power of graph neural networks with random node initialization. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2020.

[2] Beatrice Bevilacqua, Fabrizio Frasca, Derek Lim, Balasubramaniam Srinivasan, Chen Cai, Gopinath Balamurugan, Michael M Bronstein, and Haggai Maron. Equivariant subgraph aggregation networks. In *Proceedings of the International Conference on Learning Representations*, 2022.

[3] Semih Cantürk, Renming Liu, Olivier Lapointe-Gagné, Vincent Létourneau, Guy Wolf, Dominique Beaini, and Ladislav Rampášek. Graph positional and structural encoder. In *Proceedings of the International Conference on Machine Learning*, 2024.

[4] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[5] Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. Adaptive universal generalized PageRank graph neural network. In *Proceedings of the International Conference on Learning Representations*, 2021.

[6] Chenhui Deng, Zichao Yue, and Zhiru Zhang. Polynormer: Polynomial-expressive graph transformer in linear time. In *Proceedings of the International Conference on Learning Representations*, 2024.

[7] Yash Deshpande, Subhabrata Sen, Andrea Montanari, and Elchanan Mossel. Contextual stochastic block models. In *Proceedings of the Advances in Neural Information Processing Systems*, 2018.

[8] Xiangyu Dong, Xingyi Zhang, Yanni Sun, Lei Chen, Mingxuan Yuan, and Sibo Wang. SmoothGNN: Smoothing-aware GNN for unsupervised node anomaly detection. In *Proceedings of the ACM Web Conference*, 2025.

[9] Vijay Prakash Dwivedi and Xavier Bresson. A generalization of transformer networks to graphs. In *AAAI Workshop on Deep Learning on Graphs*, 2021.

[10] Fabrizio Frasca, Beatrice Bevilacqua, Michael M. Bronstein, and Haggai Maron. Understanding and extending subgraph GNNs by rethinking their symmetries. In *Proceedings of the Advances in Neural Information Processing Systems*, 2022.

[11] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2016.

[12] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the Advances in Neural Information Processing Systems*, 2017.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[14] Daniel Herbst and Stefanie Jegelka. Higher-order graphon neural networks: Approximation and cut distance. In *Proceedings of the International Conference on Learning Representations*, 2025.

[15] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. In *Proceedings of the Advances in Neural Information Processing Systems*, 2020.

[16] Michael Ito, Jiong Zhu, Dexiong Chen, Danai Koutra, and Jenna Wiens. Learning laplacian positional encodings for heterophilous graphs. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2025.

[17] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *Proceedings of the International Conference on Learning Representations*, 2017.

[18] László Lovász. *Large networks and graph limits*. American Mathematical Soc., 2012.

[19] Yuankai Luo, Lei Shi, and Xiao-Ming Wu. Classic GNNs are strong baselines: Reassessing GNNs for node classification. In *Proceedings of the Advances in Neural Information Processing Systems*, 2024.

[20] Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and Leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019.

[21] Ryan Murphy, Balasubramaniam Srinivasan, Vinayak Rao, and Bruno Ribeiro. Relational pooling for graph representations. In *Proceedings of the International Conference on Machine Learning*, 2019.

[22] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning convolutional neural networks for graphs. In *Proceedings of the International Conference on Machine Learning*, 2016.

[23] Seong Ho Pahng and Sahand Hormoz. Improving graph neural networks by learning continuous edge directions. In *Proceedings of the International Conference on Learning Representations*, 2025.

[24] Francesca Parise and Asuman Ozdaglar. Graphon games. In *Proceedings of the ACM Conference on Economics and Computation*, 2019.

[25] Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-GCN: Geometric graph convolutional networks. In *Proceedings of the International Conference on Learning Representations*, 2020.

[26] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. DeepWalk: Online learning of social representations. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2014.

[27] Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. A critical look at the evaluation of GNNs under heterophily: Are we really making progress? In *Proceedings of the International Conference on Learning Representations*, 2023.

[28] Ladislav Rampášek, Michael Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. In *Proceedings of the Advances in Neural Information Processing Systems*, 2022.

[29] Luana Ruiz, Luiz Chamon, and Alejandro Ribeiro. Graphon neural networks and the transferability of graph neural networks. In *Proceedings of the Advances in Neural Information Processing Systems*, 2020.

[30] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, 2008.

[31] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. In *Relational Representation Learning Workshop (R2L), NeurIPS*, 2018.

[32] Xiaorui Su, Pengwei Hu, Dongxu Li, Bowei Zhao, Zhaomeng Niu, Thomas Herget, Philip S Yu, and Lun Hu. Interpretable identification of cancer genes across biological networks via transformer-powered graph representation learning. *Nature Biomedical Engineering*, 2025.

[33] Yongduo Sui, Jie Sun, Shuyao Wang, Zemin Liu, Qing Cui, Longfei Li, and Xiang Wang. A unified invariant learning framework for graph classification. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2025.

[34] Zhen Tan, Ruocheng Guo, Kaize Ding, and Huan Liu. Virtual node tuning for few-shot node classification. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023.

[35] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *Proceedings of the International Conference on Learning Representations*, 2018.

[36] Zehong Wang, Zheyuan Zhang, Tianyi Ma, Nitesh V Chawla, Chuxu Zhang, and Yanfang Ye. Beyond message passing: Neural graph pattern machine. In *Proceedings of the International Conference on Machine Learning*, 2025.

[37] Boris Weisfeiler and Andrei Leman. The reduction of a graph to canonical form and the algebra which appears therein. *NTI, Series 2(9): 12-16*, 1968.

[38] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *Proceedings of the International Conference on Machine Learning*, 2019.

[39] Qitian Wu, Wentao Zhao, Chenxiao Yang, Hengrui Zhang, Fan Nie, Haitian Jiang, Yatao Bian, and Junchi Yan. SGFormer: Simplifying and empowering transformers for large-graph representations. In *Proceedings of the Advances in Neural Information Processing Systems*, 2023.

[40] Jiaming Xu. Rates of convergence of spectral methods for graphon estimation. In *Proceedings of the International Conference on Machine Learning*, 2018.

[41] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *Proceedings of the International Conference on Learning Representations*, 2019.

[42] Yao Xu, Shizhu He, Jiabei Chen, Zihao Wang, Yangqiu Song, Hanghang Tong, Guang Liu, Jun Zhao, and Kang Liu. Generate-on-Graph: Treat LLM as both agent and KG for incomplete knowledge graph question answering. In *Proceedings of the Empirical Methods in Natural Language Processing*, 2024.

[43] Yuchen Yan, Yongyi Hu, Qinghai Zhou, Lihui Liu, Zhichen Zeng, Yuzhong Chen, Menghai Pan, Huiyuan Chen, Mahashweta Das, and Hanghang Tong. PaCEr: Network embedding from positional to structural. In *Proceedings of the ACM Web Conference*, 2024.

[44] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? In *Proceedings of the Advances in Neural Information Processing Systems*, 2021.

[45] Jiaxuan You, Jonathan M Gomes-Selman, Rex Ying, and Jure Leskovec. Identity-aware graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

[46] Bohang Zhang, Shengjie Luo, Liwei Wang, and Di He. Rethinking the expressive power of GNNs via graph biconnectivity. In *Proceedings of the International Conference on Learning Representations*, 2023.

[47] Muhan Zhang and Pan Li. Nested graph neural networks. In *Proceedings of the Advances in Neural Information Processing Systems*, 2021.

[48] Yiwei Zhang. Social network user profiling for anomaly detection based on graph neural networks. In *Proceedings of the International Conference on Artificial Intelligence and Industrial Technology Applications*. IEEE, 2025.

