Beyond Node Sequences: Relational Diffusion for Unified Graph Learning

Chunyu Wei* **Xianan Wang*** **Wenji Hu** **Yueguo Chen†**
School of Information, Renmin University of China, China
weicy15@icloud.com
wangxianan@ruc.edu.cn
2024000991@ruc.edu.cn
chenyueguo@ruc.edu.cn

Abstract

Graph foundation models have emerged as a promising paradigm for generalizing across diverse structural tasks. Current approaches serialize graphs into node token sequences and apply autoregressive or masked prediction, yet this node sequential modeling disrupts the relation-centric nature of graphs and violates permutation equivariance. We propose *relational diffusion*, a framework that reconceptualizes graph modeling through edge co-occurrence modeling. By treating edges as fundamental units and learning their joint distribution via diffusion processes, our approach naturally respects the relational semantics of graphs while maintaining permutation equivariance. We introduce topological edge tokenization to enrich edge representations with multi-hop structural context, and relational prompt completion to unify node classification, link prediction, and graph classification as sequence completion problems over edge representations. Extensive experiments demonstrate that relational diffusion achieves strong performance while providing a principled foundation aligned with the relational nature of graphs. [3]

1 Introduction

Graph-structured data underpins our understanding of complex systems, from molecular interactions to social networks. While large language models have transformed sequential text processing, they lack the capacity for relational reasoning over graph structures. This gap has motivated the development of graph foundation models: systems capable of generalizing across diverse structural tasks through unified representations.

Current graph foundation models follow two main paradigms. Traditional graph neural networks excel at modeling local neighborhoods but suffer from over-smoothing and limited scalability. Transformer-based approaches have recently demonstrated improved capacity for long-range dependencies through global attention mechanisms. Building on these advances and inspired by the success of large language models, researchers have reframed graph learning as token-based sequence modeling. Such methods serialize graphs into node token sequences and apply prediction strategies such as autoregressive next-token prediction (e.g., GraphGPT Zhao et al. [2025]), treating node classification, link prediction, and graph classification as unified sequence modeling problems.

However, these **node sequential modeling** approaches suffer from two fundamental limitations. First, they disrupt the relation-centric nature of graphs. Graphs fundamentally encode relations

*Equal contribution.

†Corresponding author. He works at Big Data and Responsible Artificial Intelligence for National Governance, Renmin University of China.

[3]Code is available at https://github.com/wangxianan95/RelationalDiffusion.

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

between entities through edges, which are the atomic units of relational information. Node-centric methods reduce edges to implicit artifacts derived from node orderings, forcing models to infer relationships indirectly. Second, sequential modeling violates permutation equivariance: graph representations should remain invariant to node reordering. While existing methods impose artificial ordering schemes like Eulerian paths, no single traversal can fully capture rich graph topology without information loss. These limitations stem from a fundamental misalignment between modeling graphs as node sequences when their semantic essence lies in edge co-occurrence patterns.

To this end, we propose modeling graphs through *edge co-occurrence modeling* rather than node sequential modeling, as illustrated in Figure 1. We introduce **relational diffusion**, a framework that learns the joint distribution over edge configurations via diffusion processes. By treating edges as fundamental modeling units, we naturally respect the relation-centric semantics of graphs while avoiding arbitrary ordering constraints. The diffusion model iteratively refines complete edge configurations through self-attention, adaptively learning edge co-occurrence patterns across diverse graph structures.

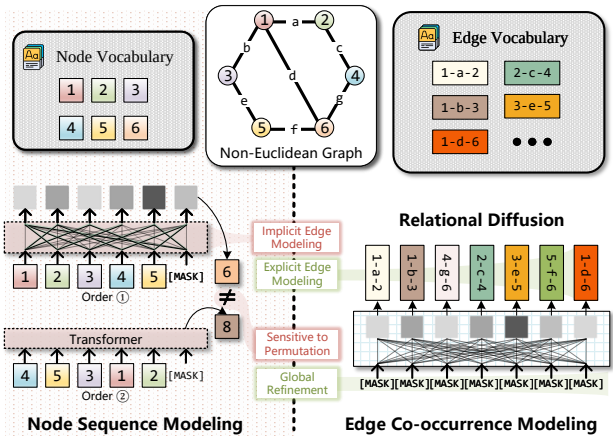

Figure 1: Node sequence modeling (left) relies on implicit edge representations derived from node orderings. Our edge co-occurrence modeling approach (right) explicitly represents edges as fundamental units and employs relational diffusion for global refinement.

Instantiating edge co-occurrence modeling for graph foundation models raises two key questions:

1. Topological enrichment: How can edge representations capture higher-order topological structure beyond pairwise connectivity?

2. Task unification: How can we unify heterogeneous graph tasks within this framework?

To achieve topological enrichment, we propose **topological edge tokenization** to enrich edge representations with multi-hop structural context. We employ graph neural networks to construct edge embeddings that incorporate neighborhood information through message passing, ensuring each edge token encodes not merely a pairwise connection but also the surrounding topological context. During pre-training, the diffusion model learns to denoise these topology-enriched representations, discovering structural regularities that generalize across graphs.

To achieve task unification, we propose **relational prompt completion** to reformulate diverse graph tasks as unified sequence completion problems over edge representations. Each input sequence comprises edge tokens encoding graph structure, task-specific prompts conditioning the prediction objective, and masked result tokens. Node classification, link prediction, and graph classification thus reduce to a common generative process: the pre-trained diffusion model iteratively denoises prompted edge sequences, leveraging learned co-occurrence patterns for multi-task generalization.

Contribution. (1) **relational diffusion**, a novel edge co-occurrence modeling framework for graph foundation models that treats edges as fundamental modeling units. By learning the joint distribution over edge configurations via diffusion processes, our approach naturally respects the relation-centric semantics of graphs while maintaining permutation equivariance; (2) **topological edge tokenization** to enrich edge representations with multi-hop structural context, and **relational prompt completion** to unify heterogeneous graph tasks as sequence completion problems over edge representations. (3) Extensive experiments across node classification, link prediction, and graph classification benchmarks, demonstrate that relational diffusion achieves strong performance.

2 Related Work

Graph Neural Networks. Graph Neural Networks established the computational foundation for learning on graph-structured data through neighborhood aggregation and message passing. Sem-

inal architectures including GCN Kipf and Welling [2017], GAT Velickovic et al. [2018], and GraphSAGE Hamilton et al. [2017] demonstrated effective learning across node classification, link prediction, and graph classification Gilmer et al. [2017], Xu et al. [2019]. While GNNs excel at modeling local structural patterns, they face known limitations including over-smoothing in deep architectures Li et al. [2018], expressiveness bounded by the Weisfeiler-Lehman test Xu et al. [2019], and challenges in capturing long-range dependencies.

Graph Foundation Models. The evolution toward general-purpose graph foundation models has been driven by transformer architectures and language-model-inspired training paradigms. Transformer-based approaches adapt self-attention to graphs through structural encodings Ying et al. [2021], Dwivedi et al. [2022], Rampásek et al. [2022], with recent work addressing scalability via sparse attention Shirzad et al. [2023] and achieving polynomial expressiveness with linear complexity Deng et al. [2024]. Language model paradigms have been transferred to graphs through autoregressive approaches such as GraphGPT Zhao et al. [2025], which serializes graphs via Eulerian paths and scales to 400M+ parameters, and masked prediction methods like GraphMAE Hou et al. [2022, 2023]. Pre-training strategies enable cross-domain generalization through contrastive learning You et al. [2020], Qiu et al. [2020], Zhu et al. [2020], Xia et al. [2022], Wei et al. [2023], generative reconstruction objectives Hu et al. [2020b], Rong et al. [2020], and prompt-based methods Sun et al. [2023b], Liu et al. [2023b], Fang et al. [2023], Zhao et al. [2024], with recent efforts toward universal graph foundation models introducing transferable vocabularies Liu et al. [2023a], Jin et al. [2024], Wang et al. [2025], cross-domain adaptation frameworks Sun et al. [2022], Zhao et al. [2024], and automated fine-tuning strategies Hu et al. [2026]. Large language models have also been equipped to reason over large-scale and textual graphs via tool chaining and hierarchical retrieval Wei et al. [2025a, 2026b]. These approaches operate through node-centric modeling, treating edges as implicit connectivity. Our work diverges by introducing edge-centric relational diffusion, where edges constitute the atomic modeling units and iterative global refinement replaces token-by-token prediction.

Diffusion Language Models. Diffusion models for continuous image generation Ho et al. [2020], Song et al. [2021] have been extended to discrete domains Austin et al. [2021], Campbell et al. [2022], Sun et al. [2023a] and text, including continuous latent diffusion Li et al. [2022], sequence-to-sequence generation Gong et al. [2023], semi-autoregressive modeling Han et al. [2023], and pre-trained-model acceleration He et al. [2023]. Recent advances have narrowed the gap with autoregressive models via score entropy Lou et al. [2024], masked diffusion Sahoo et al. [2024], and scaling to 8B parameters Nie et al. [2025], with further extensions to structured prediction Campbell et al. [2024] and accelerated sampling Zheng et al. [2024]. These works motivate our use of iterative joint denoising; however, existing diffusion models operate on token sequences and lack mechanisms to ensure topological consistency in graph generation.

3 Preliminaries

Consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = \{v_1, \ldots, v_N\}$ denotes the set of N nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ represents edges encoding pairwise relations. Each node v_i has feature vector $\mathbf{x}_i \in \mathbb{R}^{d_v}$, with complete node features $\mathbf{X} \in \mathbb{R}^{N \times d_v}$. Graph topology is encoded through adjacency matrix $\mathbf{A} \in \{0,1\}^{N \times N}$, where $\mathbf{A}_{ij} = 1$ iff $(v_i, v_j) \in \mathcal{E}$.

Graph tasks. Graph learning encompasses three fundamental categories: *node classification* assigns labels $y_i \in \mathcal{Y}_{\text{node}}$ via $f_{\text{node}} : \mathcal{G} \times v_i \to \mathcal{Y}_{\text{node}}$; *link prediction* determines edge existence through $f_{\text{link}} : \mathcal{G} \times (v_i, v_j) \to [0,1]$; and *graph classification* assigns labels to entire graphs via $f_{\text{graph}} : \mathcal{G} \to \mathcal{Y}_{\text{graph}}$.

Diffusion Models. Diffusion models Ho et al. [2020] define a two-phase process over a data distribution. The **forward process** progressively corrupts clean data $\mathbf{z}_0$ by injecting Gaussian noise over T steps, with the marginal at any step given in closed form as:

$$q(\mathbf{z}_t|\mathbf{z}_0) = \mathcal{N}(\mathbf{z}_t;\, \sqrt{\bar{\alpha}_t}\,\mathbf{z}_0,\, (1-\bar{\alpha}_t)\mathbf{I}), \quad \bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s) \tag{1}$$

where $\{\beta_t\}_{t=1}^{T}$ is a fixed variance schedule. The **reverse process** learns to recover clean data from corrupted observations via a parameterized denoising network f_θ, trained by minimizing reconstruction error between predicted and actual clean data. At inference, predictions are obtained by applying the learned reverse process to noisy inputs.

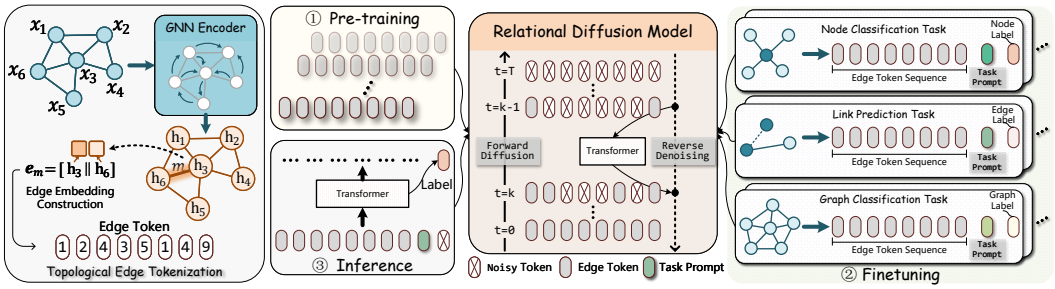

Figure 2: **Relational Diffusion framework. Left**: Topological edge tokenization via GNN encoding. **Center**: Relational diffusion model with forward corruption and transformer-based reverse denoising. ① Pre-training on edge sequences; ② Task-prompted fine-tuning for node classification, link prediction, and graph classification; ③ Single-step inference with masked result token.

4 Methodology

We present Relational Diffusion, comprising three components: (1) topological edge tokenization that enriches edge representations with multi-hop structural context, (2) a relational diffusion model that learns edge co-occurrence patterns through denoising, and (3) relational prompt completion that unifies diverse graph tasks. Figure 2 illustrates the complete pipeline.

4.1 Relational Diffusion Framework

We treat edges as primary objects of the generative process, operating directly on edge representations to respect the relation-centric nature of graphs while maintaining permutation equivariance.

Edge-centric representation. Given $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $|\mathcal{E}| = M$, we represent the graph through an edge collection $\mathbf{E} = \{e_1, \ldots, e_M\}$ where $e_m = (v_i^{(m)}, v_j^{(m)})$. We treat this as an *unordered set*, since graph structure is permutation-invariant with respect to edge ordering. Each edge is mapped to $\mathbf{e}_m \in \mathbb{R}^{d_e}$ through topological edge tokenization $\tau : \mathcal{E} \to \mathbb{R}^{d_e}$ (Section 4.2), producing the clean data $\mathbf{z}_0 = \{\mathbf{e}_1, \ldots, \mathbf{e}_M\}$ that serves as input to the diffusion process defined in Section 3.

Permutation-equivariant architecture. We adopt a Diffusion Transformer (DiT) with self-attention as the denoising network f_θ. Self-attention is permutation-equivariant: for any permutation σ of edge indices, $f_\theta(\mathbf{z}_{\sigma(1)}, \ldots, \mathbf{z}_{\sigma(M)}) = (f_\theta(\mathbf{z}_1, \ldots, \mathbf{z}_M)_{\sigma(1)}, \ldots, f_\theta(\mathbf{z}_1, \ldots, \mathbf{z}_M)_{\sigma(M)})$. We deliberately omit positional encodings on edge tokens, as they would break this property.

