# EDGE-LEVEL AUTOMORPHISM IN GNNS: A QUANTITATIVE FRAMEWORK AND EFFECTIVE DESIGNS FOR LINK PREDICTION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Graph Neural Networks (GNNs) are effective for learning node and link embeddings through permutation-equivariant aggregation. However, standard GNNs collapse automorphic nodes, i.e., those with identical structural roles (or orbits), into indistinguishable representations, leading to the *node automorphism problem*. This collapse limits their expressive power and degrades link prediction performance. Existing approaches to characterize GNN expressiveness rely primarily on Weisfeiler-Lehman (WL) analyses, but these methods are typically qualitative and often misaligned with empirical results. To address this gap, we begin by introducing a novel quantitative framework to assess GNN expressiveness for link prediction. We first formalize edge-level automorphism through *edge orbits*, which capture the set of structural role pairs for nodes that share a link. Then, we introduce the edge automorphism ratio (EAR), a scalar metric that quantifies a GNN's ability to distinguish links in a given graph. We empirically demonstrate that EAR correlates strongly with performance, validating its practical benefit. Building on this insight, we design EDGE-ORBIT EQUIVARIANT GRAPH NEURAL NETWORK (EO-GNN), a GNN architecture that addresses automorphism collapse while preserving equivariance and incurring minimal computational overhead. EO-GNN accomplishes this through two core designs combined with WL-based node hashes: (i) automorphism-aware dropouts and (ii) subgraph orbit-biased aggregation. Empirical evaluations on synthetic and real graphs show improvements of up to $42.36\%$ and $28.44\%$, respectively, in predicting links in scenarios with high automorphism.

## 1 INTRODUCTION

Link prediction (LP) estimates the likelihood of connections between node pairs in a graph and is fundamental to applications such as recommendation systems, protein interaction analysis, and knowledge graph completion [17, 35]. Most GNN-based LP methods follow an encoder-decoder paradigm, where a permutation-equivariant GNN produces node embeddings that are subsequently decoded into link probabilities [48, 49]. While permutation equivariance ensures invariance to node ordering, it also causes structurally equivalent (automorphic) nodes to receive identical representations, even when they play distinct semantic and positional roles in the graph [4, 26, 30]. Informally, two nodes are *structurally equivalent (automorphic)* if some symmetry of the graph, i.e., a relabeling of vertices that preserves every edge, maps one node onto the other; because a GNN's output cannot depend on how nodes are labeled, such nodes are architecturally indistinguishable, regardless of their intended semantic role, as illustrated in Fig. 1. We refer to this as the *Node Automorphism Problem (NAP)*, which leads to degraded link discriminability [6].

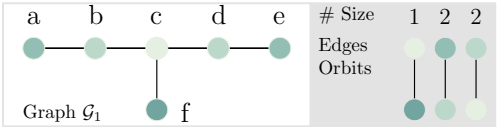



Figure 1: A standard GNN assigns identical scores to the automorphic candidate links $(a, b)$ and $(a, d)$, illustrating the Node Automorphism Problem (NAP).

Existing expressiveness analyses for GNNs primarily rely on the vertex-level Weisfeiler–Lehman (WL) hierarchy, which characterizes how well GNNs distinguish nodes using graph isomorphism



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

tests [47]. However, existing expressiveness frameworks suffer from several fundamental limitations. **(L1) Level mismatch**: WL-based analyses primarily operate at the node or subgraph level, while link prediction inherently requires fine-grained edge-centric reasoning over complex multi-node interactions, resulting in a substantial gap between theoretical expressivity characterization and practical link discrimination. **(L2) Lack of quantitative characterization**: Current frameworks remain largely qualitative, providing little insight into the extent of structural automorphism or its direct impact on link prediction performance, thereby limiting systematic analysis of model behavior under symmetry. **(L3) Missing empirical foundations**: Existing studies lack synthetic benchmarks with controllable automorphism levels, leaving theoretical claims insufficiently validated and preventing rigorous, reproducible investigation of symmetry-induced ambiguity in link prediction.

To bridge these gaps, we establish a principled edge-level framework for analyzing link prediction under structural automorphism. First, we introduce the notion of *Weisfeiler Lehman-induced Edge Orbit* to explicitly characterize edge ambiguity induced by automorphisms. Building upon this formulation, we propose the *Edge Automorphism Ratio* (EAR), a rigorous quantitative metric that measures the proportion of theoretically indistinguishable edges in a graph. We address L3 by constructing synthetic benchmarks in which EAR is an explicit, tunable generation parameter rather than an incidental property of a fixed graph. Sweeping this parameter lets us test whether performance degradation is attributable to automorphism itself (Sec. 5.2).Guided by these empirical findings, we develop EDGE-ORBIT EQUIVARIANT GRAPH NEURAL NETWORK (EO-GNN), an automorphism-aware GNN architecture designed to preserve equivariance while substantially improving edge discriminability in highly symmetric graphs. Together, our contributions are summarized as follows:

- **Quantitative Theoretical Framework:** We introduce a structural concept, the *WL-induced Edge Orbit*, that refines discrimination from the vertex to the edge level. Building on it, we characterize a GNN's expressive power through its distinguishable edge set $\mathcal{E}^{\mathcal{M}}$, letting different models be compared directly by set inclusion and set difference. We summarize this set as a single scalar, the *Edge Automorphism Ratio*, the proportion of theoretically indistinguishable edges i.e., the complement of $\mathcal{E}^{\mathcal{M}}$, giving a principled, model-agnostic measure of expressivity. (Sec. 3)
- **Automorphism-Aware GNN with Theoretical Guarantees:** We propose EDGE-ORBIT EQUIVARIANT GRAPH NEURAL NETWORK (EO-GNN), which augments standard GNN layers with two key designs: (D1) an automorphism-aware dropout and (D2) subgraph orbit-based aggregation. We prove that EO-GNN exceeds the expressivity of the standard GNN. We make our code and data available at [Anon Repo Link].(Sec. 4)
- **Extensive Empirical Validation:** We evaluate EO-GNN against state-of-the-art GNNs on synthetic graphs spanning low to high automorphism levels and on diverse real-world networks. Our ablation studies show up to a $42.36\%$ improvement over the second-best model in high-symmetry settings and consistent gains of up to $28.44\%$ across real-world benchmarks. (Sec. 5).

## 2 RELATED WORK

**Graph Expressivity Theory.** Our work relates to recent efforts on characterizing GNN expressivity [47]. Standard MPNNs are theoretically bounded by the 1-WL test in distinguishing non-isomorphic graphs [41, 43], motivating extensions based on the higher-order $k$-WL hierarchy [3, 9, 27–29]. 2D-WL further incorporates pairwise structural features for link prediction [15], while unrolling distance analyzes MPNN generalization through aligned computation trees. More recently, $k_\phi$-$k_\rho$-$m$-WL unifies message-passing link predictors by characterizing encoder expressivity and structural neighborhood radius. As summarized in Tab. 1, our framework simultaneously ensures model-agnosticism, permutation-invariance, quantitative characterization and direct alignment with the prediction task, as illustrated in Fig. 3. We provide a more comprehensive discussion of related work, MPNNs for link prediction, and comparisons with other similar designs in Sec. C.

Table 1: Graph Expressivity Frameworks for GNNs.

| Method | Level | M-Agnostic | Invariance. | Quantification. | Alignment. |
|---|---|---|---|---|---|
| 1-WL [43] | node | ✓ | ✓ | ✗ | ✗ |
| $k$-WL [5] | node | ✓ | ✓ | ✗ | ✗ |
| 2D-WL [15] | edge | ✗ | ✓ | ✗ | ✓ |
| $k_\phi$-$k_\rho$-$m$-WL [20] | edge | ✗ | ✗ | ✓ | ✗ |
| Homomorphism. [47] | subgraph | ✓ | ✓ | ✓ | ✗ |
| Unrolling Distance. [37] | node | ✓ | ✓ | ✗ | ✗ |
| **Ours** | **edge** | ✓ | ✓ | ✓ | ✓ |

# 3 QUANTIFYING EDGE DISCRIMINABILITY: A THEORETICAL FRAMEWORK

In this section, we provide key notation and present our theoretical framework before introducing EAR as a quantitative measure. We first review the fundamental concepts of graph automorphisms and node orbits. We then define the GCN-induced notion of structural equivalence, referred to as the subgraph orbit. Building on these concepts, we introduce the proposed WL-induced edge orbit and its associated edge-level equivalence relation. Finally, we define a measure for comparing the expressive power of different models based on their ability to distinguish structurally equivalent edges.

Our framework stems from a fundamental observation that the representational power of a GNN model is effectively bounded by the set of links it can structurally distinguish. We denote this set of distinguishable links as $\mathcal{E}^{\mathcal{M}}$. Consequently, we can treat $\mathcal{E}^{\mathcal{M}}$ as a direct proxy for the model's expressivity given it fully characterizes which edges will receive unique representations. Using $\mathcal{E}^{\mathcal{M}}$ is also beneficial for model comparison as it enables the expressive power of different models to be easily compared via set relations.

## 3.1 NOTATION AND PRELIMINARIES

We summarize key notation in Tab. 3 (Sec. A). Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an undirected, unweighted graph with adjacency matrix $\mathbf{A} \in \{0,1\}^{|\mathcal{V}|\times|\mathcal{V}|}$ and node feature matrix $\mathbf{X} \in \mathbb{R}^{|\mathcal{V}|\times F}$. We then consider a permutation-equivariant GNN as a mapping $\phi_w : (\mathbf{X}, \mathbf{A}) \to \mathbf{H} \in \mathbb{R}^{|\mathcal{V}|\times F}$, where $w \in \mathcal{W}$ denotes the weight set of the full link-prediction model, drawn from the weight space $\mathcal{W}$.

**Definition 3.1** (Isomorphism and Automorphism)**.** *Two graphs $\mathcal{G}$ and $\mathcal{H}$ are said to be* isomorphic *($\mathcal{G} \cong \mathcal{H}$) if there exists a bijective mapping $\pi : V_{\mathcal{G}} \to V_{\mathcal{H}}$ such that $(v,u) \in \mathcal{E}_{\mathcal{G}} \iff (\pi(v), \pi(u)) \in \mathcal{E}_{\mathcal{H}}$. An* automorphism *of $\mathcal{G}$ is the special case $\mathcal{H} = \mathcal{G}$: i.e., a relabeling of $\mathcal{G}$'s own vertices that leaves the edge set unchanged.*

**Definition 3.2** (Orbits)**.** *The automorphisms of $\mathcal{G}$ form a group $\mathrm{Aut}(\mathcal{G})$ under composition. The* orbit *of $v$ is $\mathcal{O}(v) = \{\pi(v) : \pi \in \mathrm{Aut}(\mathcal{G})\}$, i.e., the set of nodes automorphic to $v$. Orbits partition $\mathcal{V}$.*

All automorphisms induce a partition of the vertex set $\mathcal{V}_{\mathcal{G}}$ into disjoint subsets called *orbits*, where each orbit represents a unique structural role. As illustrated in Fig. 1, $\mathcal{G}_1$ has four distinct node *orbits* $a$, $b$, $c$, and $f$ shown by color. Nodes in the same orbit play identical structural roles and are therefore automorphic.

## 3.2 FORMALIZING EDGE DISCRIMINATION

Classical *orbits* are node equivalence classes of the global automorphism group; in contrast, a $k$-layer message-passing GNN observes the $k$-hop neighborhood of a node and is bounded in expressivity by the 1-WL test [29, 43]. This creates a mismatch between the global notion of automorphism and the local receptive field of GNNs [42, 46]. Thus, global automorphisms is unsuitable for NAP and fails to capture the blind spots of GNNs regarding automorphic nodes (Fig. 1).

*We formalize a structural equivalence which captures the structural role of a node as perceived by a GNN encoder.*

**Definition 3.3** (Subgraph Orbit)**.** *With subgraph $\mathcal{S}^{(k)}(u)$ induced by all nodes within distance $k$ and rooted at $u$, and representation $\phi_w(\mathcal{S}^{(k)}(u))$ of the root produced by a $k$-layer message-passing GNN. Two nodes $u, v \in \mathcal{V}$ are $k$-*equivalent*, written $u \sim_k v$, if*

$$\phi_w\big(\mathcal{S}^{(k)}(u)\big) = \phi_w\big(\mathcal{S}^{(k)}(v)\big) \quad \text{for all } w. \tag{1}$$

*A $k$-layer message-passing GNN computes the root representation from $\mathcal{S}^{(k)}(u)$. Throughout the paper, we use the equivalence obtained after 1-WL stabilizes and omit the subscript $k$. We write $\mathcal{O}_{\mathcal{S}}(u)$, or simply $\mathcal{O}_u$ when the context is clear, for the resulting* subgraph orbit *of $u$: its equivalence class $\{v \in \mathcal{V} : v \sim u\}$ under this relation.*

*Based on subgraph orbits, we introduce an edge equivalence designed to characterize the limitations of GCN-based link prediction systems.*

**Definition 3.4** (WL-Induced Edge Orbit)**.** *For a node pair $e = \{u, v\}$, $u, v \in \mathcal{V}$, not necessarily in $\mathcal{E}$, we define its* WL-induced edge orbit *as the multiset of the subgraph orbits (Def. 3.3) of its endpoints:*

$$\mathcal{O}_{\mathcal{E}}(e) \coloneqq \{\{\mathcal{O}_{\mathcal{S}}(u),\, \mathcal{O}_{\mathcal{S}}(v)\}\}\,. \tag{2}$$

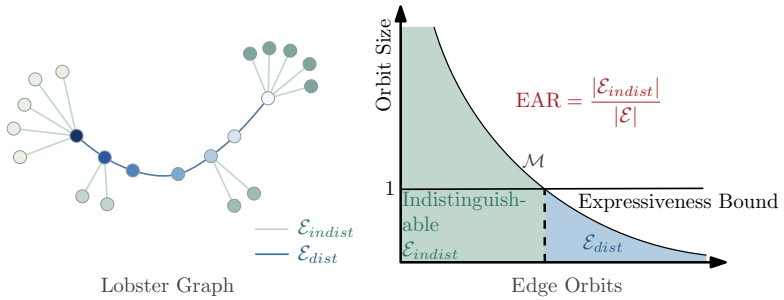



Figure 3: Illustration of EAR based on WL-induced edge equivalence. Green denotes indistinguishable edges $\mathcal{E}_{\text{indist}}$ and blue denotes distinguishable edges $\mathcal{E}_{\text{dist}}$. Expressiveness bound illustrates the theoretical discriminativity limit of a GNN model. $\mathcal{E}^{\mathcal{M}}$ provides a formal characterization of the expressivity of model $\mathcal{M}$ by capturing the set of edge pairs it can distinguish. A graph with more indistinguishable edges (high EAR) is inherently more challenging for link prediction.

*Two pairs $e, e'$ are* WL-edge-equivalent, *written $e \sim^{\mathcal{E}} e'$, iff $\mathcal{O}_{\mathcal{E}}(e) = \mathcal{O}_{\mathcal{E}}(e')$. The multiset makes the definition invariant to endpoint order while distinguishing pairs whose endpoints share a subgraph orbit from pairs whose endpoints belong to different orbits.*

**Clarification** We remark that the WL-induced edge orbit differs from classical *edge orbits* in algebraic group theory. Consider the triangular prism in Fig. 2. Because the graph is 3-regular, 1-WL assigns every vertex the same color, placing all edges in a single WL-induced edge orbit. The graph automorphism group, however, has two edge orbits: triangle edges and matching edges. Automorphisms preserve triangle membership, so no automorphism maps a triangle edge $e_1$ to a matching edge $e_2$. WL-induced equivalence edge relation proposed in our paper is a model-induced equivalence determined by GCN.

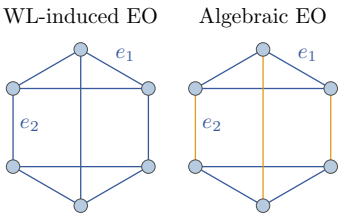



Figure 2: Graph with one WL-induced edge orbit but two algebraic edge orbits.

We next formalize when a parameterized model $\mathcal{M}_w$ discriminates a pair of such edges, and then compare models by the sets of edge pairs they can distinguish.