[49] Zaixi Zhang, Qi Liu, Hao Wang, Chengqiang Lu, and Cheekong Lee. ProtGNN: Towards self-explaining graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

[50] Chu Zhao, Enneng Yang, Yuliang Liang, Pengxiang Lan, Yuting Liu, Jianzhe Zhao, Guibing Guo, and Xingwei Wang. Graph representation learning via causal diffusion for out-of-distribution recommendation. In *Proceedings of the ACM Web Conference*, 2025.

[51] Lingxiao Zhao, Wei Jin, Leman Akoglu, and Neil Shah. From stars to subgraphs: Uplifting any GNN with local structure awareness. In *Proceedings of the International Conference on Learning Representations*, 2022.

[52] Ding-Xuan Zhou. Universality of deep convolutional neural networks. *Applied and Computational Harmonic Analysis*, 2020.

A Appendix

A.1 Integration details of Section 4.2

Prototype assignment. Eq. (9) expands as $\mathbf{s}_i^k = \mathbf{st}_{v_i}^k \mathbf{W}^k$ followed by softmax, $\alpha_{ij}^k = \exp(s_{ij}^k)/\sum_t \exp(s_{it}^k)$. The assignment can also be made discrete via Gumbel-softmax.

Transformer fusion. The full token sequence $[\mathbf{stp}_i^1, \ldots, \mathbf{stp}_i^K, \mathbf{at}_i^1, \ldots, \mathbf{at}_i^K]$ is fed to a standard Transformer encoder, producing $[\tilde{\mathbf{stp}}_i^1, \ldots, \tilde{\mathbf{at}}_i^K]$. The gate in Eq. (10) is $\mathbf{g}_i = \sigma(\mathbf{W}_g \tilde{\mathbf{at}}_i^K + \mathbf{b}_g)$ with $\mathbf{W}_g \in \mathbb{R}^{F\times F}$ and $\mathbf{b}_g \in \mathbb{R}^F$.

A.2 Discussion of Definition 1

From Definition 1, the class Σ^* is not a singleton: any two admissible orderings can disagree on graphs with non-trivial automorphism groups (e.g., they may break ties among automorphic nodes differently). Computing *any* $\sigma \in \Sigma^*$ is as hard as graph isomorphism. We therefore do not aim to realize a member of Σ^*; instead, we use Σ^* as a reference class and show (Theorem 1) that for *every* $\sigma \in \Sigma^*$, our tractable BFS+degree sorting produces an adjacency matrix whose embedding approximately satisfies $\mathcal{G} \cong \mathcal{H} \Rightarrow \mathbf{A}_{\mathcal{G}}^{\mathrm{BFS}} \approx \mathbf{A}_{\mathcal{H}}^{\mathrm{BFS}}$.

A.3 Proof of Theorem 1

We first state the full version of Theorem 1. Throughout, $\mathbf{A}^{\mathrm{BFS}}$ denotes the BFS+degree-sorted adjacency matrix of a k-hop subgraph, and d_v denotes the degree of node v within that subgraph, so that $\sum_v d_v = \mathrm{nnz}(\mathbf{A}^{\mathrm{BFS}}) = \bar{d}\, N_k$.

Theorem 1 (full version: Embedding Error Bound via Resizing Contraction). Let $\mathbf{A}_1, \mathbf{A}_2 \in \{0,1\}^{N_k\times N_k}$ be the adjacency matrices of the same k-hop subgraph with N_k nodes under two different initial orderings, and let $\mathbf{A}_1^{\mathrm{BFS}}, \mathbf{A}_2^{\mathrm{BFS}}$ be their BFS+degree-sorted versions. Let $\mathcal{T} : \mathbb{R}^{N_k\times N_k} \to \mathbb{R}^{M\times M}$ be the block-average resizing operator of Eq. (5) at fixed resolution $M < N_k$, and let $\Phi : \mathbb{R}^{M\times M} \to \mathbb{R}^F$ be L_Φ-Lipschitz with Frobenius input norm and ℓ_2 output norm. Then:

(i) Resizing contraction. For all $\mathbf{X}, \mathbf{Y} \in \mathbb{R}^{N_k\times N_k}$,

$$\left\|\mathcal{T}(\mathbf{X}) - \mathcal{T}(\mathbf{Y})\right\|_F \;\le\; \frac{1}{\lfloor N_k/M \rfloor}\left\|\mathbf{X}-\mathbf{Y}\right\|_F \;\le\; \frac{2M}{N_k}\left\|\mathbf{X}-\mathbf{Y}\right\|_F, \tag{16}$$

and the factor 2 can be dropped when M divides N_k.

(ii) Adjacency difference bound.

$$\left\|\mathbf{A}_1^{\mathrm{BFS}} - \mathbf{A}_2^{\mathrm{BFS}}\right\|_F \;\le\; \sqrt{2\bar{d}\, N_k}. \tag{17}$$

(iii) Embedding error.

$$\left\|\Phi(\mathcal{T}(\mathbf{A}_1^{\mathrm{BFS}})) - \Phi(\mathcal{T}(\mathbf{A}_2^{\mathrm{BFS}}))\right\|_2 \;\le\; 2L_\Phi\, M\sqrt{2\bar{d}}\,\cdot\,\frac{1}{\sqrt{N_k}}, \tag{18}$$

which vanishes as $N_k \to \infty$ for any fixed M and L_Φ, provided $\bar{d}$ is bounded independently of N_k.

Proof. We prove the three parts in turn.

Part (i): Resizing contraction. **Step 1 (Linearity).** $\mathcal{T}$ is linear: each output entry is a fixed uniform average of input entries. Hence

$$\mathcal{T}(\mathbf{X}) - \mathcal{T}(\mathbf{Y}) = \mathcal{T}(\mathbf{X}-\mathbf{Y}).$$

Let $\mathbf{D} = \mathbf{X} - \mathbf{Y}$. It suffices to bound $\|\mathcal{T}(\mathbf{D})\|_F$ in terms of $\|\mathbf{D}\|_F$.

Step 2 (Jensen's inequality on each block). For each output entry (i, j), applying Jensen's inequality to the convex function $f(x) = x^2$ gives

$$\big(\mathcal{T}(\mathbf{D})[i,j]\big)^2 = \left(\frac{1}{|\mathcal{B}_{i,j}|}\sum_{(u,v)\in\mathcal{B}_{i,j}} D_{uv}\right)^2 \le \frac{1}{|\mathcal{B}_{i,j}|}\sum_{(u,v)\in\mathcal{B}_{i,j}} D_{uv}^2.$$

Step 3 (Summing over all blocks). The blocks of Eq. (5) are $\mathcal{B}_{i,j} = I_i \times I_j$ with $I_i = \{\lfloor iN_k/M\rfloor, \ldots, \lfloor (i+1)N_k/M\rfloor - 1\}$, so that $|I_i| \in \{\lfloor N_k/M\rfloor, \lceil N_k/M\rceil\}$. They form a partition of all N_k^2 entries of $\mathbf{D}$, and every block is non-empty since $M < N_k$. Summing the Step 2 bound over all M^2 output entries,

$$\big\|\mathcal{T}(\mathbf{D})\big\|_F^2 = \sum_{i,j}\big(\mathcal{T}(\mathbf{D})[i,j]\big)^2 \le \sum_{i,j}\frac{1}{|\mathcal{B}_{i,j}|}\sum_{(u,v)\in\mathcal{B}_{i,j}} D_{uv}^2.$$

Since $|\mathcal{B}_{i,j}| \ge \lfloor N_k/M\rfloor^2$ for every block,

$$\big\|\mathcal{T}(\mathbf{D})\big\|_F^2 \le \frac{1}{\lfloor N_k/M\rfloor^2}\sum_{i,j}\sum_{(u,v)\in\mathcal{B}_{i,j}} D_{uv}^2 = \frac{1}{\lfloor N_k/M\rfloor^2}\big\|\mathbf{D}\big\|_F^2,$$

where the last equality uses the partition property.