Diffusion in edge embedding space. We instantiate the diffusion process of Section 3 directly in *edge embedding space* $\mathbb{R}^{d_e}$, rather than in pixel or token space as in prior work. This is a central design choice: corrupting $\mathbf{z}_0$ via Gaussian noise directly perturbs relational structure, forcing the model to learn meaningful edge co-occurrence patterns during denoising. The denoising network $f_\theta : \mathbb{R}^{M \times d_e} \times \mathbb{N} \to \mathbb{R}^{M \times d_e}$ is implemented as a Transformer, where timestep t is embedded via sinusoidal encodings and added to token embeddings. Self-attention then captures complex co-occurrence dependencies: when denoising e_m, the model attends to all other edges without bias toward any ordering.

Training objective. Following the reconstruction objective of Section 3, we optimize:

$$\mathcal{L}_{\text{relational}} = \mathbb{E}_{t \sim \mathcal{U}(1,T),\, \mathbf{z}_0,\, \mathbf{z}_t \sim q(\mathbf{z}_t|\mathbf{z}_0)} \left[\sum_{m=1}^{M} \|\mathbf{e}_m - f_\theta(\mathbf{z}_t, t)_m\|_2^2 \right]. \tag{2}$$

4.2 Topological Edge Tokenization

Effective relational diffusion requires edge representations encoding rich topological context. We propose enriching edges with multi-hop neighborhood information through GNN encoding.

GNN-based edge embeddings. Given $\mathcal{G}$ with features $\mathbf{X} \in \mathbb{R}^{N \times d_v}$, we apply a GNN encoder to obtain $\mathbf{H} = \text{GNN}_{\text{enc}}(\mathbf{X}, \mathcal{G}) \in \mathbb{R}^{N \times d_h}$ via L-hop message passing:

$$\mathbf{h}_i^{(\ell+1)} = \text{AGGREGATE}^{(\ell)}\left(\{\mathbf{h}_j^{(\ell)} : v_j \in \mathcal{N}(v_i)\}\right) \oplus \mathbf{h}_i^{(\ell)} \tag{3}$$

with $\mathbf{h}_i^{(0)} = \mathbf{W}_{\text{init}}\mathbf{x}_i$ and $\oplus$ denoting concatenation or summation. We employ Graph Attention Networks with adaptive importance weights:

$$\text{AGGREGATE}^{(\ell)}(\{\mathbf{h}_j^{(\ell)}\}) = \sum\nolimits_{v_j \in \mathcal{N}(v_i)} \alpha_{ij}^{(\ell)} \mathbf{W}^{(\ell)} \mathbf{h}_j^{(\ell)} \tag{4}$$

$$\alpha_{ij}^{(\ell)} = \frac{\exp(a^{(\ell)}(\mathbf{h}_i^{(\ell)}, \mathbf{h}_j^{(\ell)}))}{\sum_{v_k \in \mathcal{N}(v_i)} \exp(a^{(\ell)}(\mathbf{h}_i^{(\ell)}, \mathbf{h}_k^{(\ell)}))}, \quad a^{(\ell)}(\mathbf{h}_i, \mathbf{h}_j) = \text{LeakyReLU}(\mathbf{w}_a^{(\ell)\top}[\mathbf{h}_i \| \mathbf{h}_j]) \tag{5}$$

Edge embedding construction. For edge $e_m = (v_i^{(m)}, v_j^{(m)})$: $\mathbf{e}_m = \mathbf{W}_{\text{edge}} \left[\mathbf{h}_{v_i^{(m)}} \| \mathbf{h}_{v_j^{(m)}} \| \mathbf{s}_m\right] \in \mathbb{R}^{d_e}$, where $\mathbf{s}_m$ encodes structural properties (e.g., betweenness centrality, clustering coefficient) and $\mathbf{W}_{\text{edge}} \in \mathbb{R}^{d_e \times (2d_h + d_s)}$ is a learnable projection. Continuous edge embeddings serve directly as input to diffusion, avoiding the bottleneck of discrete quantization. The GNN encoder is trained end-to-end with the diffusion model.

4.3 Relational Prompt Completion

We unify diverse graph tasks via the subgraph classification perspective: node classification becomes single-node subgraph classification, link prediction becomes node-pair subgraph classification, and graph classification operates on the full graph.

Unified embedding space. We use a shared space $\mathbb{R}^{d_e}$ accommodating three token types: edge embeddings $\mathbf{e}_m$, learnable prompt embeddings $\mathbf{p}^{\mathcal{T}}$ for $\mathcal{T} \in \{\text{node}, \text{link}, \text{graph}\}$, and learnable result embeddings $\mathbf{r}^y$ for outcomes y (label tokens for classification, $\{\mathbf{r}^{\text{exist}}, \mathbf{r}^{\text{not-exist}}\}$ for link prediction). During inference, the unknown result token is initialized as Gaussian noise $\mathbf{r}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, representing an unspecified prediction to be recovered by the reverse diffusion process.

Sequence construction. Given task $\mathcal{T}$ with query q on $\mathcal{G}$, we extract task-specific subgraph $\mathcal{G}_q = (\mathcal{V}_q, \mathcal{E}_q)$, compute $\mathbf{z}_{\mathcal{G}_q} = \{\mathbf{e}_1, \ldots, \mathbf{e}_{M_q}\}$, and concatenate:

$$\mathbf{z}_{\text{input}}^{\mathcal{T}} = [\mathbf{z}_{\mathcal{G}_q}, \mathbf{p}^{\mathcal{T}}, \mathbf{r}^{\mathcal{T}}] \in \mathbb{R}^{(M_q+2)\times d_e} \tag{6}$$

Edge embeddings form an unordered set; prompt and result embeddings occupy fixed positions to condition predictions.

Task-specific subgraph extraction. Three tasks differ only in their subgraph definitions, as illustrated in Figure 2.

- For node classification on v_i, we extract the K-hop neighborhood: $\mathcal{V}_{v_i} = \{v_j : d(v_i, v_j) \le K\}$ and $\mathcal{E}_{v_i} = \{(v_j, v_k) \in \mathcal{E} : v_j, v_k \in \mathcal{V}_{v_i}\}$, yielding $\mathbf{z}_{\text{input}}^{\text{node}} = [\mathbf{z}_{\mathcal{G}_{v_i}}, \mathbf{p}^{\text{node}}, \mathbf{r}^{y_i}]$.
- For link prediction on (v_i, v_j), we extract combined K-hop neighborhoods, excluding the target edge to prevent leakage: $\mathcal{V}_{(v_i,v_j)} = \{v_k : d(v_i, v_k) \le K \text{ or } d(v_j, v_k) \le K\}$ and $\mathcal{E}_{(v_i,v_j)} = \{(v_k, v_\ell) \in \mathcal{E} \setminus \{(v_i, v_j)\} : v_k, v_\ell \in \mathcal{V}_{(v_i,v_j)}\}$, yielding $\mathbf{z}_{\text{input}}^{\text{link}} = [\mathbf{z}_{\mathcal{G}_{(v_i,v_j)}}, \mathbf{p}^{\text{link}}, \mathbf{r}^{\text{exist/not-exist}}]$.
- For graph classification, $\mathcal{G}_{\text{full}} = \mathcal{G}$ and $\mathbf{z}_{\text{input}}^{\text{graph}} = [\mathbf{z}_{\mathcal{G}}, \mathbf{p}^{\text{graph}}, \mathbf{r}^{y_{\mathcal{G}}}]$.

Two-stage training paradigm. Our pipeline mirrors LLM pre-training and fine-tuning, maintaining consistent diffusion-based learning throughout.

Stage 1: Unsupervised pre-training (Figure 2①). Given $\{\mathcal{G}_1, \ldots, \mathcal{G}_K\}$, we construct edge-only sequences $\mathbf{z}_0 = \mathbf{z}_{\mathcal{G}}$ and optimize:

$$\mathcal{L}_{\text{pretrain}} = \mathbb{E}_{\mathcal{G},\, t\sim\mathcal{U}(1,T),\, \mathbf{z}_t\sim q(\mathbf{z}_t|\mathbf{z}_{\mathcal{G}})} \left[\sum\nolimits_{m=1}^{M} \|\mathbf{e}_m - f_\theta(\mathbf{z}_t, t)_m\|_2^2\right]. \tag{7}$$

Stage 2: Task-prompted fine-tuning (Figure 2②). We sample task $\mathcal{T}$, construct $\mathbf{z}_0 = \mathbf{z}_{\text{input}}^{\mathcal{T}}$ with ground-truth result, corrupt via forward diffusion, and optimize:

$$\mathcal{L}_{\text{finetune}} = \mathbb{E}_{\mathcal{T},\, \mathbf{z}_0^{\mathcal{T}},\, t\sim\mathcal{U}(1,T),\, \mathbf{z}_t\sim q(\mathbf{z}_t|\mathbf{z}_0^{\mathcal{T}})} \left[\left\|\mathbf{z}_0^{\mathcal{T}} - f_\theta(\mathbf{z}_t, t)\right\|_2^2\right]. \tag{8}$$

Both stages employ identical diffusion-based denoising over sequences, differing only in task conditioning, ensuring seamless transfer of pre-trained structural knowledge.

4.4 Task-Specific Inference via Selective Denoising

Our inference is analogous to image inpainting: we selectively denoise only the masked result embedding while keeping edge and prompt embeddings fixed (Figure 2③). Given $\mathcal{G}$ with $\mathbf{z}_{\mathcal{G}}$, prompt $\mathbf{p}^{\mathcal{T}}$, and $\mathbf{r}^{[\texttt{MASK}]} = \mathbf{r}_T$, we construct $\mathbf{z}_T = [\mathbf{z}_{\mathcal{G}}, \mathbf{p}^{\mathcal{T}}, \mathbf{r}^{[\texttt{MASK}]}]$ and predict in a single pass: $\hat{\mathbf{r}} = f_\theta(\mathbf{z}_T, T)_{M+2}$. The prediction follows $\hat{y} = \arg\max_{y \in \mathcal{Y}} \text{sim}(\hat{\mathbf{r}}, \mathbf{r}^y)$, where $\text{sim}(\cdot, \cdot)$ is cosine similarity.

5 Theoretical Analysis

We show that learning permutation-invariant joint distributions over edge configurations never yields a looser approximation to a permutation-invariant graph distribution than autoregressive factorizations over node sequences, and yields a strictly tighter one whenever the sequential model depends on the imposed ordering.

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be drawn from an unknown distribution $P^*(\mathcal{G})$ over graphs with N nodes. We consider: (i) *node sequential modeling*, factorizing $P_{\text{seq}}(\mathcal{G}) = \sum_{\boldsymbol{\pi}} P(\boldsymbol{\pi}) \prod_{i=1}^{N} P(v_{\pi_i} | v_{\pi_1}, \ldots, v_{\pi_{i-1}})$; and (ii) *edge co-occurrence modeling*, $P_{\text{edge}}(\mathcal{E}) = P(\{e_1, \ldots, e_M\})$. Let $\mathcal{F}_{\text{seq}}, \mathcal{F}_{\text{edge}}$ denote the respective achievable function classes, and define $\epsilon(\mathcal{F}) = \inf_{P \in \mathcal{F}} D_{\text{KL}}(P^* \| P)$. For $\sigma \in S_N$, the symmetric group of $N!$ permutations, $\sigma \cdot \mathcal{G}$ denotes $\mathcal{G}$ with node labels permuted by σ.

Theorem 5.1 (Edge Co-occurrence Optimality)**.** *Let P^* be any permutation-invariant distribution over graphs $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, i.e., $P^*(\sigma \cdot \mathcal{G}) = P^*(\mathcal{G})$ for all $\sigma \in S_N$. Suppose $\mathcal{F}_{edge}$ consists of permutation-invariant distributions and contains the symmetrization $\bar{P}(\mathcal{G}) = \frac{1}{N!} \sum_{\sigma \in S_N} P(\sigma \cdot \mathcal{G})$ of every $P \in \mathcal{F}_{seq}$, and let $P^*_{seq} \in \mathcal{F}_{seq}$ attain $\epsilon(\mathcal{F}_{seq}) < \infty$. Then $\epsilon(\mathcal{F}_{edge}) \leq \epsilon(\mathcal{F}_{seq}) - \Delta_{order}$, where the ordering gap $\Delta_{order} \geq 0$ satisfies*

$$\Delta_{order} = \mathbb{E}_{\mathcal{G} \sim P^*} \left[\log \frac{1}{N!} \sum_{\sigma \in S_N} P^*_{seq}(\sigma \cdot \mathcal{G}) - \frac{1}{N!} \sum_{\sigma \in S_N} \log P^*_{seq}(\sigma \cdot \mathcal{G}) \right]. \tag{9}$$

Furthermore, $\Delta_{order} = 0$ iff $P^*_{seq}(\sigma \cdot \mathcal{G})$ does not depend on σ for P^*-almost every $\mathcal{G}$, i.e., iff the sequential model is already invariant to the node ordering on the support of P^*.