**Definition 3.5** (Edge Discrimination)**.** *Let $e_1$ and $e_2$ belong to the same* WL-induced edge orbit, *i.e., $\mathcal{O}_{\mathcal{E}}(e_1) = \mathcal{O}_{\mathcal{E}}(e_2)$. A model $\mathcal{M}_w$* discriminates *$e_1$ and $e_2$ if there exists $w \in \mathcal{W}$ such that $\mathcal{M}_w(e_1) \neq \mathcal{M}_w(e_2)$, written $e_1 \not\equiv_{\mathcal{M}} e_2$. We collect all such pairs into the* distinguishable edge set *$\mathcal{E}^{\mathcal{M}} = \{\, \{e_1, e_2\} : e_1 \not\equiv_{\mathcal{M}} e_2 \,\}$.*

**Definition 3.6** (Higher Expressiveness)**.** *We order models by inclusion of their distinguishable edge sets. Given two models $\mathcal{M}_1$ and $\mathcal{M}_2$, we say $\mathcal{M}_1$ is* at least as expressive *as $\mathcal{M}_2$, denoted $\mathcal{M}_1 \succeq \mathcal{M}_2$, if for every pair of edges $e_1, e_2 \in \mathcal{E}(\mathcal{G})$, $e_1 \not\equiv_{\mathcal{M}_2} e_2 \;\Rightarrow\; e_1 \not\equiv_{\mathcal{M}_1} e_2$; equivalently, $\mathcal{E}^{\mathcal{M}_2} \subseteq \mathcal{E}^{\mathcal{M}_1}$.*

The distinguishable edge set $\mathcal{E}^{\mathcal{M}}$ thus fully characterizes a model's expressivity, capturing exactly the links it can distinguish. We illustrate the framework in Fig. 3.

### 3.3 Quantifying Link Ambiguity: A Quantitative Measure

Building upon distinguishable edge set $\mathcal{E}^{\mathcal{M}}$, we introduce the *EAR*, a quantitative measure of theoretically indistinguishable edges under a GNN encoder $\phi_w$. EAR will be used to empirically analyze the effect of structure automorphism and motivate our principled design choices.

**Measure 3.1** (Edge Automorphism Ratio)**.** *Let $\mathcal{E}$ denote the set of all edges in the graph $\mathcal{G}$, and for an edge $e = \{u, v\}$ let $\mathcal{O}_{\mathcal{E}}(e)$ be its WL-induced edge-orbit signature (Def. 3.4). Denote by*

$$[e]_{\mathcal{E}} \;=\; \{\, e' \in \mathcal{E} \mid \mathcal{O}_{\mathcal{E}}(e') = \mathcal{O}_{\mathcal{E}}(e) \,\}$$

*the* edge orbit *of $e$, i.e., the set of edges sharing its signature (equivalently, the equivalence class of $e$ under $\sim^{\mathcal{E}}$).* EAR *is defined as the proportion of indistinguishable edges $\mathcal{E}_{indist}$, written as:*

$$\text{EAR} = \left(\frac{|\mathcal{E}_{indist}|}{|\mathcal{E}|}\right)^{\gamma}, \mathcal{E}_{indist} = \{e \in \mathcal{E} \mid |[e]_{\mathcal{E}}| > 1\}\,.$$

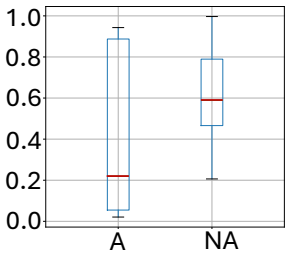
*where $\frac{|\mathcal{E}_{\text{indist}}|}{|\mathcal{E}|}$ is the proportion of indistinguishable edges in the graph, and $\gamma \in (0, 1]$ is a power-law scaling factor. For $\gamma < 1$, the exponent increases EAR values between zero and one while preserving*

*the endpoints. Indistinguishable edges are those belonging to non-singleton edge orbits, i.e., those whose* orbit size $|[e]_{\mathcal{E}}| > 1$*, because their WL-induced structural roles are not unique in the graph.*

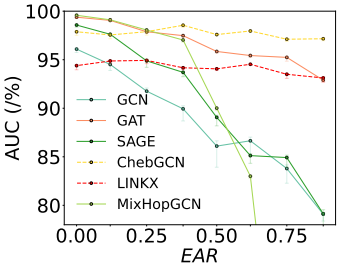
**EAR Computation.** We utilize 1-orbit-WL (Algo. 1) adapted from [29] to efficiently estimate edge orbits $\mathcal{O}_{\mathcal{E}}$. For example, in graph $\mathcal{G}_1$ of Fig. 1, four out of five edges belong to non-singleton edge orbits, yielding $\mathrm{EAR}(\mathcal{G}_1) = \frac{4}{5} = 0.8$. In contrast, all edges in $\mathcal{G}_2$ are indistinguishable, resulting in $\mathrm{EAR}(\mathcal{G}_2) = 1$. The EAR measure ranges from $[0, 1]$ and applying a power-law transformation improves its spread across this range, in contrast to the raw ratio, which typically falls within $[0, 0.4]$ for many real-world graphs.

### 3.4 Connecting EAR to GNN Performance

As a motivating example, we demonstrate in Fig. 4a that on a Lobster graph (Fig. 3) with automorphic edges detected by Algo. 1. By design, the Lobster graph features a central path of vertices with distinct orbits, while its leaves reside in identical orbits, creating structural ambiguity. We show in Fig. 4a that the automorphic edges exhibit significantly higher variance in link likelihood scores for standard GNN [19], with mean values falling below the detection threshold of 0.5. This indicates that automorphic edges are likely to be incorrectly predicted as negative links.




(a) Automorphic (A) and non-automorphic (NA) edges.




(b) Performance decay on *syn-triangle* as EAR increases.

To systematically investigate the impact of structural automorphism on link prediction, we construct synthetic graphs with controllable automorphism levels (Sec. E) and report the mean AUC of representative models, including GNNs (SAGE [11], MixHop [1]), the non-convolutional model LINKX [24], and the spectral method GCN-Cheby [12], across increasing EAR values in Fig. 4b. We observe that standard GNNs perform well under low automorphism but degrade substantially as structural automorphism increases. At $EAR = 0.88$, all GNN variants underperform LINKX by up to $15\%$, indicating severe loss of discriminability under high automorphism. Although MixHop partially alleviates this issue through higher-order neighborhoods, it still remains $15.3\%$ below LINKX under strong symmetry. In contrast, GCN-Cheby, a spectral method without equivariance constraints, maintains better robustness across all levels, highlighting that GNN-based variants that work well under low automorphism (EAR$= 0.25$) are not appropriate for networks with medium/high automorphism.

## 4 Learning over Increasing Automorphism

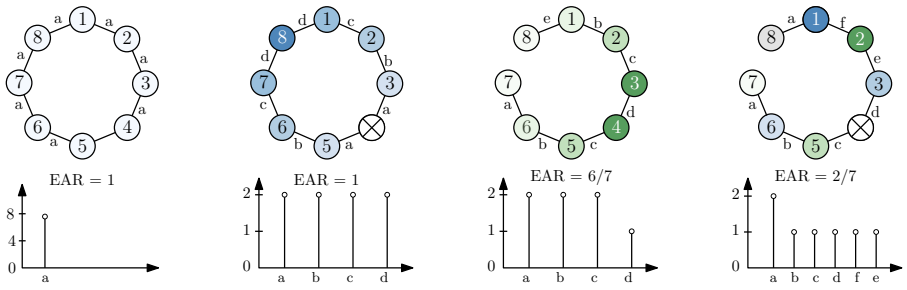
Motivated by this limitation, we discuss and theoretically justify a set of key design choices that, when appropriately incorporated in a GNN framework, can maintain robust performance across the spectrum of automorphism values.

### 4.1 D1: Automorphism-aware Dropout

*Intuition.* This design strategically perturbs the graph to disrupt otherwise indistinguishable subgraph orbits, allowing GNNs to learn more discriminative representations.
Given precomputed WL-hashes $\mathbf{O} \in \mathbb{Z}^{|\mathcal{V}|}$ from Algo. 1, we assign adaptive dropout probabilities according to normalized orbit sizes. For each node $u$, we define

$$p_u = \min\left(p_{\max},\ \alpha_p \cdot \log\left(1 + \frac{|\mathcal{O}_u|}{|\mathcal{V}|}\right)\right),$$

where $|\mathcal{O}_u|$ denotes the cardinality of the subgraph orbit associated with node $u$, $\alpha_p > 0$ is a scaling factor, and $p_{\max} < 1$ bounds the dropout probability. The resulting edge and node dropout




(a) Original $C_8$ (b) After NodeDrop (c) After EdgeDrop (d) After D1

Figure 5: Illustration of D1 Automorphism-aware Dropout on the cycle graph $C_8$, where node colors denote subgraph orbits under a 2-layer GNN. (a) All edges are automorphic ($EAR = 1$). (b) Dropping one edge breaks symmetry and reduces $EAR$ to $\frac{6}{7}$. (c) Node dropout preserves $EAR$ but makes half of the vertices distinguishable. (d) Combining node and edge dropout effectively improves GNN expressiveness (Thm. 4.1).

mechanisms are defined as

$$\widetilde{A}_{uv} = \begin{cases} \dfrac{A_{uv}}{1-p_{uv}}, & \text{with probability } 1-p_{uv}, \\ 0, & \text{with probability } p_{uv}, \end{cases} \qquad \forall (u,v) \in \mathcal{E}, \tag{3}$$

and $\widetilde{\mathbf{x}}_u = z_u \cdot \mathbf{x}_u, \quad z_u \sim \text{Bernoulli}(1-p_u),\ \forall u \in \mathcal{V}$. This design encourages the model to break structural automorphism and improve node distinguishability. Similarly, the edge dropout probability $p_{uv}$ is computed analogously from edge orbit sizes.

*Theoretical Justification.* We now formalize the role of D1 in mitigating high edge automorphism. We show that GNN can better distinguish subgraph orbits than a GNN without this design.

**Theorem 4.1.** *Let $\phi_w : \mathcal{G} \to \mathbb{R}^{|\mathcal{V}| \times F}$ be a permutation-invariant GNN. If $\mathbb{E}_{w\sim\mathcal{W}}[\phi_w(\mathcal{S})]$ exists for all $\mathcal{S} \subseteq \mathcal{V}$, then GNN-D1 $\phi(p_a, p_n)$ with $p_a, p_n > 0$, for any $\epsilon, \sigma > 0$, it suffices to sample*

$$\mathbb{L}_\alpha = \left\lceil \frac{\log(\sigma)}{|\mathcal{E}_\text{a}| \log(1-p_\text{a}) + |\mathcal{E}_\text{n}| \log(1-p_\text{n})} \right\rceil < \left\lceil \frac{\log(\sigma)}{-|\mathcal{E}| \log(1-p)} \right\rceil$$

*times such that $\mathbb{P}_{w\sim\mathcal{W}}[D(\mathcal{S}, \mathcal{S}') \geq \epsilon] \geq 1-\sigma$, where $\mathcal{S}'$ denotes the perturbed subgraph obtained by applying D1's node/edge dropout to $\mathcal{S}$, and $D(\cdot,\cdot)$ is a distance metric between their GNN embeddings. Here $\mathcal{E}_\text{a}$ and $\mathcal{E}_\text{n}$ denote the automorphic and non-automorphic edge sets, and $p_\text{a}, p_\text{n}$ the dropout probabilities applied to them. $\mathbb{L}_\alpha$ is smaller than the sample complexity required by random dropout with a single rate $p$.*

The theorem shows that GNN-D1$\phi(p_a, p_n)$ can distinguish structurally equivalent nodes for any $p > 0$, while requiring fewer repetitions than random dropout to break graph symmetries. Additional comparisons and analyses are provided in Sec. C(Fig. 8b).

### 4.2 D2: Subgraph Orbit-Biased Aggregation

*Intuition.* D2 injects stochastic orbit-aware structural priors by perturbing identical structural roles with Gaussian noise, it introduces local asymmetry while preserving orbit-level information.

Let $\mathbf{O} \in \mathbb{Z}^{|\mathcal{V}|}$ denote the structural role identifiers obtained from $K$-iteration WL subtree labels, where $\mathbf{x}_v, \mathbf{o}_v$ represents raw embedding and orbit-aware identifier of node $v$. We construct a stochastic role embedding:

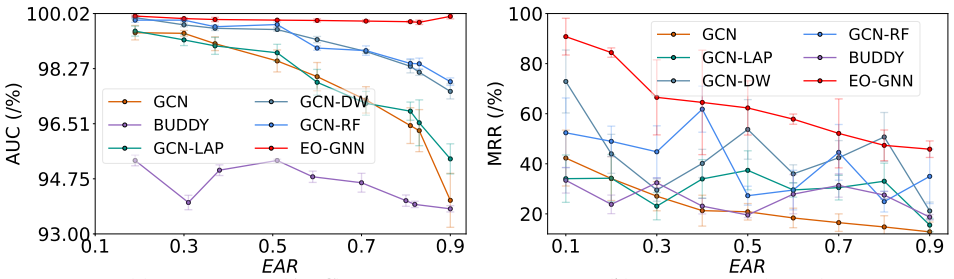
$$\mathbf{h}_v^{(0)} = \text{Emb}(\mathbf{o}_v) + \boldsymbol{\epsilon}_v, \quad \boldsymbol{\epsilon}_v \sim \mathcal{N}(0, \tau^2 \mathbf{I}), \tag{4}$$

where $\text{Emb}(\cdot)$ is a learnable embedding table and $\tau$ controls the perturbation strength. The noisy role embedding is concatenated with the initial node feature: $\mathbf{x}_v^{(0)} = [\mathbf{h}_v^{(0)} \| \mathbf{x}_v]$, and injected into each message-passing layer:

$$\mathbf{x}_v^{(k)} = \zeta^{(k)}\left(\mathbf{x}_v^{(k-1)}, \square_{u\in\mathcal{N}(v)} \phi^{(k)}\left(\mathbf{x}_u^{(k-1)}, \mathbf{x}_v^{(k-1)}\right)\right) + \mathbf{W}^{(k)} \mathbf{h}_v^{(0)}, \tag{5}$$

where $\square$ denotes a permutation-invariant aggregation operator, $\phi, \zeta$ are learnable functions, and $\mathbf{W}^{(k)}$ projects orbit-aware role embeddings into the hidden feature space.

*Theoretical Justification.* To demonstrate D2's impact on automorphism, we first define an *automorphism-dominant* vertex $v_a \in \mathcal{V}$ as a node whose neighbors are more likely to belong to the same orbit: $\mathbb{P}_{u\sim\mathcal{N}(v_a)}(\mathcal{O}_u = \mathcal{O}_{v_a}) > \mathbb{P}_{u\sim\mathcal{N}(v_a)}(\mathcal{O}_u \neq \mathcal{O}_{v_a})$, where $\mathcal{O}_v$ denotes the subgraph orbit label of node $v$ (Def. 3.3).




(a) `syn-cora` (AUC) (b) `syn-cora` (MRR)

Figure 6: Performance on the semi-synthetic benchmarks. EO-GNN remains robust under high automorphism while maintaining competitive MRR and AUC in low-automorphism regimes.

**Theorem 4.2.** *With a standard deterministic GNN with D2, the stochastic orbit-aware embedding increases their expected pairwise distance (Eq.* (4)*):*

$$\mathbb{E}\big[\|\phi_{\mathrm{D2}}(u) - \phi_{\mathrm{D2}}(v_a)\|^2 \mid \mathcal{O}_u = \mathcal{O}_{v_a}\big] > \mathbb{E}\big[\|\phi(u) - \phi(v_a)\|^2 \mid \mathcal{O}_u = \mathcal{O}_{v_a}\big]$$

*Meanwhile, this distance remains bounded by the separation from non-automorphic neighbors when appropriately combined with GNN. Thus, D2 introduces local asymmetry within automorphic neighborhoods while preserving the structural separation from non-automorphic neighbors.*

### 4.3 EO-GNN: A Framework for Relaxing Equivariance in Link Prediction

We now present Edge Orbit Equivariant Graph Neural Network (EO-GNN), which integrates a standard GCN with our proposed designs D1 and D2 to address the automorphic node problem across the full spectrum of low to high automorphism. At a high level, EO-GNN consists of three stages: **(S1)** precomputes the subgraph node orbit; **(S2)** performs automorphism-aware message passing enhanced by D1–D2; and **(S3)** performs link prediction.

