Mixed-Curvature Latent Diffusion for Graph Generation with Structural Heterogeneity

Minxue Qi
Institute of Intelligent Information Processing
Shanxi University
Shanxi, China
qiminxuert@qq.com

Xian Yang
Alliance Manchester Business School
The University of Manchester
Manchester M13 9PL, UK
xian.yang@manchester.ac.uk

Liang Bai*
Institute of Intelligent Information Processing
Shanxi University
Shanxi, China
bailiang@sxu.edu.cn

Abstract

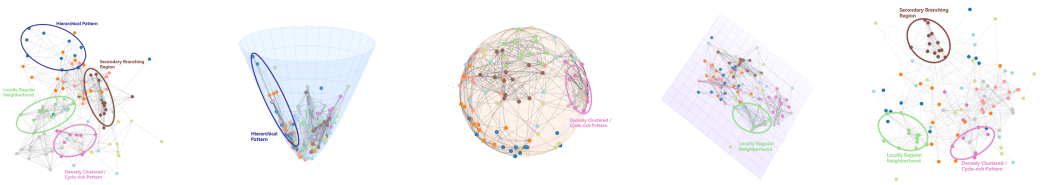
Existing latent graph diffusion models often learn graph representations and generative priors within a homogeneous latent geometry. This assumption is restrictive for graph generation with structural heterogeneity, where hierarchical, cyclic, locally regular, and densely connected patterns may coexist within the target distribution. Under such conditions, forcing all structural patterns into one latent geometry can introduce representation distortion, which may then propagate to generation. We propose CurvDiff, a latent graph generation framework based on mixed-curvature latent diffusion. The main idea is to model structurally different graph patterns in geometry-compatible latent factors rather than representing them in a single-curvature space. Specifically, CurvDiff learns graph embeddings in a product latent geometry consisting of hyperbolic, spherical, and Euclidean components, and maps the resulting representation to a shared tangent-space interface on which latent diffusion is defined. This provides a consistent latent pathway from representation learning to generation. To further preserve topology-relevant structure, CurvDiff uses Ricci-derived structural signals as topology-aware priors. These signals regularize latent representation learning and condition the diffusion prior, helping maintain structural consistency across representation learning and generation. Experiments on synthetic and real-world graph benchmarks show that CurvDiff achieves competitive or superior graph distribution matching compared with strong baselines, with improved fidelity on degree, clustering, and spectral statistics. These results support mixed-curvature latent diffusion, combined with Ricci-derived structural priors and a shared latent interface, as an effective approach to graph generation with structural heterogeneity.

1 Introduction

Representative graph diffusion and latent graph generation models often operate in homogeneous Euclidean representation or diffusion spaces [Jo et al., 2022, Luo et al., 2024]. Real-world graphs often exhibit heterogeneous structural patterns [Boguñá et al., 2021]. These include hierarchical organization [Ravasz and Barabási, 2003], cyclic motifs [Milo et al., 2002], grid-like regions [Bronstein et al.,

*Corresponding author.

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



(a) Original Layout (b) Hyperbolic Space (c) Spherical Space (d) Euclidean Space (e) Mixed Space

Figure 1: Qualitative comparison of geometric embeddings on a heterogeneous graph. Colored outlines mark hierarchical/branching (blue/brown), cyclic/dense (magenta), and locally regular (green) regions. Quantitative comparisons are reported in Sections 5.3 and 5.4.

2021], and locally dense connectivity [Topping et al., 2022]. When these patterns are compressed into a Euclidean or single-curvature latent space, topology-relevant information may be lost, and a generative model trained in that space may inherit the resulting structural distortion [Mathieu et al., 2019, Bose et al., 2020].

This problem is closely related to the geometric nature of graph structure. Hierarchical or tree-like graphs are naturally compatible with hyperbolic geometry, whose exponential volume growth supports the separation of branching structures [Nickel and Kiela, 2017, Sala et al., 2018]. Cyclic and closed patterns can benefit from a different inductive bias that is more closely related to positively curved geometry, whereas grid-like and locally flat regions are more naturally captured in Euclidean space [Skopek et al., 2020]. A single-curvature latent space may require compromises in representation quality when accommodating these different structural regimes. Despite advances in graph latent diffusion [Nguyen et al., 2025] and non-Euclidean graph generation [Bu et al., 2025, Fu et al., 2024, Wen et al., 2024], the consistent use of graph-derived structural priors across encoding and generation remains underexplored.

The above limitations motivate a latent generative space that is both geometrically heterogeneous and structurally informed. Mixed-curvature representation methods [Wang et al., 2021, Sun et al., 2022] address the first requirement by allowing different geometric components to capture different structural regimes. A product manifold composed of hyperbolic, spherical, and Euclidean factors provides complementary inductive biases for hierarchical, cyclic, and locally flat structures [Gu et al., 2019], as illustrated in Fig. 1. Recent studies suggest that topology preservation is critical for faithful graph generation [Papamarkou et al., 2024, Yan et al., 2025]. Structural signals [Lin et al., 2011, Sreejith et al., 2016] provide a natural way to characterize such patterns.

We propose CurvDiff, a unified latent graph generation framework that integrates mixed-curvature geometry with Ricci-curvature-informed guidance. CurvDiff represents graphs on a product manifold with hyperbolic, spherical, and Euclidean components, and maps the resulting representation to a shared tangent-space interface for diffusion-based generation. To connect continuous latent diffusion with discrete graph topology, CurvDiff uses Ricci curvature [Fesser et al., 2023] to form a unified geometry–topology interface rather than simple feature augmentation. We adopt Forman–Ricci curvature because it admits efficient closed-form combinatorial computation [Fesser and Weber, 2024], thereby avoiding the repeated optimal-transport computations required by Ollivier–Ricci curvature [Ollivier, 2009].

Our contributions are:

- **Mixed-curvature latent modeling for structural heterogeneity.** We propose a latent graph diffusion framework that replaces the single-geometry assumption with a mixed-curvature product manifold, enabling heterogeneous graph structures to be represented in more suitable geometric subspaces.
- **Shared tangent-space interface for topology-aware generation.** We introduce a shared tangent-space interface that maintains a consistent latent pathway across representation learning and diffusion generation, with Ricci-derived structural signals serving as topology-aware priors for consistency regularization and conditioning.
- **Improved matching of graph structural statistics.** On multiple synthetic and real-world graph generation benchmarks, CurvDiff achieves competitive or superior MMD-based fidelity and improves the matching of degree, clustering, and spectral statistics.

2 Related Work

Graph Diffusion and Flow-Based Generation. Graph diffusion has become an effective paradigm for graph generation. Continuous and score-based methods, such as GDSS [Jo et al., 2022] and GraphGDP [Huang et al., 2022], model graph distributions through denoising dynamics or stochastic differential processes, while discrete diffusion methods, such as DiGress [Vignac et al., 2023a] and EDGE [Chen et al., 2023], define noise and denoising transitions directly over discrete graph states. Recent general-purpose graph generators further include DeFoG [Qin et al., 2025], which uses discrete flow matching, and Cometh [Siraudin et al., 2025], which uses continuous-time discrete-state diffusion. Latent graph diffusion further improves efficiency by moving generation into learned latent spaces, with representative methods including Graphusion [Yang et al., 2024], LGD [Zhou et al., 2024], GLAD [Nguyen et al., 2025], and LG-Flow [Siraudin and Morris, 2026]. Despite these advances, most existing methods do not explicitly model mixed geometric structures in the latent representation, motivating our focus on latent geometry rather than modifying the diffusion backbone alone.

Geometry-Aware Graph Generation. Geometry-aware graph learning [Ganea et al., 2018] provides an important perspective for addressing structural mismatch in graph representation. This idea has also influenced graph generation, with representative hyperbolic generative models including HypDiff [Fu et al., 2024], HGDM [Wen et al., 2024], and GGBall [Bu et al., 2025]. Specifically, HypDiff constrains radial and angular diffusion for topology preservation, HGDM combines hyperbolic auto-encoding with diffusion for hierarchical graphs, and GGBall integrates a hyperbolic VQ-VAE with Riemannian flow matching on the Poincaré ball. However, a single hyperbolic geometry may be insufficient for real graphs that contain coexisting hierarchical, cyclic, grid-like, and densely clustered structures. Mixed-curvature representation methods [Wang et al., 2021, Sun et al., 2022] address this issue by combining multiple geometric components, and ProGDM [Wang et al., 2024] explores multiple Riemannian subspaces for graph generation. In contrast, CurvDiff couples a product-manifold latent space with diffusion through a shared tangent-space interface.

Topology-Aware Structural Priors. Topology-aware priors have been widely used to guide graph learning and generation. For example, GraphRNN [You et al., 2018] uses BFS orderings as node-ordering priors, SPECTRE [Martinkus et al., 2022] leverages graph Laplacian spectra for global conditioning, DiGress [Vignac et al., 2023a] incorporates local cycle counts, and MiDi [Vignac et al., 2023b] enforces molecular geometric and valency priors. While effective, these priors are often task-specific or focus on selected structural statistics. Ricci curvature offers a complementary topology-aware signal for identifying bottlenecks, clustering behavior, and community structures [Jost and Liu, 2014, Fesser and Weber, 2024]. However, Ricci-based graph methods rarely couple curvature signals with latent graph generation. In CurvDiff, Ricci-derived signals guide both mixed-curvature representation learning and the latent diffusion prior, maintaining structural consistency across representation and generation.

3 Methodology

3.1 Overview: Mixed-Curvature Latent Generation for Heterogeneous Graphs

We propose CurvDiff as a unified latent graph generation framework rather than a pipeline of independent representation and diffusion modules. As illustrated in Fig. 2, the framework first constructs a latent substrate where heterogeneous graph structures can be represented across hyperbolic, spherical, and near-Euclidean factors. It then exports this substrate to a shared tangent space on which diffusion is defined. Finally, Ricci-derived structural statistics are used as topology-aware priors that are shared by the representation learner and the diffusion prior. Ricci-E augments node features and regularizes reconstruction through FRC consistency, while Ricci-D conditions latent denoising on graph-level curvature summaries. These graph-derived signals encourage structural consistency, without imposing a fixed correspondence with latent manifold factors or guaranteeing exact topology preservation.

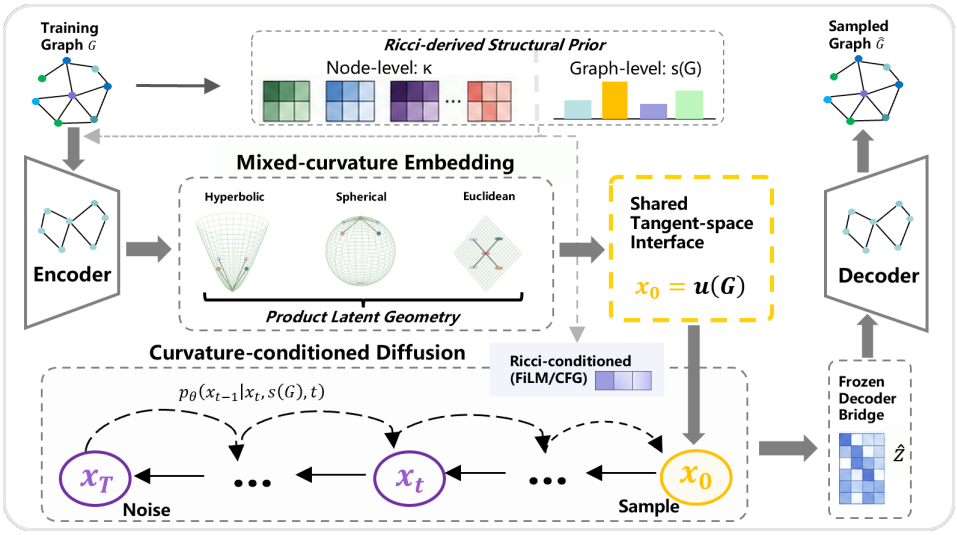


Figure 2: Overview of CurvDiff. A mixed-curvature encoder exports product-manifold embeddings to a shared tangent-space interface for diffusion. Ricci-derived structural signals provide topology-aware priors for representation regularization and diffusion conditioning.

Notation. We use κ for node-level Ricci features, and c_b for latent-factor curvatures. The encoder exports the tangent representation $u(G)$; z denotes the stochastic reconstruction latent, $x_0 = u(G)$ the diffusion target, and $\mathbf{s}(G)$ the Ricci-derived condition (see Appendix C.1 for a notation summary).

3.2 Learning a Mixed-Curvature Latent Substrate

3.2.1 Graph Representation with Structural Augmentation

Let $G = (A^{\text{raw}}, X^{\text{raw}})$ be an input graph. We use an ordered and padded representation (A, X, m), where $A \in \{0,1\}^{N_{\max} \times N_{\max}}$, $X \in \mathbb{R}^{N_{\max} \times F}$, and $m \in \{0,1\}^{N_{\max}}$ marks valid nodes. Graphs are placed under a fixed ordering and padded to a dataset-specific maximum size; implementation details are summarized in Appendix E.5.

To connect latent representation learning with discrete graph topology, we augment node features using Ricci-derived curvature channels:

$$X' = [X \,\|\, \kappa], \tag{1}$$

where $\kappa \in \mathbb{R}^{N_{\max} \times d_\kappa}$ denotes node-level curvature statistics computed from Forman–Ricci curvature. This augmentation introduces the topology-aware structural signal. We defer the detailed curvature computation to Appendix B.

3.2.2 Mixed-Curvature Graph Encoder

Mixed-curvature encoding and tangent export. Given (A, X'), the encoder first extracts node representations through GCN and graph pooling to obtain the graph-level embedding $\mathbf{h}^{(0)} \in \mathbb{R}^{d'}$. The mixed-curvature layer defines a product manifold

$$\mathcal{M} = \prod_{b=1}^{B} \mathcal{M}_{c_b}^{(d_b)}, \qquad \sum_{b=1}^{B} d_b = d', \tag{2}$$

where each factor $\mathcal{M}_{c_b}^{(d_b)}$ is a stereographic manifold with curvature c_b and dimension d_b.

The mixed-curvature encoder then maps $\mathbf{h}^{(0)}$ into the product manifold $\mathcal{M}$ by applying a linear transformation in the tangent space at the origin, followed by exponential mapping onto each curvature branch. The final representation is mapped back to the origin tangent space and exported as the

shared latent code for the diffusion stage. We move the detailed geometric operators to Appendix A to keep the main text focused on the generative mechanism.

$$u = \log_0^{\mathcal{M}}\Big(\phi_{\mathcal{M}}\Big(\exp_0^{\mathcal{M}}(W\mathbf{h}^{(0)})\Big)\Big), \qquad \mathcal{M} = \mathcal{M}_{\mathrm{hyp}} \times \mathcal{M}_{\mathrm{sph}} \times \mathcal{M}_{\mathrm{euc}}, \tag{3}$$

Adaptive curvature learning. The product structure and factor dimensions are fixed; adaptation is implemented through learnable factor curvatures, without separate scalar mixture weights. The curvature parameters c_b, including the Euclidean-like branch initialized at 10^{-3}, are optimized jointly with the network parameters:

$$\boldsymbol{c} = (c_1, \ldots, c_B), \qquad c_b \leftarrow c_b - \eta_c \frac{\partial \mathcal{L}}{\partial c_b}. \tag{4}$$

For reconstruction-based training, the encoder further parameterizes a diagonal Gaussian latent posterior $q_\phi(z \mid G)$ by reparameterization:

$$z = \mu + \sigma \odot \eta, \qquad \eta \sim \mathcal{N}(0, I), \tag{5}$$

where $\mu = f_\mu(u), \log \sigma = f_\sigma(u)$.