Step 4 (Square root). Taking square roots,

$$\big\|\mathcal{T}(\mathbf{X}) - \mathcal{T}(\mathbf{Y})\big\|_F \le \frac{\|\mathbf{D}\|_F}{\lfloor N_k/M\rfloor} \le \frac{2M}{N_k}\,\|\mathbf{D}\|_F,$$

since $\lfloor x\rfloor \ge x/2$ for $x \ge 1$. When M divides N_k, all blocks have side N_k/M and the factor 2 can be dropped. Since $\lfloor N_k/M\rfloor \ge 1$, $\mathcal{T}$ is non-expansive for $N_k \ge M$, and a strict contraction for $N_k \ge 2M$.

Part (ii): Adjacency difference bound. Let $\mathbf{D} = \mathbf{A}_1^{\mathrm{BFS}} - \mathbf{A}_2^{\mathrm{BFS}}$. An entry of $\mathbf{D}$ is nonzero only if the corresponding entry of $\mathbf{A}_1^{\mathrm{BFS}}$ or $\mathbf{A}_2^{\mathrm{BFS}}$ equals 1, and each nonzero entry is ± 1. Since $\mathbf{A}_2^{\mathrm{BFS}}$ is a row-and-column permutation of $\mathbf{A}_1^{\mathrm{BFS}}$, both matrices have $\sum_v d_v = \bar{d}\,N_k$ nonzero entries. Hence

$$\big\|\mathbf{A}_1^{\mathrm{BFS}} - \mathbf{A}_2^{\mathrm{BFS}}\big\|_F^2 \le \mathrm{nnz}(\mathbf{A}_1^{\mathrm{BFS}}) + \mathrm{nnz}(\mathbf{A}_2^{\mathrm{BFS}}) = 2\,\bar{d}\,N_k,$$

and taking square roots gives Eq. (17). This worst-case bound holds for any pair of orderings. BFS+degree sorting restricts which entries can differ—only those in rows and columns of nodes in ambiguous groups, i.e., nodes sharing both BFS distance and degree—but does not improve the worst case.

Part (iii): Embedding error. Applying Part (i) to the bound from Part (ii), then the Lipschitz property of Φ,

$$\begin{aligned}\big\|\Phi(\mathcal{T}(\mathbf{A}_1^{\mathrm{BFS}})) - \Phi(\mathcal{T}(\mathbf{A}_2^{\mathrm{BFS}}))\big\|_2 &\le L_\Phi \cdot \frac{2M}{N_k} \cdot \sqrt{2\bar{d}\,N_k} \\ &= 2L_\Phi \cdot M\sqrt{2\bar{d}} \cdot \frac{1}{\sqrt{N_k}}.\end{aligned}$$

Since M and L_Φ are fixed by the model architecture, this vanishes as $N_k \to \infty$ provided $\bar{d}$ is bounded independently of N_k. More generally, if $\bar{d} = \bar{d}(N_k) = o(N_k)$, the bound $2L_\Phi M\sqrt{2\bar{d}/N_k} \to 0$ still holds. □

Remark 1 (Case $M \ge N_k$)**.** *Here $\mathcal{T}$ upsamples by nearest-neighbor replication, each entry being replicated $\lfloor M/N_k\rfloor$ or $\lceil M/N_k\rceil$ times along each axis, so $\|\mathcal{T}(\mathbf{D})\|_F \le \lceil M/N_k\rceil\|\mathbf{D}\|_F \le (2M/N_k)\|\mathbf{D}\|_F$, with $\|\mathcal{T}(\mathbf{D})\|_F = (M/N_k)\|\mathbf{D}\|_F$ when N_k divides M. In this regime, however, $\mathcal{T}$ is expansive, so the bound exceeds the unresized bound $L_\Phi\sqrt{2\bar{d}N_k}$; resizing tightens the bound only when $N_k > 2M$ (or $N_k > M$ when M divides N_k).*

BFS+degree sorting places nodes in the order of the key (b_v, d_v). Both keys are isomorphism-invariant, and a subset of the ambiguous group shares identical key values. Let $\mathcal{U}$ be the set of nodes in ambiguous groups of size at least two, and let $\mathcal{S} \subseteq \mathcal{U}$ be the nodes whose group occupies positions in more than one interval I_i (Part (i), Step 3). Write $s_{\mathcal{U}} = \sum_{v\in\mathcal{U}} d_v$ and $s_{\mathcal{S}} = \sum_{v\in\mathcal{S}} d_v$.

Proposition 1 (Resolution-level canonicity of BFS+degree sorting)**.** *Let $\mathbf{A}_1^{\mathrm{BFS}}, \mathbf{A}_2^{\mathrm{BFS}}$ be as in Theorem 1.*

(a) If $N_k \geq M$, then

$$\left\|\mathcal{T}(\mathbf{A}_1^{\mathrm{BFS}}) - \mathcal{T}(\mathbf{A}_2^{\mathrm{BFS}})\right\|_F \leq \frac{4M}{N_k}\sqrt{s_{\mathcal{S}}},$$

where $s_{\mathcal{S}}$ is the total degree of the tied nodes whose tie-break can actually change the image. In particular, $\mathcal{T}(\mathbf{A}_1^{\mathrm{BFS}}) = \mathcal{T}(\mathbf{A}_2^{\mathrm{BFS}})$ whenever no ambiguous group straddles an interval boundary: BFS+degree sorting is then canonical at the resolution of $\mathcal{T}$.

(b) For any N_k, $\|\mathbf{A}_1^{\mathrm{BFS}} - \mathbf{A}_2^{\mathrm{BFS}}\|_F^2 \leq \min\big(4s_{\mathcal{U}},\, 2\bar{d}N_k\big)$, where $s_{\mathcal{U}}$ is the total degree of all nodes that are tied with at least one other node, whether or not their group crosses a block boundary.

Proof. **(a)** For a sorted matrix $\mathbf{A}$, let $\tau(v) \in \{1, \ldots, M\}$ be the index of the interval containing the position of node v. By Eq. (5),

$$\mathcal{T}(\mathbf{A})[i,j] = \frac{1}{|I_i|\,|I_j|} \sum_{\tau(u)=i,\ \tau(v)=j} A_{uv},$$

so $\mathcal{T}(\mathbf{A})$ depends on the ordering only through τ. A node $v \notin \mathcal{S}$ belongs to a group contained in a single interval, so $\tau_1(v) = \tau_2(v)$. Hence every interval contains the same nodes outside $\mathcal{S}$ under both orderings, and therefore the same number of free slots for nodes in $\mathcal{S}$. Let $\mathbf{A}_2'$ be obtained by placing every $v \notin \mathcal{S}$ at its position in $\mathbf{A}_1^{\mathrm{BFS}}$ and every $v \in \mathcal{S}$ at a free position in interval $\tau_2(v)$. Then $\mathbf{A}_2'$ has interval assignment τ_2, so $\mathcal{T}(\mathbf{A}_2') = \mathcal{T}(\mathbf{A}_2^{\mathrm{BFS}})$. In $\mathbf{A}_1^{\mathrm{BFS}}$ and $\mathbf{A}_2'$, the set P of positions held by nodes in $\mathcal{S}$ is the same, and every position outside P holds the same node. Thus $\mathbf{D}' = \mathbf{A}_1^{\mathrm{BFS}} - \mathbf{A}_2'$ is nonzero only in rows or columns indexed by P. These rows and columns contain at most $2s_{\mathcal{S}}$ ones in each matrix, so $\|\mathbf{D}'\|_F^2 \leq 4s_{\mathcal{S}}$. Part (i) of Theorem 1 then gives $\|\mathcal{T}(\mathbf{D}')\|_F \leq (2M/N_k)\cdot 2\sqrt{s_{\mathcal{S}}}$.