The *pre-compute stage* **(S1)** uses WL (Algo. 1) to generate $\mathbf{O} \in \mathbb{Z}^{|\mathcal{V}|}$ subgraph orbit hash labels. In the *aggregation stage* **(S2)**, the generated automorphic embeddings are concatenated with original node features and repeatedly updated for $K$ rounds of message passing steps based on the aggregation in (Eq. (5)). We collect the final embeddings in $\mathbf{X}^{(K)}$, whose row $\mathbf{x}_u^{(K)}$ is the final embedding of node $u$.

In the *prediction stage* **(S3)**, for a target edge $(u, v)$, we combine pairwise and structure features(common neighbors):

$$\begin{aligned}\mathbf{z}_{uv} = \mathrm{ReLU}\Big(&\mathbf{W}_{cn}\,\mathrm{AGG}(\mathcal{N}(u) \cap \mathcal{N}(v)) \\ &+ \mathbf{W}_h(\mathbf{x}_u^{(K)} \odot \mathbf{x}_v^{(K)}) + \mathbf{W}_o(\mathbf{o}_u^{(0)} \odot \mathbf{o}_v^{(0)})\Big).\end{aligned} \tag{6}$$

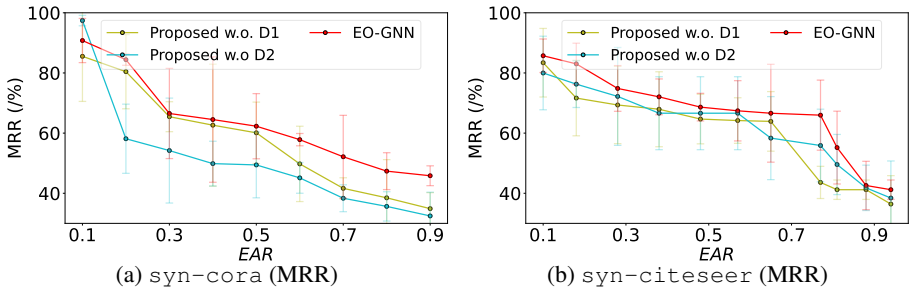
**Time complexity.** The 1-WL algorithm with $K$ iterations requires $\mathcal{O}(K|\mathcal{E}|)$ time, since each iteration aggregates neighborhood information along graph edges [36]. D1 introduces no complexity. The feature propagation complexity is $\mathcal{O}\big(L \cdot (|\mathcal{V}| \cdot p^2 + |\mathcal{E}| \cdot p)\big)$, where $p$ is the hidden feature dimension and $L$ is the number of GNN layers. Therefore, the overall complexity of EO-GNN is $\mathcal{O}\big(K|\mathcal{E}| + L \cdot (|\mathcal{V}| \cdot p^2 + |\mathcal{E}| \cdot p)\big)$ . A detailed analysis is provided in Sec. I.

## 5 Empirical Evaluation

In our empirical analysis, we first show the significance of the proposed designs in EO-GNN on both synthetic and real-world graphs with low-to-high EAR. Specifically, we consider the following research questions: **(RQ1)** In a synthetic setting where the data generation process and automorphism level are controllable, do EO-GNN distinguish the automorphic structure? **(RQ2)** As the automorphism level increases, to what extent do the EO-GNN improve the performance? **(RQ3)** On complex real-world medium- and large-scale graphs with varying levels of automorphism, to what extent do EO-GNN and other baselines improve GNN performance?

### 5.1 Experimental Setup

**Datasets.** We evaluate EO-GNN on seven real-world link prediction benchmarks from the Planetoid and OGB collections: Cora, Citeseer, Pubmed, ogbl-collab, ogbl-ddi, ogbl-ppa, and ogbl-




(a) `syn-cora` (MRR) (b) `syn-citeseer` (MRR)

Figure 7: Impact of D1 and D2 Modules in EO-GNN. When EAR $\geq$0.5, both D1 and D2 contribute substantially to the performance gains, with D2 provides slightly stronger improvements on Cora.

citation2 [14]. To study the effect of structural automorphism, we further construct semi-synthetic datasets with controllable EAR values from 0.1 to 0.9, following [25]. For the Planetoid and synthetic datasets, all methods use identical random splits (80%/15%/5%), while official OGB splits are adopted for fair comparison with prior work.

**Training Protocol and Metrics.** We report Mean Reciprocal Rank (MRR), Area Under the ROC Curve (AUC) and Hits@K depending on the benchmark setting, following the standard evaluation protocol established for each benchmark [21]. We primarily emphasize MRR due to its sensitivity to structural discrimination quality in link prediction. Although the reported metric varies by dataset, within each dataset all methods are evaluated under identical splits and the same metric, so comparisons are always apples-to-apples. All reported results are averaged over five random runs using identical training, validation and testing protocols across all methods. Detailed hyperparameter configurations and training settings are provided in Sec. G.2.

**Baseline models.** We compare EO-GNN with representative baselines spanning five categories: (1) *heuristic link predictors*, including Common Neighbor (CN), Adamic–Adar (AA), and Resource Allocation (RA) [2, 23, 50]; (2) *classical GNNs*, including GCN, GAT, GIN, GraphSAGE, and MixHop [1, 11, 19, 38, 43]; (3) *pairwise and structure-aware GNNs*, including SEAL, NBFNet, Neo-GNN, BUDDY, and NCN(C) [6, 40, 45, 48, 51]; (4) *graph-agnostic models*, such as LINKX and MLP [24]; and (5) *GNNs with positional encodings*, including GCN-DW, GCN-Lap, and GCN-RF [18, 31, 32]. The real-world evaluation (Tab. 2) covers the heuristic, classical-GNN (GCN, GAT, GIN, GraphSAGE), and pairwise/structure-aware baselines, whereas the synthetic benchmark (Tab. 5) additionally includes MixHop, the graph-agnostic models, and the positional-encoding variants. Additional dataset generation details and results are provided in Secs. E and J.

### 5.2 (RQ1) Effectiveness of EO-GNN

Figure 6a reports the mean test AUC (with standard deviation) over five runs of the top-6 performing models on the synthetic benchmark, selected for visual clarity; the complete comparison against all baselines is reported in Tab. 5. All baselines exhibit clear performance degradation as EAR increases, especially classical GNNs under highly symmetric settings (EAR $\geq$ 0.7). Positional encoding methods, including GCN-DW, GCN-RF, and GCN-Lap, improve robustness over vanilla GNNs by up to 3.71% in AUC, indicating that auxiliary structural signals partially mitigate symmetry ambiguity. Pairwise-feature methods remain relatively stable across all automorphism regimes, but often sacrifice performance under low-EAR settings. In contrast, EO-GNN (Red) consistently achieves the best AUC and MRR, reaching up to 99.7% with the lowest variance across all automorphism levels, demonstrating strong robustness under increasing structural automorphism.

### 5.3 (RQ2) Ablation Study: Significance of Design Principles

We evaluate the effectiveness of the proposed designs on `syn-cora` and `syn-citeseer` through the ablation studies shown in Fig. 7. Specifically, we consider two variants of EO-GNN: (1) removing the Automorphism-aware Dropout(D1, Eq. (3)), and (2) removing the Subgraph Orbit-Biased Aggregation(D2, Eq. (5)). These variants are compared against the full model.

*D1: Automorphism-aware Dropout.* As illustrated in Fig. 7-a and Fig. 7-b, D1 (Green) substantially improves both the expressiveness and robustness of link prediction across different automorphism levels, with particularly strong gains in highly automorphic regimes. On Syn-Cora, D1 achieves up to a 22.30% improvement at $EAR = 0.8$. Similarly, on Syn-Citeseer, a clear inflection point emerges

Table 2: Real data: mean accuracy ± stdev over different data splits. Best model per benchmark highlighted in YellowGreen. OOM indicates that the algorithm requires over 40Gb of GPU memory or more than 24 hours. We highlight the two best-performing GNN models.

| **Dataset** | **PPA** | **Pubmed** | **Collab** | **DDI** | **Cora** | **Citeseer** | **Citation2** |
|---|---|---|---|---|---|---|---|
| **EAR** | **0.001** | **0.216** | **0.505** | **0.002** | **0.139** | **0.326** | **0.311** |
| **Metric** | HR@100 | MRR | HR@50 | HR@20 | MRR | MRR | MRR |
| CN | 27.65±0.00 | 14.66±0.06 | 61.37±0.00 | 17.73±0.00 | 32.88±0.09 | 21.13±0.02 | 74.30±0.00 |
| AA | 32.45±0.00 | 19.87±0.30 | 64.17±0.00 | 18.61±0.00 | 47.33±0.09 | 24.61±0.11 | 75.96±0.00 |
| RA | 49.33±0.00 | 19.16±0.27 | 63.81±0.00 | 6.23±0.00 | 47.17±0.11 | 23.94±0.16 | 76.04±0.00 |
| GCN | 29.57±2.90 | 14.55±2.41 | 46.25±1.60 | 65.04±0.76 | 34.19±6.23 | 44.37±6.06 | 84.74±0.07 |
| GIN | OOM | 15.96±2.21 | 48.84±0.54 | 67.05±0.61 | 42.47±7.02 | 29.48±6.21 | OOM |
| SAGE | 25.80±1.94 | 11.34±2.41 | 48.10±0.81 | 63.69±1.45 | 30.81±8.07 | 44.48±8.40 | 82.60±0.36 |
| GAT | OOM | 4.85±0.91 | 48.33±0.61 | 34.51±22.49 | 36.96±4.27 | 45.69±7.19 | OOM |
| SEAL | 48.80±3.16 | 49.02±13.91 | 64.74±0.43 | 30.56±3.86 | 37.81±9.93 | 39.36±4.99 | 87.67±0.32 |
| NBFNet | OOM | 19.46±2.42 | OOM | 44.00±0.58 | 41.48±5.11 | 38.17±3.06 | OOM |
| Neo-GNN | 49.13±0.60 | 31.44±3.85 | 57.52±0.37 | 63.57±3.52 | 41.48±5.11 | 53.97±5.88 | 87.26±0.84 |
| BUDDY | 49.85±0.20 | 19.46±2.42 | 65.94±0.58 | 78.51±1.36 | 30.78±5.55 | 22.84±0.36 | 87.56±0.11 |
| NCN | 61.19±0.85 | 25.92±4.33 | 64.76±0.87 | 82.32±6.10 | 45.76±6.39 | 54.97±6.03 | 88.09±0.06 |
| NCNC | 61.42±0.73 | 20.31±6.51 | 66.61±0.71 | 84.11±3.67 | 48.68±8.60 | 64.03±3.67 | 89.12±0.40 |
| EO-GNN | 61.58±0.45 | 68.51±0.74 | 70.95±0.81 | 86.95±1.63 | 50.02±5.83 | 72.88±1.45 | 90.15±0.06 |
| **Improve.** | +0.16 ↑ | +19.49 ↑ | +4.34 ↑ | +2.84 ↑ | +1.34 ↑ | +8.85 ↑ | +1.03 ↑ |

around $EAR = 0.81$, where the performance gap reaches 42.36% in MRR.

*D2: Subgraph Orbit-Biased Aggregation.* The results further demonstrate that D2 (Blue) plays an important role in maintaining robustness under highly automorphic settings, and its impact is more correlated with the underlying dataset characteristics. Removing D2 leads to a gradual degradation in both performance and stability as $EAR$ increases, specifically, removing D2 reduces the average MRR by 26.28% on Syn-Cora and by 10.09% on Syn-Citeseer. Moreover, Syn-Citeseer appears more sensitive to high automorphism levels. Additional results and analyses are provided in Sec. F.

### 5.4 (RQ3) Evaluation on Real-World Graphs

**Significance of EO-GNN.** Tab. 2 shows that EO-GNN consistently achieves the strongest overall performance across both Planetoid and OGB link prediction benchmarks under multiple evaluation metrics. More importantly, the gains become increasingly pronounced on datasets with larger EAR values, such as Pubmed, Citeseer, and Collab, indicating that EO-GNN is particularly effective under strong structural automorphism and automorphism-induced ambiguity. In particular, on Pubmed, SEAL which also incorporates symmetry-aware structural modeling already outperforms conventional baselines by a large margin, while EO-GNN further improves upon SEAL by 19.49 MRR points(28.44%). In contrast, datasets with near-zero EAR values, such as PPA and DDI, exhibit comparatively smaller gains. We additionally observe that pairwise and subgraph-based methods generally achieve stronger robustness than classical message-passing GNNs, highlighting the importance of richer structural representations for link prediction. Nevertheless, as graph size increases, the estimation quality of subtree orbits in Algo. 1 gradually decreases, causing D2 to classify more nodes as structurally equivalent and partially reducing its discriminative capability. Despite this limitation, EO-GNN consistently maintains strong performance with low variance across all benchmarks.

## 6 Conclusion

We highlighted the limitations of node-level symmetry frameworks for link prediction and introduced an edge-centric view via the concept of *edge orbit*. We presented EO-GNN, an automorphism-aware GNN with theoretically grounded components. Together, our framework and design better align GNN inductive biases with graph symmetries, improving both the quantitative theoretical framework and empirical performance in link prediction.

### AI USE STATEMENT

In accordance with the ICLR 2027 AI Policy, we disclose that generative AI tools were used solely to aid and polish writing, specifically grammar correction, improving clarity of exposition, and minor LaTeX formatting. No research content or code was produced with the aid of AI tools, and we take full responsibility for the final content of this work.

### ETHICS STATEMENT

This work is methodological, studying the expressive power of graph neural networks for link prediction. All experiments use publicly available benchmark datasets (the Planetoid citation networks and the Open Graph Benchmark), which contain no personally identifiable or sensitive information and are standard in the community under their respective licenses. The research involves no human subjects, and we foresee no direct risks relating to discrimination, fairness, privacy, or security beyond the general dual-use considerations common to machine-learning research. The authors declare no conflicts of interest.

### REPRODUCIBILITY STATEMENT

We have taken several steps to ensure reproducibility. Our model and experimental code are available as an anonymous repository, linked in the contributions of Sec. 4. All theoretical claims state their assumptions explicitly and are accompanied by complete proofs in Sec. D. The construction of our synthetic benchmarks with controllable automorphism (EAR) is described in detail in Sec. E, and the real-world datasets, their statistics, and the splits used are described in Sec. J. Full hyperparameter settings and training protocols for all methods are reported in Sec. G.2.

## REFERENCES

[1] Sami Abu-El-Haija, Bryan Perozzi, Amol Kapoor, Hrayr Harutyunyan, Nazanin Alipourfard, Kristina Lerman, Greg Ver Steeg, and Aram Galstyan. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *The Thirty-sixth International Conference on Machine Learning (ICML)*, 2019. URL http://proceedings.mlr.press/v97/abu-el-haija19a/abu-el-haija19a.pdf.

[2] Lada Adamic and Eytan Adar. Friends and neighbors on the web. *Social Networks*, 25:211–230, 07 2003. doi: 10.1016/S0378-8733(03)00009-1.

[3] Pablo Barceló, Egor V. Kostylev, Mikaël Monet, Jorge Pérez, Juan L. Reutter, and Juan Pablo Silva. The logical expressiveness of graph neural networks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=r1lZ7AEKvB.

[4] Benjamin Bloem-Reddy and Yee Whye Teh. Probabilistic symmetries and invariant neural networks. *J. Mach. Learn. Res.*, 21:90:1–90:61, 2020. URL https://jmlr.org/papers/v21/19-322.html.

[5] Jan Böker, Ron Levie, Ningyuan Huang, Soledad Villar, and Christopher Morris. Fine-grained expressivity of graph neural networks. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/9200d97ca2bf3a26db7b591844014f00-Abstract-Conference.html.

[6] Benjamin Paul Chamberlain, Sergey Shirobokov, Emanuele Rossi, Fabrizio Frasca, Thomas Markovich, Nils Yannick Hammerla, Michael M. Bronstein, and Max Hansmire. Graph neural networks for link prediction with subgraph sketching. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=m1oqEOAozQU.