3.2.3 Reconstruction-Based Training Objective

The decoder maps the stochastic latent variable z to adjacency logits $\hat{A}$ under the same ordered and padded graph representation. We therefore regularize the reconstructed graph using node-level Forman–Ricci consistency. Using the node-validity mask m, we define

$$\mathcal{L}_{\mathrm{frc}} = \frac{1}{\sum_i m_i + \varepsilon} \sum_i m_i \left(\mathrm{FRC}_i(\hat{A}) - \mathrm{FRC}_i(A)\right)^2. \tag{6}$$

The reconstruction-based objective is

$$\mathcal{L}_{\mathrm{mae}} = \mathcal{L}_{\mathrm{recon}} + \beta\, \mathcal{L}_{\mathrm{KL}} + \gamma\, \mathcal{L}_{\mathrm{frc}}, \tag{7}$$

where $\mathcal{L}_{\mathrm{recon}}$ is a sparsity-aware weighted binary cross-entropy with logits, and $\mathcal{L}_{\mathrm{KL}}$ is the KL divergence between $q_\phi(z \mid G)$ and a standard normal prior.

3.3 Topology-Aware Latent Diffusion with Ricci Prior

3.3.1 Ricci-Derived Structural Condition

Given node-level curvature channels κ and the node-validity mask m, we define the node-level curvature summary as

$$s_{\mathrm{node}}(G) = [\mu_\kappa \parallel \sigma_\kappa], \tag{8}$$

where μ_κ and σ_κ are masked channel-wise mean and standard deviation.

We further summarize edge-level curvature values by binning them into histograms, yielding $s_{\mathrm{edge}}(G)$ as

$$s_{\mathrm{edge}}(G) = \mathrm{Hist}\big(\{\psi(e)\}_{e \in E(G)}\big). \tag{9}$$

The final Ricci-derived condition is

$$\mathbf{s}(G) = [s_{\mathrm{node}}(G) \parallel s_{\mathrm{edge}}(G)]. \tag{10}$$

Acting as a topology-aware structural prior rather than an external label, this condition encapsulates local structural information from the input graph.

3.3.2 Exported Tangent-Space Embeddings

The mixed-curvature encoder produces two related but distinct latent quantities: the deterministic tangent representation u and the stochastic decoder-facing latent z. For latent diffusion, we export

$$x_0 \triangleq u(G) \in \mathbb{R}^d. \tag{11}$$

This exported embedding is the shared latent interface between mixed-curvature representation learning and diffusion-based generation. It carries the geometric inductive bias of the product manifold while living in a common tangent space where Gaussian perturbations and denoising dynamics are well defined. Further discussion is provided in Appendix C.2.

3.3.3 Conditional Denoising on the Latent Substrate

We define a variance-preserving diffusion process over the exported tangent-space embeddings x_0. Let $\{\beta_t\}_{t=1}^T$ be the noise schedule, $\alpha_t = 1 - \beta_t$, and $\bar{\alpha}_t = \prod_{s=1}^t \alpha_s$. The forward process is

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1 - \bar{\alpha}_t}\, \epsilon, \qquad \epsilon \sim \mathcal{N}(0, I). \tag{12}$$

The denoiser predicts the injected noise from the noisy embedding, timestep, and Ricci-derived condition:

$$\hat{\epsilon} = \epsilon_\varphi(x_t, t, \tilde{\mathbf{s}}), \tag{13}$$

where $\tilde{\mathbf{s}}$ is either the true condition or a null condition used for classifier-free guidance; additional CFG and sampling details are provided in Appendix C.4.

To inject curvature information into denoising, we use feature-wise linear modulation. Let $r^{(\ell)}$ be the hidden activation at denoising block ℓ after timestep embedding fusion. The Ricci-derived condition modulates this activation as

$$r^{(\ell)}_{\mathrm{FiLM}} = \big(1 + \gamma^{(\ell)}(\tilde{\mathbf{s}})\big) \odot r^{(\ell)} + \beta^{(\ell)}(\tilde{\mathbf{s}}), \tag{14}$$

where $\gamma^{(\ell)}(\cdot)$ and $\beta^{(\ell)}(\cdot)$ are lightweight conditioning networks. This modulation does not create an unrelated conditional branch. Instead, it transfers the same Ricci-derived topology prior used by the representation learner into the latent denoising dynamics.

The diffusion prior is trained with the standard ϵ-prediction objective:

$$\mathcal{L}_{\mathrm{diff}} = \mathbb{E}_{t,\epsilon}\Big[\big\|\epsilon - \epsilon_\varphi(x_t, t, \tilde{\mathbf{s}})\big\|_2^2\Big], \tag{15}$$

where $t \sim \mathcal{U}\{1, \ldots, T\}$ and x_t is given by Eq. (12).

For completeness, the reverse sampling distribution follows the standard DDPM parameterization with mean determined by the predicted clean embedding or predicted noise. The generated tangent-space embedding is decoded back to a graph following Appendix C.3. The diffusion training and sampling procedures are summarized in Algorithms 1 and 2.

4 Theoretical Analysis

4.1 Why Heterogeneous Graph Distributions Favor Mixed-Curvature Latent Geometry

Notation. For $\kappa \in \mathbb{R}$, let $\mathcal{M}_\kappa^d$ denote the d-dimensional simply-connected constant-curvature manifold: hyperbolic if $\kappa < 0$, Euclidean if $\kappa = 0$, and spherical if $\kappa > 0$. For a graph $G = (V, E)$, let d_G denote its shortest-path metric. For a metric space $(\mathcal{M}, d_{\mathcal{M}})$, let $D(G \to \mathcal{M})$ denote the scale-invariant bi-Lipschitz distortion of embedding $(V(G), d_G)$ into $\mathcal{M}$, and define the expected distortion risk under a graph distribution p as $\mathcal{R}(\mathcal{M}; p)$.

Theorem 4.1. *Let p be a heterogeneous graph distribution that mixes hierarchical (tree-like), grid-like, and cycle motifs, parametrized by a tree depth h, a grid side-length m, and a cycle length ℓ. Let $D(G \to \mathcal{M})$ be the scale-invariant bi-Lipschitz distortion of embedding the shortest-path metric of G into a metric space $\mathcal{M}$. Then for any fixed total dimension $d \geq 5$, there exist integers d_h, d_s, d_e with $d_h, d_e \geq 2$, $d_s \geq 1$, and $d_h + d_s + d_e = d$, together with curvatures $\kappa_h < 0 < \kappa_s$, such that, with*

$$\mathcal{M}_{\mathrm{mix}} = \mathbb{H}^{d_h}_{\kappa_h} \times \mathbb{S}^{d_s}_{\kappa_s} \times \mathbb{E}^{d_e}, \tag{16}$$

one has

$$\mathcal{R}(\mathcal{M}_{\mathrm{mix}}; p) \;<\; \inf_{\kappa \in \mathbb{R}} \mathcal{R}(\mathcal{M}_\kappa^d; p), \tag{17}$$

for sufficiently large (h, m, ℓ).

Moreover, the separation can be made arbitrarily large by increasing (h, m), while the cycle component remains explicitly incorporated through $p_C^{(\ell)}$ and the spherical factor.

Proofs are provided in Appendix D.2.

Theorem 4.1 formalizes the intuition that no single geometry is consistently adequate when graph distributions combine structurally different motif families, whereas a mixed-curvature product space can reduce representation-level distortion by allocating different motifs to suitable geometric factors. Under the stated graph-family assumptions, this result provides representation-level motivation, without establishing the general optimality of the fixed factorization or identifying when individual factors are necessary (Appendix D).

4.2 An Auxiliary Guarantee for Selecting Ricci Regularization

Theorem 4.2. *Let Λ be a finite set of candidate regularization weights such that $0 \in \Lambda$. For each $\lambda \in \Lambda$, let θ_λ denote the parameters obtained by training the model with regularization weight λ. Select the final weight by minimizing the validation estimate of the base metric:*

$$\hat{\lambda} \in \arg\min_{\lambda \in \Lambda} \widehat{M}_{\mathrm{val}}(\theta_\lambda). \tag{18}$$

Then the selected model is not worse than the unregularized baseline on the chosen validation metric:

$$\widehat{M}_{\mathrm{val}}(\theta_{\hat{\lambda}}) \;\le\; \widehat{M}_{\mathrm{val}}(\theta_0). \tag{19}$$

Proofs are provided in Appendix D.3.

Theorem 4.2 should be interpreted as an auxiliary validation-level non-inferiority guarantee. The practical effect of Ricci-derived structural priors is therefore assessed empirically through the main results, ablations, and supporting analyses.

5 Experiments

5.1 Experimental Setup

We evaluate CurvDiff on four generic unlabeled graph generation datasets, including **Ego**, **Grid**, **COLLAB**, and **IMDB-BINARY**; their structural statistics are summarized in Table 1. These datasets cover branching, grid-like, and densely clustered graph regimes, providing a suitable testbed for evaluating mixed-curvature latent generation. Here, structural heterogeneity refers to the coexistence and variation of local structural patterns. We further characterize its connectivity, local-transitivity, and motif-role dimensions using graph-level degree CV, clustering standard deviation, and normalized orbit entropy (Appendix E.2). We compare against representative graph generative baselines spanning classical latent-variable and autoregressive models, diffusion and flow-based methods, recent latent-space models, and geometry-aware graph generators. Generation quality is measured using MMD between the distributions of degree, clustering, and spectral statistics computed on generated and test graphs [Gretton et al., 2012], with `Avg.` defined as their arithmetic mean; lower values indicate better distributional matching. We additionally assess local motif-role fidelity using Orbit MMD from separately trained runs (Appendix E.12). For fair comparison, generated graph sizes are sampled from the corresponding reference split. Additional dataset details, baseline descriptions, implementation settings, and extended results are provided in Appendix E.

Table 1: Dataset statistics for structural characterization. These statistics are descriptive structural proxies rather than formal geometric labels or evaluation metrics.

Dataset	# Graphs	Avg. Nodes	Density	Leaf Ratio	Clustering	Square-cycle Share
Ego	1197	13.18	0.1978	0.5886	0.0879	0.2010
Grid	100	210.25	0.0192	0.0000	0.0000	0.5575
COLLAB	5000	74.49	0.5089	0.0066	0.8908	0.0081
IMDB-BINARY	1000	19.77	0.5206	0.0000	0.9471	0.0002

5.2 Main Results on Graph Generation

Overall graph generation performance. We first evaluate the overall generation quality of CurvDiff on four generic graph generation datasets. As shown in Table 2, CurvDiff achieves the best **Avg.** MMD on all four datasets, demonstrating consistently strong structural fidelity across both synthetic and real-world graphs. It obtains the best **Deg.** scores across all datasets and remains competitive on **Clu.** and **Spe.**, although several baselines perform better on specific individual statistics. These results suggest that CurvDiff provides a more balanced structural match than Euclidean, single-curvature, and recent latent graph generative baselines. In the supplementary Orbit evaluation, CurvDiff obtains the lowest MMD among the evaluated internal variants on all four datasets and outperforms GGBall on three, with COLLAB as the exception. In a separate supplementary comparison with DeFoG and Cometh, CurvDiff obtains the lowest three-statistic `Avg.` MMD on all four datasets, whereas DeFoG gives the lowest Orbit MMD among these three models (Appendix E.14). Additional benchmark results are reported in Appendix E.6 (Table 7).

Table 2: MMD comparison on four generic graph generation datasets.

Method	Synthetic Datasets								Real-world Datasets							
	Ego				Grid				COLLAB				IMDB-BINARY			
	Deg.	Clu.	Spe.	**Avg.** ↓	Deg.	Clu.	Spe.	**Avg.** ↓	Deg.	Clu.	Spe.	**Avg.** ↓	Deg.	Clu.	Spe.	**Avg.** ↓
VGAE†	0.414	0.156	0.456	0.342	0.050	2.000	0.145	0.732	0.418	1.174	0.700	0.764	0.514	1.405	0.700	0.873
GraphRNN†	0.206	0.539	0.157	0.301	0.203	0.043	0.042	0.096	0.044	0.036	0.510	0.197	0.137	0.252	0.423	0.271
GDSS†	1.034	0.143	0.667	0.615	0.111	**0.005**	0.886	0.334	0.773	1.589	0.502	0.955	0.904	1.729	0.748	1.127
DiGress†	0.110	0.056	0.122	0.096	0.689	1.115	0.203	0.669	0.022	0.008	**0.003**	0.011	0.166	0.425	0.159	0.250
GraphGDP†	0.059	0.115	0.054	0.076	0.872	1.001	0.174	0.682	0.092	0.291	0.138	0.174	0.123	0.638	0.257	0.339
EDGE†	0.023	0.048	0.023	0.031	0.223	0.072	0.802	0.366	0.023	0.569	0.034	0.209	0.041	0.874	0.026	0.314
LGD	0.096	0.041	0.112	0.083	0.786	0.000	0.681	0.489	0.073	0.643	0.109	0.275	0.102	0.222	0.094	0.139
GLAD	0.310	0.024	0.007	0.114	1.996	0.000	**0.040**	0.679	0.260	0.914	0.004	0.393	0.819	1.655	**0.011**	0.828
LG-Flow	0.009	0.050	0.016	0.025	0.408	0.814	0.125	0.449	0.024	0.009	0.046	0.026	0.043	0.051	0.040	0.045
HypDiff†	0.075	0.090	0.015	0.060	0.422	0.665	0.137	0.408	0.016	0.023	0.009	0.016	0.034	0.257	0.030	0.107
HGDM	0.638	**0.014**	0.016	0.223	1.548	0.042	0.841	0.810	0.298	1.014	0.191	0.501	0.297	0.150	0.023	0.157
GGBall	0.567	0.090	0.140	0.266	0.408	1.762	0.574	0.915	0.093	0.055	0.127	0.092	0.129	0.402	0.149	0.227
CurvDiff	**0.002**	0.028	**0.006**	**0.012**	**0.046**	0.015	0.049	**0.037**	**0.008**	**0.004**	0.008	**0.007**	**0.033**	**0.045**	0.044	**0.041**

Note: † denotes results taken from HypDiff [Fu et al., 2024]; other results are obtained by our own runs under the same protocol. Zero-valued entries are not highlighted, as they may indicate degenerate generation. Lower is better for all metrics.

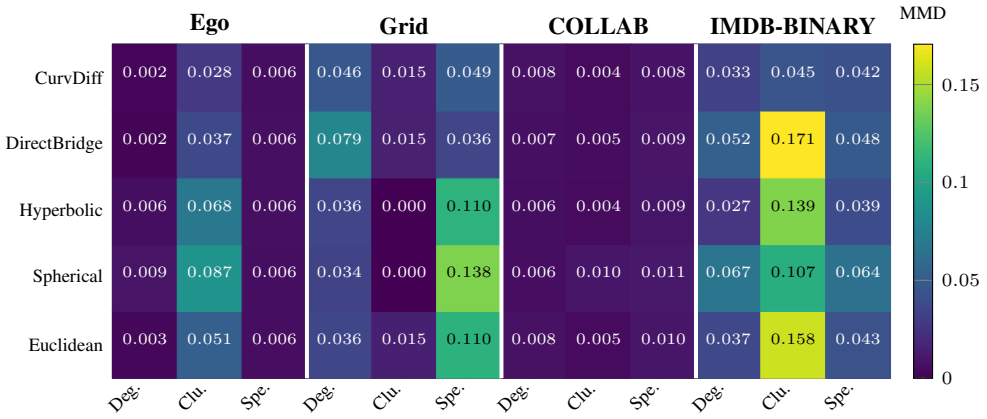


Figure 3: Per-metric MMD heatmap for geometry ablations. Darker cells generally correspond to lower MMD under the current color scale.