(b) Nodes outside $\mathcal{U}$ form singleton groups and occupy the same position under both orderings. The argument of (a), applied directly to $\mathbf{A}_1^{\mathrm{BFS}} - \mathbf{A}_2^{\mathrm{BFS}}$ with $\mathcal{U}$ in place of $\mathcal{S}$, gives the bound $4s_{\mathcal{U}}$; the bound $2\bar{d}N_k$ is Part (ii) of Theorem 1. □

A.4 Discrimination of graphs via the BFS profile (Corollary 1).

Theorem 1 establishes that our pipeline is stable under input ordering. Stability alone, however, does not imply that the pipeline is *informative*: a constant function is also stable. This section provides the complementary lower bound, showing that the pipeline separates graphs whose coarse-grained BFS structure differs.

A.4.1 The BFS profile

Definition 2 (BFS profile)**.** *Let $\mathcal{G}$ be a graph with N_k nodes, and let $L_0, L_1, \ldots, L_{D-1}$ be the BFS levels from the canonical root used by our method, where $L_i \subseteq V(\mathcal{G})$ is the set of nodes at BFS distance i. The* BFS profile *of $\mathcal{G}$ is the matrix $\pi(\mathcal{G}) \in \mathbb{R}^{D\times D}$ with entries*

$$\pi_{ij}(\mathcal{G}) \;=\; \frac{1}{|L_i|\cdot|L_j|} \sum_{u\in L_i, v\in L_j} \mathbf{A}_{uv}, \qquad 0 \leq i,j \leq D-1,$$

where $\mathbf{A}$ is a BFS+degree-ordered adjacency matrix.

The BFS profile is the natural object preserved by block-average resizing when the block partition aligns with BFS levels: under that alignment, $\mathcal{T}(\mathrm{BFS}(\mathcal{G}))$ contains $\pi(\mathcal{G})$ up to a known rescaling by level sizes.

A.4.2 Input separation in terms of the BFS profile

Lemma 1 (BFS-profile lower bound)**.** *Suppose the resolution M satisfies $M \geq D$, and the block partition of $\mathcal{T}$ aligns with the BFS-level partition (i.e., each block contains entries from a single $L_i \times L_j$ pair). Then*

$$\left\|\mathcal{T}(\mathrm{BFS}(\mathcal{G}_1)) - \mathcal{T}(\mathrm{BFS}(\mathcal{G}_2))\right\|_F \;\geq\; c\left\|\pi(\mathcal{G}_1) - \pi(\mathcal{G}_2)\right\|_F,$$

where $c \;=\; \frac{M}{N_k}\sqrt{\min_{i,j}\,|L_i||L_j|}$.

Proof sketch. Index output cells of $\mathcal{T}$ by $(\alpha, \beta) \in \{0, \ldots, M-1\}^2$ and group them by the BFS rectangle $L_i \times L_j$ that contains their preimage block. Under the alignment condition, rectangle (i, j) contains exactly $m_i m_j$ output cells, where $m_i := |L_i|\, M/N_k$.

Averaging the block-average identity (Eq. (5)) over these $m_i m_j$ cells gives

$$\frac{1}{m_i m_j} \sum_{(\alpha,\beta)\in(i,j)} \mathcal{T}(\mathrm{BFS}(\mathcal{G}))[\alpha, \beta] \;=\; \pi_{ij}(\mathcal{G}),$$

i.e. $\pi_{ij}(\mathcal{G})$ is the *mean* of the $m_i m_j$ output cells in rectangle (i, j), not the value of any individual cell. Writing $\Delta_{\alpha\beta} := \mathcal{T}(\mathrm{BFS}(\mathcal{G}_1))[\alpha, \beta] - \mathcal{T}(\mathrm{BFS}(\mathcal{G}_2))[\alpha, \beta]$, this yields

$$\pi_{ij}(\mathcal{G}_1) - \pi_{ij}(\mathcal{G}_2) \;=\; \frac{1}{m_i m_j} \sum_{(\alpha,\beta)\in(i,j)} \Delta_{\alpha\beta}.$$

Cauchy–Schwarz on the right-hand side gives $\big(\pi_{ij}(\mathcal{G}_1) - \pi_{ij}(\mathcal{G}_2)\big)^2 \leq (m_i m_j)^{-1} \sum_{(\alpha,\beta)\in(i,j)} \Delta_{\alpha\beta}^2$, i.e.

$$\sum_{(\alpha,\beta)\in(i,j)} \Delta_{\alpha\beta}^2 \;\geq\; m_i m_j \big(\pi_{ij}(\mathcal{G}_1) - \pi_{ij}(\mathcal{G}_2)\big)^2.$$

Summing over rectangles and bounding $m_i m_j \geq \min_{i,j} m_i m_j$ uniformly,

$$\left\|\mathcal{T}(\mathrm{BFS}(\mathcal{G}_1)) - \mathcal{T}(\mathrm{BFS}(\mathcal{G}_2))\right\|_F^2 \;\geq\; \Big(\min_{i,j} m_i m_j\Big)\left\|\pi(\mathcal{G}_1) - \pi(\mathcal{G}_2)\right\|_F^2.$$

Substituting $m_i = |L_i| M/N_k$ and taking square roots yields the stated bound with $c = (M/N_k)\sqrt{\min_{i,j}|L_i||L_j|}$. □

A.4.3 Discrimination corollary

Corollary 2 (Restatement of Corollary 1)**.** *Under the bi-Lipschitz assumption on Φ and the alignment condition of Lemma 1,*

$$\left\|\Phi(\mathcal{T}(\mathrm{BFS}(\mathcal{G}_1))) - \Phi(\mathcal{T}(\mathrm{BFS}(\mathcal{G}_2)))\right\|_2 \;\geq\; \ell_\Phi \left\|\mathcal{T}(\mathrm{BFS}(\mathcal{G}_1)) - \mathcal{T}(\mathrm{BFS}(\mathcal{G}_2))\right\|_F \;\geq\; \ell_\Phi\, c \left\|\pi(\mathcal{G}_1) - \pi(\mathcal{G}_2)\right\|_F.$$

A.4.4 From Theorem 1 to real graphs: Discussion of the assumption gap

The assumptions of Theorem 1 may not hold on real graphs. We report below the effective form of each and indicate where it is satisfied and where it is not on three real-world datasets (Cora, Citeseer, and Chameleon).

Table 4: Dataset statistics on node degrees, neighborhood sizes, and BFS ordering consistency.

Dataset	deg. median/p95	$\Pr[N_k < 10]$ at k=3	Kendall-τ (p05)
Cora	3 / 9	11.3%	0.93 (0.87)
Citeseer	2 / 8	43.6%	0.93 (0.78)
Chameleon	12 / 109	0.2%	0.97 (0.96)

(i) Heavy-tailed degree. The bound constrains the average degree, not the maximum degree, and $\bar{d}$ enters only through $\sqrt{2\bar{d}/N_k}$. Although the degree distributions are heavy-tailed, $\bar{d}$ is small and stable in N_k on all three graphs.

Table 5: Effect of the residual permutation on accuracy. `bfs_degree_rand` fixes the (BFS level, degree) tie groups and randomizes only within them, the ambiguity bounded by Proposition 1; `random_full` destroys the grouping as well and is included as a reference point. Δ_{tie} and Δ_{full} are differences from `bfs_degree`.

