Does Depth Really Hurt GNNs? Injective Message Passing Enables Deep Graph Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Graph Neural Networks (GNNs) have shown great promise across domains, yet their performance often degrades with increased depth, commonly attributed to the oversmoothing phenomenon. This has led to a prevailing belief that depth inherently hurts GNNs. In this paper, we challenge this view and argue that the root cause is not depth itself, but the lack of *injectivity* in standard message passing (MP) mechanisms, which fail to preserve structural information across layers. To address this matter, we propose a new message passing layer that is provably injective without requiring any training and guarantees that GNNs match the expressive power of the Weisfeiler-Lehman (WL) test by *design*. Furthermore, this injective MP enables a decoupled GNN architecture where a shallow stack of injective MP layers ensures structural expressivity, followed by a deep stack of feature learning layers for rich representation learning. We provide theoretical analysis on the required depth, width, and initialization of MP layers to ensure both expressivity and numerical stability. Empirically, we demonstrate that our architecture enables deeper GNNs without suffering from oversmoothing. Our findings suggest that depth is not the core limitation in GNNs—lack of injectivity is—and offer a new perspective on building deeper and more expressive GNNs.

1 Introduction

In recent years, Graph Neural Networks (GNNs) have emerged as a powerful framework for learning from relational data, achieving state-of-the-art results across a wide range of domains, including molecular property prediction [32, 52, 45], social network analysis [5, 17], and recommendation systems [4, 48, 11]. At their core, GNNs employ a message passing paradigm [14], where node representations are iteratively updated by aggregating information from their neighbors. This structure-aware design enables GNNs to capture both feature and topological patterns in graph-structured data. Despite their success, most GNNs face two fundamental limitations that constrain their scalability and effectiveness: (1) *unreliable expressivity*, as distinguishing graph structures requires injective aggregation, yet standard message passing provides no guarantee of injectivity during training; and (2) *depth-related degradation*, where deeper architectures suffer from oversmoothing, causing node representations to become indistinguishable across the graph.

The expressive power of GNNs is characterized by their ability to distinguish whether two graphs are topologically identical, a problem closely related to the graph isomorphism problem in graph theory, for which no known polynomial-time solution has been found yet [1]. A foundational study by [50] established a close connection between GNNs and the first-order Weisfeiler-Lehman (1-WL) test [30], a widely used graph isomorphism heuristic that distinguishes many non-isomorphic graphs [2]. Their theoretical results show that GNNs can match the discriminative capacity of the WL test if their aggregation scheme is injective. Based on this, Graph Isomorphism Network (GIN) was proposed

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

to use Multilayer Perceptrons (MLPs) to approximate such injective mappings, making GIN and its variants theoretically as expressive as the 1-WL test.

However, this expressivity is only guaranteed under the assumption that the MLPs remain injective during training. In practice, *GNNs including GIN are trained to minimize task-specific objectives (e.g., node classification), not to enforce injectivity*. As a result, the task-driven optimization of MLPs offers no guarantee that the injective properties required for theoretical expressivity are preserved. This limitation also extends to recent enhancements such as higher-order WL architectures [35] and structural feature augmentation [51, 37], which continue to rely on unverified injectivity assumptions during training. Thus, whether a trained GNN can reliably maintain expressive power equivalent to the 1-WL test (or beyond) throughout learning remains an *open* question.

Beyond expressivity, another long-standing challenge in GNNs is their unusual depth sensitivity. Empirical studies have shown that stacking more message passing layers often leads to degraded performance, a phenomenon widely known as *oversmoothing* [28, 13, 31, 36, 49], where node representations become indistinguishable across the graph. This limits the ability of GNNs to capture long-range dependencies or refine complex representations through deeper architectures. As a result, whereas depth has been considered crucial for the success of deep learning in many fields such as computer vision [20] and natural language processing [42], most GNNs used in practice remains shallow, typically using a fixed depth of just 3-5 layers [31]. While this issue has traditionally been viewed as a depth-related limitation [36, 49], we argue that oversmoothing is fundamentally a symptom of *non-injective propagation*. When message passing functions fail to distinguish structurally distinct neighborhoods, deeper layers only reinforce this loss of information, propagating homogenized features rather than preserving meaningful variations. In this light, we contend that depth is not inherently problematic. Instead, it is the lack of structure-preserving message passing that causes performance collapse as networks grow deeper.

To address these dual challenges, we propose a decoupled GNN architecture that guarantees injective message passing while enabling stable and scalable deep learning over graphs. Our framework combines: (1) a theoretically grounded message passing scheme that is provably injective without training, ensuring WL-level expressivity by design; and (2) a depth-decoupled architecture that separates structure propagation from representation learning, thereby avoiding information homogenization in deep layers, so preventing oversmoothing. Our key contributions are as follows:

- **Injective Message Passing without Training.** We introduce a new message passing layer that leverages the distinct node features in a graph. We prove that simple linear propagation is injective if the distinct node features are linearly independent. To satisfy this condition, we apply a nonlinear feature lifting function to map input features to a linearly independent space, yielding injective propagation without requiring any training.
- **Topology-Aware Depth Selection.** Inspired by the connection between GNN expressivity and the WL test, we establish a principled way to select the number of message passing layers based on graph topology. For example, in social networks with tree-like local structures, we suggest using $\mathcal{O}(\log n)$ message passing layers, as WL iteration typically stabilizes in logarithmic rounds.
- **Decoupled Architecture for Deep Representation Learning.** We identify that existing GNNs entangle structure propagation and representation learning in each layer, which limits expressivity and depth scalability. Hence, we suggest decouple these roles: a small number of injective message passing layers ensure structural expressivity, followed by a deep stack of feature learning layers dedicated to representation refinement.
- **Comprehensive Empirical Validation.** We conduct extensive experiments to validate our theoretical claims and provide detailed ablation studies analyzing the effect of key architectural choices, including network width, number of message passing layers, and number of feature learning layers.

2 Related Work

Expressivity of GNNs Since the foundational work by [50], which links GNN expressivity to the 1-WL test, many methods have attempted to improve GNN expressivity by leveraging higher-order architectures [35], positional encodings [51], or random features [37]. While these approaches are theoretically promising, they rely on MLPs trained via task-specific objectives, offering no guarantee

of maintaining injectivity throughout learning. In contrast, our method provides provably injective message passing without requiring any training, ensuring expressivity by design.

Oversmoothing and GNN Depth. Oversmoothing has been studied as a key limitation of deep GNNs [36, 49]. However, prior theoretical work yields conflicting claims—some suggest oversmoothing is inevitable on random graphs [36, 49], while others (e.g., via NTK or NNGP analysis) argue it is architecture-dependent [10]. Our view is orthogonal: we identify non-injective propagation as the root cause of oversmoothing. Without injectivity, deeper GNNs propagate homogenized features, leading to representation collapse.

Decoupled Architectures. Several works have explored decoupling message passing and feature learning, such as SGC [47] and APPNP [29], which simplify or linearize propagation. However, these models still rely on non-injective MP, so their expressivity is not theoretically guaranteed. Our work integrates a provably injective MP layer with a decoupled architecture to ensure both structural expressivity and scalable deep representation learning.

3 Preliminaries

We begin by reviewing the message passing framework for GNNs and its connection to the expressive power of the 1-dimensional Weisfeiler-Lehman (1-WL) test and setups for theoretical analysis.

Let $G = (V, E)$ be an undirected graph, where V is the set of nodes with $|V| = n$ and $E \subseteq V \times V$ is the set of edges. Each node $v \in V$ is associated with a feature vector $\boldsymbol{x}_v \in \mathbb{R}^d$, and let $\boldsymbol{A} \in \{0,1\}^{n \times n}$ denote the adjacency matrix of G, where $\boldsymbol{A}_{uv} = 1$ if $(u, v) \in E$. Let $\mathcal{N}(v)$ denote the set of neighbors of node v.

Graph Neural Networks. Modern GNNs [28, 18, 43, 50] iteratively update node embeddings by exchanging and aggregating information over graph neighborhoods. After k layers of message passing, a node's representation captures information from its k-hop neighborhood. A general form of message passing GNNs is:

$$\mathbf{m}_v^{(\ell)} = \text{AGGREGATE}^{(\ell)}\left(\{\!\{\mathbf{h}_u^{(\ell-1)} : u \in \mathcal{N}(v)\}\!\}\right), \quad \mathbf{h}_v^{(\ell)} = \text{COMBINE}^{(\ell)}\left(\mathbf{h}_v^{(\ell-1)}, \mathbf{m}_v^{(\ell)}\right), \tag{1}$$

where $\boldsymbol{h}_v^{(\ell)} \in \mathbb{R}^m$ is the embedding of node v at layer ℓ, initialized as $\boldsymbol{h}_v^{(0)} = \boldsymbol{x}_v$. We assume the embedding dimension to m across layers, simplifying theoretical analysis. The neighborhood is treated as a multiset $\{\!\{\cdot\}\!\}$ to preserve duplicate node information. As shown by [50], a message passing GNN can match the expressive power of the 1-WL test if both the AGGREGATE and COMBINE functions are injective over multisets.