Proof. Let $P = P^*_{\text{seq}}$ and $\bar{P}$ its symmetrization; $\bar{P}$ is permutation-invariant because S_N is a group. Since $D_{\text{KL}}(P^* \| P) < \infty$ and P^* is permutation-invariant, $P(\sigma \cdot \mathcal{G}) > 0$ for all σ on the support of P^*. The invariance of P^* also gives $\mathbb{E}_{P^*}[\log P(\sigma \cdot \mathcal{G})] = \mathbb{E}_{P^*}[\log P(\mathcal{G})]$ for every σ (substitute $\mathcal{G} \mapsto \sigma^{-1} \cdot \mathcal{G}$). Averaging this identity over σ,

$$D_{\text{KL}}(P^* \| P) - D_{\text{KL}}(P^* \| \bar{P}) = \mathbb{E}_{P^*}\big[\log \bar{P}(\mathcal{G}) - \log P(\mathcal{G})\big] = \Delta_{\text{order}}. \tag{10}$$

By Jensen's inequality for the concave logarithm, the logarithm of the average of $\{P(\sigma \cdot \mathcal{G})\}_\sigma$ is at least the average of their logarithms, so $\Delta_{\text{order}} \geq 0$; by strict concavity, equality holds iff these values coincide for P^*-almost every $\mathcal{G}$. Finally, $\bar{P} \in \mathcal{F}_{\text{edge}}$ by assumption, hence $\epsilon(\mathcal{F}_{\text{edge}}) \leq D_{\text{KL}}(P^* \| \bar{P}) = \epsilon(\mathcal{F}_{\text{seq}}) - \Delta_{\text{order}}$. □

The ordering gap Δ_{order} quantifies the price paid by node sequential models: they must either commit to a single ordering (losing permutation invariance) or average over orderings. Edge co-occurrence modeling sidesteps this dilemma by operating on the natural unordered representation: edge tokens are computed by a node-permutation-equivariant GNN, so relabeling nodes only permutes the edge tokens, and self-attention without positional encodings is equivariant to this permutation.

6 Experiments

Datasets. We evaluate on eight benchmarks across three tasks. *Node classification*: Cora, PubMed Sen et al. [2008], Wiki-CS Mernyei and Cangea [2020], ogbn-Arxiv Hu et al. [2020a]. *Link prediction*: WN18RR Dettmers et al. [2018], FB15K-237 Toutanova and Chen [2015]. *Graph classification*: ogbg-MolHIV, ogbg-MolPCBA Hu et al. [2020a]. Appendix B shows more details.

Unified data construction. Following OFA Liu et al. [2024], we encode each node's textual description with a frozen Llama-3.1-8B encoder Team [2024] into 4096-dimensional embeddings.

Table 1: Results across node classification (Acc. %), link prediction (MRR %), and graph classification (MolHIV: AUC %; MolPCBA: AP %). **Bold** = best; underline = runner-up. † denotes foundation methods.

Category	Method	Cora	PubMed	Wiki-CS	Arxiv	WN18RR	FB15K-237	MolHIV	MolPCBA
GNN	GCN	80.36±0.92	74.28±1.05	76.81±1.09	64.81±1.09	77.85±0.52	81.74±0.58	69.73±1.45	20.07±1.19
	GraphSAGE	81.74±0.83	75.92±0.94	77.65±1.02	64.65±1.02	80.76±0.42	88.63±0.09	57.82±1.67	18.57±0.61
	GAT	82.35±1.04	76.48±1.13	77.19±1.18	65.19±1.18	81.93±0.63	82.51±0.85	69.91±1.53	20.17±1.27
	GIN	79.28±1.17	72.86±1.26	74.35±1.29	63.35±1.29	79.62±0.67	84.72±1.24	60.05±1.58	22.20±0.78
Graph Trans.	Graphormer	83.45±1.48	80.45±0.17	79.05±1.50	68.70±0.20	84.78±0.56	85.51±0.74	**77.85±1.18**	29.74±0.92
	GraphGPS	82.80±0.78	79.58±0.85	78.65±0.95	66.40±1.05	85.83±0.50	87.24±0.43	76.12±0.94	27.56±0.89
Foundation	OFA†	71.24±0.83	77.36±0.67	68.92±0.91	72.38±0.78	86.92±0.55	88.05±0.50	75.38±1.24	22.34±1.58
	GraphMAE†	81.34±0.65	78.62±0.78	78.27±1.06	66.84±1.04	84.46±0.46	87.58±0.28	73.15±1.05	25.18±1.35
Ours	**Rel. Diffusion**	**88.71±1.21**	**81.65±0.89**	**83.56±0.97**	**74.64±1.32**	**89.42±0.53**	**89.67±0.69**	76.54±1.92	**30.63±1.64**

We extract $K = 2$-hop ego-nets centered on the query for node classification, and the union of $K = 2$-hop ego-nets around endpoints (with target edge masked) for link prediction. Graph classification uses full molecular graphs. We re-split every dataset uniformly at random into 60%/20%/20% train/validation/test partitions with a fixed seed. **All baselines are re-trained and evaluated under exactly the same splits, features, and tuning protocol**; absolute numbers are not directly comparable to prior work using official splits. Detailed statistics are in Appendix B. Results on the official ogbg-MolHIV scaffold split are reported in Appendix G.

Baselines. We compare against (1) classical GNNs: GCN Kipf and Welling [2017], GraphSAGE Hamilton et al. [2017], GAT Velickovic et al. [2018], GIN Xu et al. [2019]; (2) Graph Transformers: Graphormer Ying et al. [2021], GraphGPS Rampásek et al. [2022]; (3) Foundation models: OFA Liu et al. [2024], GraphMAE Hou et al. [2022].

Implementation. The diffusion Transformer has 4 layers, hidden dim 224, and 8 heads. Pre-training uses AdamW (lr 1×10^{-4}, 200 epochs); fine-tuning uses lr 5×10^{-5} for 100 epochs on $8\times$ A100 80GB. We report mean $\pm$ std over 5 seeds. Metrics: accuracy (node, graph), MRR (link), AUC (MolHIV), AP (MolPCBA). Appendix C shows more details.

6.1 Main Results

Table 1 consolidates results across all eight benchmarks spanning the three task categories.

Node classification. Relational diffusion achieves the strongest performance across all four datasets, with substantial gains: $+5.26\%$ on Cora, $+1.20\%$ on PubMed, $+4.51\%$ on Wiki-CS, and $+2.26\%$ on ogbn-Arxiv over the strongest baseline. Consistent gains from small (Cora) to large-scale (ogbn-Arxiv) graphs indicate edge co-occurrence modeling captures structural regularities at multiple scales. Notably, OFA dominates on ogbn-Arxiv but underperforms on smaller datasets, while Graphormer shows the opposite pattern; relational diffusion is the only method consistent across regimes.

Link prediction. Relational diffusion achieves the best test MRR on both knowledge-graph benchmarks: 89.42% on WN18RR ($+2.50\%$ over OFA) and 89.67% on FB15K-237 ($+1.04\%$ over GraphSAGE). The smaller gain on relation-heavy FB15K-237 suggests modeling relation types remains a complementary direction.

Graph classification. On multi-task PCBA benchmark, relational diffusion achieves the best AP (30.63%, $+0.89\%$ over Graphormer). On MolHIV, our method is runner-up (76.54% AUC), trailing only Graphormer (77.85%), whose spatial encoding is highly tailored to small, highly imbalanced molecular graphs.

Summary. Relational diffusion is the best-performing method on seven of eight benchmarks and runner-up on the eighth, supporting our hypothesis that edge co-occurrence modeling captures structural roles by reasoning over relational neighborhoods rather than arbitrary node orderings.

Table 2: Performance with different node features (%). Random and Degree + PageRank rows are point estimates; Llama-3.1-8B rows are the five-seed means from Table 1.

Node features	Method	Cora (Acc.)	WN18RR (MRR)	MolPCBA (AP)
Random	GAT	67.83	70.42	15.64
	Graphormer	70.91	73.25	21.13
	Rel. Diffusion	**73.62**	**76.43**	**22.74**
Degree + PageRank	GAT	72.14	74.63	17.82
	Graphormer	74.65	77.48	23.37
	Rel. Diffusion	**79.06**	**82.72**	**27.14**
Llama-3.1-8B	GAT	82.35	81.93	20.17
	Graphormer	83.45	84.78	29.74
	Rel. Diffusion	**88.71**	**89.42**	**30.63**

6.2 Ablation Studies

Component ablation. Figure 3 reveals several key insights. Replacing edge tokens with node tokens (Node-Seq) leads to an average drop of -2.73, suggesting that edge co-occurrence (E) provides more informative signals. Removing topological enrichment (T) results in substantial performance degradation, particularly on link prediction (-6.46) and graph classification (-5.43). Substituting diffusion (D) with a deterministic Transformer decreases performance by -3.80 on average, indicating that iterative refinement under noise yields more robust representations. Eliminating relational prompt completion (R) causes the largest drop on Cora (-9.31). Finally, removing pre-training (P) consistently harms performance across all tasks (average -7.47), with the most pronounced effect observed on link prediction.

Feature-source control. To assess whether the gains depend on the frozen language-model features, Table 2 compares GAT, Graphormer, and relational diffusion using random node embeddings, structural features (degree and PageRank), and Llama-3.1-8B features. Relational diffusion outperforms both baselines under all three feature sources on Cora, WN18RR, and ogbg-MolPCBA. With random features, the gains over Graphormer are $+2.71\%$, $+3.18\%$, and $+1.61\%$, respectively; with structural features, they are $+4.41\%$, $+5.24\%$, and $+3.77\%$. Thus, the advantage persists without pretrained language-model features, although these features improve the absolute scores of all three methods.

GNN encoder choice. Figure 4 shows GAT yields best performance across tasks. The largest gap to no-encoder is on link prediction ($+8.26$ MRR) and graph classification ($+7.43$ AP), confirming the value of learned multi-hop endpoint representations.

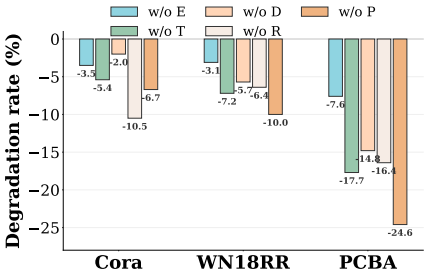

Figure 3: Relative degradation (%) when each component is removed.

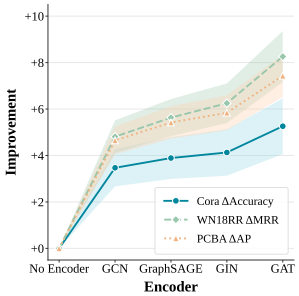

Figure 4: Gain over no-encoder vs. GNN encoder.

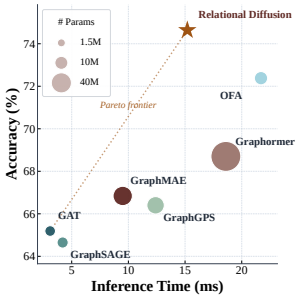

Figure 5: Accuracy vs. Inference time (size $\propto$ params).

6.3 Efficiency Analysis

We compare parameter count, latency, and accuracy on ogbn-Arxiv. As shown in Figure 5, our method dominates the Pareto frontier of accuracy versus inference time. Relational diffusion delivers the highest accuracy (74.64%) with just 2.5M trainable parameters (excluding the shared frozen text encoder), an order of magnitude smaller than Graphormer (47M). Latency is competitive at 15.2 ms per sample on a single A800, faster than OFA (21.7 ms) despite OFA being a markedly larger model.

The structural advantage of relational diffusion does not require model scale to materialize. We attribute most of the efficiency gain to two design choices examined in our ablations: (i) the edge-token sequence is asymptotically shorter than node-pair attention used in dense Graph Transformers, and (ii) the diffusion head amortizes refinement cost across only a handful of denoising steps rather than the deeper stacks typical of foundation baselines.

6.4 Permutation Equivariance Analysis

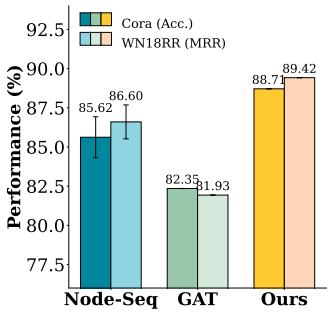

Figure 6: Prediction accuracy across random edge-ordering permutations.

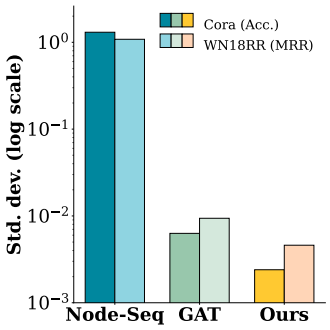

Figure 7: Standard deviation across random edge-ordering permutations.

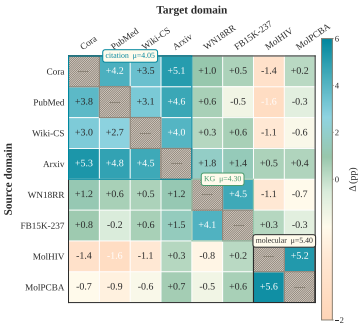

Figure 8: Pairwise cross-domain transfer matrix. Cell (i, j) reports gain (pp) on target when pre-training on source.

For each graph we generate 100 random edge-ordering permutations and measure prediction quality and standard deviation. Figure 6 and Figure 7 show the Node-Seq variant exhibits standard deviations more than two orders of magnitude larger than equivariant methods, while relational diffusion achieves both the highest mean accuracy and the tightest variance, empirically corroborating Theorem 5.1.

6.5 Pairwise Transferability Analysis

We compute the full 8×8 pairwise transfer matrix where cell (i, j) measures the test-accuracy gain on target $\mathcal{D}_j$ from Stage-1 pre-training on source $\mathcal{D}_i$ alone, with 5% target labels and 5 seeds per cell. Figure 8 visualizes the result. Three findings emerge:

(i) *Block-diagonal cluster structure*: intra-cluster transfer is strong, with citation networks averaging $+4.05$ pp, knowledge graphs $+4.30$ pp, molecular graphs $+5.40$ pp. (ii) *Negative transfer between distant clusters*: molecular $\leftrightarrow$ citation transfer yields -0.6 to -1.6 pp, attributable to encoder pull toward chemistry-specific feature regimes. (iii) *ogbn-Arxiv as a hub source*: it achieves the highest row mean ($+2.67$ pp) and is the only source with non-negative transfer to every target, including molecular benchmarks. This makes **ogbn-Arxiv a robust choice when pre-training on a single source**.