	Within-tie-group			Full permutation	
Dataset	`bfs_degree`	`bfs_degree_rand`	Δ_{tie}	`random_full`	Δ_{full}
CSL	100.0 ± 0.0	69.1 ± 4.3	-30.9	10.0 ± 1.1	-90.0
Cornell	73.0	72.6	-0.4	71.9	-1.1
Texas	73.8	72.8	-1.0	71.3	-2.5
Wisconsin	55.2	48.4	-6.8	48.5	-6.7

(ii) Small k-hop neighborhoods. We report the fraction of sampled vertices whose 3-hop neighborhood contains fewer than 10 vertices. The threshold is a conservative proxy for the condition $N_k > 2\bar{d}$. On this measure, 88.7% of Cora vertices and essentially all Chameleon vertices lie inside the regime the theorem describes, while 43.6% of Citeseer vertices lie outside it. For those vertices, the contraction mechanism does not apply: the resizing operator $\mathcal{T}$ averages over blocks of width N_k/M, and when N_k is small relative to M it upsamples by replication. This distinction motivates interpreting Theorem 1 as an asymptotic characterization of the encoding.

(iii) BFS+degree is not canonical. Degree tie-breaking does not yield a canonical order, so two initial orderings induce a residual permutation and different exact positions. What the bound requires is not that this permutation be the identity, but that it position vertices by less than the block width N_k/M, since $\mathcal{T}$ is invariant to sub-block displacement: made precise by Proposition 1. Kendall's τ between the BFS+degree order and a canonical order serves as a global stability measure for this, and is high on all three datasets ($\tau = 0.93$–0.97).

To assess the sensitivity of our construction to tie-breaking permutations end-to-end, we conduct an ablation (Table 5) on the WebKB networks (Cornell, Texas, Wisconsin) [25]. The results indicate that downstream performance exhibits bounded sensitivity to local tie permutations, with output scaling proportionally with the dataset's tie-breaking ratio. Notably, while synthetic graphs designed for isomorphism testing (such as CSL) feature near-total tie ratios by construction, practical graphs display low tie ratios ($< 25\%$), confirming that tie ambiguity has a constrained effect on real-world graphs.

A.5 Extended expressiveness of A2I

It is well-known that the Weisfeiler–Leman (1-WL) test fails to distinguish certain classes of non-isomorphic graphs, such as regular graphs with identical degree distributions but different higher-order structures. Our method, however, operates on adjacency-matrix-derived images that preserve richer structural signals beyond node-level degree refinement, and leads to the following theorem.

Theorem 3 (Expressiveness of A2I)**.** *Consider two non-isomorphic graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ that are indistinguishable by 1-WL. Since the adjacency matrices $\mathbf{A}_1$ and $\mathbf{A}_2$ differ in their global arrangement of edges, the corresponding resized images $\mathbf{A}_{1,resized}$ and $\mathbf{A}_{2,resized}$ differ on at least one pixel.*

Sketch of proof. Even though the graphs that are indistinguishable by 1-WL have identical degree configurations, $\mathcal{G}_1$ and $\mathcal{G}_2$ must still differ by at least one edge. This implies the existence of at least one pixel position (i, j) such that $\mathbf{A}_1[i, j] \neq 0$ while $\mathbf{A}_2[i, j] = 0$. Assuming that the vision encoder is a universal approximator over images [52], there exists a parameterization under which the encoded representations of $\mathcal{G}_1$ and $\mathcal{G}_2$ are distinct. □

Therefore, A2I can distinguish some graph pairs that are indistinguishable by 1-WL, as demonstrated by the examples and experiments below. The first two images of Figure 5 illustrate the difference between a 12-node cycle graph and two disjoint 6-node cycle graphs, which are representative examples that the 1-WL test cannot distinguish. Moreover, from the other images from the figure, we can observe that A2I can distinguish cospectral graphs as well.

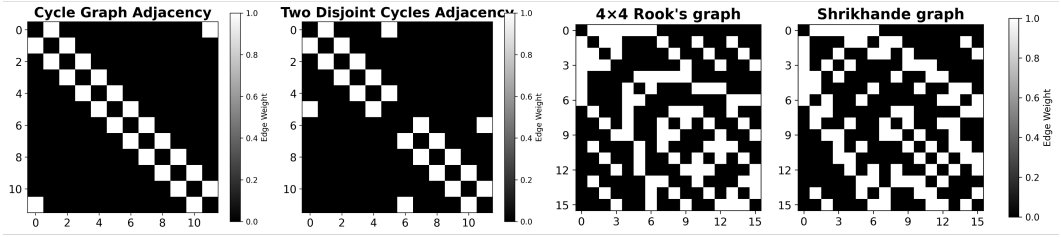


Figure 5: The image obtained from the transformed adjacency matrices of two 1-WL-indistinguishable but non-isomorphic graphs and non-isomorphic cospectral graphs.

A.6 Proof of Theorem 2

Theorem 2 (Adjacency Matrix Interpolation as Graphon Approximation) Let $\mathbf{A}_{\text{resized}} \in [0,1]^{M\times M}$ denote the resized matrix of a k-hop subgraph. Based on Eq. (5), define a mapping

$$f_M : [0,1]^2 \to [0,1], \quad f_M\Big(\frac{u}{M}, \frac{v}{M}\Big) = \mathbf{A}_{\text{resized}}[\lfloor u \rfloor, \lfloor v \rfloor], \tag{19}$$

where $u, v \in (0, M)$. Then, in the limit as $M \to \infty$, f_M converges (almost everywhere) to a measurable function

$$W(x,y) = \lim_{M\to\infty} f_M(x,y), \quad x, y \in [0,1], \tag{20}$$

which is a *graphon*, representing the continuous structural pattern of the subgraph [18].

Sketch of Proof. The interpolation operation effectively partitions the unit square $[0,1]^2$ into $M \times M$ blocks, each corresponding to a node pair in the subgraph. As $M \to \infty$, the partition becomes infinitesimally fine, and the discrete mapping f_M converges to a measurable function W that satisfies the definition of a graphon, capturing the adjacency structure in the continuous domain. □

Consequently, the resized adjacency matrix can be interpreted as a finite-sample approximation of a graphon, and applying a pre-trained vision encoder to $\mathbf{A}_{\text{resized}}$ corresponds to learning a representation over this continuous structural function.

A.7 Derivation of Eq. (15)

The attention weights without incorporating structural prototype tokens ($\mathbf{stp}$) can be expressed as follows:

$$att_i^{xy} = \frac{\exp\left((\mathbf{U}_Q\mathbf{at}_i^x)^T(\mathbf{U}_K\mathbf{at}_i^y)\right)}{\sum_{k=0}^{K}\exp\left((\mathbf{U}_Q\mathbf{at}_i^x)^T(\mathbf{U}_K\mathbf{at}_i^k)\right)}, \tag{21}$$

where $\mathbf{U}_Q, \mathbf{U}_K$ are weight matrices. Therefore, the attention value indicates the relations between the x-hop and the y-hop feature matrix.

By incorporating $\mathbf{stp}$ into the Transformer, the attention value ($with_att$) of *structural prototype tokens* is expressed as:

$$with_att_i^{x(\mathbf{stp}_i^k)} = \frac{\exp\left((\mathbf{U}_Q\mathbf{at}_i^x)^\top(\mathbf{U}_K\mathbf{stp}_i^k)\right)}{\sum_{j=1}^{K}\exp\left((\mathbf{U}_Q\mathbf{at}_i^x)^\top(\mathbf{U}_K\mathbf{stp}_i^j)\right) + \sum_{j=0}^{K}\exp\left((\mathbf{U}_Q\mathbf{at}_i^x)^\top(\mathbf{U}_K\mathbf{at}_i^j)\right)}. \tag{22}$$

and the relation between att_i^{xy} and $with_att_i^{xy}$ can be represented as:

$$with_att_i^{xy} = att_i^{xy}(1 - \sum_{k=1}^{K} with_att_i^{x(\mathbf{stp}_i^k)}), \tag{23}$$

Table 6: Hyperparameter configurations for each dataset used in our experiments.