[7] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

[8] Zhaoliang Chen, Zhihao Wu, Ylli Sadikaj, Claudia Plant, Hong-Ning Dai, Shiping Wang, Yiu-Ming Cheung, and Wenzhong Guo. Adedgedrop: Adversarial edge dropping for robust graph neural networks. *IEEE Trans. Knowl. Data Eng.*, 37(9):4948–4961, 2025. doi: 10.1109/TKDE.2025.3586369. URL https://doi.org/10.1109/TKDE.2025.3586369.

[9] Zhengdao Chen, Soledad Villar, Lei Chen, and Joan Bruna. On the equivalence between graph isomorphism testing and function approximation with gnns. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/71ee911dd06428a96c143a0b135041a4-Paper.pdf.

[10] Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric, 2019. URL https://arxiv.org/abs/1903.02428.

[11] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 1025–1035, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

[12] Mingguo He, Zhewei Wei, and Ji-Rong Wen. Convolutional neural networks on graphs with chebyshev approximation, revisited. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 7264–7276. Curran Associates, Inc., 2022. doi: 10.52202/068431-0527. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/2f9b3ee2bcea04b327c09d7e3145bd1e-Paper-Conference.pdf.

[13] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, abs/1207.0580, 2012. URL http://arxiv.org/abs/1207.0580.

[14] Weihua Hu, Matthias Fey, Hongyu Ren, Maho Nakata, Yuxiao Dong, and Jure Leskovec. OGB-LSC: A large-scale challenge for machine learning on graphs. In Joaquin Vanschoren and Sai-Kit Yeung (eds.), *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/db8e1af0cb3aca1ae2d0018624204529-Abstract-round2.html.

[15] Yang Hu, Xiyuan Wang, Zhouchen Lin, Pan Li, and Muhan Zhang. Two-dimensional weisfeiler-lehman graph neural networks for link prediction. *CoRR*, abs/2206.09567, 2022. doi: 10.48550/ARXIV.2206.09567. URL https://doi.org/10.48550/arXiv.2206.09567.

[16] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Weinberger. *Deep Networks with Stochastic Depth*, volume 9908, pp. 646–661. 09 2016. ISBN 978-3-319-46492-3. doi: 10.1007/978-3-319-46493-0_39.

[17] Zan Huang, Xin Li, and Hsinchun Chen. Link prediction approach to collaborative filtering. In *Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries*, JCDL '05, pp. 141–142, New York, NY, USA, 2005. Association for Computing Machinery. ISBN 1581138768. doi: 10.1145/1065385.1065415. URL https://doi.org/10.1145/1065385.1065415.

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

[19] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

[20] Veronica Lachi, Francesco Ferrini, Antonio Longa, Bruno Lepri, Andrea Passerini, and Manfred Jaeger. Bridging theory and practice in link representation with graph neural networks. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/b1cd72276feff8173462e3f733ac66f8-Abstract-Conference.html.

[21] Juanhui Li, Harry Shomer, Haitao Mao, Shenglai Zeng, Yao Ma, Neil Shah, Jiliang Tang, and Dawei Yin. Evaluating graph neural networks for link prediction: current pitfalls and new benchmarking. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

[22] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. AAAI'18/IAAI'18/EAAI'18. AAAI Press, 2018. ISBN 978-1-57735-800-8.

[23] David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology*, 58(7):1019–1031, 2007. doi: https://doi.org/10.1002/asi.20591. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/asi.20591.

[24] Derek Lim, Felix Hohne, Xiuyu Li, Sijia Linda Huang, Vaishnavi Gupta, Omkar Bhalerao, and Ser-Nam Lim. Large scale learning on non-homophilous graphs: new benchmarks and strong simple methods. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, Red Hook, NY, USA, 2021. Curran Associates Inc. ISBN 9781713845393.

[25] Derek Lim, Joshua Robinson, Stefanie Jegelka, and Haggai Maron. Expressive sign equivariant networks for spectral geometric learning. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/3516aa3393f0279e04c099f724664f99-Abstract-Conference.html.

[26] George Ma, Yifei Wang, Derek Lim, Stefanie Jegelka, and Yisen Wang. A canonicalization perspective on invariant and equivariant learning. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/702b67152ec4435795f681865b67999c-Abstract-Conference.html.

[27] Haggai Maron, Heli Ben-Hamu, Nadav Shamir, and Yaron Lipman. Invariant and equivariant graph networks. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=Syx72jC9tm.

[28] Haggai Maron, Ethan Fetaya, Nimrod Segol, and Yaron Lipman. On the universality of invariant networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pp. 4363–4371. PMLR, 2019. URL http://proceedings.mlr.press/v97/maron19a.html.

[29] 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 *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The*

*Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pp. 4602–4609. AAAI Press, 2019. doi: 10.1609/AAAI.V33I01.33014602. URL https://doi.org/10.1609/aaai.v33i01.33014602.

[30] Matthew Morris, Bernardo Cuenca Grau, and Ian Horrocks. Orbit-equivariant graph neural networks. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=GkJOCga62u.

[31] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: online learning of social representations. In Sofus A. Macskassy, Claudia Perlich, Jure Leskovec, Wei Wang, and Rayid Ghani (eds.), *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pp. 701–710. ACM, 2014. doi: 10.1145/2623330.2623732. URL https://doi.org/10.1145/2623330.2623732.

[32] Ladislav Rampásek, Michael Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/5d4834a159f1547b267a05a4e2b7cf5e-Abstract-Conference.html.

[33] Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. Dropedge: Towards deep graph convolutional networks on node classification. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=Hkx1qkrKPr.

[34] Harry Shomer, Yao Ma, Haitao Mao, Juanhui Li, Bo Wu, and Jiliang Tang. Lpformer: An adaptive graph transformer for link prediction. In Ricardo Baeza-Yates and Francesco Bonchi (eds.), *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2024, Barcelona, Spain, August 25-29, 2024*, pp. 2686–2698. ACM, 2024. doi: 10.1145/3637528.3672025. URL https://doi.org/10.1145/3637528.3672025.

[35] Damian Szklarczyk, Annika L. Gable, David Lyon, Alexander Junge, Stefan Wyder, Jaime Huerta-Cepas, Milan Simonovic, Nadezhda T. Doncheva, John H. Morris, Peer Bork, Lars Juhl Jensen, and Christian von Mering. STRING v11: protein-protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets. *Nucleic Acids Res.*, 47(Database-Issue):D607–D613, 2019. doi: 10.1093/NAR/GKY1131. URL https://doi.org/10.1093/nar/gky1131.

[36] Matteo Togninalli, M. Elisabetta Ghisu, Felipe Llinares-López, Bastian Rieck, and Karsten M. Borgwardt. Wasserstein weisfeiler-lehman graph kernels. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 6436–6446, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/73fed7fd472e502d8908794430511f4d-Abstract.html.

[37] Antonis Vasileiou, Timo Stoll, and Christopher Morris. Understanding generalization in node and link prediction. *CoRR*, abs/2507.00927, 2025. doi: 10.48550/ARXIV.2507.00927. URL https://doi.org/10.48550/arXiv.2507.00927.

[38] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=rJXMpikCZ.

[39] Stefan Wager, Sida Wang, and Percy Liang. Dropout training as adaptive regularization. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 26:*

*27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pp. 351–359, 2013. URL https://proceedings.neurips.cc/paper/2013/hash/38db3aed920cf82ab059bfccbd02be6a-Abstract.html.

[40] Xiyuan Wang, Haotong Yang, and Muhan Zhang. Neural common neighbor with completion for link prediction. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=sNFLN3itAd.

[41] Boris Weisfeiler and A. A. Lehman. A Reduction of a Graph to a Canonical Form and an Algebra Arising During This Reduction. *Nauchno-Technicheskaya Informatsia*, Ser. 2(N9): 12–16, 1968.

[42] Fengli Xu, Quanming Yao, Pan Hui, and Yong Li. Automorphic equivalence-aware graph neural network. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 15138–15150, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/7ffb4e0ece07869880d51662a2234143-Abstract.html.

[43] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=ryGs6iA5Km.

[44] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/3fe230348e9a12c13120749e3f9fa4cd-Abstract.html.

[45] Seongjun Yun, Seoyoon Kim, Junhyun Lee, Jaewoo Kang, and Hyunwoo J. Kim. Neo-gnns: Neighborhood overlap-aware graph neural networks for link prediction. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 13683–13694, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/71ddb91e8fa0541e426a54e538075a5a-Abstract.html.

[46] Bohang Zhang, Guhao Feng, Yiheng Du, Di He, and Liwei Wang. A complete expressiveness hierarchy for subgraph gnns via subgraph weisfeiler-lehman tests. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 41019–41077. PMLR, 2023. URL https://proceedings.mlr.press/v202/zhang23k.html.

[47] Bohang Zhang, Jingchu Gai, Yiheng Du, Qiwei Ye, Di He, and Liwei Wang. Beyond weisfeiler-lehman: A quantitative framework for GNN expressiveness. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=HSKaGOi7Ar.

[48] Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 5171–5181, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/53f0d7c537d99b3824f0f99d62ea2428-Abstract.html.

[49] Muhan Zhang, Pan Li, Yinglong Xia, Kai Wang, and Long Jin. Labeling trick: A theory of using graph neural networks for multi-node representation learning. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 9061–9073, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/4be49c79f233b4f4070794825c323733-Abstract.html.

[50] Tao Zhou, Linyuan Lü, and Yi-Cheng Zhang. Predicting missing links via local information. *The European Physical Journal B*, 71:623–630, 10 2009. doi: 10.1140/epjb/e2009-00335-8.

[51] Zhaocheng Zhu, Zuobai Zhang, Louis-Pascal Xhonneux, and Jian Tang. Neural bellman-ford networks: a general graph neural network framework for link prediction. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, Red Hook, NY, USA, 2021. Curran Associates Inc. ISBN 9781713845393.

# A NOMENCLATURE

We summarize the main symbols used in this work and their definitions below:

Table 3: Major symbols and definitions.

| Symbols | Definitions |
|---|---|
| $\mathcal{H} = (\mathcal{V}, \mathcal{E})$ | graph $\mathcal{H}$ with nodeset $\mathcal{V}$, edgeset $\mathcal{E}$ |
| $\mathbf{A}$ | $n \times n$ adjacency matrix of $\mathcal{H}$ |
| $\mathbf{X}$ | $n \times F$ node feature matrix of $\mathcal{H}$ |
| $\mathbf{x}_v$ | $F$-dimensional feature vector for node $v$ |
| $\mathbf{L}$ | unnormalized graph Laplacian matrix |
| $\mathcal{G}/\mathrm{Aut}(\mathcal{G})$ | the quotient of the automorphism when it acts on the graph $\mathcal{G}$ |
| $\mathrm{Aut}(\mathcal{G})$ | automorphism on graph $\mathcal{G}$ |
| $\pi$ | vertex permutation; $\pi \in \mathrm{Aut}^k(\mathcal{G})$ denotes a $k$-hop graph automorphism |
| $N(v)$ | general type of neighbors of node $v$ in graph $\mathcal{H}$ |
| $\bar{N}(v)$ | general type of neighbors of node $v$ in $\mathcal{H}$ *without self-loops* (i.e., excluding $v$) |
| $N_i(v), \bar{N}_i(v)$ | $i$-hop/step neighbors of node $v$ in $\mathcal{H}$ (at exactly distance $i$) maybe-with/without self-loops, resp. |
| $\mathcal{E}_2$ | set of pairs of nodes $(u, v)$ with shortest distance between them being 2 |
| $d, d_{\max}$ | node degree and maximum node degree across all nodes $v \in \mathcal{V}$, resp. |
| $K$ | the number of rounds in the neighborhood aggregation stage |
| $\mathbf{W}$ | learnable weight matrix for GNN model |
| $\rho$ | non-linear activation function |
| $[\cdot\|\cdot]$ | vector concatenation operator |
| `AGGR` | function that aggregates node feature representations within a neighborhood |
| `COMBINE` | function that combines feature representations from different neighborhoods |
| $\mathcal{P}(\mathcal{V})$ | Power set of vertex set $\mathcal{V}$ |
| $\mathcal{W}$ | Weight space of the parameterized neural network; $w \in \mathcal{W}$ |
| $\phi$ | Graph encoder |
| $\mathcal{O}(v)$ | orbit of vertex $v$ under $\mathrm{Aut}(\mathcal{G})$ |
| $O$ | $O \in \mathbb{Z}^{\lvert\mathcal{V}\rvert}$ WL-Hash value from Algo. 1 |
| $\mathbf{O}$ | $\mathbf{O} \in \mathbb{Z}^{\lvert\mathcal{V}\rvert \times d_o}$ embedded node feature based on the hash label from Algo. 1 as node feature |
| $\lvert\mathcal{O}_u\rvert$ | orbit size of vertex $u$: number of the vertices sharing the same role as $u$ |
| $\mathcal{O}_{\mathcal{S}}(u) (= \mathcal{O}_u)$ | subgraph orbit of $u$: its $k$-equivalence class under a GNN (Def. 3.3) |
| $\mathcal{O}_{\mathcal{E}}(e)$ | WL-induced edge-orbit signature of $e = \{u, v\}$: the multiset $\{\{\mathcal{O}_{\mathcal{S}}(u), \mathcal{O}_{\mathcal{S}}(v)\}\}$ (Def. 3.4) |
| $[e]_{\mathcal{E}}$ | edge orbit of $e$: the edges sharing its signature, i.e. its $\sim^{\mathcal{E}}$ equivalence class (Measure 3.1) |
| $\mathcal{O}(\cdot)$ | Complexity |
| $D$ | Decoder; typically implemented as a dot product |
| $\mathcal{M}_w$ | The full link-prediction model for weight set $w$, i.e., the GNN encoder $\phi_w$ composed with a decoder $D$; $D$'s functional form is left unspecified |
| $\mathcal{S}^{(k)}(u)$ | $k$-hop enclosing subgraph centered at node $u$ |
| $\mathcal{S}^{(k)}(u)_\epsilon$ | $k$-hop enclosing subgraph of $u$ with $\epsilon$-level structural perturbation |
| $\epsilon$ | Degree of structural alteration; an embedded distance in the representation space, i.e. the desired closeness to the unperturbed graph |
| $\alpha_{\mathcal{V}}$ | Proposed vertex-level automorphism measure (see Appendix **??**) |
| $\alpha_{\mathcal{E}}$ | Proposed edge-level automorphism measure (see Appendix **??**) |
| $\widetilde{A}$ | Adjacency matrix after random dropout |
| $\delta$ | Permissible distance from $a$ for input |
| $\sigma$ | bound on failure probability; a guarantee is stated to hold with probability at least $1 - \sigma$ (Thm. 4.1) |
| $\tau$ | standard deviation of the Gaussian perturbation noise used in D2 (Sec. 4.2) |
| $\mathbf{r}^{(0)}$ | Embedding vector from WL-Hash $\mathbf{o}$ for input in iteration 0 |
| $\mathbf{E}_e$ | weight of graph adjacency matrix |
| $\mathbf{X}^{(K)}$ | final embedding of GNN |
| $\mathbf{H}_c$ | common neighbor as feature vector, $\mathbf{H}_c(i, j) = 1$ if $(i, j) \in \mathcal{E}$ |
| $\mathbf{R}_v^{(0)}$ | embedding from embedded WL-hash Label |
| $\mathbf{H}_e$ | embedding from |
| $\mathbf{W}_f$ | weight $\mathbf{W}_f \in \mathbb{R}^{p \times 1}$ in the decoder |
| $p$ | feature dimension in the final embedding |

## B Extended Preliminary

**Definition B.1** (Graph Quotient)**.** *The* quotient *of a graph $\mathcal{G}$ is the partition of its vertex set into automorphic orbits under the action of* $\mathrm{Aut}(\mathcal{G})$*:* $\mathcal{G} \,/\, \mathrm{Aut}(\mathcal{G}) = \{\mathcal{O}(v) \mid v \in \mathcal{V}_{\mathcal{G}}\}$.