Weisfeiler-Lehman Test. The graph isomorphism problem asks whether two graphs are structurally identical. Although no known polynomial-time algorithm exists [1], the Weisfeiler-Lehman (WL) test [30] is a widely used heuristic that distinguishes many non-isomorphic graphs [2]. The 1-WL test, also known as color refinement, iteratively updates node labels by aggregating labels from neighbors:

$$c_v^{(\ell)} = \text{HASH}\left(c_v^{(\ell-1)}, \{\!\{c_u^{(\ell-1)} : u \in \mathcal{N}(v)\}\!\}\right). \tag{2}$$

Two graphs are declared non-isomorphic if their label multisets differ at any iteration. The 1-WL test forms the basis for theoretical GNN expressivity analysis [50, 35].

Assumptions for Expressivity Analysis. Following [50, 35], we assume that both the GNN and the 1-WL test start from identical node features across compared graphs. This allows us to isolate structural discriminability, as any differences must be inferred purely from topology. This setup reflects a worst-case scenario, since identical initialization typically requires more WL iterations than heterogeneous ones. Hence, our analysis in Section 5.1 thus provides an upper bound on the MP depth required for expressivity. Additionally, we assume that the final graph representation is the multiset of node embeddings $\{\!\{\boldsymbol{h}_v^{(L)} : v \in V\}\!\}$, rather than a pooled vector. This keeps our analysis focused at the node level. For graph-level tasks, a READOUT function (e.g., sum) is often used to obtain a single vector, but unless this function is also injective, it may obscure structural differences [50]. Therefore, injective message passing is a necessary condition for expressivity: once node embeddings collapse, no readout function can recover the lost structural information.

4 Training-Free Injective Message Passing

In this section, we present a new message passing (MP) layer that is provably injective without any training. As a result, a GNN using this MP layer is guaranteed to match the expressive power of the 1-WL test. Our design is based on a two-step construction: we first show that a linear MP layer is injective under linearly independent embeddings, and then demonstrate that this condition can be satisfied via a simple nonlinear feature transformation.

4.1 Injectivity from Linear Message Passing

Let $S_v^{(\ell)} := \{\!\{\boldsymbol{h}_u^{(\ell)} : u \in \mathcal{N}(v)\}\!\}$ denote the multiset of neighbor embeddings for node v at layer ℓ. We consider the following linear message passing scheme:

$$\boldsymbol{h}_v^{(\ell)} = (1+\epsilon)\boldsymbol{h}_v^{(\ell-1)} + \sum_{u \in \mathcal{N}(v)} \boldsymbol{h}_u^{(\ell-1)}, \tag{3}$$

where ϵ is a fixed irrational number[1]. The effectiveness of such linear aggregation has been observed in prior work. For instance, [47] shows that purely linear message passing performs competitively across various tasks. Moreover, this form was also used in the expressivity proof of GIN [50], where node features are encoded as n-digit scalars, making the message passing injective. Inspired by these insights, we generalize the idea and show that linear message passing is injective whenever the set of distinct node embeddings is linearly independent.

Proposition 1. *Suppose all distinct node embeddings $\{\boldsymbol{h}_v^{(\ell)}\}_{v \in V}$ are linearly independent, and ϵ is irrational. Then the linear message passing update equation 3 is injective over all neighborhood multisets $S_v^{(\ell)}$.*

Compared to prior expressivity analysis that assumes bounded multiset cardinality [50, 25, 35], this condition is weaker and more scalable, especially for large graphs. However, the linear independence assumption can easily be violated as message passing layers accumulate more diverse embeddings. To address this, we introduce a nonlinear transformation that lifts features into a linearly independent space.

4.2 Guaranteeing Linear Independence via Nonlinearity

While Proposition 1 shows that linear message passing is injective under linearly independent node embeddings, this condition is fragile in practice, especially as deeper GNN layers accumulate more structurally distinct embeddings. To address this, we propose lifting node embeddings into a linearly independent space using a simple nonlinear transformation.

Specifically, we apply a one-layer MLP of the following form:

$$\boldsymbol{a}_v = \frac{1}{\sqrt{m}}\phi(\boldsymbol{W}^\top \boldsymbol{h}_v), \quad \forall v \in V, \tag{4}$$

where $\boldsymbol{W} \in \mathbb{R}^{d \times m}$ is a weight matrix, and ϕ is a nonlinear activation function. To stabilize the output magnitudes, we apply the standard scaling factor $\frac{1}{\sqrt{m}}$ for standard random initialization schemes on the weights matrix $\boldsymbol{W}$ such as He or Xavier initialization [19, 15]:

$$\boldsymbol{W}_{ij} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2). \tag{5}$$

Let $\boldsymbol{H} \in \mathbb{R}^{k \times d}$ be a matrix composed of all k distinct input embeddings. Define the lifted feature matrix as:

$$\boldsymbol{A} = \frac{1}{\sqrt{m}}\phi(\boldsymbol{H}\boldsymbol{W}) \in \mathbb{R}^{k \times m}. \tag{6}$$

We aim to show that with high probability over the random initialization of $\boldsymbol{W}$, the rows of $\boldsymbol{A}$ are linearly independent if m is near-linear in k. Our key result is summarized as follows, and the proof, which builds on the notion of dual activation function [7] and concentration bounds from random matrix theory [44, 3], is deferred to the appendix.

[1] In [50], the authors evaluate both learned and fixed values of ϵ and find no significant difference in empirical performance.

Proposition 2. *Let ϕ be Lipschitz continuous, nonlinear, and non-polynomial. Let $\boldsymbol{H} \in \mathbb{R}^{k\times d}$ contain k distinct input embeddings and $\boldsymbol{A} \in \mathbb{R}^{k\times m}$ be the lifted feature matrix. Then, for any $\delta > 0$, if $m = \Omega\left(\frac{k}{\lambda_0}\log\left(\frac{k}{\lambda_0}\right)\log\left(\frac{k}{\delta}\right)\right)$, then with probability at least $1-\delta$, the matrix $\boldsymbol{A}$ has linearly independent rows, where $\lambda_0 := \lambda_{\min}\left(\mathbb{E}\left[\sigma(\boldsymbol{H}\boldsymbol{w})\sigma(\boldsymbol{H}\boldsymbol{w})^\top\right]\right) > 0$.*

Proposition 2 shows that linearly independent node embeddings can be achieved via a simple one-layer nonlinear transformation. As the weights $\boldsymbol{W}$ are fixed at initialization, no training is required to ensure injectivity, offering a training-free guarantee, differing fundamentally from prior MLP-based message passing schemes like GIN and its variants [50, 35]. Remarkably, the required width m scales only near-linearly with the number of distinct embeddings k, *i.e.*, $m = \tilde{\Omega}(k)$, rather than the total number of nodes n, as used in prior analysis [35, 50, 25], making this approach efficient and *scalable* for large-scale graph representation learning in practice.

4.3 WL-Expressive GNN via Injective Message Passing

By combining Proposition 1 and Proposition 2, we can construct a training-free message passing layer that is provably injective:

$$\boldsymbol{a}_v^{(\ell)} = \frac{1}{\sqrt{m}}\phi(\boldsymbol{W}^{(\ell)}\boldsymbol{h}_v^{(\ell-1)}), \tag{7}$$

$$\boldsymbol{h}_v^{(\ell)} = (1+\epsilon)\boldsymbol{a}_v^{(\ell)} + \sum_{u\in\mathcal{N}(v)} \boldsymbol{a}_u^{(\ell)} \tag{8}$$

where $\boldsymbol{W}^{(\ell)} \in \mathbb{R}^{m\times m}$ is assumed to have the same width across different layers for simplicity.

By following [50], which shows the expressive power of GNN can match 1-WL test with injective message passing, we obtain the following key results for our proposed new message passing layer.