6.6 Few-Shot Transfer

We pre-train on the union of {ogbn-Arxiv, FB15K-237, ogbg-MolPCBA} and evaluate on Cora without any target-specific pre-training, varying the number of labeled examples per class in $\{1, 3, 5\}$. Table 3 reports test accuracy averaged over 5 seeds. Relational diffusion outperforms every baseline at all shot counts, with the largest absolute gains in the extreme low-shot regime: $+7.1$ pp at 1-shot over the strongest baseline (GraphMAE), narrowing to $+5.0$ pp at 5-shot.

Table 3: Few-shot transfer test accuracy (%) on Cora node classification.

Method	1-shot	3-shot	5-shot
GAT	24.6±2.4	35.2±2.1	44.8±1.8
Graphormer	28.3±2.2	39.5±1.9	49.6±1.6
OFA	38.4±1.8	47.8±1.5	55.2±1.3
GraphMAE	40.2±1.9	49.6±1.6	57.4±1.3
Ours	**47.3±1.5**	**55.8±1.2**	**62.4±1.0**

The pattern mirrors what one would expect from a model whose pre-training task (denoising relational prompts) induces transferable structural priors that remain useful before any task-specific signal can dominate. Compared to OFA and GraphMAE, which rely on either textual alignment or feature-level masking, the edge-co-occurrence objective produces representations that retain discriminative power even when only a single example per class is available.

7 Conclusion and Limitations

We introduced relational diffusion, a graph foundation model that shifts the modeling primitive from node sequences to edge co-occurrence. By combining topological edge tokenization with a diffusion-based denoising process over edge configurations, our framework preserves permutation equivariance and captures relational structure without imposing artificial orderings. Experiments across node classification, link prediction, and graph classification demonstrate that this edge-centric paradigm achieves strong performance on most benchmarks while maintaining efficiency.

Our current framework treats all edges as homogeneous tokens, which limits its expressiveness on multi-relational graphs where distinct edge types carry different semantics, as suggested by the relatively smaller gains on FB15K-237. Additionally, the quadratic self-attention cost over edge tokens may pose scalability challenges for very dense graphs, motivating future work on sparse or hierarchical attention mechanisms, or linear-complexity selective state-space models Liu et al. [2026], for edge-level diffusion.

Acknowledgments and Disclosure of Funding

This research was supported by the National Key R&D Program of China (No. 2023YFC3304701), the National Natural Science Foundation of China (No. 62506366), the CCF-Tencent Rhino-Bird Open Research Fund, the Fundamental Research Funds for the Central Universities, and the Research Funds of Renmin University of China. It was also supported by Big Data and Responsible Artificial Intelligence for National Governance, Renmin University of China, and by the Public Computing Cloud, Renmin University of China.

References

J. Austin, D. D. Johnson, J. Ho, D. Tarlow, and R. van den Berg. Structured denoising diffusion models in discrete state-spaces. In M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 17981–17993, 2021.

A. Campbell, J. Benton, V. D. Bortoli, T. Rainforth, G. Deligiannidis, and A. Doucet. A continuous time framework for discrete denoising models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *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.

A. Campbell, J. Yim, R. Barzilay, T. Rainforth, and T. S. Jaakkola. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design. In R. Salakhutdinov, Z. Kolter, K. A. Heller, A. Weller, N. Oliver, J. Scarlett, and F. Berkenkamp, editors, *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pages 5453–5512. PMLR / OpenReview.net, 2024.

C. Deng, Z. Yue, and Z. Zhang. Polynormer: Polynomial-expressive graph transformer in linear time. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel. Convolutional 2D knowledge graph embeddings. In S. A. McIlraith and K. Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 1811–1818. AAAI Press, 2018.

V. P. Dwivedi, A. T. Luu, T. Laurent, Y. Bengio, and X. Bresson. Graph neural networks with learnable structural and positional representations. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

T. Fang, Y. Zhang, Y. Yang, C. Wang, and L. Chen. Universal prompt tuning for graph neural networks. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *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.

J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Neural message passing for quantum chemistry. In D. Precup and Y. W. Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1263–1272. PMLR, 2017.

S. Gong, M. Li, J. Feng, Z. Wu, and L. Kong. DiffuSeq: sequence to sequence text generation with diffusion models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

W. L. Hamilton, Z. Ying, and J. Leskovec. Inductive representation learning on large graphs. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 1024–1034, 2017.

X. Han, S. Kumar, and Y. Tsvetkov. SSD-LM: semi-autoregressive simplex-based diffusion language model for text generation and modular control. In A. Rogers, J. L. Boyd-Graber, and N. Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 11575–11596. Association for Computational Linguistics, 2023.

Z. He, T. Sun, Q. Tang, K. Wang, X. Huang, and X. Qiu. DiffusionBERT: improving generative masked language models with diffusion models. In A. Rogers, J. L. Boyd-Graber, and N. Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 4521–4534. Association for Computational Linguistics, 2023.

J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.-F. Balcan, and H.-T. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

Z. Hou, X. Liu, Y. Cen, Y. Dong, H. Yang, C. Wang, and J. Tang. GraphMAE: self-supervised masked graph autoencoders. In A. Zhang and H. Rangwala, editors, *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*, pages 594–604. ACM, 2022.

Z. Hou, Y. He, Y. Cen, X. Liu, Y. Dong, E. Kharlamov, and J. Tang. GraphMAE2: A decoding-enhanced masked self-supervised graph learner. In Y. Ding, J. Tang, J. F. Sequeda, L. Aroyo, C. Castillo, and G.-J. Houben, editors, *Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023*, pages 737–746. ACM, 2023.

W. Hu, M. Fey, M. Zitnik, Y. Dong, H. Ren, B. Liu, M. Catasta, and J. Leskovec. Open graph benchmark: Datasets for machine learning on graphs. In H. Larochelle, M. Ranzato, R. Hadsell, M.-F. Balcan, and H.-T. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020a.

W. Hu, X. Wang, C. Wei, S. Liu, K. Liu, Y. Wang, and Y. Chen. Test-time search for automated GFM fine-tuning. In W. Kim, J.-G. Lee, K. Shim, J. Caverlee, D. F. Gleich, X. Zhang, Q. He, F. Farooq, R. Rosales, S. Ranu, H. Wang, Y. Liu, C. Yang, L. Akoglu, N. V. Chawla, and V. Papalexakis, editors, *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2, KDD 2026, Jeju Island, Korea, August 9-13, 2026*, pages 1804–1815. ACM, 2026.

Z. Hu, Y. Dong, K. Wang, K.-W. Chang, and Y. Sun. GPT-GNN: generative pre-training of graph neural networks. In R. Gupta, Y. Liu, J. Tang, and B. A. Prakash, editors, *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 1857–1867. ACM, 2020b.

M. Jin, H. Y. Koh, Q. Wen, D. Zambon, C. Alippi, G. I. Webb, I. King, and S. Pan. A survey on graph neural networks for time series: Forecasting, classification, imputation, and anomaly detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 46(12):10466–10485, 2024.

T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

Q. Li, Z. Han, and X.-M. Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In S. A. McIlraith and K. Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 3538–3545. AAAI Press, 2018.

X. L. Li, J. Thickstun, I. Gulrajani, P. Liang, and T. B. Hashimoto. Diffusion-LM improves controllable text generation. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *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.

H. Liu, J. Feng, L. Kong, N. Liang, D. Tao, Y. Chen, and M. Zhang. One for all: Towards training one graph model for all classification tasks. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

J. Liu, C. Yang, Z. Lu, J. Chen, Y. Li, M. Zhang, T. Bai, Y. Fang, L. Sun, P. S. Yu, and C. Shi. Towards graph foundation models: A survey and beyond. *CoRR*, abs/2310.11829, 2023a.

Y. Liu, C. Wei, H. Lin, F. Xiao, J. Zhang, Y. Wang, and Y. Chen. Ramba: Selective state-space models for relational deep learning. In *Forty-third International Conference on Machine Learning*, 2026.

Z. Liu, X. Yu, Y. Fang, and X. Zhang. GraphPrompt: unifying pre-training and downstream tasks for graph neural networks. In Y. Ding, J. Tang, J. F. Sequeda, L. Aroyo, C. Castillo, and G.-J. Houben, editors, *Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023*, pages 417–428. ACM, 2023b.

A. Lou, C. Meng, and S. Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In R. Salakhutdinov, Z. Kolter, K. A. Heller, A. Weller, N. Oliver, J. Scarlett, and F. Berkenkamp, editors, *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pages 32819–32848. PMLR / OpenReview.net, 2024.

P. Mernyei and C. Cangea. Wiki-CS: A wikipedia-based benchmark for graph neural networks. *CoRR*, abs/2007.02901, 2020.

S. Nie, F. Zhu, Z. You, X. Zhang, J. Ou, J. Hu, J. Zhou, Y. Lin, J.-R. Wen, and C. Li. Large language diffusion models. In D. Belgrave, C. Zhang, L. N. Montoya, H.-T. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, N. Chen, I. V. M. Ruíz, and A. Loaiza-Bonilla, editors, *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.

C. Niu, Y. Song, J. Song, S. Zhao, A. Grover, and S. Ermon. Permutation invariant graph generation via score-based generative modeling. In S. Chiappa and R. Calandra, editors, *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*, volume 108 of *Proceedings of Machine Learning Research*, pages 4474–4484. PMLR, 2020.

J. Qiu, Q. Chen, Y. Dong, J. Zhang, H. Yang, M. Ding, K. Wang, and J. Tang. GCC: graph contrastive coding for graph neural network pre-training. In R. Gupta, Y. Liu, J. Tang, and B. A. Prakash, editors, *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 1150–1160. ACM, 2020.

L. Rampásek, M. Galkin, V. P. Dwivedi, A. T. Luu, G. Wolf, and D. Beaini. Recipe for a general, powerful, scalable graph transformer. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *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.

Y. Rong, Y. Bian, T. Xu, W. Xie, Y. Wei, W. Huang, and J. Huang. Self-supervised graph transformer on large-scale molecular data. In H. Larochelle, M. Ranzato, R. Hadsell, M.-F. Balcan, and H.-T. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

S. S. Sahoo, M. Arriola, Y. Schiff, A. Gokaslan, E. Marroquin, J. T. Chiu, A. M. Rush, and V. Kuleshov. Simple and effective masked diffusion language models. In A. Globersons, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. M. Tomczak, and C. Zhang, editors, *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.

P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad. Collective classification in network data. *AI Mag.*, 29(3):93–106, 2008.

H. Shirzad, A. Velingker, B. Venkatachalam, D. J. Sutherland, and A. K. Sinop. Exphormer: Sparse transformers for graphs. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 31613–31632. PMLR, 2023.

Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

H. Sun, L. Yu, B. Dai, D. Schuurmans, and H. Dai. Score-based continuous-time discrete diffusion models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023a.

M. Sun, K. Zhou, X. He, Y. Wang, and X. Wang. GPPT: graph pre-training and prompt tuning to generalize graph neural networks. In A. Zhang and H. Rangwala, editors, *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*, pages 1717–1727. ACM, 2022.

X. Sun, H. Cheng, J. Li, B. Liu, and J. Guan. All in one: Multi-task prompting for graph neural networks. In A. K. Singh, Y. Sun, L. Akoglu, D. Gunopulos, X. Yan, R. Kumar, F. Ozcan, and J. Ye, editors, *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023, Long Beach, CA, USA, August 6-10, 2023*, pages 2120–2131. ACM, 2023b.

L. Team. The llama 3 herd of models. *CoRR*, abs/2407.21783, 2024.

K. Toutanova and D. Chen. Observed versus latent features for knowledge base and text inference. In A. Allauzen, E. Grefenstette, K. M. Hermann, H. Larochelle, and S. W. tau Yih, editors, *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality, CVSC 2015, Beijing, China, July 26-31, 2015*, pages 57–66. Association for Computational Linguistics, 2015.

L. van der Maaten and G. Hinton. Visualizing data using t-SNE. *J. Mach. Learn. Res.*, 9(86): 2579–2605, 2008.

P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. 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.

C. Vignac, I. Krawczuk, A. Siraudin, B. Wang, V. Cevher, and P. Frossard. DiGress: discrete denoising diffusion for graph generation. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

S. Wang, J. Huang, Z. Chen, Y. Song, W. Tang, H. Mao, W. Fan, H. Liu, X. Liu, D. Yin, and Q. Li. Graph machine learning in the era of large language models (LLMs). *ACM Trans. Intell. Syst. Technol.*, 16(5):104:1–104:40, 2025.

C. Wei, Y. Wang, B. Bai, K. Ni, D. Brady, and L. Fang. Boosting graph contrastive learning via graph contrastive saliency. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 36839–36855. PMLR, 2023.

C. Wei, W. Hu, X. Hao, X. Wang, Y. Yang, Y. Wang, Y. Tian, and Y. Chen. GraphChain: large language models for large-scale graph analysis via tool chaining. In D. Belgrave, C. Zhang, L. N. Montoya, H.-T. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, N. Chen, I. V. M. Ruíz, and A. Loaiza-Bonilla, editors, *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*, 2025a.

C. Wei, H. Lin, Y. Chen, and Y. Wang. Conditional diffusion anomaly modeling on graphs. In D. Belgrave, C. Zhang, L. N. Montoya, H.-T. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, N. Chen, I. V. M. Ruíz, and A. Loaiza-Bonilla, editors, *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*, 2025b.

C. Wei, S. He, Y. Wang, Y. Chen, Y. Wang, B. Bai, Y. Zhang, Y. Xie, S. Zhang, and F. Wang. Balanced anomaly-guided ego-graph diffusion model for inductive graph anomaly detection. In S. Parthasarathy, D. F. Gleich, X. Zhang, W. H. Tok, F. Farooq, Q. He, A. K. Singh, H. Wang, and Y. Liu, editors, *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1, KDD 2026, Jeju Island, Korea, August 9-13, 2026*, pages 1542–1553. ACM, 2026a.