**Subgraph Orbit $\neq$ Orbit.** The difference between *Subgraph Orbit* and standard *Orbit* lies in the group action they are based on: an orbit is induced by is a *global* automorphism (i.e., an adjacency-preserving permutation of the entire node set), whereas subgraph orbit in link prediction is a local generalization of automorphism based on the fact that the GNN operates on a small $k$-hop neighborhood. In Fig. 1, nodes b and d belong to the same subgraph orbit under a 1-layer GNN, as their 1-hop enclosing subgraphs are structurally identical. *Motivation: It ensures that structural equivalence is rigorously characterized through the isomorphism of GNN-induced local enclosing subgraphs.*

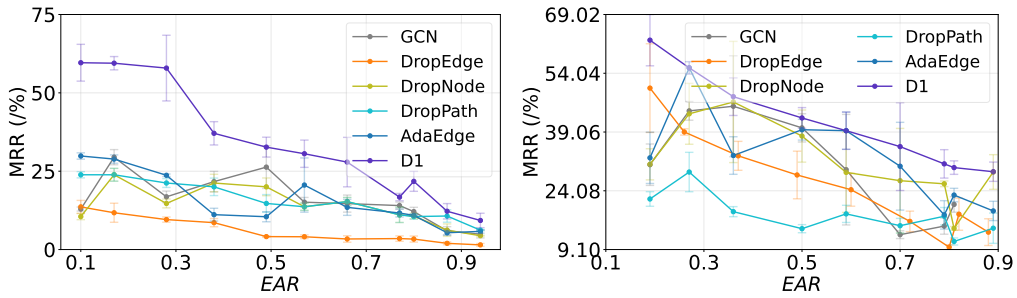
## C Extended Related Work

**MPNNs for Link Prediction.** Link prediction is commonly performed using embeddings generated by GCN-based encoders followed by simple decoders such as the dot product [11, 38, 43]. However, Zhang & Chen [48] showed that these vanilla GCNs cannot distinguish automorphic nodes due to their inherent permutation invariance. To address this limitation, Zhang et al. [49] introduced node labeling features that encode relative distances between target nodes and their neighborhoods, enhancing structural discrimination. Subsequent work incorporated manually designed structural priors, such as the Jaccard index [6, 45], while Wang et al. [40] proposed soft completion of common-neighbor features to mitigate distribution shifts and structural holes. Recent work, including the introduction of higher-order shortest path methods and PageRank attention-based frameworks, has attempted to facilitate link prediction by utilizing manually extracted local structures [34]. Our work, in contrast, aims to provide a principled solution within a permutation invariance framework.

**Expressivity Comparison.** The closest method to ours is $k_\phi$-$k_\rho$-$m$-WL. However, our approach differs in three key aspects: (i) *Motivation.* The $k_\phi$-$k_\rho$-$m$-WL framework provides a unified view for hierarchically comparing the aggregation and update designs of existing methods relative to 1-WL. In contrast, our work defines model expressiveness by explicitly identifying, quantifying and measuring indistinguishable edges in standard GNNs in a model-agnostic manner. (ii) *Objective.* While $k_\phi$-$k_\rho$-$m$-WL aims to compare the expressiveness of existing approaches for link prediction, our framework studies graph automorphisms in real-world graphs and analyzes their impact on link prediction across different categories of GNNs. (iii) *Measure and synthetic benchmark.* The $k_\phi$-$k_\rho$-$m$-WL framework proposes a vertex-level measure to compare graph automorphisms, which leads to a mismatch with the edge-level nature of the problem we study. In contrast, our proposed measure is an empirical estimator derived directly from our theoretical framework. This estimator is consistently aligned with our method design, our synthetic benchmark and our theoretical analysis.

**Dropout Comparison.** *Dropout* [13] was originally introduced to mitigate overfitting by randomly deactivating hidden units during training. Subsequent analysis showed that dropout implicitly induces a regularization term that is first-order equivalent to an $L_2$ penalty after feature scaling [39]. In graph learning, dropout primarily has three variants. *DropEdge* [33] extends dropout by randomly removing edges during message passing, effectively alleviating the over-smoothing problem in GNNs [7, 22]. *DropNode* [44] further generalizes this idea by randomly discarding vertices and their incident edges, serving as a powerful augmentation method that enhances model generalizability, transferability and robustness. *DropPath* [16] stochastically masks random-walk–based paths within the graph, reducing redundancy in local substructures. More recently, *AdaEdge* [8] optimizes the graph topology in a learnable fashion by adaptively applying dropout according to local homophily ratios during training. We empirically compare Automorphism-aware Dropout with other dropout variants on the synthetic benchmark in Sec. 5.3.

**Summary** While prior works utilized dropout to address various issues in GNN. We are the first to leverage the automorphism prior derived from the WL test to optimize the dropout process. This design choice uniquely enables us to mitigate degradations in GNN model expressiveness, which we both theoretically justify and empirically demonstrate on diverse datasets. We note that our




(a) `syn-citeseer` (MRR) (b) `syn-cora` (MRR)

Figure 9: **Empirical Comparison of Dropout Variants on Synthetic Benchmarks.** Ablation study comparing our proposed method, Automorphism-aware Dropout (D1), against three popular dropout approaches (DropEdge, DropNode, DropPath, AdaEdge and DropPath) on synthetic datasets *syn-citeseer* and *syn-cora*. We show that Automorphism-aware Dropout consistently achieves higher MRR than the other variants. This demonstrates that leveraging the WL test's automorphism prior allows Automorphism-aware Dropout to identify and prune automorphic (structurally identical) edges more effectively. It enhances the model's expressiveness to distinguish non-isomorphic local structures. (See Sec. 4 for method details and Sec. 5.3 for other empirical evaluation)

design philosophy is similar to the adaptive nature of AdaEdge, but our method specifically mitigates automorphism as opposed to optimizing for homophily labels.

# D Proofs and Discussions of Theorems

## D.1 Proof for Edge Orbit Equivariance

**Proposition D.1** (Edge-Orbit Equivariance)**.** *Let $\mathcal{G}$ be an undirected graph with automorphism group $\mathrm{Aut}(\mathcal{G})$ (Def. 3.1), and let $\mathcal{O}_{\mathcal{E}}(e) = \{\{\mathcal{O}_{\mathcal{S}}(u), \mathcal{O}_{\mathcal{S}}(v)\}\}$ be the WL-induced edge orbit of $e = \{u, v\}$ (Def. 3.4). Then the edge-orbit assignment is* invariant *under automorphisms: for every $\phi \in \mathrm{Aut}(\mathcal{G})$ and every $e = \{u, v\}$,*

$$\mathcal{O}_{\mathcal{E}}(\phi(u), \phi(v)) = \mathcal{O}_{\mathcal{E}}(u, v).$$

*For a permutation $\phi \notin \mathrm{Aut}(\mathcal{G})$ the equality need not hold, since $\phi$ may neither preserve subgraph orbits nor map edges to edges.*

*Proof.* The argument rests on the fact that automorphisms preserve subgraph orbits. Fix $\phi \in \mathrm{Aut}(\mathcal{G})$ and a node $x \in \mathcal{V}$. Because $\phi$ is adjacency-preserving, it maps the $k$-hop rooted subgraph $\mathcal{S}^{(k)}(x)$ isomorphically onto $\mathcal{S}^{(k)}(\phi(x))$; a message-passing GNN is invariant to such rooted-subgraph isomorphisms, so $\phi_w\big(\mathcal{S}^{(k)}(x)\big) = \phi_w\big(\mathcal{S}^{(k)}(\phi(x))\big)$ for all $w$. By Def. 3.3 this gives $x \sim \phi(x)$, i.e.

$$\mathcal{O}_{\mathcal{S}}(\phi(x)) = \mathcal{O}_{\mathcal{S}}(x) \qquad \text{for all } x \in \mathcal{V}. \tag{7}$$

Since $\mathcal{O}_{\mathcal{E}}$ is the multiset of the endpoints' subgraph orbits, applying Eq. (7) to both endpoints of $e = \{u, v\}$ yields

$$\mathcal{O}_{\mathcal{E}}(\phi(u), \phi(v)) = \{\{\mathcal{O}_{\mathcal{S}}(\phi(u)), \mathcal{O}_{\mathcal{S}}(\phi(v))\}\} = \{\{\mathcal{O}_{\mathcal{S}}(u), \mathcal{O}_{\mathcal{S}}(v)\}\} = \mathcal{O}_{\mathcal{E}}(u, v),$$

which proves the claim. Conversely, if $\phi \notin \mathrm{Aut}(\mathcal{G})$ then $\phi$ need not preserve the enclosing subgraphs, so Eq. (7) can fail and the two multisets may differ; indeed $\phi(u), \phi(v)$ may not even form an edge. Hence the assignment is equivariant exactly on $\mathrm{Aut}(\mathcal{G})$. □

## D.2 Detailed Analysis of Theorem 4.1

We begin by introducing key concepts to formalize subgraph orbit-distinguishability. We then define the Node Automorphism Problem and show that the proposed D1 provides a solution.

**Definition D.1** ($\epsilon$-sufficient)**.** *We define an alteration as* $\epsilon$-sufficient *if, for all subgraph* $\mathcal{S}^{(k)} \in \mathcal{G}$*, there exists an altered version* $\mathcal{S}_\epsilon^{(k)}$*, such that* $\langle \phi_w(\mathcal{S}_\epsilon^{(k)}(u)), \phi_w(\mathcal{S}^{(k)}(u))) > 0$ *under GNN encoder* $\phi_w$ *and decoder* $D$*.* $\epsilon$ *denotes the magnitude of change in the embedding space resulting from a structural perturbation in the graph.*

**Definition D.2** (Node Automorphism Problem)**.** *A GNN is said to be* $\epsilon$-complete *if there exists an* $\epsilon$-sufficient alteration *such that for all* $u \in \mathcal{V}$*, the following holds:*

$$D\left(\phi_w(\mathcal{S}_\epsilon^{(k)}(u)), \phi_w(\mathcal{S}^{(k)}(u))\right) \geq \epsilon \tag{8}$$

*Automorphic nodes are defined as nodes that are* $\epsilon$*-incomplete under the action of* $(\phi, D)_w$*.*

We reformulate the Theorem as follows:

*Proof.* For any $\epsilon > 0$ with probability $p_\epsilon$, we expect one bounded $\mathbb{L}$ exists, such that for all $l \geq \mathbb{L}$, $\mathcal{S}' = \mathcal{B}(\mathcal{S})$

$$(\phi_{w_1}(\mathcal{S}), \ldots, \phi_{w_\ell}(\mathcal{S})) \neq (\phi_{w_1}(\mathcal{S}'), \ldots, \phi_{w_\ell}(\mathcal{S}')) \tag{9}$$

holds with probability $1 - \sigma$ , where random variables $w \sim \mathcal{W}$ . It particularly implies that one closure exists $\mathcal{B}(\mathcal{S})_\epsilon$ with $\Pr(\mathcal{S} \in \mathcal{B}(\mathcal{S})_\epsilon) = \Pr_{\mathcal{S},\mathcal{S}'} = 1 - (1-p)^{|\mathcal{E}|} > 0$, such that for all subgraph orbits $\phi_\mathcal{B}(\mathcal{S}) \neq \phi_\mathcal{B}(\mathcal{S}')$
We need $\Pr(\exists i \in \{1, \ldots, \ell\} : \mathcal{S} \in \mathcal{S}') \geq 1 - \sigma$, hence

$$1 - (1 - \Pr_{\mathcal{S},\mathcal{S}'})^\ell \geq 1 - \sigma \tag{10}$$

must hold. Solving for $\ell$, we find that

$$\mathbb{L} = \left\lceil \frac{\log(1/\sigma)}{\log\left(\frac{1}{1-\Pr_{\mathcal{S},\mathcal{S}'}}\right)} \right\rceil \tag{11}$$

is sufficient to guarantee that there will be at least one $\mathcal{S}$ in $\mathcal{S}'$ with probability at least $1-\sigma$, implying $\phi_w(\mathcal{S}) \neq \phi_w(\mathcal{S}')$, which closes the proof. □

We further simplify the Eq. (11) to provide a qualitative analysis for $\mathbb{L}$ w.r.t $p$ and $|\mathcal{E}|$:

$$\mathbb{L} = \left\lceil \frac{\log(1/\sigma)}{\log\left(\frac{1}{1-\Pr_{\mathcal{S},\mathcal{S}'}}\right)} \right\rceil \tag{12}$$

Given that

$$\Pr_{\mathcal{S},\mathcal{S}'} = 1 - (1-p)^{|\mathcal{E}|}, \tag{13}$$

we substitute into the definition of $\mathbb{L}$:

$$\begin{aligned} \mathbb{L} &= \left\lceil \frac{\log(1/\sigma)}{\log\left(\frac{1}{1-(1-(1-p)^{|\mathcal{E}|})}\right)} \right\rceil \\ &= \left\lceil \frac{\log(1/\sigma)}{\log\left(\frac{1}{(1-p)^{|\mathcal{E}|}}\right)} \right\rceil. \end{aligned} \tag{14}$$

setting in $\log(1/x) = -\log(x)$, we have

$$\log\left(\frac{1}{(1-p)^{|\mathcal{E}|}}\right) = -\log\left((1-p)^{|\mathcal{E}|}\right). \tag{15}$$

Applying the power rule for logarithms $\log(x^n) = n\log(x)$, we obtain

$$-\log\left((1-p)^{|\mathcal{E}|}\right) = -|\mathcal{E}|\log(1-p). \tag{16}$$

Thus,

$$\mathbb{L} = \left\lceil \frac{\log(1/\sigma)}{-|\mathcal{E}|\log(1-p)} \right\rceil. \tag{17}$$

Since $\log(1/\sigma) = -\log(\sigma)$, it further simplifies to

$$\mathbb{L} = \left\lceil \frac{-\log(\sigma)}{-|\mathcal{E}|\log(1-p)} \right\rceil = \left\lceil \frac{\log(\sigma)}{|\mathcal{E}|\log(1-p)} \right\rceil. \tag{18}$$

**Final simplified form:**

$$\boxed{\mathbb{L} = \left\lceil \frac{\log(\sigma)}{|\mathcal{E}|\log(1-p)} \right\rceil} \tag{19}$$

**Qualitative Analysis:** Since $\log(1-p) < 0$ for $p \in (0,1)$ and $\log(\sigma) < 0$ for $\sigma \in (0,1)$, the ratio $\frac{\log(\sigma)}{|\mathcal{E}|\log(1-p)}$ is positive.

Furthermore, as $p$ increases $\mathbb{L}$ decreases; but when $|\mathcal{E}|$ is fixed and the graph becomes larger, Suppose $|\mathcal{E}|$ is fixed, but the graph becomes larger (i.e., the number of nodes increases). Since

$$(1-p)^{|\mathcal{E}|} \approx 1 - p|\mathcal{E}|, \tag{20}$$

for small $p$, we apply first-order Taylor approximation, then have

$$\Pr_{\mathcal{S},\mathcal{S}'} \approx p|\mathcal{E}|. \tag{21}$$

Thus, to keep $\Pr_{\mathcal{S},\mathcal{S}'}$ approximately constant as the graph grows, $p$ should satisfy

$$p \propto \frac{1}{|\mathcal{E}|}. \tag{22}$$

*the per-edge probability $p$* should be increased proportionally to maintain a stable success probability and avoid increasing $\mathbb{L}$ excessively.