5.3 Mechanism Analysis I: What does mixed curvature contribute?

Geometry ablation. We compare CurvDiff with Euclidean, Spherical, Hyperbolic, and DirectBridge variants to test whether a single latent geometry remains reliable across branching, grid-like, and densely clustered graph regimes. As shown in Fig. 3, CurvDiff achieves the best averaged MMD on **Ego**, **Grid**, and **IMDB-BINARY**, and remains close to the best result on **COLLAB**, where the Hyperbolic variant is slightly better. The COLLAB **Avg.** gap is 0.00030, mainly driven by the Hyperbolic variant's lower degree MMD; clustering is nearly tied, while CurvDiff has lower spectral MMD. The supplementary Orbit comparison also favors CurvDiff, indicating metric-dependent trade-offs (Appendix E.13). Although single-geometry variants can be locally competitive, their advantages are not consistent across datasets or across **Deg.**, **Clu.**, and **Spe.**; in contrast, CurvDiff provides a more balanced trade-off, supporting mixed-curvature latent representations for heterogeneous graph generation. Standalone spherical performance does not determine its utility within a jointly learned product representation. A dimension-matched comparison of $H + E$ and $H + S + E$ yields lower average MMD for the three-factor model on Ego, Grid, and IMDB-BINARY, with nearly tied results on COLLAB (Appendix E.9). Additional average-MMD comparisons and qualitative visualizations are provided in Appendix E.8.

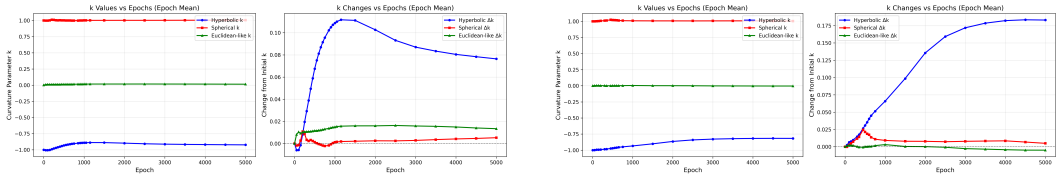
5.4 Mechanism Analysis II: Why does the unified latent interface matter?

Unified interface ablation. We use structured variants to isolate the effects of mixed-curvature modeling, encoder-side Ricci priors, diffusion-side Ricci conditioning, and the shared encoder–

Table 3: Component ablation for evaluating the unified latent interface, reported as **Avg. MMD**.

Method	Mixed	Ricci-E	Ricci-D	Shared	Ego	Grid	IMDB-BINARY
Euclidean + No Ricci	×	×	×	✓	0.0357	0.1864	0.0967
Mixed Only	✓	×	×	✓	0.0350	0.1723	0.1013
Euclidean + Full Ricci	×	✓	✓	✓	0.0169	0.0538	0.0766
Ricci Encoder Only	✓	✓	×	✓	0.0222	0.0582	<u>0.0754</u>
Ricci Diffusion Only	✓	×	✓	✓	0.0359	0.2465	0.1010
Mixed + Ricci w/o Shared	✓	✓	✓	×	<u>0.0151</u>	<u>0.0432</u>	0.0866
CurvDiff	✓	✓	✓	✓	**0.0120**	**0.0367**	**0.0407**

Note: ✓ / × denote enabled/disabled components. Mixed, Ricci-E, Ricci-D, and Shared refer to the mixed-curvature representation, encoder-side Ricci prior, diffusion-side Ricci conditioning, and shared latent interface, respectively. **Euclidean + Full Ricci** retains both Ricci pathways and the shared interface while fixing all latent curvatures to zero.



(a) **Ego** (b) **IMDB-BINARY**

Figure 4: Adaptive evolution of learnable curvature parameters.

diffusion latent interface. As shown in Table 3, CurvDiff achieves the best **Avg.** MMD on all three representative datasets, while the strongest partial variant varies across datasets. **Mixed Only** remains close to **Euclidean + No Ricci** and is slightly worse on **IMDB-BINARY**, indicating that mixed curvature alone does not yield uniform gains. With both Ricci pathways and the shared interface held fixed, CurvDiff achieves lower **Avg.** MMD than **Euclidean + Full Ricci** on all three datasets, supporting an additional benefit from mixed curvature in the full configuration. In particular, **Ricci Encoder Only** improves over **Mixed Only**, whereas **Ricci Diffusion Only** does not provide consistent gains, suggesting that Ricci signals are more effective when grounded in the learned representation. Moreover, **Mixed + Ricci w/o Shared** is competitive on **Ego** and **Grid** but still underperforms CurvDiff, especially on **IMDB-B**, indicating the importance of preserving the shared latent interface. Further per-metric results and the seven-dataset controlled comparison are reported in Appendices E.10 and E.11, respectively.

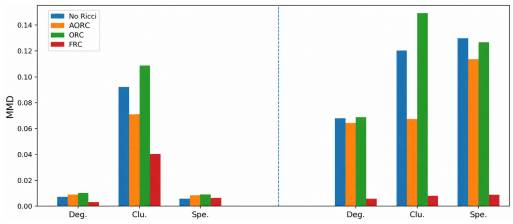
5.5 Additional Analyses

5.5.1 Adaptive Curvature Dynamics

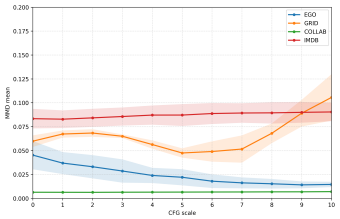
To examine whether the adaptive mixed-curvature mechanism is utilized during training, we track the learnable curvature parameters on **Ego** and **IMDB-BINARY**. As shown in Fig. 4, the curvature branches remain separated throughout training: the hyperbolic branch stays negative, the spherical branch stays positive, and the Euclidean-like branch remains near zero. Their trajectories also differ across datasets, with a larger and more persistent hyperbolic shift on **IMDB-BINARY** than on **Ego**. These observations suggest that the curvature branches do not collapse into a redundant parameterization, but retain dataset-dependent geometric roles.

5.5.2 Ricci Choice Analysis

We compare ORC, AORC, FRC, and a no-Ricci baseline while keeping all other components unchanged. AORC is a lightweight ORC approximation defined in Appendix B.2. As shown in Fig. 5a, FRC performs best overall on both **Ego** and **COLLAB**: it gives the lowest degree and clustering MMD on **Ego**, remains competitive on spectral MMD, and substantially reduces all three discrepancies on **COLLAB**. In contrast, ORC does not provide consistent gains and AORC remains weaker than FRC. Qualitative curvature case studies are provided in Appendix E.16.



(a) Ricci curvature choices.



(b) CFG-scale sensitivity.

Figure 5: Ablation analyses measured by MMD. Left: Ricci curvature choices on **Ego** and **COLLAB**. Right: CFG-scale sensitivity measured by **Avg.** MMD over five seeds.

5.5.3 Guidance-scale Sensitivity

We study classifier-free guidance (CFG) at sampling time with the trained checkpoint fixed for each seed, and use a fixed guidance scale of 3.0 in the main results to avoid dataset-specific tuning. As shown in Fig. 5b, the preferred guidance scale varies across datasets: **Ego** benefits from stronger guidance around 9–10, **Grid** favors a moderate scale around 5, **IMDB-BINARY** obtains its lowest mean under weak guidance around 1, and **COLLAB** is largely insensitive to the tested range. These results indicate that CFG is a dataset-dependent inference-time control knob. Component-wise guidance-scale results are provided in Appendix E.17.

5.5.4 Runtime Analysis

On an RTX 4090, CurvDiff's sampling stage takes 2.06–2.26 s per batch of 20 graphs across the four main datasets (103–113 ms per graph, amortized over the batch). Reverse diffusion dominates this cost; detailed sampling timings and one-time Ricci-preprocessing measurements are provided in Appendix E.15.

6 Conclusion

We proposed CurvDiff, a geometry–topology-aware latent generative framework for generic unlabeled graph generation. Rather than treating mixed-curvature representation learning and Ricci-conditioned diffusion as separate modules, CurvDiff links mixed-curvature embeddings and Ricci-derived structural priors through a shared tangent-space interface. This interface makes product-manifold representations compatible with latent diffusion while preserving topology-aware guidance across representation learning and generation. Experiments on representative benchmarks show that CurvDiff achieves competitive aggregate MMD-based structural fidelity, with improvements that depend on the dataset and graph statistic. Future work will extend this framework beyond structure-only generation to attributed and domain-specific graphs, such as molecular generation.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No.62432006, 62276159) and the Fundamental Research Program of Shanxi Province (No. 202303021223004). The authors declare no competing interests.

References

Marián Boguñá, Ivan Bonamassa, Manlio De Domenico, Shlomo Havlin, Dmitri Krioukov, and M. Ángeles Serrano. Network geometry. *Nature Reviews Physics*, 3(2), 2021.

Avishek Joey Bose, Ariella Smofsky, Renjie Liao, Prakash Panangaden, and William L. Hamilton. Latent variable modelling with hyperbolic normalizing flows. In *International Conference on Machine Learning*, 2020.

Michael M Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *CoRR*, abs/2104.13478, 2021.

Tianci Bu, Chuanrui Wang, Hao Ma, Haoren Zheng, Xin Lu, and Tailin Wu. GGBall: Graph generative model on Poincaré ball. *CoRR*, abs/2506.07198, 2025.

Xiaohui Chen, Jiaxing He, Xu Han, and Liping Liu. Efficient and degree-guided graph generation via discrete diffusion modeling. In *International Conference on Machine Learning*, 2023.

Lukas Fesser and Melanie Weber. Mitigating over-smoothing and over-squashing using augmentations of Forman-Ricci curvature. In *Learning on Graphs Conference*, 2024.

Lukas Fesser, Sergio Serrano de Haro Iváñez, Karel Devriendt, Melanie Weber, and Renaud Lambiotte. Augmentations of Forman's Ricci curvature and their applications in community detection. *CoRR*, abs/2306.06474, 2023.

Robin Forman. Bochner's method for cell complexes and combinatorial Ricci curvature. *Discrete & Computational Geometry*, 29(3), 2003.

Xingcheng Fu, Yisen Gao, Yuecen Wei, Qingyun Sun, Hao Peng, Jianxin Li, and Xianxian Li. Hyperbolic geometric latent diffusion model for graph generation. In *International Conference on Machine Learning*, 2024.

Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic neural networks. In *Advances in Neural Information Processing Systems*, 2018.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(1), 2012.

Albert Gu, Frederic Sala, Beliz Gunel, and Christopher Ré. Learning mixed-curvature representations in product spaces. In *International Conference on Learning Representations*, 2019.

Han Huang, Leilei Sun, Bowen Du, Yanjie Fu, and Weifeng Lv. GraphGDP: Generative diffusion processes for permutation invariant graph generation. In *IEEE International Conference on Data Mining*, 2022.

Jaehyeong Jo, Seul Lee, and Sung Ju Hwang. Score-based generative modeling of graphs via the system of stochastic differential equations. In *International Conference on Machine Learning*, 2022.

Jürgen Jost and Shiping Liu. Ollivier's Ricci curvature, local clustering and curvature-dimension inequalities on graphs. *Discrete & Computational Geometry*, 51(2), 2014.

Thomas N. Kipf and Max Welling. Variational graph auto-encoders. *CoRR*, abs/1611.07308, 2016.

Yong Lin, Linyuan Lu, and Shing-Tung Yau. Ricci curvature of graphs. *Tohoku Mathematical Journal*, 63(4), 2011.

Tianze Luo, Zhanfeng Mo, and Sinno Jialin Pan. Fast graph generation via spectral diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(5), 2024.

Karolis Martinkus, Andreas Loukas, Nathanaël Perraudin, and Roger Wattenhofer. SPECTRE: Spectral conditioning helps to overcome the expressivity limits of one-shot graph generators. In *International Conference on Machine Learning*, 2022.

Emile Mathieu, Charline Le Lan, Chris J. Maddison, Ryota Tomioka, and Yee Whye Teh. Continuous hierarchical representations with Poincaré variational auto-encoders. In *Advances in Neural Information Processing Systems*, 2019.

Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri Chklovskii, and Uri Alon. Network motifs: simple building blocks of complex networks. *Science*, 298(5594), 2002.

Van Khoa Nguyen, Yoann Boget, Frantzeska Lavda, and Alexandros Kalousis. GLAD: improving latent graph generative modeling with simple quantization. In *AAAI Conference on Artificial Intelligence*, 2025.

Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. *Advances in Neural Information Processing Systems*, 2017.

Yann Ollivier. Ricci curvature of Markov chains on metric spaces. *Journal of Functional Analysis*, 256(3), 2009.

Theodore Papamarkou, Tolga Birdal, Michael M Bronstein, Gunnar E Carlsson, Justin Curry, Yue Gao, Mustafa Hajij, Roland Kwitt, Pietro Lio, Paolo Di Lorenzo, et al. Position: Topological deep learning is the new frontier for relational learning. In *International Conference on Machine Learning*, 2024.

Yiming Qin, Manuel Madeira, Dorina Thanou, and Pascal Frossard. DeFoG: Discrete flow matching for graph generation. In *International Conference on Machine Learning*, 2025.

Erzsébet Ravasz and Albert-László Barabási. Hierarchical organization in complex networks. *Physical Review E*, 67(2), 2003.

Frederic Sala, Chris De Sa, Albert Gu, and Christopher Ré. Representation tradeoffs for hyperbolic embeddings. In *International Conference on Machine Learning*, 2018.

Antoine Siraudin and Christopher Morris. Principled latent diffusion for graphs via laplacian autoencoders. *CoRR*, abs/2601.13780, 2026.

Antoine Siraudin, Fragkiskos D. Malliaros, and Christopher Morris. Cometh: A continuous-time discrete-state graph diffusion model. *Transactions on Machine Learning Research*, 2025.

Ondrej Skopek, Octavian-Eugen Ganea, and Gary Bécigneul. Mixed-curvature variational autoencoders. In *International Conference on Learning Representations*, 2020.

R. P. Sreejith, Karthikeyan Mohanraj, Jürgen Jost, Emil Saucan, and Areejit Samal. Forman curvature for complex networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2016(6), 2016.

Li Sun, Zhongbao Zhang, Junda Ye, Hao Peng, Jiawei Zhang, Sen Su, and Philip S. Yu. A self-supervised mixed-curvature graph neural network. In *AAAI Conference on Artificial Intelligence*, 2022.

Jake Topping, Francesco Di Giovanni, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M. Bronstein. Understanding over-squashing and bottlenecks on graphs via curvature. In *International Conference on Learning Representations*, 2022.

Clément Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. DiGress: Discrete denoising diffusion for graph generation. In *International Conference on Learning Representations*, 2023a.

Clément Vignac, Nagham Osman, Laura Toni, and Pascal Frossard. MiDi: Mixed graph and 3D denoising diffusion for molecule generation. In *Machine Learning and Knowledge Discovery in Databases: Research Track*, 2023b.

Shen Wang, Xiaokai Wei, Cícero Nogueira dos Santos, Zhiguo Wang, Ramesh Nallapati, Andrew O. Arnold, Bing Xiang, Philip S. Yu, and Isabel F. Cruz. Mixed-curvature multi-relational graph neural network for knowledge graph completion. In *ACM Web Conference*, 2021.

Yujie Wang, Shuo Zhang, Junda Ye, Hao Peng, and Li Sun. A mixed-curvature graph diffusion model. In *ACM International Conference on Information and Knowledge Management*, 2024.