C. Wei, H. Qin, S. He, Y. Wang, and Y. Chen. T-retriever: Tree-based hierarchical retrieval augmented generation for textual graphs. In S. Koenig, C. Jenkins, and M. E. Taylor, editors, *Fortieth AAAI Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pages 15924–15932. AAAI Press, 2026b.

J. Xia, L. Wu, J. Chen, B. Hu, and S. Z. Li. SimGRACE: A simple framework for graph contrastive learning without data augmentation. In F. Laforest, R. Troncy, E. Simperl, D. Agarwal, A. Gionis, I. Herman, and L. Médini, editors, *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, pages 1070–1079. ACM, 2022.

K. Xu, W. Hu, J. Leskovec, and S. 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.

C. Ying, T. Cai, S. Luo, S. Zheng, G. Ke, D. He, Y. Shen, and T.-Y. Liu. Do transformers really perform badly for graph representation? In M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 28877–28888, 2021.

Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, and Y. Shen. Graph contrastive learning with augmentations. In H. Larochelle, M. Ranzato, R. Hadsell, M.-F. Balcan, and H.-T. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

X. Zhang, X. Deng, H. Yuan, C. Wei, Y. Fan, and J. Zhang. Graph-based diffusion model for service recommendation. *IEEE Trans. Serv. Comput.*, 19(1):396–409, 2026.

H. Zhao, A. Chen, X. Sun, H. Cheng, and J. Li. All in one and one for all: A simple yet effective method towards cross-domain graph pretraining. In R. Baeza-Yates and F. Bonchi, editors, *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2024, Barcelona, Spain, August 25-29, 2024*, pages 4443–4454. ACM, 2024.

Q. Zhao, W. Ren, T. Li, H. Liu, X. He, and X. Xu. GraphGPT: generative pre-trained graph eulerian transformer. In A. Singh, M. Fazel, D. Hsu, S. Lacoste-Julien, F. Berkenkamp, T. Maharaj, K. Wagstaff, and J. Zhu, editors, *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025.

H. Zheng, W. Nie, A. Vahdat, and A. Anandkumar. Fast training of diffusion models with masked transformers. *Trans. Mach. Learn. Res.*, 2024, 2024.

Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang. Deep graph contrastive representation learning. *CoRR*, abs/2006.04131, 2020.

A Extended Theoretical Analysis

We provide extended theoretical results supporting our relational diffusion framework, including detailed proofs, concrete examples quantifying the ordering gap, and additional theoretical guarantees.

A.1 Detailed Proof of Theorem 5.1

We expand on the proof in the main text with complete technical details.

Lemma A.1 (Invariance of the Expected Log-Likelihood)**.** *Let P^* be permutation-invariant and let P satisfy $D_{KL}(P^*\|P) < \infty$. Then $P(\sigma \cdot \mathcal{G}) > 0$ for every $\sigma \in S_N$ and P^*-almost every $\mathcal{G}$, and $\mathbb{E}_{P^*}[\log P(\sigma \cdot \mathcal{G})] = \mathbb{E}_{P^*}[\log P(\mathcal{G})]$. Consequently, $D_{KL}(P^*\|P(\sigma\cdot)) = D_{KL}(P^*\|P)$ for every σ.*

Proof. Finite KL divergence implies $P(\mathcal{G}) > 0$ whenever $P^*(\mathcal{G}) > 0$. Since $P^*(\sigma \cdot \mathcal{G}) = P^*(\mathcal{G})$, the support of P^* is closed under relabeling, which gives positivity of $P(\sigma \cdot \mathcal{G})$. For the identity, $\mathbb{E}_{P^*}[\log P(\sigma \cdot \mathcal{G})] = \sum_{\mathcal{G}} P^*(\mathcal{G}) \log P(\sigma \cdot \mathcal{G}) = \sum_{\mathcal{G}'} P^*(\sigma^{-1} \cdot \mathcal{G}') \log P(\mathcal{G}') = \mathbb{E}_{P^*}[\log P(\mathcal{G}')]$, using the substitution $\mathcal{G}' = \sigma \cdot \mathcal{G}$ and invariance of P^*. □

Lemma A.2 (Symmetrization Gap)**.** *Under the conditions of Lemma A.1, let $\bar{P}(\mathcal{G}) = \frac{1}{N!}\sum_\sigma P(\sigma\cdot\mathcal{G})$. Then*

$$D_{KL}(P^*\|\bar{P}) = D_{KL}(P^*\|P) - \Delta_{order}(P), \tag{11}$$

$$\Delta_{order}(P) = \mathbb{E}_{P^*}\Big[\log \tfrac{1}{N!}\textstyle\sum_\sigma P(\sigma\cdot\mathcal{G}) - \tfrac{1}{N!}\sum_\sigma \log P(\sigma\cdot\mathcal{G})\Big] \geq 0, \tag{12}$$

with equality iff $P(\sigma \cdot \mathcal{G})$ is constant in σ for P^*-almost every $\mathcal{G}$.

Proof. The terms $\mathbb{E}_{P^*}[\log P^*]$ cancel, so $D_{\mathrm{KL}}(P^*\|P) - D_{\mathrm{KL}}(P^*\|\bar{P}) = \mathbb{E}_{P^*}[\log \bar{P}(\mathcal{G})] - \mathbb{E}_{P^*}[\log P(\mathcal{G})]$. By Lemma A.1, the second expectation equals $\mathbb{E}_{P^*}[\frac{1}{N!}\sum_\sigma \log P(\sigma\cdot\mathcal{G})]$, which yields the identity. Non-negativity and the equality condition follow pointwise from Jensen's inequality for the strictly concave logarithm. □

Proposition A.3 (Complete Proof of Theorem 5.1)**.** *Apply Lemma A.2 to $P = P^*_{seq}$, for which $D_{KL}(P^*\|P^*_{seq}) = \epsilon(\mathcal{F}_{seq}) < \infty$. The symmetrization $\bar{P}^*_{seq}$ is permutation-invariant and belongs to $\mathcal{F}_{edge}$ by assumption, so $\epsilon(\mathcal{F}_{edge}) \leq D_{KL}(P^*\|\bar{P}^*_{seq}) = \epsilon(\mathcal{F}_{seq}) - \Delta_{order}(P^*_{seq})$, with $\Delta_{order}(P^*_{seq}) \geq 0$ and the stated equality condition.*

Remark. *The ordering gap is positive exactly when the fitted sequential model assigns different probabilities to relabelings of the same graph, e.g., when it is trained with a fixed or data-dependent traversal and its finite-capacity conditionals fit some orderings better than others. A sequential model that is exactly invariant (e.g., an exact chain-rule factorization of P^*) has $\Delta_{order} = 0$; the theorem does not claim that every autoregressive model is strictly worse.*

A.2 Concrete Example: Quantifying the Ordering Gap

We provide a concrete example demonstrating the ordering gap for a simple graph family.

Example 1 (Triangle Graphs)**.** *Consider the distribution P^* over 3-node graphs where the graph is either a complete triangle (all 3 edges present) or empty (no edges), each with probability $\frac{1}{2}$.*

True distribution:

$$P^*(\mathcal{E}) = \frac{1}{2}\mathbf{1}[|\mathcal{E}| = 3] + \frac{1}{2}\mathbf{1}[|\mathcal{E}| = 0] \tag{13}$$

Sequential model analysis: *For any ordering $\boldsymbol{\pi} = (\pi_1, \pi_2, \pi_3)$, the autoregressive factorization is:*

$$P_{seq}(\mathcal{E}|\boldsymbol{\pi}) = P(e_{\pi_1\pi_2}) \cdot P(e_{\pi_1\pi_3}|e_{\pi_1\pi_2}) \cdot P(e_{\pi_2\pi_3}|e_{\pi_1\pi_2}, e_{\pi_1\pi_3}) \tag{14}$$

To capture the true distribution, we need:

- $P(e_{\pi_1\pi_2} = 1) = \frac{1}{2}$
- $P(e_{\pi_1\pi_3} = 1|e_{\pi_1\pi_2} = 1) = 1$, $P(e_{\pi_1\pi_3} = 1|e_{\pi_1\pi_2} = 0) = 0$

- $P(e_{\pi_2\pi_3} = 1 | e_{\pi_1\pi_2} = 1, e_{\pi_1\pi_3} = 1) = 1$

A perfect sequential model can represent this distribution for any fixed ordering.

Edge co-occurrence model: *Directly models:*

$$P_{edge}(e_{12}, e_{13}, e_{23}) = \frac{1}{2}\mathbf{1}[e_{12} = e_{13} = e_{23} = 1] + \frac{1}{2}\mathbf{1}[e_{12} = e_{13} = e_{23} = 0] \tag{15}$$

This is naturally permutation-invariant and requires no ordering.

Gap quantification: *For this simple example, both models can achieve zero error with sufficient capacity. An exact sequential model assigns the same probability to every relabeling of a graph, so $\Delta_{order} = 0$ by Theorem 5.1; a positive gap arises only when an approximate sequential model fits the conditionals differently for different orderings.*

Example 2 (Motif Co-occurrence)**.** *Consider graphs where edges form one of K possible motifs $\{M_1, \ldots, M_K\}$, with $P^*(M_k) = \pi_k$. Each motif M_k specifies a subset of edges that must all be present.*

The edge co-occurrence model naturally captures motif structure through joint edge distributions. The sequential model must decompose motif membership into sequential decisions, potentially losing the holistic motif structure.

If a finite-capacity sequential model captures a motif more accurately under some orderings than others, then its probabilities differ across relabelings of the same graph and Theorem 5.1 gives $\Delta_{order} > 0$. A permutation-invariant edge model is not subject to this ordering gap.

A.3 Expressiveness of Edge Co-occurrence Diffusion

We analyze the expressiveness of our diffusion-based edge co-occurrence model.

We do not claim a universal approximation guarantee for the complete model. Such a result, i.e., $D_{\mathrm{KL}}(P^* \| p_\theta) < \epsilon$ for every permutation-invariant P^* with bounded edge count and every $\epsilon > 0$, would require establishing three conditions that are not proven here:

1. the Transformer denoiser class is dense in the relevant class of permutation-equivariant functions over edge-token sets;
2. the diffusion model with sufficiently many timesteps T and an accurate denoiser approximates the target distribution in KL divergence;
3. the edge tokenization is injective on graph structures, so that the distribution over tokens determines the distribution over graphs.

A.4 Computational Complexity Analysis

We compare the computational complexity of sequential versus edge co-occurrence approaches.

Proposition A.4 (Complexity Comparison)**.** *For a graph with N nodes and M edges:*

- *Node sequential generation: $O(N^2 \cdot C_{AR})$ for autoregressive steps, where C_{AR} is the per-step complexity.*
- *Edge co-occurrence diffusion (single-step inference): $O(M^2 \cdot d)$ for self-attention over M edge tokens with dimension d.*

For sparse graphs where $M = O(N)$, edge co-occurrence achieves $O(N^2 d)$ versus $O(N^2 C_{AR})$ for sequential methods. The advantage becomes more pronounced when $C_{AR} \gg d$ due to complex autoregressive dependencies.

A.5 Connection to Graph Isomorphism

We establish connections between our edge co-occurrence framework and graph isomorphism testing.

Proposition A.5 (WL-Expressiveness)**.** *The node representations produced by the L-layer message-passing GNN of our topological edge tokenization are at most as expressive as the L-iteration*

Weisfeiler-Lehman (1-WL) test: if L-WL assigns the same colors to two graphs with identical initial node features, the GNN produces the same multisets of node and endpoint-pair embeddings.

Proof. Each GNN layer updates a node from its own representation and the multiset of its neighbors' representations, so by induction on L, nodes with identical L-WL colors receive identical representations Xu et al. [2019]. Edge embeddings combine endpoint representations and therefore inherit this upper bound. Matching 1-WL requires injective aggregation and update functions Xu et al. [2019], which the attention-weighted aggregation of GAT does not guarantee. □

The structural features $\mathbf{s}_m$ and random-walk encodings can distinguish some graphs that 1-WL cannot (e.g., through triangle-dependent clustering coefficients), but we do not claim a general lower bound on the expressiveness of the complete tokenizer.

B Dataset Details

This appendix provides full details of our evaluation corpus: raw dataset sources and statistics, the OFA-style text-to-embedding pipeline, subgraph extraction procedures, the unified 60/20/20 splitting protocol, and a visualization of the resulting embedding space. All baselines are re-trained and evaluated under identical conditions.

B.1 Dataset Statistics

Table 4 summarizes the raw statistics for all eight benchmarks spanning three task categories. For graph classification datasets, node and edge counts represent per-graph averages over the corpus.

Table 4: **Dataset statistics.** For graph classification, node/edge counts are per-graph averages. #Cls denotes the number of classes; binary tasks are marked as 2.

Task	Dataset	#Graphs	#Nodes	#Edges	#Cls	Domain
Node Cls.	Cora	1	2,708	5,429	7	Citation
	PubMed	1	19,717	44,338	3	Citation
	Wiki-CS	1	11,701	216,123	10	Wikipedia
	ogbn-Arxiv	1	169,343	1,166,243	40	Citation
Link Pred.	WN18RR	1	40,943	93,003	–	Lexical KG
	FB15K-237	1	14,541	310,116	–	Knowledge Graph
Graph Cls.	ogbg-MolHIV	41,127	25.5 (avg)	27.5 (avg)	2	Molecular
	ogbg-MolPCBA	437,929	26.0 (avg)	28.1 (avg)	128	Molecular

B.2 Dataset Sources and Textual Descriptions

To unify heterogeneous node-feature modalities into a shared input space, each node v is associated with a natural-language description text(v) that is fed into the OFA-style text encoder described in Section B.3. Below we detail the description construction for each dataset.

Node Classification Datasets. **Cora & PubMed** [Sen et al., 2008]. Citation networks where nodes are scientific papers and edges are citations. Each node's textual description consists of the paper *title* concatenated with its *abstract*. Labels encode research topic categories (7 classes for Cora, 3 for PubMed).