### D.3 Detailed Analysis of Theorem 4.2

**Theorem D.1.** *Let $G$ be a lobster graph containing an automorphism-dominant node $v_a$ and let $\phi(\cdot)$ denote the embedding function of a $K$-layer GNN. In standard GNNs, the average embedding distance between $v_a$ and its automorphic neighbors is greater than that between $v_a$ and its non-automorphic neighbors:*

$$\begin{aligned} &\mathbb{E}_{u\in\mathcal{N}(v)}\left[\|\phi(u)-\phi(v)\|^2 \mid \mathcal{O}_u \neq \mathcal{O}_v\right] \\ &\quad - \mathbb{E}_{u\in\mathcal{N}(v_a)}\left[\|\phi(u)-\phi(v_a)\|^2 \mid \mathcal{O}_u = \mathcal{O}_{v_a}\right] > \delta \end{aligned} \tag{23}$$

*for some $\delta > 0$. A GNN equipped with D2 reduces this gap caomparing to standard GNN without.*

We start with a concrete example: lobster graph illustrated in Fig. 4a

**Lemma D.1.** *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an undirected graph with $|\mathcal{V}| = 10$ nodes and let $\mathbf{X} \in \mathbb{R}^{10\times 1}$ be the node feature matrix where all node features are identical, i.e.,*

$$\mathbf{X} = \mathbf{1}_N \cdot \mathbf{x}_0^\top$$

*for some fixed vector $\mathbf{x}_0 \in \mathbb{R}^d$. Let $\hat{\mathbf{A}} \in \mathbb{R}^{N\times N}$ be the normalized adjacency matrix used in a one-layer GCN. Then the output embedding $\mathbf{H} \in \mathbb{R}^{N\times d'}$ satisfies:*

$$\mathbf{H} = \hat{\mathbf{A}}\mathbf{X}\mathbf{W} = \mathbf{c} \cdot (\mathbf{x}_0^\top \mathbf{W}),$$

*where $\mathbf{c} \in \mathbb{R}^N$ is a vector with entries $c_i = \sum_{j=1}^N \hat{A}_{ij}$ and $\mathbf{W} \in \mathbb{R}^{d\times d'}$ is the trainable weight matrix.*

*Proof.* By the standard GCN layer definition, we have:

$$\mathbf{H} = \hat{\mathbf{A}}\mathbf{X}\mathbf{W}.$$

We replace $\mathbf{X} = \mathbf{1}_N \cdot \mathbf{x}_0^\top$, then:

$$\hat{\mathbf{A}}\mathbf{X} = \hat{\mathbf{A}}(\mathbf{1}_N \cdot \mathbf{x}_0^\top) = (\hat{\mathbf{A}}\mathbf{1}_N) \cdot \mathbf{x}_0^\top.$$

We represent $\mathbf{c}$ in matrix form $\mathbf{c} = \hat{\mathbf{A}}\mathbf{1}_N$. Then:

$$\hat{\mathbf{A}}\mathbf{X} = \mathbf{c} \cdot \mathbf{x}_0^\top.$$

Therefore:

$$\mathbf{H} = (\hat{\mathbf{A}}\mathbf{X})\mathbf{W} = (\mathbf{c} \cdot \mathbf{x}_0^\top)\mathbf{W} = \mathbf{c} \cdot (\mathbf{x}_0^\top \mathbf{W}).$$

□

We see that three vertices are automorphic; a special case is when two nodes $u, v \in \mathcal{V}$ have identical rows in $\hat{\mathbf{A}}$ in the lobster graph. We characterize such intuition as $\hat{A}_{u,:} = \hat{A}_{v,:}$. Assuming we have no dropout, injected noise and all weights $\mathbf{W}$ are shared, we have the following deviation, substituting $\mathbf{X} = \mathbf{1}_N \cdot \mathbf{x}_0^\top$:

$$\mathbf{H} = \hat{\mathbf{A}}(\mathbf{1}_N \cdot \mathbf{x}_0^\top)\mathbf{W}.$$
$$\mathbf{H} = (\hat{\mathbf{A}}\mathbf{1}_N) \cdot (\mathbf{x}_0^\top \mathbf{W}).$$

Define $\mathbf{z} = \hat{\mathbf{A}}\mathbf{1}_N \in \mathbb{R}^{N\times 1}$ and $\mathbf{v} = \mathbf{x}_0^\top \mathbf{W} \in \mathbb{R}^{1\times d'}$, $\mathbf{H} = \mathbf{z} \cdot \mathbf{v}$ which implies that each row of $\mathbf{h}$ is given by:

$$\mathbf{h}_i = z_i \cdot \mathbf{v} = \sum_{j=1}^{N} \hat{A}_{ij}\mathbf{v} = k\mathbf{D}_i\mathbf{v}.$$

We now compute the two terms in Eq. (23) to illustrate the embedding difference between automorphic and non-automorphic neighbors of an automorphism-dominant node.

$$\begin{aligned} &\mathbb{E}_{u\in\mathcal{N}(v)}\left[\|\phi(u) - \phi(v)\|^2 \mid \mathcal{O}_u \neq \mathcal{O}_v\right] \\ &= \frac{k(\mathbf{D}_u - \mathbf{D}_v)\mathbf{v}}{N_{nauto}} \\ &= 2/7 = 0.2857 \end{aligned} \tag{24}$$

where $\mathbf{D}_u$ and $\mathbf{D}_v$ represent the degrees of the neighboring and center nodes respectively, $\mathbf{v}$ is the learned weight vector and $N_{\text{non-auto}} = 7$ is the number of non-automorphic neighbors. This reflects the average squared embedding distance due to degree gaps, most automorphic vertices have identical degrees except two tail nodes.

Next, we calculate the expected distance among automorphic neighbors:

$$\begin{aligned} &\mathbb{E}_{u\in\mathcal{N}(v_a)}\left[\|\phi(u) - \phi(v_a)\|^2 \mid \mathcal{O}_u = \mathcal{O}_v\right] \\ &= \frac{k(\mathbf{D}_u - \mathbf{D}_v)\mathbf{v}}{N_{nauto}} \\ &= 1 + 3 + 3 + 3/4 \\ &= 2.5 \end{aligned} \tag{25}$$

where the degree difference is 3 for automorphic neighbors (e.g., between vertex 1 and other symmetric vertices) and 1 for the tail node. These results indicate that standard GNNs assign larger embedding differences to automorphic edges compared to non-automorphic ones—contradicting structural automorphism. This directly supports Thm. 4.2.

We have now

$$\begin{aligned} &\mathbb{E}_{u\in\mathcal{N}(v_a)}\left[\|\phi(u) - \phi(v_a)\|^2 \mid \mathcal{O}_u = \mathcal{O}_v\right] < 0.2857 \\ &= \frac{k(\mathbf{D}_u - \mathbf{D}_v)\mathbf{v}}{N_{nauto}} \\ &= 1w_1 + 3w_2 + 3w_3 + 0.75w_4 < 0.2857 \end{aligned} \tag{26}$$

We provide one valid solution to the inequality using negative weights:

$$w_1 = 0.1, \quad w_2 = 0.05, \quad w_3 = -0.1, \quad w_4 = 0.4$$

Substituting into the left-hand side of the inequality:

$$\begin{aligned} &1w_1 + 3w_2 + 3w_3 + 0.75w_4 \\ &= 1(0.1) + 3(0.05) + 3(-0.1) + 0.75(0.4) \\ &= 0.1 + 0.15 - 0.3 + 0.3 = 0.25 < 0.2857 \end{aligned}$$

Thus, this set of weights satisfies the inequality.

# E Synthetic Datasets: Details

In this section, we present our data generation process, dataset details, the sources of our baselines and the corresponding hyperparameters.

## E.1 Detailed Results on Semi-Synthetic Benchmarks

**Automorphic Real Synthetic Graph Construction** We utilize a real world graph $\mathcal{H}$ (Cora and Citeseer). Then we form a larger graph $\mathcal{H}^2$ that contains two disjoint copies of $\mathcal{H}$, along with 1000 uniformly-randomly added edges (both between (inter) and within (intra) copies of $\mathcal{H}$). Without the random edges, each node in one copy of $\mathcal{H}$ is automorphic to the corresponding node in the other copy, as increasing inter, intra probability and number of edges, $EAR$ increases. We use three parameters inter $p_i$, intra $p_s$, introduced automorphic nodes $a$ and number of perturbed edges $|\mathcal{E}|$ to control the $EAR$. The utilized parameters for Syn-Cora & Syn-Citeseer are provided in Tab. 4.

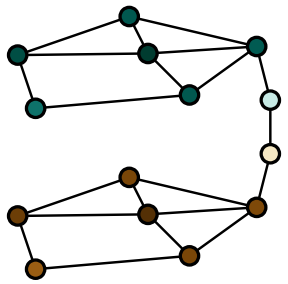


Figure 10: Syn-Cora & Syn-Citeseer

Figure 11: Symmetric automorphic graph construction to change the automorphism in real world graph.

Table 4: Statistics for Synthetic Datasets

| **Benchmark Name** | `syn-cora` | `syn-citeseer` |
|---|---|---|
| **# Nodes** | 5416 | 6654 |
| **# Edges** | 10556 to 17549 | 9104 to 23098 |
| **automorphism** $EAR$ | [0, 0.1, . . ., 0.9] | [0, 0.1, . . ., 0.9] |
| **Degree Range** | 1 to 168 | 1 to 107 |
| **Average Degree** | 3.89 to 6.48 | 2.73 to 6.94 |
| **Inter Prob.** | 0.1 | 0.1 |
| **Intra Prob.** | 0.5 | 0.5 |
| **Number of Edges** | [0.2, 1, 4, 7, 12, 18, 20, 28]×250 | [0.2, 1, 2, 3, 4, 5, 7, 8, 10, 14]×1000 |
| $EAR$ | [0.19, 0.30, 0.37, 0.51, 0.60, 0.71, 0.81, 0.83, 0.90] | [0.10, 0.18, 0.28, 0.38, 0.48, 0.57, 0.65, 0.77, 0.81, 0.88, 0.94] |

Table 5: SYN-CORA: Mean metrics (MRR) and standard deviation for each method on synthetic datasets with varying automorphism ratios $EAR$.

| Method | 0.19 | 0.30 | 0.37 | 0.51 | 0.60 | 0.71 | 0.81 | 0.83 | 0.90 |
|---|---|---|---|---|---|---|---|---|---|
| GCN | 99.41±.23 | 99.39±.09 | 99.06±.22 | 98.51±.35 | 98.01±.45 | 97.27±.42 | 96.45±.57 | 96.29±.67 | 94.07±.86 |
| GAT | 99.28±.16 | 98.97±.23 | 98.99±.13 | 98.75±.20 | 98.48±.25 | 97.98±.22 | 97.72±.50 | 97.51±.50 | 96.42±.39 |
| GIN | 91.83±.70 | 95.76±.78 | 95.53±.59 | 94.91±.90 | 84.81±.14 | 82.49±.14 | 70.24±.78 | 72.01±.96 | 65.24±.38 |
| GraphSAGE | 97.91±.62 | 97.96±.57 | 97.96±.60 | 97.05±.85 | 95.93±.93 | 93.38±.99 | 90.27±1.93 | 90.37±2.24 | 85.82±2.03 |
| MixHopGCN | 99.55±.17 | 99.28±.09 | 98.99±.13 | 98.75±.20 | 98.48±.25 | 97.98±.22 | 97.72±.50 | 97.51±.50 | 96.42±.39 |
| ChebGCN | 98.41±.13 | 98.19±.22 | 97.82±.21 | 96.56±.79 | 96.76±.32 | 94.09±.56 | 91.18±1.02 | 86.24±1.77 | 84.88±1.42 |
| GCN-DW | 99.89±.04 | 99.66±.06 | 99.55±.07 | 99.51±.05 | 99.19±.10 | 98.81±.06 | 98.34±.21 | 98.15±.10 | 97.54±.24 |
| GCN-RF | 99.81±.07 | 99.81±.03 | 99.60±.05 | 99.67±.08 | 98.92±.07 | 98.84±.15 | 98.43±.15 | 98.42±.18 | 97.85±.12 |
| GCN-LAP | 99.46±.16 | 99.18±.18 | 98.99±.27 | 98.77±.27 | 97.83±.41 | 97.16±.38 | 96.91±.29 | 96.54±.73 | 95.39±.48 |
| BUDDY | 95.34±.17 | 94.00±.23 | 95.03±.17 | 95.34±.04 | 94.82±.18 | 94.63±.30 | 94.06±.18 | 93.94±.10 | 93.80±.10 |
| LINKX | 97.30±.32 | 96.99±.15 | 96.81±.18 | 94.28±.15 | 92.82±.13 | 89.63±.29 | 86.69±.15 | 85.47±.12 | 83.48±.18 |
| Proposed w.o. D1 | 99.89±.08 | 99.83±.10 | 99.77±.11 | 99.76±.07 | 99.76±.05 | 99.74±.07 | 99.67±.12 | 99.63±.07 | 99.62±.22 |
| Proposed w.o D2 | 99.90±.05 | 99.80±.05 | 99.79±.07 | 99.78±.12 | 99.75±.03 | 99.74±.07 | 99.73±.04 | 99.71±.03 | 99.90±.05 |
| Proposed | **99.94**±.03 | **99.86**±.03 | **99.83**±.07 | **99.81**±.00 | **99.80**±.05 | **99.78**±.04 | **99.76**±.03 | **99.74**±.07 | **99.93**±.08 |

Table 6: SYN-CORA: Mean Metrics (MRR) and standard deviation for each method on synthetic datasets with varying automorphism ratios $EAR$.

| Method | 0.19 | 0.30 | 0.37 | 0.51 | 0.60 | 0.71 | 0.81 | 0.83 | 0.90 |
|---|---|---|---|---|---|---|---|---|---|
| GCN | 42.28±11.15 | 34.09±8.75 | 27.06±5.83 | 21.29±6.17 | 20.83±3.29 | 18.37±3.90 | 16.50±3.50 | 14.78±4.48 | 12.82±3.91 |
| GAT | 30.45±6.74 | 25.24±7.63 | 21.77±7.00 | 20.28±5.60 | 19.74±9.83 | 17.06±6.54 | 15.60±3.41 | 14.80±4.85 | 13.28±3.10 |
| GIN | 15.80±2.59 | 15.57±5.25 | 14.43±3.99 | 10.88±4.81 | 9.19±3.93 | 6.87±4.00 | 4.42±3.89 | 4.06±3.72 | 3.07±3.30 |
| GraphSAGE | 24.79±8.27 | 24.20±4.74 | 17.53±5.70 | 16.30±5.24 | 15.45±3.57 | 13.66±4.13 | 12.26±5.25 | 11.01±3.77 | 8.94±2.54 |
| MixHopGCN | 65.87±14.64 | 58.62±11.24 | 17.65±29.31 | 0.12±0.00 | 0.12±0.00 | 0.12±0.00 | 0.12±0.00 | 0.12±0.00 | 0.12±0.00 |
| ChebGCN | 28.71±4.29 | 24.71±5.39 | 21.38±5.21 | 21.01±4.68 | 19.82±5.16 | 16.75±4.98 | 15.35±3.71 | 10.26±2.32 | 9.77±2.48 |
| GCN-DW | 72.87±12.57 | 43.98±7.79 | 29.52±4.48 | 40.14±5.64 | 53.73±11.90 | 35.98±3.61 | 42.41±6.86 | **50.71**±9.77 | 21.16±3.57 |
| GCN-RF | 52.38±13.90 | 48.97±6.06 | 44.79±10.38 | 61.81±9.20 | 27.26±3.79 | 29.51±7.91 | 44.57±11.12 | 24.89±4.14 | 34.98±10.87 |
| GCN-LAP | 34.06±9.46 | 34.24±9.45 | 23.08±5.45 | 33.95±7.70 | 37.39±7.80 | 29.56±2.86 | 30.49±4.94 | 33.04±7.21 | 15.52±2.65 |
| BUDDY | 33.41±5.10 | 23.80±3.71 | 32.42±1.96 | 23.09±3.11 | 19.55±1.05 | 27.87±7.83 | 31.36±4.70 | 27.51±1.46 | 18.71±1.04 |
| LINKX | 32.99±9.10 | 23.91±0.00 | 19.90±0.00 | 18.60±0.00 | 17.44±0.00 | 15.08±0.00 | 14.99±0.00 | 11.67±3.83 | 10.86±0.00 |
| Proposed w.o. D1 | 85.53±14.99 | 80.43±12.34 | 65.46±4.98 | 62.64±20.27 | 60.10±10.21 | 49.77±12.52 | 41.64±3.52 | 38.49±12.70 | 34.89±5.32 |
| Proposed w.o D2 | **97.42**±1.74 | 58.13±11.50 | 54.20±17.44 | 49.86±7.44 | 49.43±10.93 | 45.12±5.11 | 38.35±4.47 | 35.64±4.88 | 32.47±7.89 |
| Proposed | 90.78±7.34 | **84.41**±1.87 | **66.53**±14.99 | **64.51**±20.85 | **62.30**±10.83 | **57.83**±2.04 | **52.14**±13.77 | 47.36±6.13 | **45.83**±3.30 |

Table 7: SYN-CITESEER: Mean metrics (AUC) and standard deviation for each method on synthetic datasets with varying automorphism ratios $EAR$ (1/2, $EAR \in [0.10, 0.57]$). The best method per column is bolded.