Lingfeng Wen, Xuan Tang, Mingjie Ouyang, Xiangxiang Shen, Jian Yang, Daxin Zhu, Mingsong Chen, and Xian Wei. Hyperbolic graph diffusion model. In *AAAI Conference on Artificial Intelligence*, 2024.

Zuoyu Yan, Qi Zhao, Ze Ye, Tengfei Ma, Liangcai Gao, Zhi Tang, Yusu Wang, and Chao Chen. Enhancing graph representation learning with localized topological features. *Journal of Machine Learning Research*, 26 (5), 2025.

Ling Yang, Zhilin Huang, Zhilong Zhang, Zhongyi Liu, Shenda Hong, Wentao Zhang, Wenming Yang, Bin Cui, and Luxia Zhang. Graphusion: Latent diffusion for graph generation. *IEEE Transactions on Knowledge and Data Engineering*, 36(11), 2024.

Jiaxuan You, Rex Ying, Xiang Ren, William L. Hamilton, and Jure Leskovec. GraphRNN: Generating realistic graphs with deep auto-regressive models. In *International Conference on Machine Learning*, 2018.

Cai Zhou, Xiyuan Wang, and Muhan Zhang. Unifying generation and prediction on graphs with latent graph diffusion. In *Advances in Neural Information Processing Systems*, 2024.

A Geometric Operations on Constant-Curvature Manifolds

For completeness, we collect the standard constant-curvature operations used by CurvDiff. These operations are applied component-wise in the mixed-curvature product latent space to move between manifold representations and Euclidean tangent spaces. We use the c-Poincaré ball model for the hyperbolic component, i.e.,

$$\mathbb{B}_c^d = \{\mathbf{x} \in \mathbb{R}^d : \ c\|\mathbf{x}\|^2 < 1\},$$

which has sectional curvature $-c$ for $c > 0$. The associated conformal factor is

$$\lambda_{\mathbf{x}} = \frac{2}{1 - c\|\mathbf{x}\|^2}. \tag{20}$$

The Euclidean case is recovered in the limit $c \to 0$. For the spherical component, CurvDiff uses the analogous positively curved stereographic formulas.

A.1 Exponential and Logarithmic Maps

The exponential and logarithmic maps provide the interface between the manifold and its tangent spaces. In CurvDiff, they are used to perform Euclidean operations in tangent spaces and then map the resulting representations back to the corresponding manifold factors.

At the origin, for $\mathbf{v} \in T_{\mathbf{0}}\mathbb{B}_c^d \cong \mathbb{R}^d$ and $\mathbf{x} \in \mathbb{B}_c^d$,

$$\exp_{\mathbf{0}}^c(\mathbf{v}) = \tanh\big(\sqrt{c}\,\|\mathbf{v}\|\big)\,\frac{\mathbf{v}}{\sqrt{c}\,\|\mathbf{v}\|}, \qquad \log_{\mathbf{0}}^c(\mathbf{x}) = \operatorname{artanh}\big(\sqrt{c}\,\|\mathbf{x}\|\big)\,\frac{\mathbf{x}}{\sqrt{c}\,\|\mathbf{x}\|}. \tag{21}$$

For an arbitrary base point $\mathbf{x}$, the maps are

$$\exp_{\mathbf{x}}^c(\mathbf{v}) = \mathbf{x} \oplus_c \left(\tanh\Big(\frac{\sqrt{c}\,\lambda_{\mathbf{x}}\,\|\mathbf{v}\|}{2}\Big)\,\frac{\mathbf{v}}{\sqrt{c}\,\|\mathbf{v}\|}\right), \tag{22}$$

for $\mathbf{v} \in T_{\mathbf{x}}\mathbb{B}_c^d$, and

$$\log_{\mathbf{x}}^c(\mathbf{y}) = \frac{2}{\sqrt{c}\,\lambda_{\mathbf{x}}}\,\operatorname{artanh}\big(\sqrt{c}\,\|(-\mathbf{x}) \oplus_c \mathbf{y}\|\big)\,\frac{(-\mathbf{x}) \oplus_c \mathbf{y}}{\|(-\mathbf{x}) \oplus_c \mathbf{y}\|}, \tag{23}$$

for $\mathbf{y} \in \mathbb{B}_c^d$. We use the convention $\mathbf{u}/\|\mathbf{u}\| = \mathbf{0}$ when $\mathbf{u} = \mathbf{0}$.

As $c \to 0$, these maps reduce to the Euclidean expressions

$$\exp_{\mathbf{x}}^c(\mathbf{v}) \to \mathbf{x} + \mathbf{v}, \qquad \log_{\mathbf{x}}^c(\mathbf{y}) \to \mathbf{y} - \mathbf{x}.$$

For the spherical component with positive curvature $+c$ under stereographic projection, the corresponding formulas use $(\tan, \arctan)$ in place of $(\tanh, \operatorname{artanh})$ and

$$\lambda_{\mathbf{x}} = \frac{2}{1 + c\|\mathbf{x}\|^2}.$$

A.2 Parallel Transport

Parallel transport moves a tangent vector between tangent spaces while preserving the Riemannian inner product. In CurvDiff, it is used when tangent vectors associated with different base points must be compared or combined consistently across manifold factors.

Define the gyration operator

$$\operatorname{gyr}[\mathbf{a}, \mathbf{b}]\,\mathbf{v} = \ominus_c(\mathbf{a} \oplus_c \mathbf{b}) \oplus_c \big(\mathbf{a} \oplus_c (\mathbf{b} \oplus_c \mathbf{v})\big). \tag{24}$$

Then the parallel transport of $\mathbf{v} \in T_{\mathbf{x}}\mathbb{B}_c^d$ from $\mathbf{x}$ to $\mathbf{y}$ is

$$\mathrm{PT}_{\mathbf{x}\to\mathbf{y}}^c(\mathbf{v}) = \frac{\lambda_{\mathbf{x}}}{\lambda_{\mathbf{y}}}\,\operatorname{gyr}[\mathbf{y}, \ominus_c\mathbf{x}]\,\mathbf{v}. \tag{25}$$

A useful special case is transport from the origin:

$$\mathrm{PT}_{\mathbf{0}\to\mathbf{x}}^c(\mathbf{v}) = \frac{\lambda_{\mathbf{0}}}{\lambda_{\mathbf{x}}}\,\mathbf{v} = \big(1 - c\|\mathbf{x}\|^2\big)\,\mathbf{v}, \qquad \lambda_{\mathbf{0}} = 2. \tag{26}$$

All of the above operations are applied independently to the hyperbolic, Euclidean, and spherical factors in the mixed-curvature latent space.

B Discrete Ricci Curvatures on Graphs

This appendix summarizes three edge-based curvature quantities used in our study: the exact Ollivier–Ricci curvature (ORC), a bound-averaged approximation to ORC (AORC), and the Forman–Ricci curvature (FRC). In all cases, edge-level curvature can be aggregated to obtain node-level curvature.

B.1 Ollivier–Ricci Curvature (ORC)

Let $G = (V, E)$ be an unweighted simple graph equipped with the shortest-path metric $d(\cdot, \cdot)$. For each node $i \in V$, let $\mathcal{N}(i)$ denote its neighbor set and let $d_i = |\mathcal{N}(i)|$ be its degree. Following the standard non-lazy definition of Ollivier–Ricci curvature on graphs [Ollivier, 2009], we assign to node i the uniform probability measure

$$m_i(k) = \begin{cases} \frac{1}{d_i}, & k \in \mathcal{N}(i), \\ 0, & \text{otherwise.} \end{cases} \tag{27}$$

For an edge $(i, j) \in E$, the Ollivier–Ricci curvature is defined as

$$\kappa_{ij}^{\mathrm{ORC}} = 1 - \frac{W_1(m_i, m_j)}{d(i, j)}, \tag{28}$$

B.2 Bound-Averaged Approximation to Ollivier–Ricci Curvature (AORC)

Computing $\kappa_{ij}^{\mathrm{ORC}}$ exactly requires solving a local optimal transport problem. To obtain a lightweight surrogate, we use the lower and upper bounds derived for ORC in terms of local triangle counts by Jost and Liu [2014]. For an edge $(i, j) \in E$, define

$$T_{ij} = |\mathcal{N}(i) \cap \mathcal{N}(j)|, \tag{29}$$

which equals the number of triangles incident to edge (i, j). We further define

$$d_{ij}^{\min} = \min(d_i, d_j), \qquad d_{ij}^{\max} = \max(d_i, d_j), \tag{30}$$

and use the shorthand $[x]_+ = \max(x, 0)$.

Following Jost and Liu [2014], we write the lower and upper bounds as

$$\underline{\kappa}_{ij}^{\mathrm{ORC}} = -\Big[1 - \frac{1}{d_i} - \frac{1}{d_j} - \frac{T_{ij}}{d_{ij}^{\min}}\Big]_+ - \Big[1 - \frac{1}{d_i} - \frac{1}{d_j} - \frac{T_{ij}}{d_{ij}^{\max}}\Big]_+ + \frac{T_{ij}}{d_{ij}^{\max}}, \tag{31}$$

and

$$\overline{\kappa}_{ij}^{\mathrm{ORC}} = \frac{T_{ij}}{d_{ij}^{\max}}. \tag{32}$$

We define the approximated edge curvature as the midpoint of these two bounds:

$$\kappa_{ij}^{\mathrm{AORC}} = \frac{1}{2}\left(\underline{\kappa}_{ij}^{\mathrm{ORC}} + \overline{\kappa}_{ij}^{\mathrm{ORC}}\right). \tag{33}$$

This approximation preserves the triangle-aware structural bias of ORC while avoiding explicit optimal transport computation.

B.3 Forman–Ricci Curvature (FRC)

Forman–Ricci curvature is a combinatorial notion of discrete curvature introduced for cell complexes by Forman [2003] and later adapted to complex networks by Sreejith et al. [2016]. For an unweighted simple graph, we use the commonly adopted edge curvature

$$\kappa_{ij}^{\mathrm{FRC}} = 4 - d_i - d_j, \qquad (i, j) \in E. \tag{34}$$

More general formulations exist for weighted graphs and higher-order cell complexes, but the above expression is widely used in graph learning due to its simplicity and efficiency.

B.4 Node-level Curvature Aggregation

Given any edge curvature $\kappa_{ij}^{(\cdot)}$, its node-level counterpart is obtained by averaging over incident edges. For node i, we define

$$\kappa_i^{(\cdot)} = \frac{1}{d_i} \sum_{j \in \mathcal{N}(i)} \kappa_{ij}^{(\cdot)}, \tag{35}$$

where $(\cdot) \in \{\mathrm{ORC}, \mathrm{AORC}, \mathrm{FRC}\}$. For isolated nodes, $\kappa_i^{(\cdot)}$ is set to 0 by convention.

B.5 Computational Cost and Practical Choice

The three Ricci curvature choices have different computational costs. For a graph with m edges, exact ORC solves a local optimal transport problem for every edge; its cost depends on neighborhood sizes and the chosen optimal-transport solver, and the OTD implementation used here is evaluated empirically. AORC avoids optimal transport and instead relies on local degree and triangle-count information; its main cost is computing common-neighbor counts for each edge, which has expected cost $O(\min(d_i, d_j))$ per edge with hash-based neighbor sets and approximately $O(\sum_{(i,j)\in E} \min(d_i, d_j))$ in total, after which each AORC value is obtained in constant time. This estimate assumes sparse neighbor-set processing; our AORC implementation additionally materializes and scans a dense adjacency matrix, incurring $O(|V|^2)$ time and memory overhead. FRC admits closed-form evaluation: after node degrees are computed, each edge curvature is obtained in constant time, giving $O(m)$ cost for edge evaluation and $O(|V| + |E|)$ time and memory for the full sparse feature pipeline, including degree computation and node aggregation. Measured preprocessing costs are reported in Appendix E.15. We use FRC as the default Ricci signal in CurvDiff because it provides a scalable closed-form structural prior while still retaining topology-relevant information for representation learning and diffusion conditioning.

C Additional Methodological Details

C.1 Notation Summary

Table 4 summarizes the main symbols used in the methodology and theoretical analysis. Repeated symbols are interpreted within their stated contexts.

In Appendix B, $m_i(\cdot)$ denotes a neighbor probability measure, whereas the mask entry m_i is a binary indicator; m in computational-cost expressions denotes the number of edges. In the hyperbolic formulas of Appendix A, $c > 0$ denotes curvature magnitude, giving sectional curvature $-c$.

C.2 Why Diffusion Operates on Deterministic Latent Representations

The stochastic variable $z \sim q_\phi(z \mid G)$ introduces posterior noise that is useful for regularized reconstruction, but this noise can obscure the deterministic structural identity of each graph. In contrast, $u(G)$ provides one stable mixed-curvature tangent representation per graph and therefore forms a cleaner empirical distribution for latent diffusion.

For this reason, the diffusion prior is trained over exported embeddings $x_0 = u(G)$. The posterior heads f_μ and f_σ are retained as a bridge between the generated tangent-space embedding and the decoder-facing latent space. This separation clarifies the role of each variable: u defines the shared mixed-curvature tangent representation, x_0 denotes the same representation as the diffusion target, and z remains the stochastic latent used by the decoder.

C.3 Decoding Generated Embeddings Back to Graphs

After latent diffusion generates an embedding $\hat{x}_0$, it is mapped back to the decoder-facing latent space through the frozen posterior heads, ensuring compatibility with the reconstruction-trained decoder:

$$\hat{\mu} = f_\mu(\hat{x}_0), \qquad \hat{\log \sigma} = f_\sigma(\hat{x}_0). \tag{36}$$

For the main sampling path, we use the posterior mean as the decoder input, i.e., $\hat{z} = \hat{\mu}$.

Table 4: Notation summary.

Symbol	Meaning
A, X, m	Ordered and padded adjacency matrix, node-feature matrix, and binary node-validity mask.
$N_{\max}, F, d_\kappa$	Maximum padded node count, node-feature dimension, and Ricci-feature dimension.
κ (methodology)	Matrix of node-level Ricci-derived features used for feature augmentation and structural conditioning.
$\mathcal{M}, c_b$	Product latent manifold and signed curvature of its b-th factor.
B, d_b, d'	Number of manifold factors, factor dimensions, and their total dimension, with $\sum_{b=1}^{B} d_b = d'$.
$u(G), x_0$	Deterministic exported tangent representation and diffusion target, with $x_0 = u(G)$.
z, μ, σ	Stochastic reconstruction latent and its Gaussian posterior mean and standard deviation.
x_t, t, T, ϵ	Noisy tangent embedding, diffusion timestep, total number of steps, and standard Gaussian noise.
$\mathbf{s}(G), \tilde{\mathbf{s}}$	Graph-level Ricci-derived condition and its condition-dropped version used for classifier-free guidance.
ϵ_φ	Conditional noise-prediction network.
$\beta_t, \alpha_t, \bar{\alpha}_t$	Diffusion noise schedule and associated factors: $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$.
$G = (V, E), d_G$	Graph in vertex–edge notation and its shortest-path metric.
$\kappa, \mathcal{M}_\kappa^d$ (theory)	Scalar signed manifold curvature and the d-dimensional constant-curvature manifold; this κ differs from the Ricci-feature matrix.
h, m, ℓ (Theorem 4.1)	Tree depth, grid side length, and cycle length; this m differs from the node-validity mask.
$p, \alpha_A, \alpha_B, \alpha_C$	Heterogeneous graph distribution and its tree, grid, and cycle mixture weights.
$D(G \to \mathcal{M})$	Scale-invariant bi-Lipschitz embedding distortion.
$\mathcal{R}(\mathcal{M}; p)$	Expected embedding distortion under graph distribution p.
$\lambda, \Lambda, \hat{\lambda}$	Regularization weight, finite candidate set, and validation-selected weight.
$\theta_\lambda, \widehat{M}_{\mathrm{val}}$	Parameters trained with weight λ and the validation estimate of the base metric.