Wiki-CS [Mernyei and Cangea, 2020]. A Wikipedia graph where nodes are computer-science articles and edges are hyperlinks. We use the article title concatenated with the *first paragraph* of the article as the node description. Labels encode 10 subject areas.

ogbn-Arxiv [Hu et al., 2020a]. A large-scale arXiv CS citation network. Each node carries the paper *title and abstract*. Labels are 40 fine-grained CS subject areas (MAG taxonomy).

Link Prediction Datasets. **WN18RR** [Dettmers et al., 2018]. A WordNet-based knowledge graph with inverse-relation leakage removed. Each node corresponds to a WordNet synset; we use the synset *gloss* as the textual description, e.g.,

"dog.n.01: a member of the genus Canis (probably descended from the common wolf) that has been domesticated by man since prehistoric times."

FB15K-237 [Toutanova and Chen, 2015]. A Freebase entity graph with redundant inverse relations removed. Each entity carries a *name* and a textual description harvested from Freebase metadata; we use their concatenation as the node text.

Graph Classification Datasets. **ogbg-MolHIV & ogbg-MolPCBA** [Hu et al., 2020a]. Molecular property-prediction benchmarks. Each graph is a molecule with atoms as nodes and bonds as edges. Each atom's textual description is constructed from its chemical attributes: element symbol, formal charge, hybridization, hydrogen count, and aromaticity. For example:

"aromatic carbon, sp2 hybridization, with one hydrogen, zero formal charge."

This converts the discrete OGB atom-feature vector into a natural-language sentence that maps into the same unified embedding space as text-attributed datasets, enabling true cross-domain pre-training.

B.3 OFA-Style Text-to-Embedding Pipeline

Following the text-attributed graph paradigm of OFA [Liu et al., 2024], we use a large language model as a frozen feature extractor to embed all node descriptions into a shared 4096-dimensional space.

$$\mathbf{x}_v \;=\; \mathrm{LM}_{\mathrm{enc}}\big(\mathrm{text}(v)\big) \;\in\; \mathbb{R}^{d_v}, \qquad d_v = 4096. \tag{16}$$

We use `Llama-3.1-8B` [Team, 2024] as the encoder: we extract *final-layer* hidden states, *mean-pool* over all input tokens, and *L2-normalize* the resulting vector. LM weights are **frozen** throughout pre-training and fine-tuning — only the GAT encoder, diffusion Transformer, and prompt/result embeddings are updated. Embeddings are pre-computed and cached per dataset; peak GPU memory during embedding computation is approximately 32 GB.

Table 5: **Pipeline configuration.** Summary of the OFA-style embedding procedure applied uniformly to all eight datasets.

Component	Configuration
LM backbone	Llama-3.1-8B (frozen)
Pooling	Mean pooling over token hidden states
Normalization	L2 normalization
Output dimension	4096
Computation	Pre-computed & cached per dataset
Peak memory	$\approx$32 GB during encoding

B.4 Visualization of the Unified Embedding Space

To illustrate the structure of the shared representation space, we apply t-SNE [van der Maaten and Hinton, 2008] to a random subsample of node embeddings from all eight datasets. Figure 9 shows the two-dimensional projection (perplexity 30, 1000 iterations), colored by source dataset.

The clear separation between molecular and citation domains explains the *negative transfer* (-0.6 to -1.6 pp) observed when pre-training on molecular graphs and evaluating on citation networks, and vice versa. The adjacency between Cora, PubMed, and ogbn-Arxiv explains why ogbn-Arxiv is the strongest universal source domain (row mean $+2.67$ pp), as it lies near the centroid of the academic-text manifold while covering the largest and most diverse set of nodes.

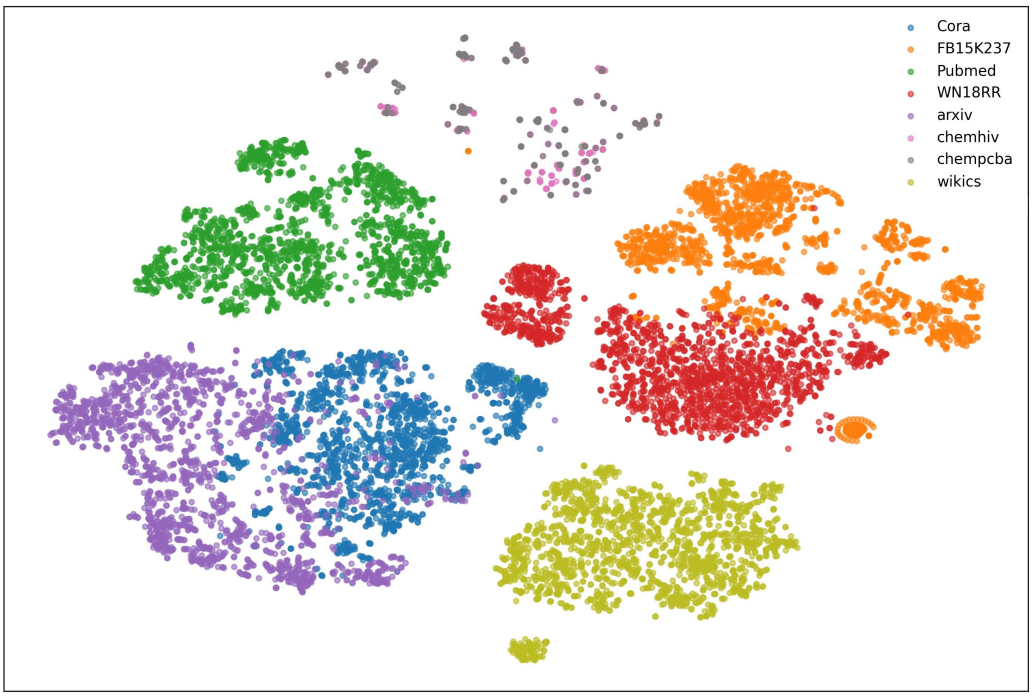

Figure 9: **t-SNE projection of Llama-3.1-8B node embeddings across all eight datasets.** Each point is a node colored by its source dataset. The embedding space exhibits two key properties. **(1) Domain separation:** each dataset occupies a distinguishable region, confirming that domain identity is preserved in the LM-induced space. **(2) Semantic adjacency:** citation networks (Cora, PubMed, ogbn-Arxiv) cluster together; lexical and entity-based knowledge graphs (WN18RR, FB15K-237) form a neighboring group; molecular graphs (ogbg-MolHIV, ogbg-MolPCBA) form an isolated cluster reflecting their distinct chemical-text vocabulary; Wiki-CS occupies its own encyclopedic-text region. This organized structure helps explain the block-diagonal transfer behavior observed in Section 6.5.

B.5 Subgraph Extraction

Each task requires extracting a task-specific subgraph $\mathcal{G}_q$ around a query q. We use $K = 2$ hops uniformly across all node and link prediction tasks.

Node classification. For query node v_i:

$$\mathcal{V}_{v_i} = \{v_j : d(v_i, v_j) \leq 2\},$$
$$\mathcal{E}_{v_i} = \{(v_j, v_k) \in \mathcal{E} : v_j, v_k \in \mathcal{V}_{v_i}\}.$$

For high-degree nodes in ogbn-Arxiv, per-hop neighbor count is capped at 25 (following GraphSAGE [Hamilton et al., 2017]) to limit subgraph size.

Link prediction. For query pair (v_i, v_j), we extract the union of 2-hop ego-networks and **remove the target edge** to prevent leakage:

$$\mathcal{V}_{(v_i,v_j)} = \{v_k : d(v_i, v_k) \leq 2 \text{ or } d(v_j, v_k) \leq 2\},$$
$$\mathcal{E}_{(v_i,v_j)} = \{(v_k, v_\ell) \in \mathcal{E} \setminus \{(v_i, v_j)\} : v_k, v_\ell \in \mathcal{V}_{(v_i,v_j)}\}.$$

Graph classification. The full molecular graph $\mathcal{G}$ is used directly. No subgraph sampling is applied, since molecular graphs are small (median $\approx$25 atoms) and global topology is essential for property prediction.

B.6 Train/Validation/Test Splits

Protocol. Every dataset is split *uniformly at random* into 60 / 20 / 20 train / validation / test partitions with a fixed seed of 42. Split granularity differs by task:

- **Node classification (transductive).** Split over *nodes*; the full graph structure is shared across all splits, and loss is computed only on nodes assigned to the current split.
- **Link prediction.** Split over *positive edges*. For each validation/test edge, 50 negatives are sampled by corrupting one endpoint; MRR is computed over the resulting 51-candidate ranking. Validation/test edges are masked from message-passing to prevent leakage.
- **Graph classification.** Split over *graphs*.

Departure from official splits and motivation. We deliberately depart from the official OGB and Planetoid splits for three reasons:

1. **Sufficient training data.** Diffusion-based learning requires enough examples to capture edge co-occurrence patterns; the 5% training fraction in Planetoid splits (e.g., 140 nodes for Cora) is too sparse.
2. **Uniform protocol.** A single splitting rule eliminates dataset-specific bookkeeping across eight benchmarks spanning very different scales.
3. **Fair internal comparison.** Every baseline observes the same training/validation/test data, enabling clean apples-to-apples comparisons.

△ **Disclosure.** The absolute numbers reported in this paper are **not directly comparable** to results from prior work using official splits — in particular, our ogbn-Arxiv, ogbg-MolHIV, and ogbg-MolPCBA numbers cannot be compared to OGB-leaderboard entries, since the OGB scaffold/temporal splits are deliberately harder than random 60/20/20 splits. *All comparisons in this paper are internally consistent*: every baseline is re-trained and evaluated under identical splits, features, hyperparameter tuning, and 5-seed protocol. No numbers are imported from prior publications.

C Implementation Details

This appendix provides a complete account of the architectural configurations, training procedures, and engineering choices underlying Relational Diffusion. All experiments are implemented in PyTorch with PyTorch Geometric and run on $8\times$ NVIDIA A100 80 GB GPUs.

C.1 Overall Architecture

Relational Diffusion is composed of three tightly coupled modules: (i) a **GNN-based topological edge tokenizer** that maps each edge to a topology-enriched continuous embedding; (ii) a **diffusion Transformer** that learns the joint distribution over edge configurations via iterative denoising; and (iii) a **relational prompt completion** head that conditions generation on task-specific prompts for unified multi-task inference. Table 6 summarises the key architectural hyper-parameters.

C.2 Topological Edge Tokenization

Each edge $e_m = (v_i^{(m)}, v_j^{(m)})$ is mapped to a d_e-dimensional embedding through the following pipeline.

GNN node encoding. We apply a 2-layer Graph Attention Network (GAT) [Velickovic et al., 2018] to obtain contextualised node representations $\mathbf{H} = \mathrm{GAT}(\mathbf{X}, \mathcal{G}) \in \mathbb{R}^{N\times d_h}$, with $d_h = 112$ and 4 attention heads. The attention coefficient between nodes v_i and v_j at layer ℓ is

$$\alpha_{ij}^{(\ell)} = \frac{\exp\big(\mathrm{LeakyReLU}(\mathbf{w}_a^{(\ell)\top}[\mathbf{h}_i^{(\ell)} \| \mathbf{h}_j^{(\ell)}])\big)}{\sum_{v_k\in\mathcal{N}(v_i)} \exp\big(\mathrm{LeakyReLU}(\mathbf{w}_a^{(\ell)\top}[\mathbf{h}_i^{(\ell)} \| \mathbf{h}_k^{(\ell)}])\big)}. \tag{17}$$

Table 6: **Architectural hyper-parameters of Relational Diffusion.** All settings are shared across the three task types unless noted otherwise.

Module	Hyper-parameter	Value
GNN Encoder	Architecture	GAT
	Number of layers	2
	Hidden dimension	112
	Attention heads	4
Diffusion Transformer	Number of layers	4
	Hidden dimension d_e	224
	Attention heads	8
	Feed-forward dim	448
	Dropout	0.1
Diffusion Process	Noise schedule	Cosine
	Diffusion steps T	1,000
	Edge sequence length	up to 2,048
Edge Tokenization	Structural features	Betweenness centrality, clustering coefficient
Structural Encoding	Method	Random-Walk PE (RWPE)
	Walk steps	8
Total trainable params	(excl. frozen text encoder)	$\approx$2.5 M

Node representations are initialised from the frozen Llama-3.1-8B embeddings ($d_v = 4096$) projected to d_h via a learnable linear layer $\mathbf{W}_{\text{init}}$.

Edge embedding construction. For edge e_m, the final token is

$$\mathbf{e}_m = \mathbf{W}_{\text{edge}}\big[\mathbf{h}_{v_i^{(m)}} \,\|\, \mathbf{h}_{v_j^{(m)}} \,\|\, \mathbf{s}_m\big] \;\in\; \mathbb{R}^{d_e}, \tag{18}$$

where $\mathbf{s}_m$ encodes per-edge structural scalars (betweenness centrality and clustering coefficient of both endpoints) and $\mathbf{W}_{\text{edge}} \in \mathbb{R}^{d_e \times (2d_h + d_s)}$ is a learnable projection with $d_s = 4$.

Random-Walk Positional Encoding (RWPE). To inject topological context without breaking permutation equivariance, we compute node-level RWPE with $r = 8$ steps. The k-step landing probability of a random walk starting at node v is

$$\text{RWPE}(v)_k = \big[(\mathbf{D}^{-1}\mathbf{A})^k\big]_{vv}, \quad k = 1, \ldots, r, \tag{19}$$

where $\mathbf{A}$ is the adjacency matrix and $\mathbf{D}$ is the degree matrix. Edge-level positional encodings are obtained by averaging the RWPE vectors of the two endpoint nodes and projecting to d_e via a linear layer. These encodings are added to edge token embeddings before Transformer processing.

Edge sequence construction. Edges are sorted lexicographically by (source, destination) node index and packed into a sequence of length up to $L_{\max} = 2{,}048$. Shorter sequences are zero-padded; a Boolean padding mask tracks valid positions. For graphs exceeding $L_{\max}$ edges (only ogbn-Arxiv in our setup), we apply neighbourhood sampling: we first sample target edges uniformly, then greedily expand to their 2-hop neighbours until the budget is reached.