Theorem 1. *Suppose ϕ is Lipschitz continuous and nonlinear but non-polynomial, and ϵ is an irrational number. Let k denote the total number of distinct rooted subtree structures encountered during the 1-WL test across both graphs. For any $\delta > 0$, if the width m satisfies $m = \Omega\left(\frac{k}{\lambda_0}\log\left(\frac{k}{\lambda_0}\right)\log\left(\frac{k}{\delta}\right)\right)$, then with probability at least $1-\delta$ over the random initialization, a GNN using the message passing layer in Eq. (8) is as expressive as the 1-WL test. That is, it produces different embeddings for two graphs if and only if the 1-WL test distinguishes them.*

Remark 1. *While the total number of nodes n provides a loose upper bound on network width m, our result shows that the required width m depends only on the number of distinct rooted subtrees k observed during 1-WL iterations. In many practical datasets, $k \ll n$, leading to a significantly smaller width requirement. This reflects the fact that 1-WL distinguishability depends on structural diversity rather than graph size.*

Since expressivity is already ensured by design, the focus of training can now shift mainly toward representation learning, just as in classical deep networks. This motivates a clean architectural separation between structure propagation and representation learning. In the next section, we introduce a decoupled GNN design that leverages this separation to enable deeper and more stable feature learning while maintaining expressive power.

5 Layer Design: Decoupling Message Passing and Feature Learning

Most existing GNNs adopt a shallow and fixed number of message passing (MP) layers (e.g., 3–5) [31, 36, 49], as performance often degrades with increased depth due to oversmoothing. Our key insight is that depth itself is not the root cause of this degradation; rather, it stems from non-injective aggregation schemes, which fail to preserve structural variations and instead propagate homogenized features. In contrast, the training-free injective MP scheme introduced in Section 4 guarantees expressivity by design. Once structural information is sufficiently captured through a small number of such injective MP layers, the model can shift its focus to representation learning. This motivates a decoupled architecture: a shallow but expressive MP block for structural encoding, followed by a deep and fully trainable feature learning (FL) block for downstream tasks.

5.1 Topology-Aware Message Passing Depth

We first examine how many MP layers are required to sufficiently explore structural variations in a graph. Inspired by the WL test, we argue that the number of MP layers should not be fixed across datasets, but instead determined by the graph's topology. Specifically, the depth should align with the number of WL iterations needed to distinguish node labels.

While the 1-WL test stabilizes in just $\mathcal{O}(1)$ iterations for fully connected graphs, the worst-case complexity is $\mathcal{O}(n)$, as the number of distinct node labels may grow linearly. Recent theoretical results confirm this bound is tight: Kiefer and McKay [26] construct an infinite family of graphs requiring n iterations, and Grohe et al. [16] further characterize families with high iteration counts. Motivated by this, we state the following general upper bound:

Proposition 3. *For two n-node graphs with identical initial node features, an expressive GNN with $\mathcal{O}(n)$ injective MP layers returns different embeddings if and only if the 1-WL test determines them non-isomorphic.*

While informative, this worst-case bound is rarely encountered in practice. Most real-world graphs exhibit structural regularities that allow for sublinear MP depths, as shown in the following proposition:

Proposition 4. *For two n-node graphs with identical initial node features, suppose the 1-WL test determines them non-isomorphic. Then:*

- *If the graphs are balanced binary trees, then $\mathcal{O}(\log n)$ injective MP layers suffice.*
- *If the graphs are 2D grids, then $\mathcal{O}(\sqrt{n})$ injective MP layers suffice.*

Remark 2. *Social networks often exhibit small-world properties, where the graph diameter is much smaller than n. Local structures tend to be either tree-like (e.g., with hub nodes) [24], where $\mathcal{O}(\log n)$ layers are sufficient, or grid-like with uniform connectivity [46], where $\mathcal{O}(\sqrt{n})$ layers are more appropriate.*

Remark 3. *Molecular graphs generally have bounded degree and a small number of atom types, resulting in low-diameter and often planar structures [14, 9]. We recommend using $\mathcal{O}(\log n)$ MP layers for such datasets.*

These observations highlight the importance of *topology-aware* MP depth selection. Rather than using a fixed shallow depth, practitioners can estimate graph diameter or WL iteration count via breadth-first search (BFS) or symbolic labeling to guide the number of MP layers.

5.2 Stability of Deep Message Passing

Although many real-world graphs only require shallow MP depth, the worst-case $\mathcal{O}(n)$ bound motivates analyzing stability when stacking many injective MP layers. Since our MP is training-free, the variance of hidden representations is fully governed by initialization. Let $\boldsymbol{H}^{(\ell)}$ denote the node embedding matrix at layer ℓ. We derive the following recursive bound on its norm.

Lemma 1. *If ϕ is L-Lipschitz, then*

$$\mathbb{E}\left[\|\boldsymbol{H}^{(\ell)}\|^2\right] \leq \{\sigma_\ell L\left[(1+\epsilon)+\|\boldsymbol{A}\|\right]\}^2 \mathbb{E}\left[\|\boldsymbol{H}^{(\ell-1)}\|^2\right], \tag{9}$$

where $\boldsymbol{A} \in \{0,1\}^{n\times n}$ is the adjacency matrix.

To maintain stable propagation, we recommend initializing MP layers with:

$$\sigma_\ell = \frac{1}{L\left[(1+\epsilon)+\|\boldsymbol{A}\|\right]}. \tag{10}$$

Although our architecture does not require training the MP layers to ensure expressivity, we empirically observe that training the ML layers can further improve performance (see Figure 1 and 2). Moreover, even when MP layers are trained, this initialization remains beneficial, since parameters typically remain close to their initial values during training [23].

5.3 Decoupled GNN Design: Injective Propagation + Deep Learning

Standard GNNs couple MP and FL within each layer. This entanglement introduces two key limitations. First, MP functions are optimized indirectly via task-specific objectives, which—as discussed in Section 4—cannot guarantee injectivity and thus undermine expressive power. Second, as non-injective MP layers accumulate, they tend to propagate homogenized features rather than preserve structural distinctions, leading to oversmoothing and degraded performance in deep models.

In contrast, deep models in other domains, such as CNNs in vision [20, 21] and Transformers in NLP [42], succeed by separating early structural encoding from deep feature refinement. For example, CNNs resolve local edges and textures in early layers, while deeper layers capture abstract, task-specific features.

We bring this principle to graph learning through a decoupled architecture:

- A shallow stack of injective MP layers to encode structural variation and ensure WL-level expressivity.
- A deep stack of fully trainable FL layers to learn rich representations for downstream tasks, leveraging the expressive embeddings from the MP block.

This separation yields multiple benefits:

- **Guaranteed Expressivity**: The injective MP layers encode sufficient structural variations by design, making the GNN WL-expressive by construction.
- **Focused Training**: Since expressivity is ensured upfront, all optimization effort is devoted to learning useful representations, not propagation.
- **Deeper and Stable Architectures**: The feature learning block can incorporate standard stabilization and regularization techniques (e.g., residual connections [20], batch normalization [22], and dropout [41]), enabling significantly deeper GNNs than previously feasible.
- **Scalability**: As established in Section 5.1, the depth and width of MP layers only need to scale with the number of distinct rooted subtree structures k and graph diameter D, respectively, both of which are much smaller than the total number of nodes n in most real-world datasets, while the FL block can be made much deeper.

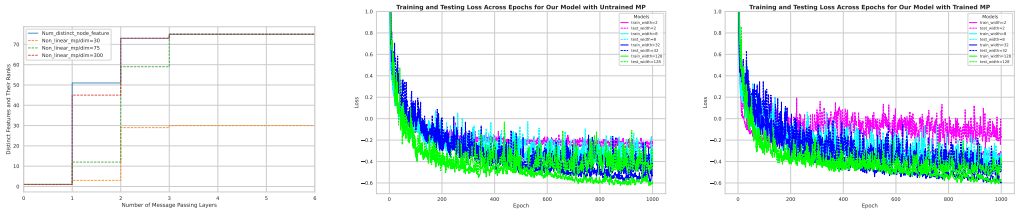
Models like APPNP [29] and SGC [47] also decouple structure and learning, but use non-injective linear MPs. Consequently, they still suffer from oversmoothing and under-expressivity. In contrast, our decoupled design pairs injective propagation with deep learning, enabling both expressivity and scalability.

6 Experiments

Table 1: Node classification results over eight datasets (%). The best results are highlighted in blue.