Dataset	Hops	Dimension of Transformer	# of Transformer Layers	Heads	Dropout	learning rate	P
Cora	3	256	1	1	0.9	0.001	20
Citeseer	3	256	4	1	0.7	0.001	50
Chameleon	3	64	2	1	0.7	0.001	200
Squirrel	3	64	4	1	0.7	0.001	50
Roman-Empire	3	512	4	16	0.8	0.001	20
Ogbn-Arxiv	3	512	1	16	0.9	0.0001	20

A.8 Extended Related Work

A.8.1 Graphon

A graphon is a bounded, symmetric, measurable function $\mathbf{W} : [0, 1]^2 \to [0, 1]$. Graphons have been applied in game theory [24], mathematics [18], and statistics [40]. Recently, graphons have been applied in graph representation learning. A work [29] establishes the transferability of GNNs by showing that GNNs trained with the same graphon family can be transferable. Similarly, [14] extends to higher-order GNNs via signal-weighted homomorphism densities. Furthermore, [33] extracts invariant structural information of graphs by utilizing the graphon. Nonetheless, graphons have not been exploited to provide structural signals directly usable in embeddings for node classification tasks.

A.8.2 Comparison with graph prototype-based approaches

Unlike most existing prototype-based approaches, which typically compare prototypes against GNN-derived node representations that rely on both structural and feature information [49], our method learns prototypes directly from structural embeddings alone. This design allows the model to capture meaningful structural regularities without requiring node feature information, making the prototype learning more broadly applicable and less dependent on features.

A.8.3 Comparison with graph transformer-based approaches

Unlike conventional graph Transformer architectures [44], where each node in the graph is typically treated as an individual token, often resulting in scalability challenges due to the quadratic complexity of self-attention, our approach introduces a more compact tokenization scheme. Specifically, for each node v_i, we construct K structural prototype tokens and $K + 1$ aggregation tokens, including one token for the original node feature, yielding a sequence of $2K + 1$ tokens. This design reduces the token space per node while still preserving both structural and aggregation-based information, thereby offering a more scalable alternative. In terms of complexity, conventional graph Transformers require $\mathcal{O}(N^2)$ self-attention operations for a graph with N nodes, whereas our formulation reduces this to $\mathcal{O}\big(N(2K + 1)^2\big)$, which is significantly more efficient given that $(2K + 1)^2 \ll N$ in practice. Moreover, our method can be applied in a batch-wise operation.

A.8.4 Comparison with subgraph-based approaches

A line of work including ESAN [2], GNN-AK [51], ID-GNN [45], and Nested GNN [47] extracts subgraphs around each target node and processes them with an additional encoder. A2I shares the high-level "extract subgraph $\to$ encode $\to$ integrate" template but differs in three respects. **Encoder.** Subgraph GNNs enhance expressive power by running a trainable GNN encoder on many subgraphs, which incurs considerable computational cost, but exceeds the 1-WL limit of message passing. In contrast, A2I leverages a fixed pre-trained vision encoder applied to images rendered from sorted adjacency matrices, which is not bound by the per-subgraph 1-WL limit even when paired with conventional lightweight GNN encoders (Theorem 3). **Subgraph representation.** Subgraph GNNs operate on the variable-size raw subgraph and pay a cost that scales with subgraph size. A2I renders each subgraph to a fixed $M \times M$ image, decoupling encoder cost from subgraph size. **Integration.** Subgraph GNNs typically pool subgraph embeddings via summation or concatenation. A2I integrates K structural tokens with K aggregation tokens through a Transformer with prototypes, which we show in Table 10 outperforms simple concatenation of the same structural tokens.

Table 7: Expressiveness and performance comparison on CSL and EXP synthetic benchmarks.

Method	CSL	EXP
GCN	10.0	~ 50
ESAN	100.00	100.00 ± 0.00
ID-GNN-Full	100.00	50.50 ± 1.50
Nested GNN	100.00	99.9 ± 0.24
GNN-AK+	100.00	100.00 ± 0.00
A2I	**100.00**	$\mathbf{92.75 \pm 2.84}$

For the expressiveness comparison with subgraph-based approaches, A2I extracts structural subgraphs from the adjacency matrix rather than rooted subtrees. With an admissible canonical sorting, the ordered adjacency matrix of each extracted subgraph identifies it up to isomorphism (Definition 1), so A2I is not inherently subject to the 3-WL bound of subgraph GNNs with node-based policies [10].[4] For example, A2I distinguishes the 4×4 rook's graph and the Shrikhande graph with 92.3% accuracy (Figure 4(b)), whereas 3-WL cannot distinguish them since both are strongly regular graphs with identical parameters, $\mathrm{SRG}(16, 6, 2, 2)$ [10]. Table 7 further compares A2I with subgraph-based approaches on CSL and EXP [1]. A2I perfectly distinguishes CSL graphs but drops slightly on EXP. We attribute this decline to tie-breaking ambiguities in our BFS and degree-based sorting, which, unlike an admissible canonical sorting, leaves ties among nodes with the same BFS distance and degree unresolved (Appendix A.4.4).

A.9 Hyperparameter settings

The best-performing hyperparameter configuration is reported in Table 6. These hyperparameters were selected via a grid-search procedure over the predefined parameter space, based on validation performance. The image size M depends on the vision encoder we utilize. For example, for the ResNet-50 model, M is set to 512.

Table 8: Benchmark dataset properties and statistics.

Dataset	Classes	Features	Nodes	Edges
Cora	7	1,433	2,708	5,278
Citeseer	6	3,703	3,327	4,552
Squirrel	5	2,089	5,201	198,353
Chameleon	5	2,325	2,277	31,371
Roman-Empire	18	300	22,662	32,927
Ogbn-Arxiv	40	128	169,343	1,166,243

A.10 Experimental settings

We conduct experiments on six well-known benchmark datasets for node classification, including homophilous network datasets (Cora, Citeseer) [30], heterophilous datasets (Squirrel, Chameleon, Roman-Empire) [25, 27], and one large-scale dataset (Ogbn-Arxiv) [15]. The statistics of each dataset are provided in Table 8. We utilize the standard data split provided by the PyTorch Geometric library or 1:1:8 (train:valid:test) for the dataset that is not provided for the node classification task. We run the experiments 5 times by fixing the seed for a fair comparison. The node classification performance is validated by comparing the mean accuracy and the standard deviation. Every experiment was run on a single NVIDIA A100 GPU.

Thirteen baselines are compared with A2I for node classification, which can be grouped into five categories: (1) Aggregation-based methods, including GCN [17], GAT [35], and GraphSAGE [12], which perform feature propagation through neighborhood averaging aggregation, and GPRGNN [5], which generalizes message passing by learning adaptive propagation weights across multiple hops; (2) Graph

[4] We follow the k-WL convention in [20], under which 3-WL is equivalent to the two-dimensional folklore WL test (2-FWL).

Transformer-based models targeted for node classification, such as Polynormer [6], which incorporates polynomial filters within Transformer layers to capture multi-scale dependencies, and SGFormer [39], which employs a simple global attention; (3) A recent walk-based method, PaCEr [43], which defines positional embedding as a walk-based method and incorporates it with the aggregation-based method; (4) subgraph-based baselines, ESAN [2], ID-GNN [45], GNN-AK [51], and Nested GNN [47]; and (5) a recent baseline, CoED [23], and GPM [36], where CoED models continuous edge weights by representing each edge as a directed connection and incorporating fuzzy membership functions to capture nuanced relational dependencies, and GPM treats walks as a graph pattern to avoid message passing.