The frozen decoder then converts $\hat{z}$ into adjacency probabilities $\hat{A}$ via a sigmoid transformation of the decoder output, from which a discrete graph is sampled by Bernoulli edge sampling. This bridge maintains compatibility between the deterministic diffusion substrate and the reconstruction-trained decoder.

C.4 Classifier-Free Guidance and Graph Generation

To support classifier-free guidance, the Ricci-derived condition is randomly dropped during diffusion training:

$$\tilde{\mathbf{s}} = \begin{cases} \mathbf{0}, & \text{with probability } p_{\mathrm{drop}}, \\ \mathbf{s}, & \text{otherwise.} \end{cases} \tag{37}$$

At sampling time, unconditional and conditional denoising predictions are combined as

$$\hat{\epsilon} = \epsilon_\varphi(x_t, t, \mathbf{0}) + s_{\mathrm{cfg}}\Big(\epsilon_\varphi(x_t, t, \mathbf{s}) - \epsilon_\varphi(x_t, t, \mathbf{0})\Big), \tag{38}$$

where $s_{\mathrm{cfg}} \geq 0$ controls the strength of topology-aware guidance.

Starting from $x_T \sim \mathcal{N}(0, I)$, the reverse process iteratively denoises toward a generated tangent-space embedding $\hat{x}_0$ under the sampled Ricci-derived condition. The resulting embedding is decoded into a discrete graph following Appendix C.3. Overall, graph generation follows a single latent pathway in which Ricci-derived statistics guide denoising on the shared tangent-space interface.

C.5 Algorithm

We summarize the diffusion-side training and sampling procedures of CurvDiff. The model learns a Ricci-conditioned diffusion prior over the exported mixed-curvature tangent embeddings and decodes generated embeddings through the frozen posterior heads and decoder.

Algorithm 1 CurvDiff Training

Input: Training graphs $\{G\}$.
Parameters: Denoiser parameters φ.
Output: Trained denoiser and exported embeddings $\{x_0(G)\}$.
for each graph G **do**
  Compute $\mathbf{s}(G)$ using Eq. (10).
  Export embedding $x_0(G)$ using Eq. (11).
end for
for each training iteration **do**
  Sample $(x_0(G), \mathbf{s}(G))$, timestep t, and noise ϵ.
  Obtain embedding x_t using Eq. (12).
  Predict noise using Eq. (13).
  Compute the loss $\mathcal{L}_{\text{diff}}$ using Eq. (15) and update φ.
end for

Algorithm 2 CurvDiff Sampling

Input: Ricci condition source, diffusion schedule.
Output: Generated adjacency matrix $\hat{A}$.
Sample a Ricci-derived condition vector $\mathbf{s}$.
Initialize x_T from Gaussian noise.
for reverse diffusion timesteps **do**
  Compute the classifier-free guided noise prediction.
  Apply one reverse DDPM update.
end for
Obtain the generated tangent embedding $\hat{x}_0$.
Map $\hat{x}_0$ to decoder-facing latent $\hat{z}$.
Decode $\hat{z}$ and sample the adjacency matrix $\hat{A}$.

D Theoretical Analysis and Proofs

D.1 Scope and Role of the Analysis

This section states the scope of our theoretical analysis and provides proofs for the theorems. The goal is not to establish an end-to-end guarantee for the full graph generation process, nor to directly analyze finite-sample optimization, decoder approximation, diffusion sampling dynamics, or MMD-based generation performance. Instead, the analysis clarifies two design choices. Theorem 4.1 motivates the use of a mixed-curvature latent substrate for heterogeneous graph distributions. This comparison is restricted to the stated tree, grid, and cycle families and does not establish general optimality of the fixed factorization or characterize the necessity of individual geometric factors. Automatic selection of factor types and dimensional allocation is outside the scope of the present formulation. Theorem 4.1 uses an idealized zero-curvature Euclidean factor; implementation details for the learnable Euclidean-like branch are given in Appendix E.5. Theorem 4.2 provides an auxiliary validation-level non-inferiority statement for selecting the Ricci regularization strength when the unregularized model is included among the candidates. It should therefore be viewed as a safety property of the selection protocol, rather than as a claim that Ricci regularization necessarily improves test-time performance.

D.2 Proof of Theorem 4.1

Proof. We define the scale-invariant bi-Lipschitz distortion by

$$D(G \to \mathcal{M}) \;:=\; \inf_{f:V\to\mathcal{M}} \inf_{s>0} \Big(\sup_{u\neq v} \frac{d_{\mathcal{M}}(f(u), f(v))}{s\, d_G(u,v)} \Big) \Big(\sup_{u\neq v} \frac{s\, d_G(u,v)}{d_{\mathcal{M}}(f(u), f(v))} \Big), \tag{39}$$

and define the expected distortion risk under a graph distribution p as

$$\mathcal{R}(\mathcal{M}; p) \;:=\; \mathbb{E}_{G\sim p}\big[D(G \to \mathcal{M}) \big]. \tag{40}$$

This distortion measures the best global multiplicative mismatch between the graph shortest-path metric and the target-space metric, up to a single optimal rescaling.

We instantiate the heterogeneous distribution by three canonical graph families:

$$p = \alpha_A p_A^{(h)} + \alpha_B p_B^{(m)} + \alpha_C p_C^{(\ell)}, \qquad \alpha_A, \alpha_B > 0,\ \alpha_C \geq 0,\ \alpha_A + \alpha_B + \alpha_C = 1,$$

where:

- $p_A^{(h)}$ is supported on binary trees T_h of depth h;
- $p_B^{(m)}$ is supported on $m \times m$ grid graphs Q_m;
- $p_C^{(\ell)}$ is supported on cycle graphs C_ℓ of length ℓ.

These three families represent, respectively, the hierarchical, flat grid-like, and cycle structural components in the mixed distribution.

We prove the claim in two parts.

Part I: Lower bound for every single-curvature space. We show that any fixed single-curvature space must incur a distortion risk that diverges with (h, m, ℓ).

Case 1: $\kappa < 0$. Let $\mathcal{M}_\kappa^d$ be a negatively curved target. We first use the grid family. It is standard that the Gromov hyperbolicity of the grid satisfies

$$\delta(Q_m) = \Theta(m),$$

while any fixed hyperbolic manifold $\mathcal{M}_\kappa^d$ has uniformly bounded hyperbolicity,

$$\delta(\mathcal{M}_\kappa^d) = O(1).$$

Now suppose that $f : V(Q_m) \to \mathcal{M}_\kappa^d$ has distortion at most D. Then there exists a scale $s > 0$ such that

$$\frac{s}{D} d_{Q_m}(u, v) \leq d_{\mathcal{M}}(f(u), f(v)) \leq sD\, d_{Q_m}(u, v), \qquad \forall u, v \in V(Q_m).$$

By the standard stability of Gromov hyperbolicity under bi-Lipschitz embeddings, there exists an absolute constant $C_1 > 0$ such that

$$\delta(Q_m) \leq C_1 D\big(\delta(\mathcal{M}_\kappa^d) + 1\big).$$

Combining $\delta(Q_m) = \Theta(m)$ with $\delta(\mathcal{M}_\kappa^d) = O(1)$ yields

$$D(Q_m \to \mathcal{M}_\kappa^d) \geq c_B m$$

for some constant $c_B > 0$ independent of m.

Next consider the cycle family. The cycle graph satisfies

$$\delta(C_\ell) = \Theta(\ell).$$

Applying the same bi-Lipschitz stability argument to any embedding $f : V(C_\ell) \to \mathcal{M}_\kappa^d$ with distortion D gives

$$\delta(C_\ell) \leq C_2 D\big(\delta(\mathcal{M}_\kappa^d) + 1\big)$$

for an absolute constant $C_2 > 0$. Since again $\delta(\mathcal{M}_\kappa^d) = O(1)$, we obtain

$$D(C_\ell \to \mathcal{M}_\kappa^d) \geq c_C \ell$$

for some constant $c_C > 0$ independent of ℓ.

Therefore, for $\kappa < 0$,

$$\begin{aligned}
\mathcal{R}(\mathcal{M}_\kappa^d; p) &= \mathbb{E}_{G \sim p}\big[D(G \to \mathcal{M}_\kappa^d)\big] \\
&= \alpha_A \mathbb{E}_{G \sim p_A^{(h)}}[D(G \to \mathcal{M}_\kappa^d)] + \alpha_B \mathbb{E}_{G \sim p_B^{(m)}}[D(G \to \mathcal{M}_\kappa^d)] + \alpha_C \mathbb{E}_{G \sim p_C^{(\ell)}}[D(G \to \mathcal{M}_\kappa^d)] \\
&\geq \alpha_B\, D(Q_m \to \mathcal{M}_\kappa^d) + \alpha_C\, D(C_\ell \to \mathcal{M}_\kappa^d) \\
&\geq \alpha_B c_B m + \alpha_C c_C \ell.
\end{aligned} \tag{41}$$

Although both terms are derived via hyperbolicity mismatch, they capture different structural obstructions: the grid term corresponds to large flat two-dimensional patches, whereas the cycle term corresponds to long closed-loop geometry. The latter is precisely the part that can later be routed to a geodesic circle inside the spherical factor.

Case 2: $\kappa \geq 0$. Let $\mathcal{M}_\kappa^d$ be a nonnegative-curvature target. We use the hierarchical family. Let L_h be the set of leaves of the binary tree T_h. Then

$$|L_h| = 2^h.$$

Suppose $f : V(T_h) \to \mathcal{M}_\kappa^d$ has distortion at most D. Then there exists a scale $s > 0$ such that

$$\frac{s}{D} d_{T_h}(u, v) \leq d_{\mathcal{M}}(f(u), f(v)) \leq sD\, d_{T_h}(u, v), \qquad \forall u, v \in V(T_h).$$

Fix the root r of T_h. For every leaf $x \in L_h$, one has $d_{T_h}(r, x) = h$, hence

$$d_{\mathcal{M}}(f(r), f(x)) \leq sDh.$$

Thus all embedded leaf vertices lie inside the metric ball

$$B_{\mathcal{M}}(f(r),\, sDh).$$

On the other hand, for two distinct leaves $x, y \in L_h$, one has

$$d_{T_h}(x, y) \geq 2,$$

so

$$d_{\mathcal{M}}(f(x), f(y)) \geq \frac{2s}{D}.$$

Hence the 2^h embedded leaves form a $\frac{2s}{D}$-separated set inside the ball $B_{\mathcal{M}}(f(r), sDh)$.

For any d-dimensional manifold with nonnegative curvature, Bishop–Gromov comparison implies that the packing number of a radius-R ball by ρ-separated points is at most polynomial:

$$N_{\mathrm{pack}}(R, \rho) \leq C_d \Big(1 + \frac{R}{\rho}\Big)^d$$

for a constant C_d depending only on d. Applying this with

$$R = sDh, \qquad \rho = \frac{2s}{D},$$

we obtain

$$2^h \leq C_d \Big(1 + \frac{sDh}{2s/D}\Big)^d = C_d \Big(1 + \frac{D^2 h}{2}\Big)^d.$$

Therefore

$$D^2 h \geq c_d' 2^{h/d}$$

for some constant $c_d' > 0$, and hence

$$D(T_h \to \mathcal{M}_\kappa^d) \geq c_T\, 2^{\frac{h}{2d}} h^{-\frac{1}{2}}$$

for a constant $c_T > 0$ depending only on d.

It follows that for $\kappa \geq 0$,

$$\mathcal{R}(\mathcal{M}_\kappa^d; p) \geq \alpha_A\, D(T_h \to \mathcal{M}_\kappa^d) \geq \alpha_A c_T\, 2^{\frac{h}{2d}} h^{-\frac{1}{2}}. \tag{42}$$

We do not use the cycle component in the nonnegative-curvature lower bound, since long cycle metrics admit $O(1)$-distortion embeddings into nonnegative-curvature spaces. Thus, in the present framework, the asymptotically growing obstruction for $\kappa \geq 0$ remains driven by the hierarchical component. The strict separation in the present argument is inherited from the tree/grid submixture, while the spherical factor provides an additional bounded-distortion route for the cycle family.

Combining (41) and (42), we conclude that for every $\kappa \in \mathbb{R}$,

$$\mathcal{R}(\mathcal{M}_\kappa^d; p) \;\geq\; \min\Big\{\alpha_B c_B m + \alpha_C c_C \ell,\; \alpha_A c_T\, 2^{\frac{h}{2d}} h^{-\frac{1}{2}}\Big\} \;=:\; \varepsilon(h, m, \ell). \tag{43}$$

Since $\varepsilon(h, m, \ell) \to \infty$ as $(h, m, \ell) \to \infty$, we obtain

$$\inf_{\kappa \in \mathbb{R}} \mathcal{R}(\mathcal{M}_\kappa^d; p) \;\geq\; \varepsilon(h, m, \ell). \tag{44}$$

Part II: Upper bound for the mixed-curvature product space. Choose integers d_h, d_s, d_e such that

$$d_h, d_e \geq 2, \qquad d_s \geq 1, \qquad d_h + d_s + d_e = d,$$

and choose curvatures $\kappa_h < 0 < \kappa_s$. Define

$$\mathcal{M}_{\text{mix}} = \mathbb{H}^{d_h}_{\kappa_h} \times \mathbb{S}^{d_s}_{\kappa_s} \times \mathbb{E}^{d_e}$$

with the product metric

$$d^2_{\text{mix}} = d^2_{\mathbb{H}} + d^2_{\mathbb{S}} + d^2_{\mathbb{E}}.$$

We now route each graph family to its geometrically matched factor.

Hierarchical component. A standard fact from hyperbolic embedding theory is that finite binary trees admit uniformly bounded-distortion embeddings into hyperbolic space. Hence there exists a constant $C_A > 0$, independent of h, such that

$$D(T_h \to \mathbb{H}^{d_h}_{\kappa_h}) \leq C_A.$$

Grid component. Embed the grid Q_m into $\mathbb{E}^2 \subseteq \mathbb{E}^{d_e}$ by its canonical integer-lattice coordinates. For any two vertices, the graph shortest-path distance equals the ℓ_1 distance of their coordinates, whereas the Euclidean metric is ℓ_2. Thus

$$\|x - y\|_2 \leq d_{Q_m}(x, y) \leq \sqrt{2}\,\|x - y\|_2,$$

which implies

$$D(Q_m \to \mathbb{E}^{d_e}) \leq \sqrt{2}.$$

Cycle component. Embed the cycle C_ℓ into a geodesic great circle of $\mathbb{S}^{d_s}_{\kappa_s}$ by placing the ℓ vertices at equal arc-length intervals. The geodesic distance along that circle agrees exactly with the graph shortest-path metric on C_ℓ up to the global scale factor. Therefore

$$D(C_\ell \to \mathbb{S}^{d_s}_{\kappa_s}) = 1.$$

Because the product metric preserves distances within the active factor when the other coordinates are fixed basepoints, the same distortion bounds hold for the embeddings into $\mathcal{M}_{\text{mix}}$. Consequently,

$$\begin{aligned} \mathcal{R}(\mathcal{M}_{\text{mix}}; p) &= \alpha_A D(T_h \to \mathcal{M}_{\text{mix}}) + \alpha_B D(Q_m \to \mathcal{M}_{\text{mix}}) + \alpha_C D(C_\ell \to \mathcal{M}_{\text{mix}}) \\ &\leq \alpha_A C_A + \sqrt{2}\,\alpha_B + \alpha_C \;=:\; \varepsilon'. \end{aligned} \tag{45}$$

Here $\varepsilon' = O(1)$ is independent of (h, m, ℓ).