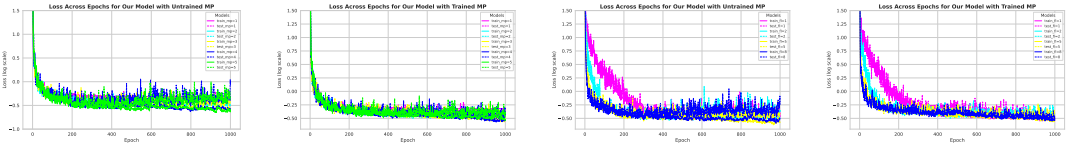
	Cora	Citeseer	Pubmed	Wikics	Computer	Physics	CS	Photo
# Nodes	2,708	3,327	19,717	11701	13,752	34,493	18,333	7,650
# Edges	5,278	4,732	44,324	216,123	245,861	247,962	81,894	119,081
Metric	Accuracy	Acccuracy	Accuracy	Accuracy	Accuracy	Accuracy	Accuracy	Accuracy
GCN	$84.08_{\pm 0.37}$	$72.28_{\pm 0.74}$	$80.48_{\pm 0.81}$	$80.41_{\pm 0.57}$	$93.99_{\pm 0.26}$	$97.38_{\pm 0.07}$	$95.91_{\pm 0.05}$	$95.73_{\pm 0.19}$
SAGE	$83.36_{\pm 0.34}$	$71.96_{\pm 1.11}$	$78.34_{\pm 0.53}$	$80.60_{\pm 0.18}$	$93.14_{\pm 0.13}$	$97.19_{\pm 0.10}$	$96.26_{\pm 0.07}$	$96.34_{\pm 0.57}$
GAT	$82.92_{\pm 1.48}$	$71.94_{\pm 0.99}$	$80.32_{\pm 0.60}$	$81.02_{\pm 0.36}$	$93.77_{\pm 0.13}$	$97.30_{\pm 0.08}$	$96.18_{\pm 0.12}$	$96.56_{\pm 0.39}$
Ours	$85.06_{\pm 0.83}$	$72.19_{\pm 0.81}$	$81.64_{\pm 0.63}$	$81.10_{\pm 0.29}$	$93.99_{\pm 0.15}$	$97.54_{\pm 0.08}$	$96.27_{\pm 0.23}$	$96.21_{\pm 0.10}$

In this section, we conduct a series of experiments to support our theoretical findings and design principles. We focus our experiments on studying the expressivity of the proposed method through checking the injectivity of message passing and the scalability of the depth to prevent oversmoothing. The exact details about the experiments and choice of datasets are deferred to the appendix due to page limit.



(a) (b) (c)

Figure 1: Effect of **MP width** on expressivity and training: (a) Larger widths yield higher feature matrix rank, closely tracking the number of distinct node features; (b) Wider MPs reduce training loss but risk overfitting; (c) Training MP parameters alleviates overfitting.



(a) (b) (c) (d)

Figure 2: Effect of **MP depth** and **FL depth** on learning: (a) On molecular graphs, deeper frozen MPs reduce loss but cause overfitting; (b) Training MP layers mitigates overfitting on molecular graphs; (c) With frozen MP layers, deeper FL improves training loss but risks overfitting; (d) Training MP layers stabilizes learning and improves generalization as FL depth increases.

Expressive GNNs and Injective Message Passing. As shown by [50], a GNN achieves 1-WL expressivity if its MPs are injective. In Section 4, we proposed a new MP layer in Eq. (8) that is provably injective without training, based on Propositions1 and 2. We empirically validate our construction by tracking the number and rank of distinct node features across layers. As shown in Figure 1(a), even with identical initial features, our MP steadily increases both the number of distinct node embeddings and their rank, provided the hidden dimension m exceeds the number of distinct features k. Due to numerical instability (e.g., near-zero singular values), the rank may briefly lag when $m \approx k$, but eventually matches it.

Importantly, this injectivity is achieved without training. Moreover, the required hidden width scales with the number of distinct structures, *i.e.*, $m = \tilde{\Omega}(k)$, not the total number of nodes n, with $k \ll n$ in practice. For example, in Figure 1(a), a graph with $n = 218$ nodes and $k = 75$ distinct rooted subtree structures reaches full expressivity using $m = 75$ and only 3 MP layers, demonstrating the scalability of our method to large graphs.

Impact of Message Passing Width. To assess how MP width affects model performance, we vary the width of our injective MP layer Eq. (8) while keeping its parameters frozen and only training the feature learning layers, following our decoupled design in Section 5. As shown in Figure 1(b), increasing the width consistently lowers training loss, demonstrating that wider MPs can improve learning, even without being trained.

However, excessive width introduces overfitting, likely because randomly initialized MP layers inject noise that is memorized by the downstream feature learner. Figure 1(c) shows that training the MP layers mitigates this issue, especially for large widths. Notably, smaller-width models exhibit unstable performance when *MP and FL are entangled*, encouraging a decoupled GNN design as we introduced in Section 5.3. It also highlights that large-width MPs may not only guarantee injectivity at initialization but also preserve it during training—a promising direction for future investigation.

Impact of Message Passing Depth. As discussed in Section 5.1, the number of MP layers should match the number of 1-WL iterations required to distinguish structural patterns. For molecular graphs, which are typically tree-like, $\mathcal{O}(\log n)$ MP layers are sufficient. Figure 2(a) shows that increasing MP depth initially improves training loss, but excessive depth eventually degrades test performance, an effect resembling overfitting. This mirrors our earlier observation with large MP width: frozen, randomly initialized MP layers introduce noise that downstream FL layers may overfit. As shown in

Figure 2(b), allowing MP layers to be trained mitigates this issue and improves both training and test performance. While our injective MP design does not require training for expressivity, these results suggest that training can further enhance performance, possibly because injectivity is preserved throughout optimization.

Decoupled Deep Representation Learning. Traditional GNNs entangle structure propagation and representation learning within each layer, limiting the model's capacity to learn expressive representations, especially as deeper architectures often lead to oversmoothing. Our proposed decoupling strategy from Section 5 separates these two components: a small number of injective MP layers ensures structural expressivity, while deep FL layers refine node representations for downstream tasks.

To evaluate this design, we fix the number of MP layers based on the dataset's graph topology, as suggested in Section 5.1, and vary the number of FL layers. As shown in Figure 2(c), increasing the FL depth reduces training loss, but frozen MP layers (initialized randomly) can introduce noise, leading to overfitting for deeper FL models. In Figure 2(b), this overfitting is mitigated when we train the MP layers alongside FL layers, resulting in both stable training and improved generalization.

At first glance, training the MP layers may seem to reintroduce the entanglement of propagation and feature learning seen in traditional GNNs. However, we observe no oversmoothing, even with increased depth. This key difference lies in our use of provably injective MP layers. Unlike prior GNNs where non-injective propagation leads to homogenized features and expressivity loss, our MP layers preserve structural distinctions throughout training. This supports our central claim: depth is not the issue, but non-injective message passing is. By addressing this, our design enables deep, expressive, and stable GNNs.

Comparison with Classic GNNs. To further validate the practical effectiveness of our architecture, we compare our model against several state-of-the-art GNNs. Recent studies (e.g., [33]) have shown that classic GNNs can achieve remarkably strong performance. Following their setup, we evaluate our model under comparable conditions. Notably, our message passing depth and width are selected based on theoretical insights from Section 4 and 5, ensuring sufficient structural exploration without excessive overhead. Although our comparison does not involve exhaustive hyperparameter tuning, the results in Table 1 show that our model achieves competitive or even superior performance. We attribute this to two key factors: (1) our injective MP layers preserve expressivity throughout training by design, and (2) the decoupled architecture enables deep and stable feature learning. Together, these components allow our model to scale effectively while maintaining high expressivity, bridging a critical gap between depth, learnability, and structural awareness in GNN design.

7 Conclusion

This paper revisits two long-standing challenges in GNNs: unreliable expressivity and depth-induced degradation. While depth is often blamed for oversmoothing, we argue that the true bottleneck lies in non-injective message passing, which causes structural information to vanish across layers. To address this, we propose a provably injective message passing scheme that requires no training and matches the expressive power of the 1-WL test by design. Built on this foundation, we introduce a decoupled GNN architecture that separates structural propagation from feature learning. This design allows shallow, injective MP layers to encode topology and supports arbitrarily deep feature learning blocks for expressive representation learning. We further develop theory-guided criteria for selecting MP depth based on graph topology and propose a variance-stabilized initialization scheme to ensure robustness across depths. Extensive empirical studies validate our claims: the proposed decoupled GNN with injective MPs avoids oversmoothing, remains expressive at scale, and enables deep, stable architectures. By disentangling expressivity from trainability, our framework bridges a critical gap in GNN design, bringing the scalability of deep learning to graph representation learning. We hope this work inspires future exploration into principled, modular GNN architectures that are both expressive and trainable, especially in large-scale or structure-sensitive domains.

References

[1] László Babai. Graph isomorphism in quasipolynomial time. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 684–697, 2016.

[2] László Babai and Ludik Kucera. Canonical labelling of graphs in linear average time. In *20th annual symposium on foundations of computer science (sfcs 1979)*, pages 39–46. IEEE, 1979.

[3] Zhi-Dong Bai and Yong-Qua Yin. Limit of the smallest eigenvalue of a large dimensional sample covariance matrix. In *Advances In Statistics*. World Scientific, 2008.