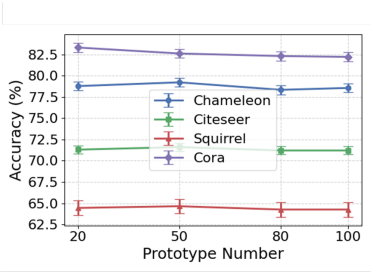


Figure 6: Parameter sensitivity of the number of prototypes.

A.11 Extended experiments including ablation & sensitivity (RQ4)

We investigate the sensitivity of A2I with respect to the number of prototypes in Figure 6 and perform an ablation study across multiple datasets in Table 10.

A.11.1 Hyperparameter sensitivity

Figure 6 depicts the sensitivity of A2I to the number of prototypes in the *structural prototype tokens*. Across four datasets, A2I demonstrates consistently robust performance, indicating that the model is relatively insensitive to the exact choice of prototype count. Notably, on smaller graphs such as Cora, employing a smaller number of prototypes results in slightly improved performance.

A.11.2 Extended results of expressiveness of A2I

As shown in Figure 7, a similar trend can be observed with Figure 4-(b).

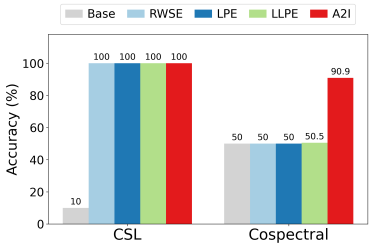


Figure 7: Expressiveness of A2I when the backbone model is GCN [17].

A.11.3 Transferability in Real-World Graphs

To demonstrate that the structural representations learned by A2I generalize beyond controlled synthetic distributions (e.g., CSBMs), we evaluate transferability across real-world graph benchmarks under two settings: (i) cross-dataset fine-tuning across heterogeneous domain spaces, and (ii) zero-shot temporal transfer on a disjoint graph split.

Cross-dataset fine-tuning across heterogeneous features. We evaluate cross-dataset transfer on five real-world graph pairs featuring distinct topologies, incompatible feature dimensions, and non-overlapping label sets. To accommodate varying feature dimensions and class counts, we re-initialize only the input and output linear layers while preserving the core backbone.

Zero-shot transfer on disjoint temporal graphs. To evaluate transferability without layer re-initialization, we design a zero-shot experiment on the `Ogbn-Arxiv` citation network partitioned temporally. Models are pre-trained on the historical subgraph (papers published prior to 2016) and evaluated directly on the disjoint test subgraph (papers published post-2017). Because the feature dimension and 40-class schema are shared across splits, no parameter re-initialization is required.

A.11.4 Full Results of Table 1.

Table 9 reports the results of Table 1 with standard deviations over five runs.

Table 9: Full results of Table 1, reported as mean ± standard deviation over five runs.

Method	Cora		Citeseer		Chameleon		Squirrel		Roman-Empire		Ogbn-Arxiv	
	Full	Miss.	Full	Miss.	Full	Miss.	Full	Miss.	Full	Miss.	Full	Miss.
General Baselines												
GAT	78.6 ±0.2	76.5 ±0.6	69.8 ±0.2	66.7 ±0.2	70.9 ±1.1	71.7 ±0.9	56.1 ±1.9	54.6 ±2.1	59.0 ±0.3	56.1 ±0.5	68.1 ±1.0	67.0 ±1.2
GraphSAGE	81.1 ±0.1	79.5 ±0.1	71.5 ±0.1	68.7 ±0.1	68.5 ±0.6	65.4 ±0.1	46.6 ±0.2	42.9 ±0.2	79.5 ±0.5	76.2 ±0.6	69.2 ±0.8	67.0 ±1.1
GPRGNN	83.2 ±0.2	77.5 ±0.1	71.3 ±0.6	67.2 ±0.8	66.3 ±2.2	58.4 ±3.1	53.4 ±1.3	43.6 ±0.7	71.0 ±0.6	64.5 ±0.5	67.6 ±0.8	64.1 ±0.6
SGFormer	81.2 ±0.8	78.5 ±0.9	69.6 ±0.4	66.3 ±1.2	64.6 ±1.7	63.0 ±2.0	40.3 ±2.2	36.5 ±1.8	65.7 ±0.4	63.1 ±0.3	72.6 ±0.2	71.5 ±0.4
PaCEr	79.3 ±1.3	78.0 ±0.2	69.4 ±0.7	67.0 ±0.6	70.0 ±1.8	70.0 ±0.3	50.4 ±0.9	51.5 ±0.3	64.9 ±0.7	62.8 ±0.4	64.4 ±0.4	63.0 ±0.2
CoED	82.2 ±0.7	78.5 ±0.4	66.6 ±1.8	65.3 ±0.3	78.7 ±3.7	78.1 ±1.4	**65.3 ±1.1**	63.3 ±1.2	81.5 ±0.5	79.1 ±0.3	OOM	OOM
GPM	78.8 ±0.3	78.0 ±0.2	64.4 ±1.9	63.2 ±0.3	55.2 ±1.7	55.0 ±1.4	37.8 ±0.7	36.7 ±0.4	48.3 ±0.6	48.0 ±0.8	71.9 ±0.4	71.5 ±0.6
Subgraph-based Baselines												
ID-GNN-Fast	81.6 ±0.2	80.7 ±0.4	68.0 ±0.7	67.6 ±1.2	70.2 ±0.5	68.8 ±0.6	48.3 ±0.5	46.5 ±0.2	32.8 ±0.3	32.2 ±0.5	OOM	OOM
Nested GNN	77.9 ±1.4	76.1 ±1.0	66.6 ±1.0	66.4 ±1.1	67.9 ±2.4	63.5 ±2.4	OOM	OOM	57.8 ±3.2	52.5 ±4.5	OOM	OOM
GNN-AK	66.6 ±3.0	63.1 ±2.2	60.4 ±1.6	57.8 ±2.2	61.4 ±0.9	59.5 ±0.6	OOM	OOM	80.2 ±0.3	81.1 ±0.3	OOM	OOM
ESAN	50.5 ±6.6	51.6 ±5.0	50.3 ±3.8	45.2 ±4.2	OOM	OOM	OOM	OOM	63.0 ±0.9	61.6 ±1.3	OOM	OOM
Positional/Structural Encoding (Polynormer Backbone)												
Polynormer-Base	80.8 ±0.9	78.6 ±1.3	68.1 ±1.1	65.9 ±1.0	71.6 ±3.4	68.9 ±3.8	61.2 ±1.4	59.8 ±2.2	**91.9 ±0.2**	90.8 ±0.4	72.5 ±0.1	71.7 ±0.3
Polynormer+RWSE	81.0 ±0.1	78.8 ±0.5	67.3 ±0.2	66.0 ±0.8	70.4 ±1.9	67.6 ±2.1	60.7 ±0.9	58.8 ±0.7	91.8 ±0.4	90.9 ±0.3	OOM	OOM
Polynormer+LPE	81.1 ±0.2	79.2 ±0.1	68.9 ±0.1	66.2 ±0.2	70.8 ±1.1	67.3 ±0.8	60.5 ±0.1	58.8 ±0.1	**91.9 ±0.4**	90.9 ±0.4	71.9 ±0.2	71.4 ±0.1
Polynormer+LLPE	80.4 ±0.8	78.7 ±0.5	67.7 ±0.6	65.7 ±1.4	72.1 ±1.5	67.4 ±0.8	59.8 ±0.1	59.5 ±0.1	90.9 ±0.4	90.6 ±0.4	OOM	OOM
Polynormer+A2I	82.4 ±0.5	81.0 ±0.1	70.5 ±0.4	67.6 ±0.5	72.4 ±0.3	75.1 ±0.4	61.5 ±0.8	61.5 ±0.2	91.8 ±0.5	**91.0 ±0.7**	**72.8 ±0.5**	**72.5 ±0.4**
Positional/Structural Encoding (GCN Backbone)												
GCN-Base	80.5 ±0.4	79.7 ±0.8	68.5 ±0.3	66.9 ±0.4	63.7 ±0.7	64.5 ±1.1	51.2 ±2.3	43.5 ±2.0	51.2 ±0.3	46.8 ±0.4	70.3 ±0.3	66.8 ±0.4
GCN+RWSE	81.8 ±0.1	80.2 ±0.3	70.1 ±0.2	69.1 ±0.1	71.3 ±0.4	71.1 ±0.7	51.2 ±0.1	48.5 ±0.1	51.4 ±0.4	47.2 ±0.3	OOM	OOM
GCN+LPE	81.6 ±0.7	80.3 ±0.5	69.2 ±0.3	69.0 ±0.3	71.5 ±0.9	70.5 ±1.1	50.9 ±0.2	48.6 ±0.1	51.1 ±0.3	46.7 ±0.5	67.8 ±0.1	66.7 ±0.1
GCN+LLPE	81.4 ±0.3	79.3 ±0.1	67.0 ±0.2	68.6 ±1.1	73.3 ±1.3	68.6 ±0.6	55.1 ±0.5	46.3 ±0.4	60.3 ±0.5	59.9 ±0.4	OOM	OOM
GCN+A2I	**83.3 ±0.5**	**82.7 ±0.1**	**71.6 ±0.4**	**71.2 ±0.5**	**79.4 ±0.3**	**78.7 ±0.4**	64.7 ±0.8	**63.4 ±0.4**	76.8 ±0.2	75.1 ±0.3	72.7 ±0.5	72.2 ±0.4