| $EAR$ | 0.10 | 0.18 | 0.28 | 0.38 | 0.48 | 0.57 |
|---|---|---|---|---|---|---|
| GCN | 99.65±0.08 | 99.50±0.14 | 99.42±0.16 | 98.92±0.24 | 98.61±0.33 | 98.03±0.22 |
| GAT | 99.42±0.14 | 98.94±0.32 | 99.17±0.14 | 98.28±0.71 | 98.31±0.74 | 97.89±1.07 |
| GIN | 95.43±1.10 | 95.40±0.71 | 94.69±1.14 | 92.73±0.79 | 90.53±1.34 | 89.78±1.16 |
| GraphSAGE | 96.19±2.00 | 95.99±1.77 | 94.18±1.73 | 93.26±3.49 | 91.69±2.63 | 91.81±2.91 |
| MixHopGCN | 99.69±0.06 | 94.41±1.60 | 99.03±0.25 | 98.77±0.07 | 98.22±0.27 | 97.99±0.23 |
| ChebGCN | 97.59±0.46 | 97.76±0.41 | 97.20±0.44 | 94.90±0.58 | 92.50±1.31 | 88.75±1.86 |
| GCN-DW | 99.86±0.06 | 99.80±0.08 | 99.60±0.10 | 99.37±0.12 | 99.09±0.16 | 98.58±0.23 |
| GCN-RF | 99.87±0.04 | 99.69±0.06 | 99.75±0.06 | 99.60±0.08 | 99.39±0.09 | 99.27±0.11 |
| GCN-LAP | 99.79±0.07 | 99.65±0.09 | 99.35±0.19 | 99.04±0.13 | 98.78±0.14 | 98.23±0.23 |
| BUDDY | 95.20±0.39 | 95.83±0.18 | 95.84±0.26 | 94.37±0.53 | 95.10±0.51 | 95.79±0.48 |
| LINKX | 97.30±0.08 | 96.99±0.14 | 96.81±0.16 | 94.28±0.24 | 92.82±0.33 | 89.63±0.22 |
| Proposed | **99.94**±0.03 | **99.86**±0.03 | **99.83**±0.07 | **99.81**±0.00 | **99.80**±0.05 | **99.78**±0.04 |

Table 8: SYN-CITESEER: Mean metrics (AUC) and standard deviation for each method on synthetic datasets with varying automorphism ratios $EAR$ (2/2, $EAR \in [0.65, 0.94]$). The best method per column is bolded.

| $EAR$ | 0.65 | 0.77 | 0.81 | 0.88 | 0.94 |
|---|---|---|---|---|---|
| GCN | 97.65±0.47 | 96.32±0.56 | 96.49±0.74 | 95.03±1.60 | 91.81±2.05 |
| GAT | 97.81±0.41 | 96.56±1.26 | 95.87±1.41 | 93.11±4.54 | 91.93±2.09 |
| GIN | 86.46±2.00 | 78.42±1.10 | 82.29±1.19 | 75.34±1.54 | 61.25±1.04 |
| GraphSAGE | 86.72±4.49 | 85.62±5.20 | 82.41±6.02 | 77.59±3.97 | 74.68±5.96 |
| MixHopGCN | 96.28±0.44 | 93.72±0.76 | 94.55±0.52 | 91.21±1.95 | 88.35±3.18 |
| ChebGCN | 84.72±1.89 | 79.21±1.42 | 76.92±1.49 | 72.16±2.07 | 68.81±2.76 |
| GCN-DW | 98.45±0.17 | 97.92±0.20 | 97.83±0.23 | 97.36±0.24 | 96.23±0.24 |
| GCN-RF | 99.27±0.13 | 98.46±0.15 | 98.48±0.19 | 98.16±0.20 | 97.01±0.37 |
| GCN-LAP | 98.29±0.29 | 97.81±0.25 | 97.51±0.24 | 97.16±0.25 | 95.80±0.45 |
| BUDDY | 96.03±0.32 | 94.86±0.42 | 94.70±0.31 | 95.50±0.39 | 95.57±0.41 |
| LINKX | 86.69±0.47 | 85.47±0.56 | 83.48±0.74 | 81.28±1.42 | 79.68±1.49 |
| Proposed | **99.76**±0.03 | **99.74**±0.07 | 99.74±0.075 | 99.74±0.0775 | **99.90**±0.08 |

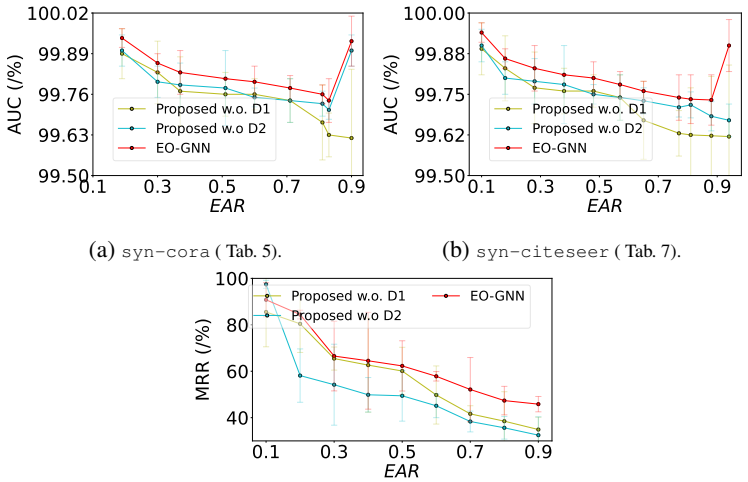
Table 9: SYN-CITESEER: Mean MRR and standard deviation for each method on synthetic datasets with varying automorphism ratios $EAR$ (1/2, $EAR \in [0.10, 0.57]$). The best method per column is bolded.

| $EAR$ | 0.10 | 0.18 | 0.28 | 0.38 | 0.48 | 0.57 |
|---|---|---|---|---|---|---|
| GCN | 49.84±4.78 | 48.61±13.29 | 48.34±10.77 | 43.70±10.66 | 39.74±9.66 | 27.99±7.70 |
| GAT | 31.93±8.23 | 27.72±7.61 | 25.82±4.77 | 24.61±6.22 | 23.06±3.30 | 21.82±8.21 |
| GIN | 20.37±3.99 | 18.09±4.48 | 15.77±2.23 | 14.17±3.56 | 11.37±2.29 | 10.79±2.36 |
| GraphSAGE | 24.01±15.47 | 20.28±12.22 | 20.13±10.96 | 16.45±4.91 | 15.82±5.18 | 12.57±5.13 |
| MixHopGCN | 70.99±27.71 | 52.17±4.79 | 51.84±9.30 | 43.61±6.49 | 39.02±2.52 | 36.87±10.14 |
| ChebGCN | 28.40±6.49 | 27.45±5.14 | 27.07±10.19 | 23.05±3.27 | 22.76±4.13 | 15.23±4.23 |
| GCN-DW | 65.92±9.59 | 53.50±9.29 | **75.96**±10.76 | 61.99±8.48 | 50.64±10.31 | 32.26±3.79 |
| GCN-RF | 60.49±9.57 | 49.02±12.14 | 59.79±12.07 | 58.83±11.46 | 59.68±11.20 | 65.63±13.42 |
| GCN-LAP | 63.75±8.76 | 64.01±8.85 | 74.99±8.55 | 60.41±9.69 | 49.12±7.91 | 30.90±4.64 |
| BUDDY | 29.06±7.08 | 31.40±4.33 | 26.07±8.50 | 37.07±5.59 | 40.97±4.09 | 38.80±1.98 |
| LINKX | 61.82±5.53 | 56.80±7.94 | 51.21±6.15 | 50.44±15.06 | 44.59±15.02 | 39.72±9.74 |
| Proposed | **85.72**±5.57 | **83.02**±6.87 | 74.85±7.60 | **72.06**±5.95 | **68.64**±4.59 | **67.42**±10.02 |

# F ABLATION STUDY IN DETAILS

Table 10: SYN-CITESEER: Mean MRR and standard deviation for each method on synthetic datasets with varying automorphism ratios $EAR$ (2/2, $EAR \in [0.65, 0.94]$). The best method per column is bolded.

| $EAR$ | 0.65 | 0.77 | 0.81 | 0.88 | 0.94 |
|---|---|---|---|---|---|
| GCN | 21.32±4.65 | 20.78±3.08 | 18.08±3.87 | 13.18±2.00 | 10.56±2.93 |
| GAT | 21.16±4.89 | 19.10±4.48 | 14.75±2.74 | 13.61±5.15 | 10.22±1.75 |
| GIN | 10.45±1.82 | 9.61±3.08 | 7.51±3.36 | 5.91±2.72 | 2.82±2.89 |
| GraphSAGE | 10.61±5.20 | 10.56±5.06 | 10.34±3.02 | 6.89±2.12 | 5.16±2.18 |
| MixHopGCN | 36.12±4.80 | 35.69±9.47 | 34.84±6.12 | 25.08±11.08 | 14.32±2.42 |
| ChebGCN | 12.33±3.00 | 8.83±2.23 | 7.57±2.49 | 7.09±0.94 | 4.81±0.62 |
| GCN-DW | 37.06±4.30 | 35.14±4.28 | 26.22±2.46 | 29.03±3.65 | 25.11±5.44 |
| GCN-RF | 54.62±9.97 | 30.14±6.63 | 34.67±4.98 | 41.84±5.38 | 18.56±3.56 |
| GCN-LAP | 37.15±6.31 | 38.19±6.80 | 24.42±2.50 | 26.08±4.46 | 23.36±4.97 |
| BUDDY | 22.80±4.00 | 37.20±3.34 | 32.22±3.14 | 35.62±5.89 | 34.86±5.28 |
| LINKX | 36.21±4.30 | 32.68±8.60 | 32.09±12.39 | 18.60±2.41 | 12.34±4.56 |
| Proposed | **66.61**±16.27 | **65.97**±11.65 | **55.19**±12.10 | **42.59**±8.04 | **41.19**±3.23 |




(a) syn-cora ( Tab. 5). (b) syn-citeseer ( Tab. 7).

(c) syn-cora ( Tab. 6).

Figure 12: Significance of Design Choices D1-D2 via ablation studies. Under high automorphism, D1 exhibits a larger performance gain compared to low automorphism, whereas D2 demonstrates greater robustness across varying levels of automorphism.

# G Empirical Setup & Hyperparameter Tuning

## G.1 Setup

**EO-GNN Implementation** For loss function, we calculate the contrastive learning loss between the predicted and the ground-truth edge connections.

$$L(v) = -\sum_{v \in N(v)} \log \delta(\mathbf{r}_v, \mathbf{r}_u) - \sum_{w \notin N(v)} \log(1 - \delta(\mathbf{r}_v, \mathbf{r}_w)). \tag{27}$$

**Baseline Implementations** For all baselines besides MLP, we used the implementation from the public library pytorch geometric [10].

- **GCN** [19]: `https://github.com/tkipf/gcn`

Table 11: Ablation study showing the effect of different components. Mean metrics (MRR) and standard deviation for each method on synthetic Cora with varying automorphism ratios $EAR$.

| Method | 0.19 | 0.30 | 0.37 | 0.51 | 0.60 | 0.71 | 0.81 | 0.83 | 0.90 |
|---|---|---|---|---|---|---|---|---|---|
| Proposed w.o. D1 | $99.89_{\pm 0.08}$ | $99.83_{\pm 0.10}$ | $99.77_{\pm 0.11}$ | $99.76_{\pm 0.07}$ | $99.76_{\pm 0.05}$ | $99.74_{\pm 0.07}$ | $99.67_{\pm 0.12}$ | $99.63_{\pm 0.07}$ | $99.62_{\pm 0.22}$ |
| Proposed w.o D2 | $99.90_{\pm 0.05}$ | $99.80_{\pm 0.05}$ | $99.79_{\pm 0.07}$ | $99.78_{\pm 0.12}$ | $99.75_{\pm 0.03}$ | $99.74_{\pm 0.07}$ | $99.73_{\pm 0.04}$ | $99.71_{\pm 0.03}$ | $99.90_{\pm 0.05}$ |
| Proposed | $99.94_{\pm 0.03}$ | $99.86_{\pm 0.03}$ | $99.83_{\pm 0.07}$ | $99.81_{\pm 0.00}$ | $99.80_{\pm 0.05}$ | $99.78_{\pm 0.04}$ | $99.76_{\pm 0.03}$ | $99.74_{\pm 0.07}$ | $99.93_{\pm 0.08}$ |

Table 12: Ablation study showing the effect of different components. Mean metrics (MRR) and standard deviation for each method on synthetic Cora with varying automorphism ratios $EAR$.

| Method | 0.19 | 0.30 | 0.37 | 0.51 | 0.60 | 0.71 | 0.81 | 0.83 | 0.90 |
|---|---|---|---|---|---|---|---|---|---|
| Proposed w.o. D1 | $85.53_{\pm 14.99}$ | $80.43_{\pm 12.34}$ | $65.46_{\pm 4.98}$ | $62.64_{\pm 20.27}$ | $60.10_{\pm 10.21}$ | $49.77_{\pm 12.52}$ | $41.64_{\pm 3.52}$ | $38.49_{\pm 12.70}$ | $34.89_{\pm 5.32}$ |
| Proposed w.o D2 | $97.42_{\pm 1.74}$ | $58.13_{\pm 11.50}$ | $54.20_{\pm 17.44}$ | $49.86_{\pm 7.44}$ | $49.43_{\pm 10.93}$ | $45.12_{\pm 5.11}$ | $38.35_{\pm 4.47}$ | $35.64_{\pm 4.88}$ | $32.47_{\pm 7.89}$ |
| Proposed | $90.78_{\pm 7.34}$ | $84.41_{\pm 1.87}$ | $66.53_{\pm 14.99}$ | $64.51_{\pm 20.85}$ | $62.30_{\pm 10.83}$ | $57.83_{\pm 2.04}$ | $52.14_{\pm 13.77}$ | $47.36_{\pm 6.13}$ | $45.83_{\pm 3.30}$ |

- **GraphSAGE** [44]: `https://github.com/williamleif/graphsage-simple` (PyTorch implementation)
- **GIN** [43]: `https://pytorch-geometric.readthedocs.io/en/latest/_modules/torch_geometric/nn/conv/gin_conv.html#GINConv`
- **GAT** [38]: `https://github.com/PetarV-/GAT`.

For DECODER, we used our own implementation of MLP with 1 to 3-hidden layers. We use the same loss function as EO-GNN for training MLP.

**Hardware Specifications** We run experiments on synthetic benchmarks using the HoreKa Green cluster, which consists of nodes equipped with dual-socket Intel Xeon Platinum 8368 CPUs, each with 76 cores (152 threads per node) and 512 GB of main memory. Each node features four NVIDIA A100 GPUs, each with 40 GB of GPU memory.

**Dataset Statistic** Random splits allocate 80%/15%/5% of the edges for the training, validation and test sets, respectively. The Collab dataset permits the use of validation edges as input during testing. For all reported results in the tables, we perform target link removal during training and do not utilize validation edges in the training stage.

### G.2 HYPERPARAMETER TUNING

To avoid bias, we tuned the hyperparameters of each method (EO-GNN and baseline models) on each benchmark. Below we list the hyperparameters tested on each benchmark per model. As the hyperparameters defined by each baseline model differ significantly, we list the combinations of non-default command line arguments we tested, without explaining them in detail. For the adam optimizer, we perform a grid search over the parameters reported in the original source code and tune the Adam optimizer through the following ranges: learning rate ($10^{-2,-3,-4}$), dropout rate (0, 0.1, 0.2, 0.3) and batch size ($2^{5,7,\ldots,15}$).