Finally, combining (44) and (45), we get

$$\mathcal{R}(\mathcal{M}_{\text{mix}}; p) \leq \varepsilon', \qquad \inf_{\kappa \in \mathbb{R}} \mathcal{R}(\mathcal{M}^d_\kappa; p) \geq \varepsilon(h, m, \ell).$$

Since $\varepsilon(h, m, \ell) \to \infty$ while $\varepsilon' = O(1)$, for sufficiently large (h, m, ℓ) we have

$$\varepsilon' < \varepsilon(h, m, \ell),$$

and therefore

$$\mathcal{R}(\mathcal{M}_{\text{mix}}; p) < \inf_{\kappa \in \mathbb{R}} \mathcal{R}(\mathcal{M}^d_\kappa; p).$$

Moreover, the separation

$$\inf_{\kappa \in \mathbb{R}} \mathcal{R}(\mathcal{M}^d_\kappa; p) - \mathcal{R}(\mathcal{M}_{\text{mix}}; p)$$

can be made arbitrarily large by increasing (h, m, ℓ). This proves the theorem. □

D.3 Proof of Theorem 4.2

Proof. Let $\mathcal{D} = \mathcal{D}_{\mathrm{tr}} \cup \mathcal{D}_{\mathrm{val}}$ be a disjoint split of graphs, where $\mathcal{D}_{\mathrm{tr}}$ is used to fit model parameters and $\mathcal{D}_{\mathrm{val}}$ is used only for selecting hyperparameters. Throughout, θ denotes all trainable parameters of the mixed-curvature generative model. Let $\mathcal{L}_0(\theta; \mathcal{D}_{\mathrm{tr}})$ be the *original* training objective on the training set, corresponding to Eq. (7) with the $\mathcal{L}_{\mathrm{frc}}$ removed.

Let $R(G)$ be Ricci-type statistics computed from a graph G (e.g., ORC/AORC/FRC at node/edge level, or their aggregates). Let $\mathcal{L}_{\mathrm{curv}}(\theta; \mathcal{D}_{\mathrm{tr}}) \geq 0$ be a *curvature mismatch* penalty ($\mathcal{L}_{\mathrm{frc}}$) on the training set:

$$\mathcal{L}_{\mathrm{curv}}(\theta; \mathcal{D}_{\mathrm{tr}}) := \mathbb{E}_{G \sim \mathcal{D}_{\mathrm{tr}}} \Big[\Delta_{\mathrm{curv}} \big(R(G), R(\hat{G}_\theta) \big) \Big], \qquad \Delta_{\mathrm{curv}}(\cdot, \cdot) \geq 0, \tag{46}$$

where $\hat{G}_\theta$ denotes a graph sampled/generated by the model with parameters θ.

This can be seen as a general form of the curvature penalty in Eq. (6). Then for each regularization weight $\lambda \geq 0$, we define the *regularized* training objective

$$\mathcal{L}_\lambda(\theta; \mathcal{D}_{\mathrm{tr}}) := \mathcal{L}_0(\theta; \mathcal{D}_{\mathrm{tr}}) + \lambda \, \mathcal{L}_{\mathrm{curv}}(\theta; \mathcal{D}_{\mathrm{tr}}). \tag{47}$$

Let $\mathcal{A}$ denote the training procedure (e.g., SGD with fixed schedule/early stopping rule), and define

$$\theta_\lambda := \mathcal{A}(\mathcal{D}_{\mathrm{tr}}, \lambda), \tag{48}$$

i.e., θ_λ is the parameter vector produced by running the same training pipeline with weight λ.

Let $M(\theta; \mathcal{D}_{\mathrm{val}})$ be the *base evaluation metric* used to compare models. Because M is typically estimated empirically, define its validation estimate

$$\widehat{M}_{\mathrm{val}}(\theta) := \widehat{M}(\theta; \mathcal{D}_{\mathrm{val}}), \tag{49}$$

We expand all symbols used in the argument.

(i) Candidate set contains the baseline. By assumption, $0 \in \Lambda$. The model trained with $\lambda = 0$ uses the objective

$$\mathcal{L}_0(\theta; \mathcal{D}_{\mathrm{tr}}) = \mathcal{L}_0(\theta; \mathcal{D}_{\mathrm{tr}}) + 0 \cdot \mathcal{L}_{\mathrm{curv}}(\theta; \mathcal{D}_{\mathrm{tr}}),$$

so $\theta_0 = \mathcal{A}(\mathcal{D}_{\mathrm{tr}}, 0)$ is exactly the baseline training run (i.e., no ORC/FRC regularization).

(ii) Definition of the selected hyperparameter. By (18), $\hat{\lambda}$ is chosen to minimize the scalar quantity $\widehat{M}_{\mathrm{val}}(\theta_\lambda)$ over $\lambda \in \Lambda$, where:

- θ_λ is the trained parameter vector output by the fixed training pipeline $\mathcal{A}$ for weight λ,
- $\widehat{M}_{\mathrm{val}}(\theta_\lambda)$ is the validation estimate of the base metric computed on $\mathcal{D}_{\mathrm{val}}$ using a fixed evaluation protocol.

(iii) Minimality implies dominance over any candidate, in particular the baseline. Since $\hat{\lambda}$ attains (or ties for) the minimum of $\widehat{M}_{\mathrm{val}}(\theta_\lambda)$ over the finite set Λ, we have, by the definition of $\arg\min$,

$$\widehat{M}_{\mathrm{val}}(\theta_{\hat{\lambda}}) = \min_{\lambda \in \Lambda} \widehat{M}_{\mathrm{val}}(\theta_\lambda) \leq \widehat{M}_{\mathrm{val}}(\theta_0), \tag{50}$$

where the inequality holds because $\lambda = 0$ is one admissible element of Λ.

This is exactly (19), completing the proof.

□

E Experimental Details and Additional Benchmarks

This appendix provides supplementary details for Sec. 5, including dataset information, baseline descriptions, evaluation protocol, additional benchmark coverage, and implementation settings.

Table 5: Additional benchmark statistics for structural characterization.

Dataset	# Graphs	Avg. Nodes	Density	Leaf Ratio	Clustering	Square-cycle Share
Community	1000	7.85	0.1056	0.7658	0.0700	0.0683
Planar	200	64.00	0.0883	0.0000	0.4617	0.0256
MUTAG	188	17.93	0.1385	0.2022	0.0000	0.0000
PROTEINS	1045	30.52	0.2244	0.0053	0.5321	0.1346

Table 6: Complementary measures of structural heterogeneity. Each entry reports the mean±population standard deviation of the corresponding per-graph statistic across the analyzed real graphs in each dataset.

Dataset	Degree CV	Clustering Std.	Orbit Entropy
Ego	0.894 ± 0.290	0.143 ± 0.178	0.685 ± 0.092
Grid	0.133 ± 0.010	0.000 ± 0.000	0.670 ± 0.001
COLLAB	0.479 ± 0.320	0.139 ± 0.086	0.580 ± 0.277
IMDB-BINARY	0.380 ± 0.223	0.138 ± 0.063	0.669 ± 0.201

E.1 Datasets and Additional Benchmarks

The main text evaluates CurvDiff on four representative generic graph generation datasets: **Ego**, **Grid**, **COLLAB**, and **IMDB-BINARY**. To broaden the evaluation while keeping the main paper concise, we additionally report results on **Community**, **Planar**, **MUTAG**, and **PROTEINS**.

Community contains two-community synthetic graphs and is used to assess whether a model captures explicit community structure. **Planar** evaluates generation under planar structural constraints. **MUTAG** and **PROTEINS** are used only as supplementary structure-distribution benchmarks under the same MMD protocol as the generic graph datasets. We therefore do not treat them as full molecular or biological generation evaluations, and do not report domain-specific validity, novelty, or property metrics. The structural statistics of these additional benchmarks are summarized in Table 5.

E.2 Quantifying Structural Heterogeneity

We complement Table 1 with three graph-level measures. Degree CV characterizes relative variation in node degrees, clustering standard deviation characterizes variation in node-level clustering coefficients, and normalized orbit entropy characterizes the diversity of aggregate orbit counts.

For each real graph G, we use ORCA in `node 4` mode to obtain all 15 node-orbit counts (orbits 0–14). Let c_{vk} denote the count of node v in orbit k. We define

$$\begin{aligned} p_k &= \frac{\sum_{v \in V(G)} c_{vk}}{\sum_{\ell=0}^{14} \sum_{v \in V(G)} c_{v\ell}}, \qquad k = 0, \dots, 14, \\ H_{\mathrm{orbit}}(G) &= -\frac{\sum_{k:p_k>0} p_k \log p_k}{\log 15}. \end{aligned} \tag{51}$$

This entropy lies in $[0, 1]$ and is computed from the graph-level orbit-count distribution.

Table 6 reveals distinct structural profiles across datasets. Ego exhibits the largest mean degree CV, while COLLAB shows the largest across-graph variation in orbit entropy. Grid combines regular degree and clustering patterns with nonzero orbit entropy, illustrating that node-level variation and aggregate motif-role diversity capture different aspects of structure. We use these measures as complementary descriptive proxies, without assuming that any single statistic monotonically predicts the benefit of mixed curvature.

E.3 Baselines

We compare CurvDiff with representative graph generative baselines covering classical latent-variable, autoregressive, diffusion and flow-based, latent-space, and geometry-aware paradigms. **VGAE** [Kipf and Welling, 2016] and **GraphRNN** [You et al., 2018] serve as classical latent-variable and autoregressive baselines. For diffusion-based graph generation, we include **GDSS** [Jo et al., 2022],

DiGress [Vignac et al., 2023a], **GraphGDP** [Huang et al., 2022], and **EDGE** [Chen et al., 2023]. We additionally evaluate DeFoG [Qin et al., 2025] and Cometh [Siraudin et al., 2025] as recent general-purpose discrete flow-matching and continuous-time discrete-state diffusion baselines, respectively, on the four main datasets (Appendix E.14). We further include recent latent-space graph generative models, including **LGD** [Zhou et al., 2024], **GLAD** [Nguyen et al., 2025], and **LG-Flow** [Siraudin and Morris, 2026]. For geometry-aware graph generation, we compare against **HypDiff** [Fu et al., 2024], **HGDM** [Wen et al., 2024], and **GGBall** [Bu et al., 2025]. Due to the lack of reproducible public implementations, we do not include **Graphusion** [Yang et al., 2024] and **ProGDM** [Wang et al., 2024] in the quantitative comparison, but both are discussed in the related work.

E.4 Evaluation Protocol

We evaluate generation quality using the maximum mean discrepancy (MMD) between the empirical distributions of three graph statistics computed from the generated and test sets: node degree, clustering coefficient, and spectral statistics [Gretton et al., 2012, You et al., 2018, Jo et al., 2022]. These statistics characterize local connectivity, local transitivity, and global structural organization, respectively. We report `Avg.` as the arithmetic mean of the three MMD scores, with lower values indicating closer agreement between the generated and test distributions. For a fair comparison, the target number of nodes for each generated graph is sampled from the corresponding reference split. Unless otherwise stated, we use the same evaluation protocol for all experiments reported in the main paper and the appendix. The complementary Orbit MMD evaluation is reported separately in Appendix E.12 and is not included in `Avg`.

E.5 Implementation Settings

All graphs are treated as undirected, ordered by breadth-first search, and zero-padded to the dataset-specific maximum size for adjacency reconstruction under a fixed node ordering. The encoder uses two GraphConv layers followed by a mixed-curvature layer with 64 latent dimensions approximately evenly allocated across hyperbolic, spherical, and near-Euclidean factors; the decoder comprises four fully connected layers. Factor curvatures are learned jointly with the network parameters, starting from $c_{\mathrm{hyp}} \approx -1$, $c_{\mathrm{sph}} \approx 1$, and $c_{\mathrm{euc}} = 10^{-3}$. Both training stages use a learning rate of 10^{-4}. Reconstruction training uses 5000 epochs and batch size 512, except on **COLLAB**, where these are 500 and 256, respectively. Encoder-side Forman–Ricci consistency regularization, where enabled, has weight 0.02. The diffusion denoiser has 10 layers and hidden dimension 512, and is trained with an ϵ-prediction objective for 5000 epochs with batch size 512. We use $T = 1000$ diffusion steps with a linear noise schedule from $\beta_1 = 10^{-4}$ to $\beta_T = 0.02$. Unless otherwise stated, sampling uses an exponential moving average of the model parameters with decay 0.9999. Classifier-free guidance uses condition dropout probability 0.1 during training and a common guidance scale of 3.0 across datasets in the main comparisons. Training and sampling were performed on an NVIDIA RTX 4090 GPU.

Controlled geometry configuration. To isolate the contribution of latent curvature, the **Euclidean + Full Ricci** control fixes all latent-factor curvatures to zero and excludes them from curvature optimization. It retains the architecture, latent dimensionality and factor allocation, encoder-side Ricci regularization, diffusion-side Ricci conditioning, and shared latent interface of CurvDiff. Within each dataset, data splits, preprocessing, training budgets, sampling settings and sample counts, and evaluation and checkpoint-selection procedures are matched between the two models. The Euclidean variant in Section 5.3 and **Euclidean + Full Ricci** share this zero-curvature configuration but were evaluated using separate training runs with different random seeds.

E.6 Additional Benchmark Results

Table 7 extends the evaluation to **Community**, **Planar**, **MUTAG**, and **PROTEINS**, covering additional synthetic and real-world graph families. We retain the MMD metrics and **Avg.** definition of Table 2, with dataset-specific batch sizes and sampling guidance scales to accommodate the additional benchmarks. CurvDiff achieves the lowest **Avg.** MMD on **Community**, **MUTAG**, and **PROTEINS**, and the second-lowest on **Planar**. These results support strong aggregate structural matching beyond the main benchmarks, while individual statistics exhibit method-specific trade-offs.

Table 7: MMD comparison on additional graph generation benchmarks.

Method	Synthetic Datasets								Real-world Datasets							
	Community				Planar				MUTAG				PROTEINS			
	Deg.	Clu.	Spe.	**Avg.** ↓	Deg.	Clu.	Spe.	**Avg.** ↓	Deg.	Clu.	Spe.	**Avg.** ↓	Deg.	Clu.	Spe.	**Avg.** ↓
GDSS	0.094	0.031	0.052	0.059	0.151	1.667	0.587	0.802	0.074	0.021	**0.003**	0.033	1.463	0.168	0.013	0.548
DiGress	0.226	0.158	0.194	0.193	**0.004**	**0.155**	**0.012**	**0.057**	0.100	0.351	0.082	0.178	0.108	0.062	0.079	0.083
GraphGDP	0.046	0.016	0.042	0.035	0.276	1.033	0.086	0.465	0.127	0.057	0.050	0.078	0.103	0.240	0.088	0.144
EDGE	0.021	0.013	0.040	0.025	0.038	1.564	0.811	0.804	0.024	0.597	0.468	0.363	**0.033**	0.523	0.024	0.193
LGD	0.213	0.065	0.084	0.121	0.776	1.295	0.641	0.904	0.415	2.000	0.204	0.873	0.322	1.016	0.109	0.482
GLAD	0.408	0.039	0.038	0.162	1.984	1.978	0.031	1.331	0.577	0.021	0.011	0.203	1.804	1.117	0.045	0.989
LG-Flow	0.028	0.009	0.013	0.017	0.472	0.442	0.320	0.411	0.069	0.087	0.065	0.074	0.261	0.211	0.159	0.210
HypDiff	**0.002**	0.010	0.028	0.013	0.748	0.313	0.303	0.455	0.048	**0.001**	0.040	0.030	0.133	**0.004**	**0.012**	0.050
HGDM	0.036	0.062	0.021	0.040	0.080	1.792	0.731	0.868	0.014	0.054	0.070	0.046	0.108	0.197	**0.012**	0.106
GGBall	0.234	0.325	0.130	0.230	0.741	0.400	0.397	0.513	0.415	0.196	0.198	0.270	0.326	0.974	0.125	0.475
CurvDiff	0.014	**0.002**	**0.008**	**0.008**	0.419	0.390	0.191	0.333	**0.005**	0.021	0.023	**0.016**	0.057	0.049	0.026	**0.044**

Note: Bold and underlined entries indicate the best and second-best reported scores, respectively. Lower is better for all metrics.