C.3 Diffusion Transformer

Forward process. We corrupt edge embeddings with Gaussian noise following a cosine schedule:

$$q(\mathbf{z}_t \mid \mathbf{z}_0) = \mathcal{N}\big(\mathbf{z}_t;\; \sqrt{\bar{\alpha}_t}\,\mathbf{z}_0,\; (1 - \bar{\alpha}_t)\mathbf{I}\big), \quad \bar{\alpha}_t = \prod_{\tau=1}^{t}(1 - \beta_\tau), \tag{20}$$

with cosine schedule $\beta_t = \min\big(1 - g(t)/g(t-1),\, 0.999\big)$, where $g(t) = \cos^2\big(\frac{t/T+s}{1+s} \cdot \frac{\pi}{2}\big)$, $s = 0.008$ and $T = 1{,}000$; without clipping, this gives $\bar{\alpha}_t = g(t)/g(0)$.

Denoising network. The reverse process is parameterised by a Transformer f_θ with 4 layers, hidden dimension $d_e = 224$, 8 attention heads, and feed-forward dimension 448. Crucially, **no positional encodings are added to edge tokens** so that the model remains permutation-equivariant with respect to edge ordering. Timestep t is injected via sinusoidal embeddings added to every token before the first layer. The denoising objective is

$$\mathcal{L}_{\text{relational}} = \mathbb{E}_{t,\,\mathbf{z}_0,\,\mathbf{z}_t}\Big[\sum_{m=1}^{M} \|\mathbf{e}_m - f_\theta(\mathbf{z}_t, t)_m\|_2^2\Big]. \tag{21}$$

Permutation equivariance by design. Because self-attention satisfies $f_\theta(\mathbf{Z}_\sigma) = f_\theta(\mathbf{Z})_\sigma$ for any permutation σ of edge indices, and because we omit positional encodings on edge tokens, the model output is invariant to the ordering in which edges are presented. This is a deliberate architectural choice that enforces the relational inductive bias described in Section 4.1.

C.4 Relational Prompt Completion

We unify node classification, link prediction, and graph classification through a shared sequence-completion interface. Each input sequence has the form

$$\mathbf{z}^{\mathcal{T}}_{\text{input}} = [\underbrace{\mathbf{e}_1, \ldots, \mathbf{e}_{M_q}}_{\text{edge tokens}},\ \underbrace{\mathbf{p}^{\mathcal{T}}}_{\text{task prompt}},\ \underbrace{\mathbf{r}^{y}}_{\text{result token}}\ ] \in \mathbb{R}^{(M_q+2)\times d_e}. \tag{22}$$

Learnable embeddings. Both prompt embeddings $\mathbf{p}^{\mathcal{T}}$ and result embeddings $\mathbf{r}^y$ live in $\mathbb{R}^{d_e}$ and are initialised randomly then optimised during fine-tuning. A dedicated mask embedding $\mathbf{r}^{[\texttt{MASK}]}$ is used at inference to represent the unknown prediction target.

Task-specific subgraph extraction. The subgraph $\mathcal{G}_q$ and thus the edge token prefix differ by task:

Table 7: **Task-specific sequence construction** for relational prompt completion. All three tasks share the same diffusion-based inference procedure; only the subgraph definition and result token vocabulary differ.

Task	Subgraph $\mathcal{G}_q$	Result token space
Node classification	K = 2-hop ego-net of v_i	One token per class label $y \in \mathcal{Y}_{\text{node}}$
Link prediction	Union of K = 2-hop ego-nets of v_i, v_j; target edge masked	$\{\mathbf{r}^{\text{exist}}, \mathbf{r}^{\text{not-exist}}\}$
Graph classification	Full graph $\mathcal{G}$	One token per graph label $y \in \mathcal{Y}_{\text{graph}}$

Inference via selective denoising. At test time we construct $\mathbf{z}_T = [\mathbf{z}_{\mathcal{G}_q},\, \mathbf{p}^{\mathcal{T}},\, \mathbf{r}^{[\texttt{MASK}]}]$ and perform a *single forward pass* through the denoising network, keeping edge and prompt tokens fixed:

$$\hat{\mathbf{r}} = f_\theta(\mathbf{z}_T, T)_{M_q+2}, \qquad \hat{y} = \arg\max_{y \in \mathcal{Y}} \text{sim}(\hat{\mathbf{r}},\, \mathbf{r}^y), \tag{23}$$

where $\text{sim}(\cdot,\cdot)$ denotes cosine similarity. This is analogous to image inpainting: the model fills in only the masked result position while the context (graph structure and task identity) is fully observed.

C.5 Training Procedure

Training follows a two-stage paradigm that mirrors LLM pre-training and fine-tuning, with identical diffusion-based denoising used in both stages.

Objective. Edge-only sequences $\mathbf{z}_0 = \mathbf{z}_\mathcal{G}$ are corrupted via forward diffusion and the model is trained to reconstruct the original edge embeddings:

$$\mathcal{L}_{\text{pretrain}} = \mathbb{E}_{\mathcal{G},\, t,\, \mathbf{z}_t}\Big[\sum_{m=1}^{M} \|\mathbf{e}_m - f_\theta(\mathbf{z}_t, t)_m\|_2^2\Big].$$

Data. All eight datasets are used jointly. For node and link prediction, we extract up to 500 subgraphs per dataset; for graph classification the full graph corpus is used.

Optimiser. AdamW, learning rate 1×10^{-4}, ExponentialLR decay $\gamma = 0.95$, 200 epochs, batch size 8 per GPU ($\times$8 GPUs).

Objective. Full prompted sequences $\mathbf{z}_0 = \mathbf{z}^{\mathcal{T}}_{\text{input}}$ (including ground-truth result tokens) are used:

$$\mathcal{L}_{\text{finetune}} = \mathbb{E}_{\mathcal{T},\, \mathbf{z}^{\mathcal{T}}_0,\, t,\, \mathbf{z}_t}\big[\|\mathbf{z}^{\mathcal{T}}_0 - f_\theta(\mathbf{z}_t, t)\|_2^2\big].$$

Backbone freezing schedule. In Stage 2, we optionally freeze the Transformer encoder for an initial warm-up phase (Stage 1 epochs of fine-tuning loop) to stabilise the task-head, then unfreeze all parameters for joint optimisation.

Optimiser. Adam, learning rate 5×10^{-5} for the backbone and 1×10^{-4} for the task head, 100 epochs, batch size 32.

△ **Note on stage consistency.** Both pre-training and fine-tuning employ *identical* diffusion-based denoising over edge sequences — they differ only in whether task-conditioning tokens are appended. This design ensures seamless transfer of pre-trained structural knowledge to downstream tasks without any architectural modifications.

C.6 Baseline Re-implementation

All baselines are re-implemented and evaluated under the same protocol as Relational Diffusion: identical 60/20/20 splits, identical frozen Llama-3.1-8B node features, and the same 5-seed evaluation with mean $\pm$ std reporting. No numbers are imported from prior publications. Table 8 summarises the configuration used for each baseline.

Table 8: **Baseline configurations** used in all experiments. All models receive the same frozen Llama-3.1-8B node features as input.

Category	Method	Key configuration
GNN	GCN [Kipf and Welling, 2017]	2 layers, hidden 256, Adam lr 1×10^{-3}, 200 epochs
	GraphSAGE [Hamilton et al., 2017]	2 layers, hidden 256, mean aggregation, same schedule
	GAT [Velickovic et al., 2018]	2 layers, hidden 256, 8 heads, same schedule
	GIN [Xu et al., 2019]	2 layers, hidden 256, sum aggregation, same schedule
Graph Trans.	Graphormer [Ying et al., 2021]	6 layers, $d = 256$, 8 heads; spatial/degree encodings
	GraphGPS [Rampásek et al., 2022]	4 layers, $d = 256$, RWPE + Transformer + MPNN
Foundation	OFA [Liu et al., 2024]	Official codebase; Llama-3.1-8B features; same splits
	GraphMAE [Hou et al., 2022]	GATv2 encoder, masking ratio 0.5; same features/splits

For all GNN baselines, we perform a grid search over learning rates $\{10^{-2}, 10^{-3}, 10^{-4}\}$, hidden dimensions $\{128, 256\}$, and dropout rates $\{0.0, 0.1, 0.3\}$, selecting the configuration with the best

validation performance. The same search budget is applied to Relational Diffusion to ensure a fair comparison.

D Full Main Results with Standard Deviations

For space, the main paper reports test accuracy without standard deviations. Here we provide the complete results, including standard deviations over 5 random seeds. Tables 9, 10, and 11 mirror the main-paper tables exactly, with standard deviations included.

Table 9: Node classification: validation and test accuracy (%) with standard deviations over 5 seeds.

		Cora		PubMed		Wiki-CS		ogbn-Arxiv	
Category	Method	Valid	Test	Valid	Test	Valid	Test	Valid	Test
GNN	GCN	81.47 ± 0.68	80.36 ± 0.92	75.34 ± 0.81	74.28 ± 1.05	77.93 ± 0.87	76.81 ± 1.09	69.93 ± 0.87	64.81 ± 1.09
	GraphSAGE	82.81 ± 0.59	81.74 ± 0.83	76.85 ± 0.72	75.92 ± 0.94	78.76 ± 0.79	77.65 ± 1.02	70.76 ± 0.79	64.65 ± 1.02
	GAT	83.19 ± 0.77	82.35 ± 1.04	77.62 ± 0.89	76.48 ± 1.13	78.42 ± 0.96	77.19 ± 1.18	71.42 ± 0.96	65.19 ± 1.18
	GIN	80.62 ± 1.03	79.28 ± 1.17	74.15 ± 1.14	72.86 ± 1.26	75.68 ± 1.21	74.35 ± 1.29	70.68 ± 1.21	63.35 ± 1.29
Graph Trans.	Graphormer	85.76 ± 0.14	83.45 ± 1.48	81.62 ± 0.53	80.45 ± 0.17	81.45 ± 0.30	79.05 ± 1.50	77.20 ± 0.60	68.70 ± 0.20
	GraphGPS	84.90 ± 0.55	82.80 ± 0.78	80.74 ± 0.62	79.58 ± 0.85	79.85 ± 0.72	78.65 ± 0.95	72.50 ± 0.85	66.40 ± 1.05
Foundation	OFA	72.85 ± 0.76	71.24 ± 0.83	78.45 ± 0.61	77.36 ± 0.67	70.18 ± 0.84	68.92 ± 0.91	75.63 ± 0.71	72.38 ± 0.78
	GraphMAE	82.88 ± 0.73	81.34 ± 0.65	79.83 ± 0.88	78.62 ± 0.78	80.71 ± 0.99	78.27 ± 1.06	72.50 ± 0.87	66.84 ± 1.04
Ours	**Rel. Diffusion**	$\mathbf{89.82 \pm 0.93}$	$\mathbf{88.71 \pm 1.21}$	$\mathbf{82.74 \pm 1.05}$	$\mathbf{81.65 \pm 0.89}$	$\mathbf{85.13 \pm 1.14}$	$\mathbf{83.56 \pm 0.97}$	$\mathbf{77.93 \pm 0.87}$	$\mathbf{74.64 \pm 1.32}$

Table 10: Link prediction: MRR (%) with standard deviations over 5 seeds.

		WN18RR		FB15K-237	
Category	Method	Valid	Test	Valid	Test
GNN	GCN	78.62 ± 0.44	77.85 ± 0.52	82.53 ± 0.50	81.74 ± 0.58
	GAT	82.74 ± 0.55	81.93 ± 0.63	83.38 ± 0.76	82.51 ± 0.85
	GraphSAGE	81.58 ± 0.34	80.76 ± 0.42	$\underline{89.28 \pm 0.06}$	$\underline{88.63 \pm 0.09}$
	GIN	80.46 ± 0.58	79.62 ± 0.67	85.60 ± 1.10	84.72 ± 1.24
Graph Trans.	Graphormer	85.62 ± 0.48	84.78 ± 0.56	86.40 ± 0.62	85.51 ± 0.74
	GraphGPS	86.74 ± 0.41	85.83 ± 0.50	88.10 ± 0.35	87.24 ± 0.43
Foundation	OFA	87.85 ± 0.45	86.92 ± 0.55	88.95 ± 0.46	88.05 ± 0.50
	GraphMAE	85.34 ± 0.38	84.46 ± 0.46	88.45 ± 0.20	87.58 ± 0.28
Ours	**Relational Diffusion**	$\mathbf{90.18 \pm 0.33}$	$\mathbf{89.42 \pm 0.53}$	$\mathbf{90.49 \pm 0.60}$	$\mathbf{89.67 \pm 0.69}$

Table 11: Graph classification: ROC-AUC (%) on ogbg-MolHIV and Average Precision (%) on ogbg-MolPCBA, with standard deviations over 5 seeds.

		ogbg-MolHIV (AUC)		ogbg-MolPCBA (AP)	
Category	Method	Valid	Test	Valid	Test
GNN	GCN	70.18 ± 1.29	69.73 ± 1.45	20.61 ± 1.06	20.07 ± 1.19
	GIN	63.92 ± 1.24	60.05 ± 1.58	23.59 ± 1.80	22.20 ± 0.78
	GraphSAGE	61.47 ± 1.35	57.82 ± 1.67	25.23 ± 0.53	18.57 ± 0.61
	GAT	71.65 ± 1.38	69.91 ± 1.53	20.43 ± 1.22	20.17 ± 1.27
Graph Trans.	Graphormer	$\mathbf{78.42 \pm 0.95}$	$\mathbf{77.85 \pm 1.18}$	30.85 ± 0.78	29.74 ± 0.92
	GraphGPS	77.38 ± 0.57	76.12 ± 0.94	28.86 ± 0.94	27.56 ± 0.89
Foundation	OFA	76.85 ± 1.12	75.38 ± 1.24	24.76 ± 1.42	22.34 ± 1.58
	GraphMAE	74.20 ± 0.85	73.15 ± 1.05	26.45 ± 1.20	25.18 ± 1.35
Ours	**Relational Diffusion**	77.15 ± 1.50	$\underline{76.54 \pm 1.92}$	$\mathbf{31.27 \pm 1.56}$	$\mathbf{30.63 \pm 1.64}$

E Additional Ablation Results

E.1 Component Ablation: Full Results with Standard Deviations

Table 12 reproduces the component ablation from the main paper (Figure 3) with standard deviations and computed deltas relative to the full model.