[4] Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. *Advances in neural information processing systems*, 29, 2016.

[5] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and deep locally connected networks on graphs. In *2nd International Conference on Learning Representations, ICLR 2014*, 2014.

[6] Jinsong Chen, Kaiyuan Gao, Gaichao Li, and Kun He. Nagphormer: A tokenized graph transformer for node classification in large graphs. *arXiv preprint arXiv:2206.04910*, 2022.

[7] Amit Daniely, Roy Frostig, and Yoram Singer. Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. *Advances in neural information processing systems*, 2016.

[8] Chenhui Deng, Zichao Yue, and Zhiru Zhang. Polynormer: Polynomial-expressive graph transformer in linear time. *arXiv preprint arXiv:2403.01232*, 2024.

[9] David K. Duvenaud, Dougal Maclaurin, Jorge Aguilera-Iparraguirre, Rafael Gomez-Bombarelli, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P. Adams. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

[10] Bastian Epping, Alexandre René, Moritz Helias, and Michael T Schaub. Graph neural networks do not always oversmooth. *arXiv preprint arXiv:2406.02269*, 2024.

[11] Chen Gao, Xiang Wang, Xiangnan He, and Yong Li. Graph neural networks for recommender system. In *Proceedings of the fifteenth ACM international conference on web search and data mining*, pages 1623–1625, 2022.

[12] Tianxiang Gao, Hailiang Liu, Jia Liu, Hridesh Rajan, and Hongyang Gao. A global convergence theory for deep reLU implicit networks via over-parameterization. In *International Conference on Learning Representations*, 2022.

[13] Johannes Gasteiger, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In *International Conference on Machine Learning*, 2018.

[14] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.

[15] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 9, pages 249–256, 2010.

[16] Martin Grohe, Moritz Lichter, and Daniel Neuen. The iteration number of the weisfeiler-leman algorithm. *ACM Transactions on Computational Logic*, 26(1):6:1–6:31, 2025.

[17] Zhiwei Guo and Heng Wang. A deep graph neural network-based mechanism for social recommendations. *IEEE Transactions on Industrial Informatics*, 17(4):2776–2783, 2020.

[18] William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, 2015.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 630–645. Springer, 2016.

[22] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, volume 37, pages 448–456. PMLR, 2015.

[23] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 2018.

[24] Kevin J. Lang Jure Leskovec and Michael Mahoney. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. In *Internet Mathematics*, volume 6, pages 29–123, 2009.

[25] Nicolas Keriven, Gilles Blanchard, Théo Mai, and Nicolas Vayatis. Neural injective functions for multisets, measures and graphs via a finite witness theorem. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pages 645–659, 2022.

[26] Sandra Kiefer and Brendan D. McKay. The iteration number of colour refinement. In *47th International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 168 of *LIPIcs*. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2020.

[27] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[28] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Machine Learning*, 2016.

[29] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In *International Conference on Learning Representations (ICLR)*, 2019.

[30] Andrei Leman and Boris Weisfeiler. A reduction of a graph to a canonical form and an algebra arising during this reduction. *Nauchno-Technicheskaya Informatsiya*, 2(9):12–16, 1968.

[31] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[32] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. In *International Conference on Learning Representations*, 2016.

[33] Yuankai Luo, Lei Shi, and Xiao-Ming Wu. Classic GNNs are strong baselines: Reassessing GNNs for node classification. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024.

[34] Péter Mernyei and Cătălina Cangea. Wiki-cs: A wikipedia-based benchmark for graph neural networks. *arXiv preprint arXiv:2007.02901*, 2020.

[35] Christopher Morris, Martin Ritzert, Matthias Fey, William L Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4602–4609, 2019.

[36] Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. In *International Conference on Learning Representations*, 2020.

[37] Ryoma Sato, Makoto Yamada, and Hisashi Kashima. Random features strengthen graph neural networks. In *Proceedings of the 2021 SIAM international conference on data mining (SDM)*, pages 333–341. SIAM, 2021.

[38] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008.

[39] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. *arXiv preprint arXiv:1811.05868*, 2018.

[40] Hamed Shirzad, Ameya Velingker, Balaji Venkatachalam, Danica J Sutherland, and Ali Kemal Sinop. Exphormer: Sparse transformers for graphs. In *International Conference on Machine Learning*, pages 31613–31632. PMLR, 2023.

[41] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. In *Journal of Machine Learning Research (JMLR)*, volume 15, pages 1929–1958. JMLR.org, 2014.

[42] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[43] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations (ICLR)*, 2018.

[44] Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

[45] Zhepeng Wang, Runxue Bao, Yawen Wu, Guodong Liu, Lei Yang, Liang Zhan, Feng Zheng, Weiwen Jiang, and Yanfu Zhang. Self-guided knowledge-injected graph neural network for alzheimer's diseases. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 378–388. Springer, 2024.

[46] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998.

[47] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *International conference on machine learning*, pages 6861–6871. Pmlr, 2019.

[48] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. Graph neural networks in recommender systems: a survey. *ACM Computing Surveys*, 55(5):1–37, 2022.

[49] Xinyi Wu, Zhengdao Chen, William Wei Wang, and Ali Jadbabaie. A non-asymptotic analysis of oversmoothing in graph neural networks. In *The Eleventh International Conference on Learning Representations*, 2023.

[50] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations (ICLR)*, 2019.

[51] Jiaxuan You, Rex Ying, Xiang Ren, William L Hamilton, and Jure Leskovec. Position-aware graph neural networks. In *International Conference on Machine Learning*, pages 7134–7143. PMLR, 2019.

[52] Zhaoning Yu and Hongyang Gao. Molecular representation learning via heterogeneous motif graph neural networks. In *International Conference on Machine Learning*, pages 25581–25594. PMLR, 2022.

A Mathematic Proofs

B Useful Mathematical Results

Lemma 2 (Matrix Chernoff inequalities)**.** *Consider a finite sequence $\{\boldsymbol{X}_k\}$ of independent, random, self-adjoint matrices with dimension d. Assume that each random matrix satisfies $0 \leq X_k \leq RI_n$. Denote*

$$\mu_{\min} = \lambda_{\min}\left\{\mathbb{E}\left(\sum_k X_k\right)\right\}, \quad \mu_{\max} = \lambda_{\max}\left\{\mathbb{E}\left(\sum_k X_k\right)\right\}. \tag{11}$$

Then

$$\mathbb{P}\left\{\lambda_{\max}\left(\sum_k X_k\right) \geq (1+\delta)\mu_{\max}\right\} \leq n\left[\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right]^{\mu_{\max}/R}, \quad \forall \delta \geq 0. \tag{12}$$

$$\mathbb{P}\left\{\lambda_{\min}\left(\sum_k X_k\right) \leq (1-\delta)\mu_{\min}\right\} \leq n\left[\frac{e^{-\delta}}{(1-\delta)^{1-\delta}}\right]^{\mu_{\min}/R}, \quad \forall 0 < \delta < 1. \tag{13}$$

C Missing Proofs for Section 4

This section includes the missing proofs in Section 4 that design and prove injectivity of message passing, which is further be leveraged to show the expressive power of GNNs.

C.1 Linear Injective Message Passing

In this subsection, we prove Proposition 1. Recall that the message-passing layer is defined as follows

$$\boldsymbol{h}_v^{\ell} = \phi(\boldsymbol{h}_v^{\ell-1}, f(\{\!\{\boldsymbol{h}_u^{\ell-1} : u \in \mathcal{N}_v\}\!\})) \tag{14}$$

where f is the aggregation operation and ϕ is the combination operation.

Let us suppose a graph with n vertices, and there are totally k **distinct** features. Additionally, we assume all distinct features are linearly independent. Then we can show both f and ϕ are injective by using a simple linear map.

First, we consider the aggregation operation f defined as follows

$$f(\{\!\{\boldsymbol{h}_u^{\ell-1} : u \in \mathcal{N}_v\}\!\}) = \sum_{u \in \mathcal{N}_v} \boldsymbol{h}_u^{\ell-1}. \tag{15}$$

Let $\boldsymbol{H} \in \mathbb{R}^{k \times d}$ be the distinct node features. Then we have

$$f(\{\!\{\boldsymbol{h}_u^{\ell-1} : u \in \mathcal{N}_v\}\!\}) = \boldsymbol{H}^{\top}\alpha_v, \tag{16}$$

where $\boldsymbol{\alpha}_v$ is an integer vector whose entries are all nonnegative integers and their sum equals the degree of node v. If the two nodes have the same aggregated messages, then we have

$$\boldsymbol{H}^{\top}\boldsymbol{\alpha}_v = \boldsymbol{H}^{\top}\boldsymbol{\alpha}_u \tag{17}$$

or equivalently

$$\boldsymbol{H}^{\top}(\boldsymbol{\alpha}_v - \boldsymbol{\alpha}_u) = 0. \tag{18}$$

Recall that we have distinct features that are linearly independent. Hence, $\boldsymbol{H}^{\top}$ is full column rank and $\boldsymbol{\alpha}_u = \boldsymbol{\alpha}_v$. This discussion proves the following result.