E.7 Stage Consistency Results

To further probe the stage-consistency claim of CurvDiff, we construct five diffusion-side counterfactual variants while keeping the encoder fixed to the default FRC-based design. Specifically, No Diffusion-side Ricci removes Ricci conditioning from the diffusion stage altogether, testing whether the second stage still benefits from topology-aware guidance once the encoder is fixed. Shuffled FRC Prior preserves the marginal distribution of FRC condition vectors but breaks graph-wise alignment between latent representations and their own topology priors. Random FRC Prior introduces a stronger mismatch by replacing the matched FRC condition with an independently sampled one. Matched AORC Prior and Matched ORC Prior further replace the matched FRC condition with matched conditions derived from alternative Ricci variants, in order to examine whether consistency in prior family also matters across stages.

Table 8 reports the detailed results. Overall, CurvDiff remains strongest on **Ego** and **Grid**, indicating that the matched full design is more reliable than these perturbed alternatives on these datasets. At the same time, the relative ranking of the counterfactual variants is dataset-dependent. In particular, on **IMDB-BINARY**, Matched ORC Prior slightly outperforms the current CurvDiff result in the average metric. Therefore, this experiment should be interpreted as supportive rather than definitive evidence for the matched-prior hypothesis: it shows that removing, disrupting, or altering diffusion-side topology priors can degrade performance, while also suggesting that the benefit of stage consistency depends on the graph family and the specific curvature prior used.

Table 8: Detailed results for the stage consistency experiment.

Method	Ego				Grid				IMDB-BINARY			
	Deg.	Clu.	Spe.	Avg.	Deg.	Clu.	Spe.	Avg.	Deg.	Clu.	Spe.	Avg.
No Diffusion-side Ricci	**0.0019**	0.0345	0.0075	0.0146	0.0036	0.0365	0.1417	0.0606	0.0288	0.1663	0.0431	0.0794
Shuffled FRC Prior	0.0038	0.0512	0.0067	0.0206	0.0039	0.0343	0.1369	0.0584	0.0300	0.2031	0.0362	0.0898
Random FRC Prior	0.0026	0.0289	**0.0057**	0.0124	0.0055	0.0378	0.1049	0.0494	0.0238	0.1620	0.0416	0.0758
Matched AORC Prior	0.0028	**0.0275**	0.0067	0.0123	0.0023	0.0328	0.1540	0.0630	0.0214	0.1055	0.0378	0.0549
Matched ORC Prior	0.0027	0.0298	0.0060	0.0128	**0.0014**	0.0334	0.1550	0.0633	**0.0148**	0.0713	**0.0279**	**0.0380**
CurvDiff	0.0020	0.0280	0.0060	**0.0120**	0.0460	**0.0150**	**0.0490**	**0.0367**	0.0330	**0.0450**	0.0440	0.0407

E.8 Additional Geometry Ablation Results

Figure 6 summarizes supplementary **Avg.** MMD comparisons for geometry and interface variants. The configuration of each variant and the evaluation protocol match Fig. 3, but the results come from independently trained checkpoints using different training seeds. Consistent with the per-metric comparisons, CurvDiff achieves the lowest **Avg.** MMD on **Ego**, **Grid**, and **IMDB-BINARY**, and remains close to the best result on **COLLAB**, where the Hyperbolic variant performs slightly better.

Figure 7 presents representative reference and generated graphs from **Ego**, **Grid**, and **IMDB-BINARY**. The examples illustrate branching, grid-like, and locally dense patterns, providing qualitative context for the complementary geometric biases motivating mixed-curvature modeling.

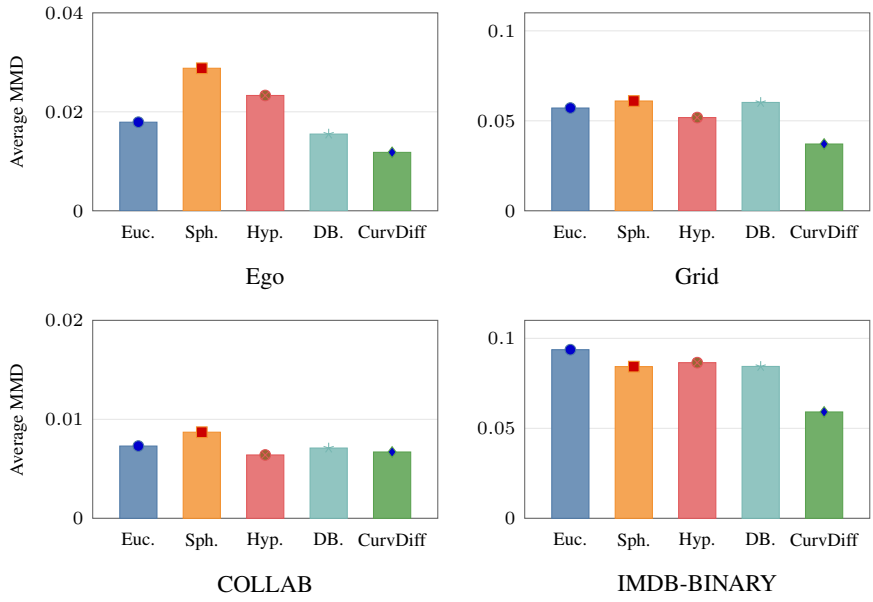


Figure 6: Per-dataset **Avg.** MMD comparison for geometry and interface variants.

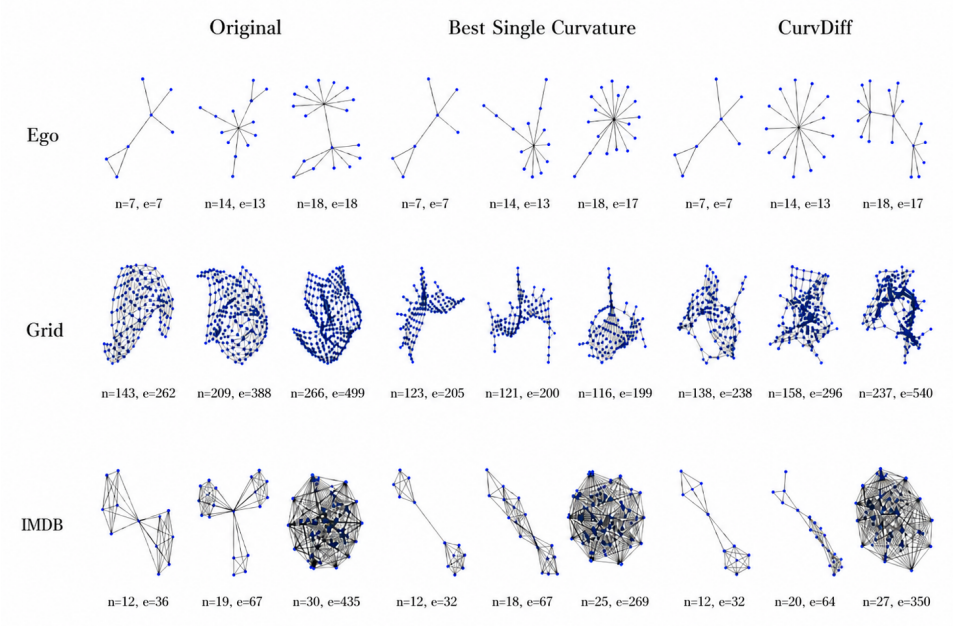


Figure 7: Representative graph samples for the geometry ablation.

E.9 Role of the Spherical Factor

We compare CurvDiff with a variant using only hyperbolic and Euclidean factors, retaining the Ricci pathways, shared interface, and evaluation settings. Both models use 64 latent dimensions, approximately evenly allocated across their respective factors. Table 9 shows lower **Avg.** MMD for CurvDiff on Ego, Grid, and IMDB-BINARY, with nearly tied results on COLLAB. This comparison supports a complementary, dataset-dependent role for the spherical factor within the jointly learned representation.

Table 9: Spherical-factor ablation: **Avg.** MMD (↓).

Dataset	Without spherical factor	Full model
Ego	0.01377	**0.01200**
Grid	0.06455	**0.03670**
COLLAB	**0.00665**	0.00673
IMDB-BINARY	0.10027	**0.04070**

E.10 Detailed Unified Interface Ablation Results

We report degree, clustering, and spectral MMD for six unified-interface ablation variants on **Ego**, **Grid**, and **IMDB-BINARY** in Table 10; their averages correspond to the **Avg.** MMD values in Table 3. CurvDiff achieves the lowest average MMD on all three datasets, while the strongest partial configuration varies across datasets. The variant without the shared interface remains competitive on Ego and Grid but shows a larger performance gap on IMDB-BINARY, suggesting a dataset-dependent benefit of the shared interface.

Table 10: Detailed per-metric results for the unified interface ablation.

	Ego			Grid			IMDB-BINARY		
Method	Deg.	Clu.	Spe.	Deg.	Clu.	Spe.	Deg.	Clu.	Spe.
Euclidean + No Ricci	0.0058	0.0955	<u>0.0058</u>	0.2878	0.1646	0.1069	0.1166	0.0677	0.1058
Mixed Only	0.0072	0.0918	0.0060	0.2674	0.1622	0.0872	0.1195	0.0715	0.1129
Ricci Encoder Only	0.0032	0.0586	**0.0047**	**0.0346**	**0.0050**	0.1351	0.0477	0.1231	0.0554
Ricci Diffusion Only	0.0089	0.0902	0.0086	0.2805	0.4144	<u>0.0445</u>	0.1223	<u>0.0654</u>	0.1153
Mixed + Ricci w/o Shared	**0.0016**	<u>0.0374</u>	0.0064	0.0791	0.0151	**0.0355**	<u>0.0398</u>	0.1774	**0.0425**
CurvDiff	<u>0.0020</u>	**0.0280**	0.0060	<u>0.0460</u>	<u>0.0150</u>	0.0490	**0.0330**	**0.0450**	<u>0.0440</u>

E.11 Extended Controlled Analysis of Mixed-Curvature Geometry

We extend the controlled geometry comparison to seven datasets with matched runs (Table 11), excluding Planar because its matched control was not evaluated. Enabling both Ricci pathways improves the Euclidean baseline on every reported dataset. With these pathways and the shared interface held fixed, CurvDiff further lowers average MMD on every dataset relative to **Euclidean + Full Ricci**, whose latent-factor curvatures are fixed to zero. Geometry-only and single-pathway variants show dataset-dependent benefits, supporting the complementary use of mixed-curvature representations and Ricci-derived structural guidance.

Table 11: Extended controlled geometry comparison using **Avg.** MMD (↓).

Method	Ego	Grid	COLLAB	IMDB-B	Community	MUTAG	PROTEINS
Euclidean + No Ricci	0.0357	0.1864	0.1058	0.0967	0.0364	0.4182	0.1587
Mixed Only	0.0350	0.1723	0.1049	0.1013	0.0213	0.3974	0.1575
Euclidean + Full Ricci	<u>0.0169</u>	<u>0.0538</u>	0.0088	0.0766	<u>0.0174</u>	<u>0.1978</u>	<u>0.0595</u>
Ricci Encoder Only	0.0222	0.0582	<u>0.0082</u>	<u>0.0754</u>	0.0298	0.2789	0.0624
Ricci Diffusion Only	0.0359	0.2465	0.1079	0.1010	0.0188	0.4114	0.1621
CurvDiff	**0.0120**	**0.0367**	**0.0067**	**0.0407**	**0.0080**	**0.0160**	**0.0440**

Note: Bold and underlined entries indicate the best and second-best results, respectively.

E.12 Supplementary Orbit-Count Evaluation

We assess local motif-role fidelity using Orbit MMD on the four main datasets. ORCA in `node 4` mode produces 15-dimensional node-orbit counts, which are averaged over nodes without unit-mass normalization. We report biased empirical squared MMD using a Gaussian kernel based on half the ℓ_1 distance, with bandwidth $\sigma = 30$. CurvDiff configurations use newly trained checkpoints,

Table 12: Supplementary Orbit MMD on the four main datasets (↓).

Method	Ego	Grid	COLLAB	IMDB-BINARY
GGBall	0.4386	1.0480	**0.0062**	0.1495
Euclidean + No Ricci	0.0184	0.0892	0.0173	0.1675
Mixed Only	0.0166	0.0987	0.0171	0.1981
Ricci Encoder Only	0.0174	0.0714	0.0306	0.1274
Ricci Diffusion Only	0.0206	0.0963	0.0168	0.1988
Euclidean + Full Ricci	0.0222	0.0809	0.0228	0.1371
CurvDiff	**0.0113**	**0.0479**	0.0144	**0.1145**

separately from the original main-benchmark runs in Table 2; all methods share the same data splits, generated-sample counts, and Orbit evaluator. Orbit MMD is excluded from the three-metric **Avg**.

Table 12 shows that CurvDiff achieves the lowest Orbit MMD among the internal variants on all four datasets and outperforms GGBall on Ego, Grid, and IMDB-BINARY, while GGBall performs better on COLLAB. These results support the complementary use of mixed geometry and Ricci guidance for local motif-role matching, with dataset-dependent effects from individual components.

E.13 Metric-Dependent Geometry Trade-offs on COLLAB

We compare the Hyperbolic variant and CurvDiff on COLLAB to examine metric-dependent geometry trade-offs (Table 13). Both models use matched data splits and evaluator settings, with all four statistics computed from the same generated graph pool for each model. Orbit MMD follows Appendix E.12. `Avg.-3` follows the original mean of degree, clustering, and spectral MMD, while `Avg.-4` additionally includes Orbit MMD as a supplementary summary.

Table 13: Geometry trade-offs on COLLAB (MMD ↓).

Method	Deg.	Clu.	Spe.	Avg.-3	Orbit	Avg.-4
Hyperbolic	**0.0060**	**0.0043**	0.0090	**0.00643**	0.0187	0.00950
CurvDiff	0.0075	0.0044	**0.0083**	0.00673	**0.0144**	**0.00865**

Hyperbolic's advantage in degree MMD outweighs CurvDiff's spectral improvement under the original three-metric average, while clustering scores are nearly tied. CurvDiff nevertheless achieves lower spectral and Orbit MMD, indicating that the average-MMD ranking does not extend uniformly to global spectral and local motif-role statistics.

E.14 Comparison with Recent General-Purpose Graph Generators

We broaden the evaluation with DeFoG [Qin et al., 2025] and Cometh [Siraudin et al., 2025] on Ego, Grid, COLLAB, and IMDB-BINARY, using matched data splits, generated-sample counts, and evaluator settings. For each dataset, all four CurvDiff statistics are computed from the same generated graph pool produced by a newly trained checkpoint, separately from the original benchmark in Table 2. `Avg.-3` follows the original **Avg.** definition, while `Avg.-4` additionally includes Orbit MMD as a supplementary equal-weight summary.