Table 12: Component ablation across three representative benchmarks: Cora (node classification, accuracy %), WN18RR (link prediction, MRR %), and ogbg-MolPCBA (graph classification, Average Precision %). Numbers in parentheses are absolute drops relative to the full model.

Variant	Cora (Acc.)	WN18RR (MRR)	PCBA (AP)
Full Relational Diffusion	**88.71**	**89.42**	**30.63**
w/o Edge Co-occurrence (Node-Seq)	85.60 (-3.11)	86.66 (-2.76)	28.30 (-2.33)
w/o Topological Enrichment	83.90 (-4.81)	82.96 (-6.46)	25.20 (-5.43)
w/o Diffusion	86.90 (-1.81)	84.36 (-5.06)	26.10 (-4.53)
w/o Relational Prompt Completion (no Stage 2)	79.40 (-9.31)	83.66 (-5.76)	25.60 (-5.03)
w/o Pre-training (no Stage 1)	82.80 (-5.91)	80.46 (-8.96)	23.10 (-7.53)

Cross-task observations. Three patterns emerge across the three benchmarks. First, among the architectural components (E, T, D), *topological enrichment* (using GAT-encoded multi-hop endpoints rather than raw features) yields the largest performance drops on link prediction and graph classification, where edge structure is most directly tied to the prediction objective. Second, *relational prompt completion* (i.e., the Stage-2 task-prompted fine-tuning) is the most impactful component on Cora node classification (-9.31), suggesting that mapping pre-trained structural knowledge to specific tasks via prompts is essential when edges are not the prediction target itself. Third, *pre-training* (Stage 1) consistently provides 5–9 point gains, with the largest impact on link prediction and graph classification. The Node-Seq variant drops more uniformly across all three tasks, supporting our claim that edge tokens are more informative than node sequences regardless of the downstream task type.

E.2 Sensitivity to Subgraph Hop K

Table 13 examines the effect of the ego-subgraph hop K. We find $K = 2$ is the sweet spot: smaller K deprives the diffusion model of edge co-occurrence context, while larger K introduces noise on heterophilic neighborhoods.

Table 13: Effect of subgraph hop K (avg. test acc. across the four node classification datasets).

K	1	2	3	4
Avg. Acc.	81.97	**83.21**	82.84	80.05

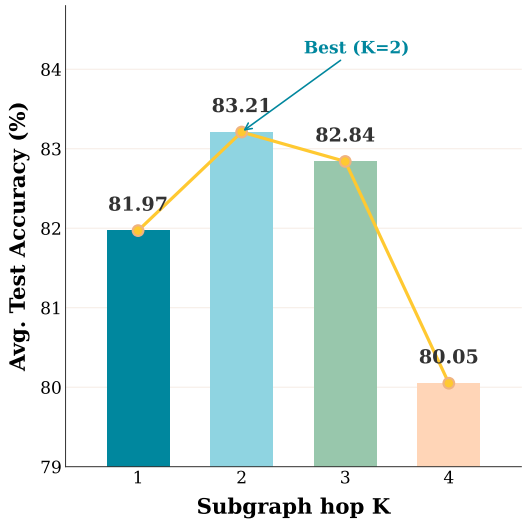

Figure 10: Effect of subgraph hop K on average node classification accuracy. Performance peaks at $K = 2$ and degrades sharply at $K = 4$ (-3.16 from peak), where the ego-subgraph grows large enough that distant, weakly-related nodes contribute noisy edge tokens. The asymmetric drop on either side of $K = 2$ confirms that two-hop neighborhoods provide an effective trade-off between expressive context and signal-to-noise.

F Scaling Behavior

We analyze how performance scales with model size and pre-training data volume.

Model size scaling. Table 14 reports the effect of varying the diffusion Transformer depth and hidden dimension. We observe consistent improvements with model scale, suggesting our framework benefits from compute analogous to LLMs in language.

Table 14: Model size scaling on ogbn-Arxiv. #Params reflects diffusion Transformer parameters only (excluding the GAT encoder).

Configuration	Layers	Hidden Dim	#Params	Test Acc. (%)
Tiny	2	128	0.4M	70.81
Small	3	160	0.9M	73.27
Base (ours)	4	224	**2.4M**	**74.64**
Large	6	320	7.4M	75.36
XLarge	8	448	19.3M	75.95

Pre-training data scaling. Table 15 reports the effect of pre-training corpus size, measured in total edge tokens. Performance follows a roughly log-linear trajectory, with no clear saturation in our experimental range.

Table 15: Pre-training data scaling on Cora node classification. We sample subsets of the source corpus by total edge count.

Pre-training Edges	Test Acc. on Cora (%)
0 (no pre-training)	84.93
100K	86.18
1M	87.45
5M	88.21
All (10.4M)	**88.71**

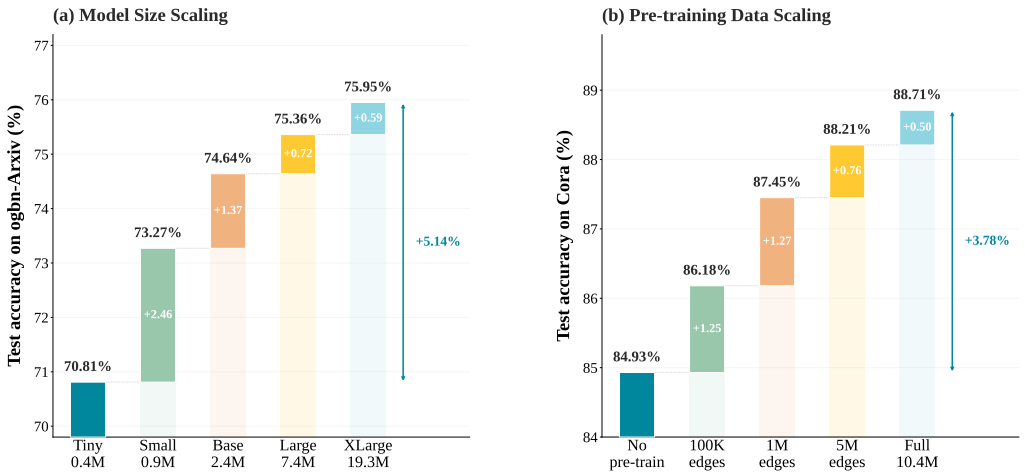

Figure 11: Visualization of the two scaling experiments. **(a) Model size scaling** on ogbn-Arxiv: each step up the configuration ladder yields diminishing but consistent gains, with the largest jump from Tiny ($+2.46$) and rapidly diminishing returns beyond Base. The Base configuration (highlighted) sits at the knee of the curve, providing the best efficiency-accuracy trade-off and motivating its use as the paper default. **(b) Pre-training data scaling** on Cora: accuracy increases monotonically with corpus size, with cumulative pre-training contribution reaching $+3.78\%$ over the no-pre-training baseline. The progression shows no clear saturation within our experimental range, suggesting further gains may be achievable with larger corpora.

Figure 11 jointly visualizes both scaling experiments. The model-size panel (left) shows a saturation pattern characteristic of well-regularized architectures: the marginal gain per parameter doubling decreases sharply, justifying our choice of the 2.4M-parameter Base configuration as the practical operating point. The data-scaling panel (right) shows an approximately log-linear improvement with pre-training corpus size, which together with the absence of saturation suggests that the framework's full potential is data-bounded rather than capacity-bounded at our current scale.

G Evaluation on the Official MolHIV Split

To complement the random-split comparison in Table 1, we evaluate on the official ogbg-MolHIV scaffold split [Hu et al., 2020a], keeping the shared frozen Llama-3.1-8B node features. Only training-split graphs are used for pre-training. Table 16 reports ROC-AUC for OFA, GraphMAE, and relational diffusion. Relational diffusion achieves 75.82%, outperforming OFA (74.62%) and GraphMAE (72.41%) by $+1.20\%$ and $+3.41\%$, respectively, showing that its advantage carries over to the harder scaffold split.

Table 16: ROC-AUC (%) on the official ogbg-MolHIV scaffold split (mean $\pm$ std) with shared frozen Llama-3.1-8B features.

Method	MolHIV ROC-AUC (%)
OFA	74.62 ± 1.18
GraphMAE	72.41 ± 1.03
Relational Diffusion	$\mathbf{75.82 \pm 1.37}$

H Additional Discussion

H.1 Relation to Existing Diffusion-on-Graph Approaches

A line of prior work applies diffusion to *graph generation*, e.g., DiGress Vignac et al. [2023] performs discrete diffusion over node and edge categories, while EDP-GNN Niu et al. [2020] performs score-based diffusion over adjacency matrices. Our work differs in three ways:

1. **Purpose.** These works target unconditional graph generation; we target representation learning for downstream classification and prediction tasks.
2. **Token granularity.** They diffuse over node-attribute matrices or full adjacency tensors; we diffuse over compact edge-token sequences encoding multi-hop topology.
3. **Task interface.** They lack a task-conditioning mechanism; relational prompt completion provides a unified interface for node, link, and graph-level prediction.

Diffusion models on graphs have also been developed for specific downstream tasks, such as graph anomaly detection Wei et al. [2025b, 2026a] and graph-based recommendation Zhang et al. [2026]; these models are trained for a single task, whereas relational diffusion provides a shared pre-training and task-prompting interface across node-, link-, and graph-level prediction. This positions relational diffusion as the first edge-centric *diffusion-based foundation model* for graph reasoning, distinct from generative graph diffusion.

H.2 Broader Impact

Graph foundation models have applications across drug discovery, social network analysis, recommender systems, and scientific research. Our work advances the state of the art in unified graph learning, which may amplify both the benefits and risks of these applications. On the benefits side, more accurate molecular property prediction could accelerate drug discovery and materials science. On the risks side, improved link prediction over social networks raises privacy concerns when applied without consent, and stronger graph foundation models may exacerbate biases present in training graphs (e.g., systematic under-representation of certain communities in citation networks). We encourage practitioners to perform task-specific bias audits before deploying foundation models trained with our method.

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 accurately state our three contributions: relational diffusion framework, topological edge tokenization with relational prompt completion, and experimental validation across eight benchmarks. All claims are supported by the theoretical analysis in Section 5 and experiments in Section 6.

2. **Limitations**

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

   Answer: [Yes]

   Justification: Limitations are discussed in Section 7 (Conclusion and Limitations), including: (1) homogeneous edge treatment limiting expressiveness on multi-relational graphs, and (2) quadratic self-attention cost posing scalability challenges for dense graphs.

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: Theorem 5.1 in Section 5 states all assumptions explicitly (permutation-invariant distribution P^*, an edge model class containing the symmetrizations of the sequential model class) and provides a complete proof, with the detailed derivation of Δ_{order} in Appendix A.

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: Section 6 provides full implementation details including model architecture (2-layer GAT encoder, 4-layer DiT with hidden dim 224, 8 heads), optimizer (AdamW), learning rates, training epochs, hardware (8$\times$ A100 80GB), and random seeds. Code is available at `https://github.com/wangxianan95/RelationalDiffusion`.

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 is publicly available at `https://github.com/wangxianan95/RelationalDiffusion` (see footnote in abstract). All datasets used are publicly available benchmarks (Cora, PubMed, Wiki-CS, ogbn-Arxiv, WN18RR, FB15K-237, ogbg-MolHIV, ogbg-MolPCBA) with citations provided.

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Section 6 specifies data splits (60%/20%/20% train/val/test with fixed seed), hyperparameters (hidden dims, number of layers, heads, learning rates, epochs), optimizer (AdamW), hardware, and evaluation metrics. Additional dataset statistics are provided in Appendix B.

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: All results in Table 1 and Table 3 are reported as mean $\pm$ std over 5 independent runs with different random seeds, as stated in Section 6.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Section 6 states that experiments are conducted on 8$\times$ A100 80GB GPUs. Section 6.3 additionally reports per-sample inference latency (e.g., 15.2 ms on a single A800) and parameter counts for all compared methods.

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: This work involves graph representation learning on standard public benchmarks with no human subjects, sensitive data, or dual-use concerns. We have reviewed the NeurIPS Code of Ethics and confirm full compliance.

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: The proposed method is a general graph foundation model with broad applicability (e.g., molecular property prediction, knowledge graph reasoning, citation network analysis). Positive impacts include enabling better generalization in scientific discovery. As a general-purpose method, potential misuse (e.g., applied to surveillance or social network analysis) is possible; however, no direct negative application path is introduced by this work beyond what existing GNN methods already enable.

11. **Safeguards**

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

    Answer: [N/A]

    Justification: This paper does not release large-scale pre-trained models or scraped datasets. The model is trained on standard academic benchmarks and does not pose a high risk for misuse.

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: [N/A]

    Justification: All datasets and baselines are properly cited in Section 6 and the references. We use publicly available academic benchmarks (ogb datasets under MIT license, Cora/PubMed in the public domain) and do not redistribute any assets.

13. **New assets**

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

    Answer: [N/A]

Justification: This paper does not introduce new datasets. The released code implementing the proposed method is documented in its public repository (`https://github.com/wangxianan95/RelationalDiffusion`).

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

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

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing or research with human subjects.

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

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

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing or research with human subjects, so IRB approval is not applicable.

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?

    Answer: [Yes]

    Justification: A frozen Llama-3.1-8B encoder is used as a feature extractor for node textual descriptions (Section 6, Unified data construction). This usage is explicitly described. The LLM is not part of the core proposed method (relational diffusion), but serves as a standard feature encoding baseline following OFA Liu et al. [2024].