Lemma 3. *If distinct node features are linearly independent, then the linear aggregation function f is injective.*

Next, let us recall the linear message passing layer:

$$\boldsymbol{h}_v^{\ell} = (1+\epsilon)\boldsymbol{h}_v^{\ell-1} + \sum_{u \in \mathcal{N}_v} \boldsymbol{h}_u^{\ell-1} \tag{19}$$

To simplify the notation, we denote $\{\!\{\boldsymbol{h}_v^{\ell-1}\}\!\} := \{\!\{\boldsymbol{h}_u^{\ell-1} : u \in \mathcal{N}_v\}\!\}$. Given two inputs $(\boldsymbol{h}_v^{\ell-1}, \{\!\{\boldsymbol{h}_v^{\ell-1}\}\!\})$ and $(\boldsymbol{h}_{\bar{v}}^{\ell-1}, \{\!\{\boldsymbol{h}_{\bar{v}}^{\ell-1}\}\!\})$, we consider

$$\boldsymbol{h}_v^{\ell} = \boldsymbol{h}_{\bar{v}}^{\ell}, \tag{20}$$

or equivalently

$$(1+\epsilon)(\boldsymbol{h}_v^{\ell-1} - \boldsymbol{h}_{\bar{v}}^{\ell-1}) = \boldsymbol{H}_k^{\top}(\boldsymbol{\alpha}_{\bar{v}} - \boldsymbol{\alpha}_v) \tag{21}$$

If $\boldsymbol{h}_v^{\ell-1} = \boldsymbol{h}_{\bar{v}}^{\ell-1}$, then we have $\boldsymbol{H}^{\top}(\boldsymbol{\alpha}_{\bar{v}} - \boldsymbol{\alpha}_v) = 0$ implies $\boldsymbol{\alpha}_{\bar{v}} = \boldsymbol{\alpha}_v$ as we have proved before.

Now we will show $\boldsymbol{h}_v^{\ell-1} \neq \boldsymbol{h}_{\bar{v}}^{\ell-1}$ is impossible if we chose ε as a irrational number. Note that the LHS is the linear span of $\boldsymbol{h}_v^{\ell}$ and $\boldsymbol{h}_{\bar{v}}^{\ell}$. The RHS can only be their linear span as they are linearly independent. Hence, there exist nonzero integers a and b such that

$$\boldsymbol{H}^{\top}(\boldsymbol{\alpha}_{\bar{v}} - \boldsymbol{\alpha}_v) = a\boldsymbol{h}_v^{\ell-1} + b\boldsymbol{h}_{\bar{v}}^{\ell-1}. \tag{22}$$

Additionally, we will have $b = -a$. Otherwise, $\boldsymbol{h}_v$ and $\boldsymbol{h}_{\bar{v}}$ becomes linearly dependent. Then we obtain

$$(1+\epsilon)(\boldsymbol{h}_v^{\ell-1} - \boldsymbol{h}_{\bar{v}}^{\ell} - 1) = a(\boldsymbol{h}_v^{\ell-1} - \boldsymbol{h}_{\bar{v}}^{\ell-1}) \tag{23}$$

Recall that a is an integer, while ϵ is irrational. Hence, we must have $\boldsymbol{h}_v = \boldsymbol{h}_{\bar{v}}$.

Lemma 4. *Suppose distinct node features are linearly independent. If we choose $\epsilon > 0$ to be irrational, then the linear message passing is injective.*

Combining Lemma 3 and Lemma 4 yields the desired result in Proposition 1.

C.2 Nonlinear Lifting Enables Linearly Independence

We have shown that linear message passing is injective if distinct node feature vectors are linearly independent. However, this linear message passing cannot be done in an iterative manner since after one linear message passing, the distinct feature vectors become likely linearly dependent. Hence, in this section, we will show how to use a nonlinear transform to lift the distinct feature vectors to become linearly independent.

Let us consider that we have a totally of k distinct feature vectors and they can be linearly dependent or not. For simplicity, we denote them as $\boldsymbol{X} \in \mathbb{R}^{k\times d}$. Then we consider the following nonlinear transform

$$\boldsymbol{H} = \frac{1}{\sqrt{m}}\phi(\boldsymbol{X}\boldsymbol{W}) \in \mathbb{R}^{k\times m}, \tag{24}$$

where $\boldsymbol{W} \in \mathbb{R}^{d\times m}$. This is a one-layer MLP. Moreover, we will assume $\|\boldsymbol{x}\| = \mathbf{1}$ and we random initialize $\boldsymbol{W}$ such that

$$\boldsymbol{W}_{ij} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0,1). \tag{25}$$

Hence, to show $\boldsymbol{H}$ has linearly independent rows, it is equivalent to study the smallest eigenvalues of the following Gram matrix

$$\boldsymbol{G} = \boldsymbol{H}\boldsymbol{H}^{\top} = \frac{1}{m}\phi(\boldsymbol{X}\boldsymbol{W})\phi\boldsymbol{X}\boldsymbol{W})^{\top} = \frac{1}{m}\sum_{r=1}^{m}\phi(\boldsymbol{X}\boldsymbol{w}_r)\phi(\boldsymbol{X}\boldsymbol{w}_r)^{\top}, \tag{26}$$

where $\boldsymbol{w}_r \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$. With law of large number argument, as $m \to \infty$, we have $\boldsymbol{G}$ converges to

$$\boldsymbol{G}^{\infty} = \mathbb{E}[\phi(\boldsymbol{X}\boldsymbol{w})\phi(\boldsymbol{X}\boldsymbol{w})^{\top}]. \tag{27}$$

Then we can show that as long as ϕ is nonlinear but non-polynomial, the least eigenvalue of $\boldsymbol{K}^{\infty}$ is strictly positive definite.

The proof is based on the notion of dual activation [7]. The determinsitic matrix $\boldsymbol{G}$ has the following expansion form:

$$\boldsymbol{G}^{\infty} = \mathbb{E}\phi(\boldsymbol{X}\boldsymbol{w})\phi(\boldsymbol{X}\boldsymbol{w})^{\top} = \sum_{n=0}^{\infty} a_n^2(\boldsymbol{X}\boldsymbol{X}^{\top})^{\odot n}, \tag{28}$$

where a_n is the n-th Hermitian coefficients of ϕ and $\odot$ is the element-wise product. Then it is followed by [12, Lemma A.9] that the least eigenvalue is strictly positive definite.

Lemma 5. *Suppose $\boldsymbol{x}_i \in \mathbb{S}^{d-1}$ for all $i \in [n]$. If $\boldsymbol{x}_i \neq \boldsymbol{x}_j$ and ϕ is non-polynomial, then $\lambda_0 = \lambda_{\min}(\boldsymbol{G}^\infty) > 0$.*

Remarkably, the matrix $\boldsymbol{G}^\infty$ is deterministic, and it is the limit of $\boldsymbol{G}$ as m tends to infinity. Next, we will show that the least eigenvalue of $\boldsymbol{G}$ is highly likely to be positive even if m is near-linear in k.

We have

$$\boldsymbol{A}\boldsymbol{A}^\top = \sum_{r=1}^{m} \phi(\boldsymbol{X}\boldsymbol{w}_r)\phi(\boldsymbol{X}\boldsymbol{w}_r)^\top \tag{29}$$

where we denote $\boldsymbol{A} = \phi(\boldsymbol{X}\boldsymbol{W})$. As $\boldsymbol{w}_r$ are i.i.d., we have

$$\mathbb{E}\boldsymbol{A}\boldsymbol{A}^\top = m\boldsymbol{G}^\infty. \tag{30}$$

And so we obtain

$$\lambda_{\min}(\mathbb{E}\boldsymbol{A}\boldsymbol{A}^\top) = m\lambda_0. \tag{31}$$

For any positive $t > 0$, we consider the matrix $\boldsymbol{B}_t$ with each column defined

$$\boldsymbol{b}_r = \phi(\boldsymbol{X}\boldsymbol{w}_r)1\{\|\phi(\boldsymbol{X}\boldsymbol{w}_r)\| \leq t\}. \tag{32}$$

Then we have

$$\lambda_{\max}(\boldsymbol{b}_r\boldsymbol{b}_r^\top) \leq t^2 \tag{33}$$

and

$$\lambda_{\min}(\boldsymbol{A}\boldsymbol{A}^\top) \geq \lambda_{\min}(\boldsymbol{B}_t\boldsymbol{B}_t^\top), \quad \forall t > 0. \tag{34}$$