**Synthetic Benchmark Tuning** For `syn-cora`, we test the following command-line arguments for each baseline method:

- **EO-GNN-1 & EO-GNN-2**:
  - Dimension of Feature Embedding $p$: 64
  - Non-linearity Function $\rho$: ReLU
  - Dropout Rate: $a \in \{0, 0.5\}$

  We report the best performance, for $a = 0$.
- **GCN** [19]:
  - `hidden1`: $a \in \{16, 32, 64\}$

Table 13: Ablation study showing the effect of different components. Mean metrics (MRR) and standard deviation for each method on synthetic Citeseer with varying automorphism ratios $EAR$.

| $EAR$ | 0.10 | 0.18 | 0.28 | 0.38 | 0.48 | 0.57 |
|---|---|---|---|---|---|---|
| Proposed w.o. D1 | $99.89_{\pm 0.08}$ | $99.83_{\pm 0.10}$ | $99.77_{\pm 0.11}$ | $99.76_{\pm 0.07}$ | $99.76_{\pm 0.05}$ | $99.74_{\pm 0.07}$ |
| Proposed w.o D2 | $99.90_{\pm 0.05}$ | $99.80_{\pm 0.05}$ | $99.79_{\pm 0.07}$ | $99.78_{\pm 0.12}$ | $99.75_{\pm 0.03}$ | $99.74_{\pm 0.07}$ |
| Proposed | $99.94_{\pm 0.03}$ | $99.86_{\pm 0.03}$ | $99.83_{\pm 0.07}$ | $99.81_{\pm 0.00}$ | $99.80_{\pm 0.05}$ | $99.78_{\pm 0.04}$ |

| $EAR$ | 0.65 | 0.77 | 0.81 | 0.88 | 0.94 |
|---|---|---|---|---|---|
| Proposed w.o. D1 | $99.67_{\pm 0.12}$ | $99.63_{\pm 0.07}$ | $99.63_{\pm 0.15}$ | $99.63_{\pm 0.18}$ | $99.62_{\pm 0.22}$ |
| Proposed w.o D2 | $99.73_{\pm 0.04}$ | $99.71_{\pm 0.03}$ | $99.75_{\pm 0.04}$ | $99.84_{\pm 0.05}$ | $99.73_{\pm 0.05}$ |
| Proposed | $99.76_{\pm 0.03}$ | $99.74_{\pm 0.07}$ | $99.74_{\pm 0.08}$ | $99.73_{\pm 0.08}$ | $99.90_{\pm 0.08}$ |

Table 14: Ablation study showing the effect of different components. Mean MRR and standard deviation for each method on synthetic Citeseer with varying automorphism ratios $EAR$.

| $EAR$ | 0.10 | 0.18 | 0.28 | 0.38 | 0.48 | 0.57 |
|---|---|---|---|---|---|---|
| Proposed w.o. D1 | $83.40_{\pm 11.40}$ | $71.65_{\pm 12.54}$ | $69.35_{\pm 12.85}$ | $67.94_{\pm 12.46}$ | $64.67_{\pm 8.23}$ | $64.18_{\pm 7.58}$ |
| Proposed w.o D2 | $79.99_{\pm 12.23}$ | $76.28_{\pm 7.76}$ | $72.20_{\pm 16.27}$ | $66.62_{\pm 12.13}$ | $66.62_{\pm 12.13}$ | $66.62_{\pm 12.13}$ |
| Proposed | $85.72_{\pm 5.57}$ | $83.02_{\pm 6.87}$ | $74.85_{\pm 7.60}$ | $72.06_{\pm 5.95}$ | $68.64_{\pm 4.59}$ | $67.42_{\pm 10.02}$ |

| $EAR$ | 0.65 | 0.77 | 0.81 | 0.88 | 0.94 |
|---|---|---|---|---|---|
| Proposed w.o. D1 | $63.90_{\pm 9.91}$ | $43.61_{\pm 5.35}$ | $42.21_{\pm 3.54}$ | $41.19_{\pm 3.23}$ | $36.42_{\pm 9.45}$ |
| Proposed w.o D2 | $58.34_{\pm 13.80}$ | $55.88_{\pm 12.07}$ | $49.56_{\pm 10.00}$ | $41.81_{\pm 7.54}$ | $38.42_{\pm 12.32}$ |
| Proposed | $66.61_{\pm 16.27}$ | $65.97_{\pm 11.65}$ | $55.19_{\pm 12.10}$ | $42.59_{\pm 8.04}$ | $41.19_{\pm 3.23}$ |

  - `early_stopping`: $b \in \{40, 100, 200\}$
  - `epochs`: 2000

  We report the best performance, for $a = 32, b = 40$.

- **GraphSAGE** [11]:
  - `hid_units`: $a \in \{64, 128\}$
  - `lr`: $b \in \{0.1, 0.7\}$
  - `layers`: $l \in \{1, 3\}$
  - `dropout`: $l \in \{1, 3\}$
  - `epochs`: 500

  We report the performance with $layers = 3, dropout = 0.5, epochs = 800$.

- **GAT** [38]:
  - `hid_units`: $a \in \{8, 16, 32, 64\}$
  - `n_heads`: $b \in \{1, 4, 8\}$
  - `n_layers`: $b \in \{1, 2, 3\}$
  - `dropout`: $l \in \{1, 3\}$
  - `epochs`: 500

  We report the performance with $a = 8, b = 8$.

- **DECODER**
  - Dimension of Feature Embedding $p$: 64
  - Non-linearity Function $\rho$: ReLU
  - Dropout Rate: 0.5

## H WL LABELING SCHEME WITH PERFECT HASH

Consider a graph $\mathcal{H} = (\mathcal{V}, \mathcal{E})$. Let $\ell^{(0)}(v) = \ell(v)$ be the initial label assigned to each node $v \in \mathcal{V}$ (e.g., based on node features or degree). Let $H$ denote the number of WL iterations. Then, for each iteration $h = 1, \ldots, H$, the WL labeling scheme updates node labels recursively by considering the multiset of neighboring labels from the previous iteration.

Formally, define the multiset of neighbors' labels for node $v$ at iteration $h$ as:

$$\mathcal{N}^{(h)}(v) = \left\{ \ell^{(h)}(u) \mid u \in \bar{N}(v) \right\}$$

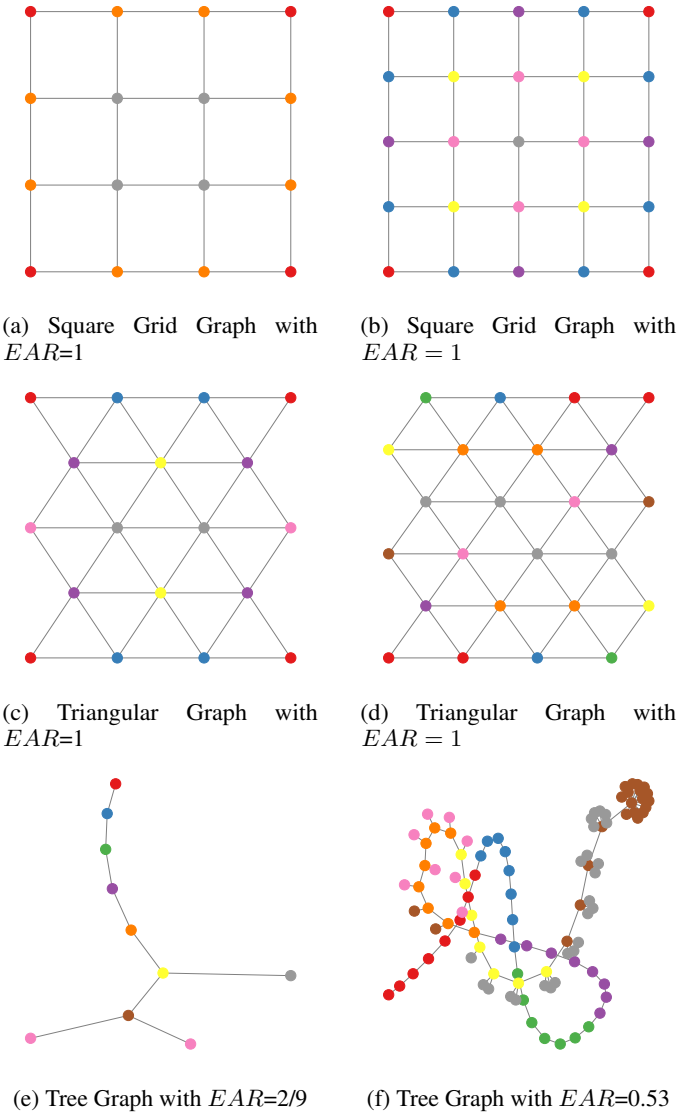
The updated label at iteration $h+1$ is computed as:

$$\ell^{(h+1)}(v) = \texttt{hash}\left( \ell^{(h)}(v), \mathcal{N}^{(h)}(v) \right) \tag{28}$$

Following the original formulation [41], we use a perfect hashing function, ensuring that two nodes receive the same label at iteration $h+1$ if and only if their own label and the multiset of their neighbors' labels were identical at iteration $h$.

**Algorithm 1:** Weisfeiler-Lehman Forward Pass

**Input:** Node feature matrix $\mathbf{X} = \mathbf{I} \in \mathbb{Z}^{|\mathcal{V}| \times 1}$
Adjacency matrix $\mathbf{A}$
Hash function $\mathcal{H}$
**Train Parameters:** Empty
**Output:** Updated node hash labels $h \in \mathbb{Z}^{|\mathcal{V}|}$
**begin**
  `/* Stage S1: Preprocessing */`
  **if** $\mathbf{A}$ *is sparse* **then**
    Convert to $(s, d)$ format;
  **else**
    Sort $\mathbf{A}$ by destination node;
  Initialize $\mathcal{N} \leftarrow \text{EmptyList}_N$;
  `/* Stage S2: Neighbor Aggregation */`
  **foreach** $(s, d) \in \mathbf{A}$ **do**
    Append $\mathbf{X}[s]$ to $\mathcal{N}[d]$;
  `/* Stage S3: Weisfeiler-Lehman Hash Computation */`
  Initialize $O \leftarrow \text{EmptyList}_{|}\mathcal{V}|$;
  **foreach** $i \in [0, |\mathcal{V}|)$ **do**
    Compute unique hash $i$ using:;
      - Node feature $\mathbf{X}[i]$;
      - Sorted features of $\mathcal{N}[i]$;
    **if** $i \notin \mathcal{H}$ **then**
      Assign a new unique ID to $i$;
    Store hashed value in $O[i]$;
  `/* Stage S4: Edge Orbit Hash Computation */`
  **foreach** $(u, v) \in \mathcal{E}$ **do**
    Compute edge hash using the multiset $\{O[u], O[v]\}$;
    Store hashed edge orbit ID in $\mathcal{O}_{\mathcal{E}}[(u, v)]$;
  **Output:** Updated node hash labels $O \in \mathbb{Z}^{|\mathcal{V}|}$
  Edge orbit labels $\mathcal{O}_{\mathcal{E}}$



(a) Square Grid Graph with $EAR$=1

(b) Square Grid Graph with $EAR = 1$

(c) Triangular Graph with $EAR$=1

(d) Triangular Graph with $EAR = 1$

(e) Tree Graph with $EAR$=2/9

(f) Tree Graph with $EAR$=0.53

Figure 13: Illustration of the automorphism test approximated by Algo. 1.

# I EO-GNN: Time Complexity in Detail

The 1-dimensional Weisfeiler-Lehman (1-WL) algorithm with $h$ iterations for precomputing subtree-based WL labels incurs a complexity of $\mathcal{O}(|\mathcal{E}| \cdot K)$ [36]. The D1 augmentation, which involves randomly dropping edges and selecting subgraph orbits, introduces no computational overhead and is therefore considered to have $\mathcal{O}(1)$ complexity.

For feature encoding, we apply a GNN with $L$ layers, each involving a dense transformation and neighborhood aggregation. This results in a total embedding complexity of $\mathcal{O}\left(L \cdot (|\mathcal{V}| \cdot p^2 + |\mathcal{E}| \cdot p)\right)$, where $p$ denotes the feature dimensionality. For each node with a performed linear transformation $\mathbf{H}^l\mathbf{W}^l$, where $\mathbf{H}^l \in \mathbb{R}^{|\mathcal{V}| \times p}$, $\mathbf{W}^l \in \mathbb{R}^{p \times p}$ refer to node and weight matrix in $l$ layer. The complexity is $\mathcal{O}\left(|\mathcal{V}| \cdot p^2\right)$. The edge wise aggregation is $\mathcal{O}\left(|\mathcal{E}| \cdot p\right)$. Combining both components, the total computational complexity of EO-GNN is:

$$\mathcal{O}\left(|\mathcal{E}| \cdot K + L \cdot (|\mathcal{V}| \cdot p^2 + |\mathcal{E}| \cdot p)\right).$$

Table 15: Statistics of standard benchmark graphs

| | Cora | Citeseer | Pubmed | Collab | PPA | Citation2 | DDI |
|---|---|---|---|---|---|---|---|
| **Split Ratio** | 80/15/5 | 80/15/5 | 80/15/5 | 92/4/4 | 70/20/10 | 98/1/1 | 80/10/10 |
| **Split Scheme** | R | R | R | Time | Throughput | Time | Protein target |
| **#Nodes** $\lvert\mathcal{V}\rvert$ | 2708 | 3327 | 19716 | 235868 | 576289 | 2927963 | 4267 |
| **#Edges** $\lvert\mathcal{E}\rvert$ | 7392 | 6374 | 62056 | 1935264 | 42463862 | 60703760 | 2135822 |
| **Avg Deg (G)** | 7.85 | 3.62 | 11.55 | 24.86 | 149.17 | 89.75 | **804.51** |
| **Avg Deg (G2)** | 5.63 | 3.28 | 7.85 | 20.70 | 133.13 | 68.21 | **796** |
| **Clustering** | 0.12 | 0.07 | 0.03 | **0.73** | 0.22 | 0.18 | 0.51 |
| **Transitivity** | 0.06 | 0.09 | 0.04 | 0.36 | 0.22 | 0.06 | **0.47** |
| **Deg Gini** | 0.45 | 0.50 | **0.63** | 0.55 | 0.55 | 0.57 | 0.47 |
| **Coreness Gini** | 0.27 | 0.36 | 0.45 | 0.43 | **0.45** | 0.42 | 0.35 |
| **Heterogeneity** | 0.14 | 0.11 | 0.23 | 0.21 | 0.13 | **0.31** | -0.08 |
| **Power Law** $\alpha$ | **1.89** | **2.13** | 1.97 | 1.89 | 1.33 | 1.39 | 1.21 |

## J REAL DATASETS: DETAILS

Considerable work has demonstrated that local and global structural characterization's are more effective for LP. To translate the homophily assumption, local and global graph heuristics, small-world phenomenon and scale-free network properties into task-specific statistics, we provide the following graph metrics.

1. **Graph Density**: Number of Nodes, Edges, Arithmetic Deg are used to measure the graph's size, density and sparsity. Average degree of each central node $v \in \mathcal{V}$ and its of 2-order neighborhood $\mathcal{N}_v$' average degree measures the graph's local connectivity.
2. **Graph Locality**: We utilize two metrics to quantify the locality of one graph. *Transitivity*: Transitivity measures the fraction of all possible triangles in the graph. It quantifies the likelihood that if two nodes are connected to a common node, they will also be connected to each other. The formula for transitivity is given by:

$$T = \frac{3 \times \#\text{triangles}}{\#\text{triads}} \tag{29}$$

   where, the numerator represents the number of triangles in the graph; the denominator represents the number of possible triads (sets of three nodes that are connected by at least two edges). Transitivity gives an overall measure of how many triangles (closed 3-node subgraphs) exist relative to the total number of possible connections between three nodes in the graph. *Average Clustering Coefficient*: It measures the fraction of possible triangles through that node that actually exist. It can be computed for a node $i$ as:

$$C_i = \frac{2 \times T(i)}{\deg(i)(\deg(i) - 1)} \tag{30}$$

   $T(i)$ is the number of triangles through node $i$ and $\deg(i)$ is the degree of node $i$. The *average clustering coefficient* is simply the average value of $C_i$ for all nodes in the graph. It gives a measure of how close the graph is to a complete clique, i.e., how often neighbors of a node are connected to each other.
3. **Hierarchical level**: We leverage k-Core graph's fraction and degree distribution to calculate Gini and Coreness Gini.
4. **Scale-free**: If its node degree distribution $P(d)$ follows a power law $P(d) \sim d^{-\gamma}$, where $\gamma$ typically lies within the range $2 < \gamma < 3$. We approximate power law $\alpha$ based on the following estimator. Citeseer is scale-free networks.

$$\hat{\alpha} = 1 + N \left( \sum_{i=1}^{n} \log \left( \frac{d_i + 1}{d_{\min} + 1} \right) \right)^{-1} \tag{31}$$