A.11.5 Ablation study

Table 10 illustrates the ablation study of A2I. First, we examine the effect of incorporating *structural prototype tokens* on the model's performance. As shown in the table, omitting these tokens leads to a noticeable decline in accuracy, highlighting their importance in capturing structural information. The impact is particularly pronounced on heterophilous datasets, where neighboring nodes share limited information, thereby making structural cues critical for accurate classification. A similar observation can be found when we do not apply BFS + degree-based sorting, leading to a noticeable degradation in performance. This suggests that our sorting approach yields an ordering closer to the ideal canonical ordering than random ordering.

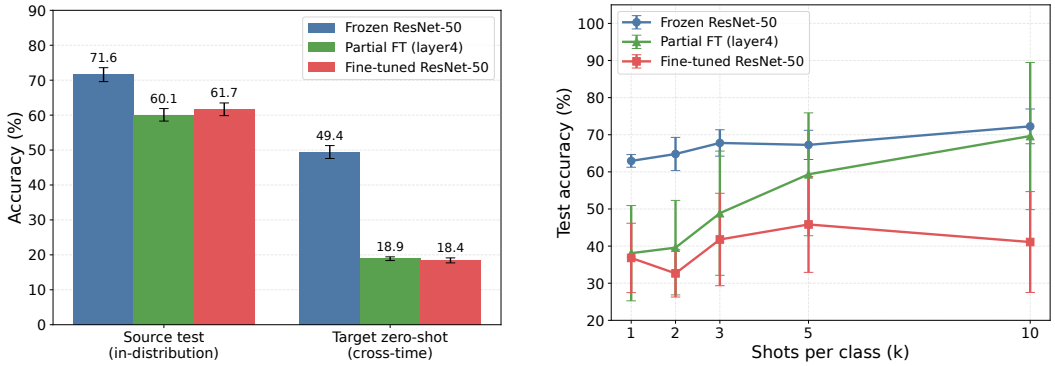
Furthermore, we validate the efficacy of the proposed structural tokens and the necessity of the Transformer architecture. Compared to a vanilla GCN [17], the simple concatenation of structural tokens $\mathbf{st}$ yields consistent performance gains, confirming the value of the added structural information. Crucially, however, A2I surpasses the simple concatenation version. This result indicates that the Transformer layer is essential for effectively harmonizing the two modalities, thereby mitigating potential interference between the structural and feature embeddings.

Table 10: Ablation study of A2I.

	Cora	Citeseer	Chameleon	Squirrel
Ours	83.3	71.6	79.4	64.7
w/o $\mathbf{stp}$	82.3	69.4	74.2	60.9
w/o sorting	81.5	69.4	70.5	51.5
GCN	80.5	68.5	63.7	51.2
GCN + w/ $\mathbf{st}$ (concat)	81.6	69.5	71.3	51.5
w/ DINO-ViT	82.9	71.6	78.8	64.5
w/ CoED	82.2	70.3	80.3	66.4

Next, we examine the impact of the pre-trained visual encoder by replacing the ResNet-50 backbone with DINO-ViT [4], enabling a comparison between CNN- and Transformer-based feature extraction. The results indicate minimal differences in performance between the two encoders. This finding suggests that A2I does not heavily rely on highly complex visual encoders, providing flexibility in choosing computationally efficient architectures without sacrificing performance. Furthermore, as the theoretical guarantees of Theorem 1 and Corollary 1 extend to any (bi-)Lipschitz encoder, the integration of a pre-trained vision model constitutes a strategic inductive bias rather than a fundamental representational necessity.

In A2I, the aggregation tokens at_i^k can be instantiated by *any* node encoder. To examine whether our structural pathway provides gains beyond a vanilla GCN backbone, we replace the GCN-based $\mathbf{h}_i^K$ with the embeddings produced by CoED, while keeping others unchanged. As shown in Table 10, this substitution yields clear improvements on heterophilous benchmarks, indicating that A2I's structural prototype tokens are complementary to stronger feature encoders.



(a) Temporal transfer on Ogbn-Arxiv

(b) CSBM graph

Figure 8: Effect of pre-trained vision encoder on transferability of A2I. Results are obtained using 2-hop images, in contrast to the 3-hop images used in the main experiments in Figure 4-(a) and Table 3.

Additionally, we evaluated the impact of freezing the pre-trained vision encoder in our transferability experiments. The experiments in Figure 8 are run in a reduced setting, rather than the one in Table 3 and Figure 4-(a), by using only up to 2-hop images instead of 3-hop images to reduce the memory peak of the fine-tuning baseline. As shown in Figure 8, A2I with a frozen pre-trained vision encoder consistently outperforms variants using partially or fully fine-tuned encoders. The partially fine-tuned variant updates only layer-4, the last residual stage. Given that utilizing a frozen encoder is also computationally efficient, these findings justify our choice to keep the vision encoder frozen.

A.12 Broader impact

A2I is foundational research on structural encodings for graph learning. Its structural tokens are computed from adjacency alone and shared across graphs through a frozen encoder and learnable prototypes. This makes A2I a possible building block for graph foundation models that encode structure independently of domain-specific node features. The same property carries risks. Structural priors learned on one graph distribution may also carry spurious or biased patterns into new domains; our transfer evidence covers only node classification on a small set of benchmarks.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Theorems and experimental results are presented to substantiate the main claims.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 7.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: Appendix A.4, A.3, A.6, A.5

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Table 6.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provide the link `https://github.com/steve30572/adj2img_NeurIPS2026` to reproduce the results.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Table 6, Figure 6.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Error bars are included in Table 9 and other experimental results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Section A.10.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the Code of ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Broader impacts are discussed in Abstract, Introduction, Limitations, Conclusion, and Appendix A.12.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite the model and datasets used in this paper.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We included the README file in `https://github.com/steve30572/adj2img_NeurIPS2026`.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper is irrelevant to Crowdsourcing.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core method in this research does not involve LLMs.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.