For any $\epsilon > 0$, we apply the Matrix Chernoff inequality to the matrices $\{\boldsymbol{b}_r\boldsymbol{b}_r^\top\}$ and get

$$\mathbb{P}\left\{\lambda_{\min}(\boldsymbol{B}_t\boldsymbol{B}_t^\top) \leq (1-\epsilon)\lambda_{\min}(\mathbb{E}\boldsymbol{B}_t\boldsymbol{B}_t^\top)\right\} \leq ke^{-\epsilon^2\lambda_{\min}(\mathbb{E}\boldsymbol{B}_t\boldsymbol{B}_t^\top)/(2t^2)}. \tag{35}$$

Chose $\epsilon = 1/2$ and define

$$\boldsymbol{G}_t := \mathbb{E}\phi(\boldsymbol{X}\boldsymbol{w})\phi(\boldsymbol{X}\boldsymbol{w})^\top 1\{\|\phi(\boldsymbol{X}\boldsymbol{w})\| \leq t\}. \tag{36}$$

The inequality becomes

$$\mathbb{P}\left\{\lambda_{\min}(\boldsymbol{B}_t\boldsymbol{B}_t^\top) \leq m\lambda_{\min}(\boldsymbol{G}_t)/2\right\} \leq ke^{-m\lambda_{\min}(\boldsymbol{G}_t)/(8t^2)}. \tag{37}$$

If we choose $m \geq \frac{8t^2}{\lambda_{\min}(\boldsymbol{G}_t)}\log(\frac{k}{\delta})$, then with probability at least $1-\delta$, we have

$$\lambda_{\min}(\boldsymbol{B}_t\boldsymbol{B}_t^\top) \geq m\lambda_{\min}(\boldsymbol{G}_t)/2 \tag{38}$$

Note that

$$\begin{aligned}
\|\boldsymbol{G}^\infty - \boldsymbol{G}_t\| =& \|\mathbb{E}\left[\phi(\boldsymbol{X}\boldsymbol{w})\phi(\boldsymbol{X}\boldsymbol{w})^T\right] - \phi(\boldsymbol{X}\boldsymbol{w})\phi(\boldsymbol{X}\boldsymbol{w})^T 1\{\|\phi(\boldsymbol{X}\boldsymbol{w})\| \leq t\}\right]\| && (39)\\
\leq& \mathbb{E}\|\left[\phi(\boldsymbol{X}\boldsymbol{w})\phi(\boldsymbol{X}\boldsymbol{w})^T\right] - \phi(\boldsymbol{X}\boldsymbol{w})\phi(\boldsymbol{X}\boldsymbol{w})^T 1\{\|\phi(\boldsymbol{X}\boldsymbol{w})\| \leq t\}\right]\|, \quad (i) && (40)\\
=& \mathbb{E}\left[\|\phi(\boldsymbol{X}\boldsymbol{w})\|^2 \cdot 1\{\|\phi(\boldsymbol{X}\boldsymbol{w})\| > t\}\right] && (41)\\
\leq& \int_0^\infty \mathbb{P}\left(\|\phi(\boldsymbol{X}\boldsymbol{w})\|^2 \cdot 1\{\|\phi(\boldsymbol{X}\boldsymbol{w})\| > t\} \geq s\right) ds && (42)
\end{aligned}$$

Since ϕ is Lipschitz continuous, $\|\phi(\boldsymbol{X}\boldsymbol{w})\|$ is a sub-Gaussian random variable with coefficient $C\|\boldsymbol{X}\|$, where the constant $C > 0$ only depends on the Lipschitz coefficient of ϕ. Then we obtain

$$\|\boldsymbol{G}^\infty - \boldsymbol{G}_t\| \leq \int_0^\infty e^{-c_0(s+t^2)/(2\|\boldsymbol{X}\|^2)}ds \leq e^{-c_0t^2/(2\|\boldsymbol{X}\|^2)} \cdot \frac{2\|\boldsymbol{X}\|^2}{c_0} \tag{43}$$

where $c_0 > 0$ is some absolute constant.

We can choose $t > 0$ small so that the RHS is less than $\lambda_0/2$. Specifically, we choose

$$t = \sqrt{\frac{2\|\boldsymbol{X}\|^2}{c_0}\log\left(\frac{4\|\boldsymbol{X}\|^2}{c_0\lambda_0}\right)} \tag{44}$$

and obtain

$$\|\boldsymbol{G}^{\infty} - \boldsymbol{G}_t\| \leq \lambda_0/2. \tag{45}$$

This yields

$$\lambda_{\min}(\boldsymbol{G}_t) \geq \lambda_{\min}(\boldsymbol{G}^{\infty}) - \|\boldsymbol{G}^{\infty} - \boldsymbol{G}_t\| \geq \lambda_0/2. \tag{46}$$

Altogether, we have

$$\lambda_{\min}(\boldsymbol{A}\boldsymbol{A}^{\top}) \geq \lambda_{\min}(\boldsymbol{B}_t\boldsymbol{B}_t) \geq m\lambda_{\min}(\boldsymbol{G}_t) \geq m\lambda_0/4. \tag{47}$$

Lemma 6. *Suppose ϕ is Lipschitz continuous and $\lambda_0 > 0$. Then, for any $\delta > 0$, if $m = \Omega(\frac{k}{\lambda_0}\log(\frac{k}{\lambda_0})\log(\frac{k}{\delta}))$, then with probability at least $1-\delta$, we have $\lambda_{\min}(\boldsymbol{G}) \geq \lambda_0/4$. Hence, the feature matrix $\boldsymbol{H}$ has linearly independent rows.*

Therefore, combining Lemma 5 with Lemma 6 yields the desired result in Proposition 2.

C.3 Expressive GNNs with Injective Message Passing

Recall the foundational result from [50], as restated below.

Theorem 2. *[50, Lemma 2 and Theorem 3] Suppose the aggregation and combination functions are injective. Then a GNN returns different embeddings for two given graphs if and only if the 1-WL test decides non-isomorphic.*

Combining Propostion 1, Propostion 2, and Theorem 2 yields Theorem 1.

D Missing Proofs for Section 5

D.1 Depth Analysis for Message Passing Layers

Let us first consider the fully connected graph K_n. We will assume each node initially has the same node features or colors. Since all nodes are connected, each u receives an identical multiset of neighbor colors. Combining with its unique color, the resulting color remains the same for all u. No further refinement occurs because the colors are stabilized. This implies that GNN achieves the maximal expressive power using one message passing scheme, since it simulates the 1-WL test iteration using an injective message passing operation.

Lemma 7. *For the complete graph K_n with identical initial node colors, the 1-WL test stabilizes after one iteration. Consequently, a GNN with one message-passing layer achieves maximal expressive power on K_n.*

Now, let us consider the path graph P_n with nodes $\{v_1, \cdots, v_n\}$ and edges (v_i, v_{i+1}). Assume all nodes start with their identical initial colors, *i.e.*, $c_u^0 = c$ for all $u \in V$. At the first iteration, endpoints v_1 and v_n (degree 1) receive multiset $\{c^{(0)}\}$, while the internal nodes $\{v_2, \cdots, v_{n-1}\}$ (degree 2) receive multiset $\{c^{(0)}, c^{(0)}\}$. Hence, endpoints and internal nodes are assigned distinct colors after hashing: the endpoints have new colors while the internal nodes retain c, not refined yet. At iteration k, nodes within k hops of an endpoint have refined their colors. The "new endpoints" v_k and v_{n-k+1} are assigned new colors, while the rest internal nodes remain unchanged. Hence, the color propagates inward until the midpoints stabilize. Hence, we need $\lceil n/2 \rceil$ iterations to stabilize all colors. Consequently, GNNs with injective message passing need $\lceil n/2 \rceil$ message passes to distinguish all nodes in P_n, achieving the maximal expressive power.

Lemma 8. *For the path graph P_n with identical initial colors, the 1-WL test stabilizes after $\lceil n/2 \rceil$ iterations. Consequently, a GNN needs $\lceil n/2 \rceil$ message-passing layers to achieve maximal expressive power on P_n.*

Consider a complete binary tree with $n = 2^h - 1$ nodes, where h is the height of the tree. When all nodes begin with identical initial colors, the 1-WL test exhibits a characteristic bottom-up refinement process. At the first iteration, the leaves (degree 1) become distinguishable from internal nodes (degree 2 or 3 for the root), receiving new colors. In contrast, internal nodes retain their original color (except probably the root). This creates the first level of differentiation at the tree's lowest level.

The refinement proceeds upward through subsequent iterations. At each step k, nodes at height k from the bottom become distinguishable because they now see distinct color patterns in their subtrees. Specifically, parents of already-distinguished child nodes receive new colors based on their children's unique color configurations, while higher-level nodes remain unchanged until their turn in this propagation process.

At the second iteration, the parents-of-leaves nodes now see their children have a special leaf color. Hence, these parents get a new color while the rest internal nodes remain unchanged. Remarkably, it essentially treats the parents-of-leaves nodes as "new leaves" and assigns new "leaf" colors to them. Hence, each iteration reveals one more level of the hierarchy, and the refinement proceeds upward from leaves to root. Hence, the total number of iterations equals the tree height $h = \Theta(\log n)$.

Lemma 9. *For a balanced binary tree graph with n nodes with identical initial colors, the 1-WL test stabilizes after $\Theta(\log n)$ iterations. Consequently, a GNN needs $\Theta(\log n)$ message-passing layers to achieve maximal expressive power.*

Consider a $\sqrt{n} \times \sqrt{n}$ grid graph, where all nodes start with identical initial colors, *i.e.*, $c_u^{(0)} = c$ for all $u \in V$. At the first iteration, the corner nodes (degree 2) receive a multiset $\{c^{(0)}, c^{(0)}\}$ and the boundary nodes (degree 3) receive a multiset $\{c^{(0)}, c^{(0)}, c^{(0)}\}$ from their neighbors, while the rest internal nodes receive $\{c^{(0)}, c^{(0)}, c^{(0)}, c^{(0)}\}$. Hence, we can assign two new colors to the corner and boundary nodes, while the color of the internal nodes remains unchanged. At the k-th iteration, nodes at distance $k-1$ from the boundary are refined. The new color refinement propagates inward from the boundary at a rate of one layer per iteration. Hence, the process stabilizes when it reaches the center of the grid. As a result, we need $\lceil \sqrt{n}/2 \rceil$ iterations to stabilize the colors, since the maximum distance from the boundary to the center is $\lceil \sqrt{n}/2 \rceil$.

Lemma 10. *For a $\sqrt{n} \times \sqrt{n}$ gride graph with identical initial colors, the 1-WL test stabilizes after $\Theta(\sqrt{n})$ iterations. Consequently, a GNN needs $\Theta(\log n)$ message-passing layers to achieve maximal expressive power.*

D.2 Stability of Deep Message Passing

In this section, we prove the iterative relation of $\mathbb{E}[\|\boldsymbol{H}^{(\ell)}\|^2]$ in Lemma 1. Recall that the proposed ML has the following matrix form:

$$\boldsymbol{H}^{(\ell)} = ((1+\epsilon)\boldsymbol{I}_n + \boldsymbol{A}) \frac{1}{\sqrt{m}} \phi(\boldsymbol{H}^{(\ell-1)}\boldsymbol{W}^{(\ell)}). \tag{48}$$

As each layer uses independent $\boldsymbol{W}^{\ell}$, we can first assume $\boldsymbol{H}^{(\ell-1)}$ is fixed. Then we have

$$\begin{aligned}
&\mathbb{E}\left[\|\boldsymbol{H}^{(\ell)}\|_F^2 \mid \boldsymbol{H}^{(\ell-1)}\right] \\
=&\frac{1}{m}\operatorname{Tr}\left(\boldsymbol{T}^2 \mathbb{E}\phi(\boldsymbol{H}^{(\ell-1)}\boldsymbol{W}^{(\ell)})\phi(\boldsymbol{H}^{(\ell-1)}\boldsymbol{W}^{(\ell)})^\top\right) \\
=&\frac{1}{m}\sum_{r=1}^{m}\operatorname{Tr}\left(\boldsymbol{T}^2 \mathbb{E}\phi(\boldsymbol{H}^{(\ell-1)}\boldsymbol{w}_r)\phi(\boldsymbol{H}^{(\ell-1)}\boldsymbol{w}_r)^\top\right) \\
\leq&\frac{K^2}{m}\sum_{r=1}^{m}\operatorname{Tr}\left(\boldsymbol{T}^2 \boldsymbol{H}^{(\ell-1)} \mathbb{E}\left[\boldsymbol{w}_r\boldsymbol{w}_r^\top\right] \boldsymbol{H}^{(\ell-1)\top}\right) \\
=&\sigma_\ell^2 K^2 \operatorname{Tr}\left(\boldsymbol{T}^2 \boldsymbol{H}^{(\ell-1)}\boldsymbol{H}^{(\ell-1)\top}\right) \\
\leq&\sigma_\ell^2 K^2 \|\boldsymbol{T}\|^2 \|\boldsymbol{H}^{(\ell-1)}\|_F^2,
\end{aligned}$$

where $\boldsymbol{T} = (1+\epsilon)\boldsymbol{I}_n + \boldsymbol{A}$ and we use the Lipschitz continuity of ϕ. Adding the expectation on both sides for $\boldsymbol{H}^{\ell-1}$ yields the desired result.

E Experiments Setup and Additional Results

Since the supplemental submission deadline is one week after the full paper deadline, we only include the experimental setups here. Please refer to the additional results in the supplemental files.

Following [33], we select eight node classification datasets to evaluate our method. Cora, CiteSeer, and PubMed are widely used benchmark datasets for evaluating citation networks [38]. Following the semi-supervised learning setup described in [28], we apply the same data splits and evaluation metrics. Additionally, we include the Computer and Photo datasets from [39], which represent co-purchase networks where nodes correspond to products and edges indicate frequent co-purchases. We also consider the CS and Physics datasets from [39], which are co-authorship networks where nodes represent authors, and edges signify collaborative publications. For these datasets, we adopt the standard 60%/20%/20% split for training, validation, and testing, using accuracy as the evaluation metric [6, 40, 8, 33]. Lastly, we evaluate on the WikiCS dataset, leveraging its official data splits and metrics as specified in [34]. We perform hyperparameter tuning for all experiments, following the search space defined in [8, 33]. Specifically, we employ the Adam optimizer [27] with learning rates selected from 0.001, 0.005, 0.01 and a maximum of 2500 training epochs. The hidden dimension is tuned over 64, 256, 512, while dropout rates are chosen from 0.2, 0.3, 0.5, 0.7. We also explore the number of message-passing layers and feature-learning layers within the range of 1, 2, 3, 4, 5, 6, 7, 8, 9, 10. All reported results represent the mean and standard deviation across five independent runs with different random initializations. For baseline comparisons, we utilize the official code provided by [33]. All experiments are tested on a server with a CPU AMD Threadripper 2990WX, a GPU Nvidia RTX 4090, and 128GB of memory.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction clearly summarize the paper's contributions: (1) identifying the lack of injectivity as the cause of oversmoothing, (2) introducing injective message passing without training, and (3) proposing a decoupled architecture that enables deep GNNs with theoretical guarantees.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We discuss assumptions for our theoretical analysis in Section 3 and empirical limitations due to the noise introduced from random initialization in Section 6.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [Yes]

Justification: Each proposition includes clearly stated assumptions, and proofs are provided in Appendix C and D.

Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: All experimental setups, training details, hyperparameters, and architectures are detailed in Section 5 and the Appendix E.

Guidelines:

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

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

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

   Answer: [Yes]

   Justification: Code will be made available via an anonymized GitHub link in the supplementary material upon acceptance.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: See Section 6 and Appendix E for full hyperparameter settings, dataset splits, optimizers, and evaluation protocols.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: Results include mean and standard deviation over five random seeds for all key experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Appendix E specifies the GPU, maximum training epochs, and memory setup.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: Our research does not involve human data, misinformation, or malicious use. It adheres to all ethical guidelines.

   Guidelines:

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

   Yes, that's a reasonable and honest justification—especially for theoretical papers where societal impact is indirect or speculative. Here's how you can phrase it more clearly for the checklist:

   A: [Yes]

10. **Broader impacts**

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

    Answer: [Yes]

Justification: While we do not include a standalone Broader Impacts section, the Introduction briefly mentions that GNNs are widely used in domains such as molecular property prediction. Our theoretical contributions may benefit applications in these areas by enabling deeper and more expressive graph models.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

Answer: [NA]

Justification: No high-risk models or datasets are released.

Guidelines:

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

12. **Licenses for existing assets**

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

Answer: [Yes]

Justification: We use standard benchmark datasets (e.g., ZINC, Reddit) and properly cite their original sources.

Guidelines:

- The answer NA means that the paper does not use existing assets.

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

13. **New assets**

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

    Answer: [NA]

    Justification: We do not introduce new datasets or pre-trained models.

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: No human subjects or user studies are involved.

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: No human subjects were involved in this work.

Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [NA]

Justification: LLMs were not used in developing the method or experiments.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.