We find that CurvDiff achieves the lowest degree MMD and both aggregate scores on all four datasets in Table 14, supporting competitive aggregate structural fidelity against the two baselines. DeFoG consistently achieves the lowest Orbit MMD among the three models, while Cometh leads on selected clustering and spectral statistics, highlighting metric-specific trade-offs.

E.15 Runtime and Ricci-Preprocessing Analysis

Sampling runtime. We profile sampling with 1000 reverse-diffusion steps and a CFG scale of 3.0. Table 15 shows that reverse diffusion dominates the measured runtime, while decoding and graph construction contribute relatively little overhead. Timing covers these three stages and excludes model and feature loading, warm-up, metric computation, and file writing.

Table 14: Supplementary MMD comparison with DeFoG and Cometh (↓).

Dataset	Method	Deg.	Clu.	Spe.	Orbit	Avg.-3	Avg.-4
Ego	Cometh	0.0153	0.0638	**0.0057**	0.0118	0.0283	0.0242
	DeFoG	0.0077	0.0465	0.0074	**0.0050**	0.0205	0.0167
	CurvDiff	**0.0019**	**0.0282**	0.0059	0.0113	**0.0120**	**0.0118**
Grid	Cometh	0.1333	**0.0007**	**0.0288**	0.1436	0.0543	0.0766
	DeFoG	0.2170	0.0345	0.0452	**0.0349**	0.0989	0.0829
	CurvDiff	**0.0460**	0.0152	0.0487	0.0479	**0.0366**	**0.0395**
COLLAB	Cometh	0.0610	0.1759	0.0384	0.0145	0.0918	0.0725
	DeFoG	0.0439	0.3681	0.0615	**0.0073**	0.1578	0.1202
	CurvDiff	**0.0075**	**0.0044**	**0.0083**	0.0144	**0.0067**	**0.0087**
IMDB-BINARY	Cometh	0.0499	**0.0399**	0.0446	0.1795	0.0448	0.0785
	DeFoG	0.0538	0.0581	0.0527	**0.0965**	0.0549	0.0653
	CurvDiff	**0.0333**	0.0446	**0.0439**	0.1145	**0.0406**	**0.0591**

Table 15: Sampling-stage runtime.

Dataset	Reverse diffusion (s)	Decode + graph construction (s)	Total (s)	Amortized time/graph (ms)
Ego	2.057 ± 0.005	0.003 ± 0.000	2.060 ± 0.005	102.98 ± 0.26
Grid	2.241 ± 0.298	0.020 ± 0.001	2.261 ± 0.299	113.03 ± 14.94
COLLAB	2.062 ± 0.045	0.015 ± 0.004	2.078 ± 0.043	103.88 ± 2.14
IMDB-BINARY	2.059 ± 0.002	0.004 ± 0.000	2.063 ± 0.002	103.15 ± 0.10

Note: Values are mean±std over five runs, each generating a batch of 20 graphs. Per-graph time is the batch runtime divided by 20.

One-time Ricci preprocessing. We benchmark node- and edge-level Ricci preprocessing on Intel Xeon Gold 6326 CPUs. Static FRC features are computed once before reconstruction training and cached, together with the derived graph-level diffusion conditions, for subsequent reuse. Table 16 shows substantially lower measured preprocessing time for FRC than for ORC, with compact feature caches supporting low-cost reuse of structural information. This comparison is implementation-specific because numerical-library thread counts were not controlled.

Table 16: Ricci preprocessing on complete datasets.

Dataset	Graphs	Nodes	Edges	ORC time (s)	FRC time (s)	Cached features (MiB)
Ego	1,197	15,777	16,608	670.789	0.849	0.155
Grid	100	21,025	39,150	82.731	0.092	0.287
IMDB-BINARY	1,000	19,773	96,531	560.296	0.714	0.887

Note: Times are means over three runs, including feature allocation and bookkeeping but excluding dataset loading. Graphs are processed sequentially; ORC uses separate node- and edge-feature solver calls with optimal transport distance (OTD), $\alpha = 0.5$, and up to eight worker processes per graph.

Scaling of static FRC computation. We profile static FRC preprocessing on synthetic sparse graphs with average degree 10, measuring runtime, incremental peak resident set size (RSS), and cached-feature size. For sparse inputs, the closed-form feature pipeline has $O(|V| + |E|)$ time and memory cost. Table 17 shows increasing runtime with graph size and an approximately linear incremental peak RSS trend across the three largest graph sizes. These measurements cover static preprocessing and exclude end-to-end generation and the batchwise reconstruction-side Ricci regularizer.

Table 17: Static FRC preprocessing on synthetic sparse graphs.

Nodes	Edges	FRC time (s)	Incremental peak RSS (MiB)	Cached features (MiB)
1,000	5,000	0.0021 ± 0.0003	$0.14 \pm 0.16^{\dagger}$	0.023
10,000	50,000	0.0065 ± 0.0010	0.35 ± 0.04	0.229
50,000	250,000	0.0411 ± 0.0051	15.03 ± 0.02	1.144
100,000	500,000	0.0925 ± 0.0143	30.24 ± 0.03	2.289
200,000	1,000,000	0.1409 ± 0.0115	64.92 ± 0.02	4.578

Note: Runtime and incremental peak RSS are reported as mean$\pm$std. †The RSS increment for the smallest graph is near the profiling noise floor and is excluded from the memory-scaling interpretation.

E.16 Qualitative Analysis of Ricci Curvature Signals

We provide a qualitative comparison of ORC, AORC, and FRC on representative graphs from **Ego** and **COLLAB**. For each dataset, the overview plot visualizes edge-level curvature patterns on a fixed graph layout, together with raw edge-curvature histograms and the corresponding conditioning histograms used by the diffusion model. This allows us to inspect whether different Ricci variants highlight structurally meaningful regions, such as locally supported edges, hub-incident edges, and peripheral connectors.

The edge-role boxplots provide a complementary role-based view by grouping edges into interpretable structural categories, including triangle-supported, hub-incident, peripheral, and bridge edges when present. As shown in Fig. 8 and Fig. 9, FRC produces a coarser and differently scaled signal compared with ORC and AORC, but it still preserves non-trivial separation across structural edge roles. These observations suggest that FRC provides useful topology-relevant structural discrimination in addition to its computational efficiency. They complement the dataset-level MMD comparison in Fig. 5a.

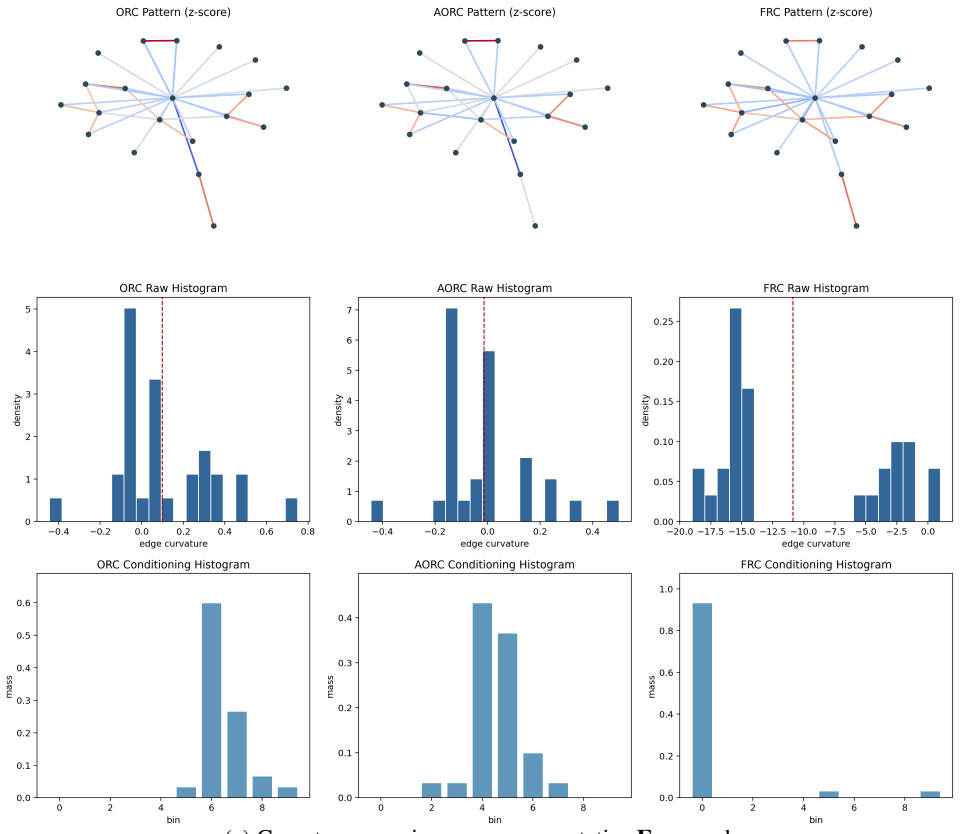
E.17 Component-wise Guidance-Scale Sensitivity

To complement Fig. 5b, Fig. 10 decomposes the effect of classifier-free guidance into degree, clustering, and spectral MMD on all four representative datasets. The component-wise trends clarify why the preferred guidance scale is dataset dependent.

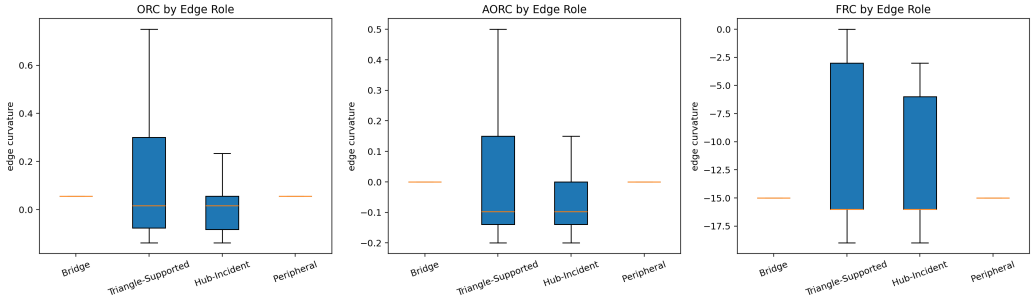
On **Ego**, increasing guidance mainly reduces clustering MMD, while degree MMD remains nearly unchanged and spectral MMD increases slightly. On **Grid**, the effect is more clearly a trade-off: low guidance gives the best degree and clustering scores, whereas higher guidance substantially improves spectral MMD before harming the other two statistics. On **COLLAB**, low-to-moderate guidance yields a small improvement in clustering MMD, but degree MMD gradually worsens and spectral MMD remains nearly flat, so stronger conditioning brings only limited net benefit. On **IMDB-BINARY**, stronger guidance steadily worsens clustering MMD, while degree and spectral statistics vary only mildly; accordingly, weak guidance is generally preferable on this dataset. Overall, these curves indicate that the fixed guidance scale used in the main tables should be interpreted as a shared evaluation protocol rather than a dataset-specific optimum.

F Visualization

We provide qualitative visualizations of graphs from the training sets and the corresponding generated graphs in Figure 11. For each dataset, including Ego, Grid, and IMDB-BINARY, we show representative original graphs and generated counterparts. These examples offer an intuitive comparison between real and generated samples, and show that the proposed model is able to reproduce characteristic structural patterns across different graph datasets.

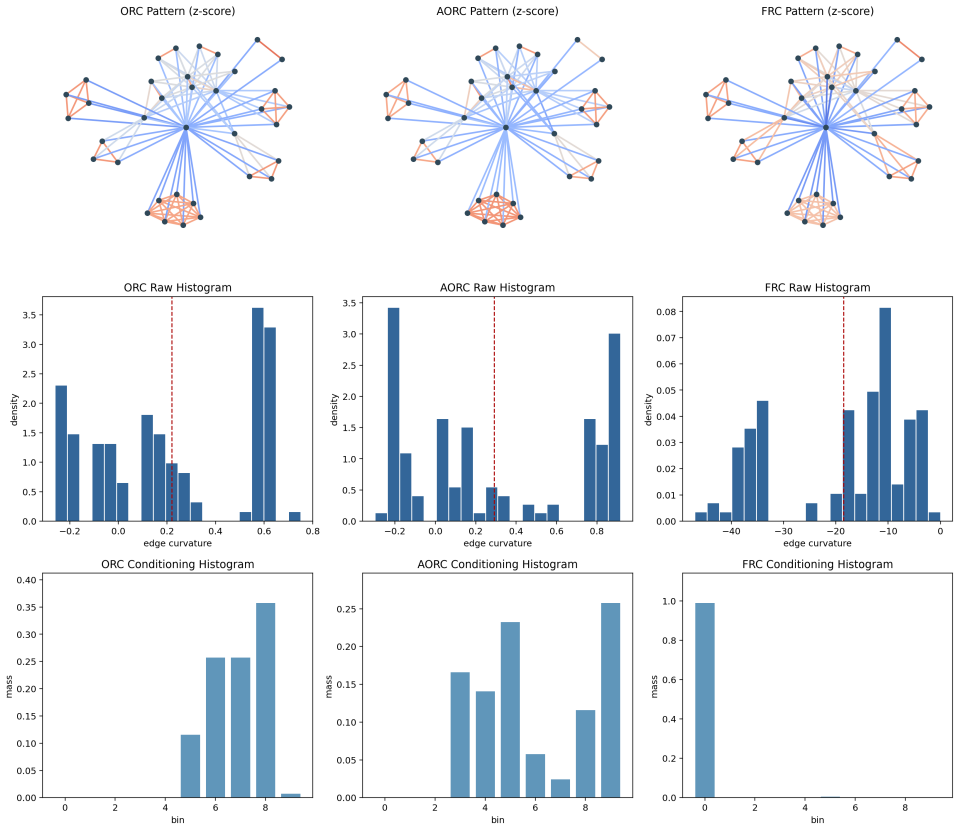


(a) Curvature overview on a representative **Ego** graph.

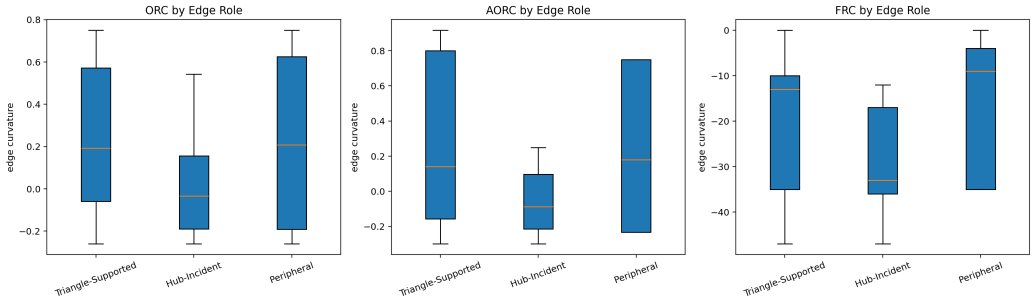


(b) Edge-role boxplots on the same **Ego** graph.

Figure 8: Qualitative comparison of ORC, AORC, and FRC curvature signals on a representative **Ego** graph. The overview plot shows edge-level curvature patterns, raw edge-curvature histograms, and diffusion-side conditioning histograms. The edge-role boxplots summarize curvature distributions across interpretable structural roles. FRC yields a coarser and differently scaled signal, but still provides meaningful role-level structural discrimination.



(a) Curvature overview on a representative **COLLAB** graph.



(b) Edge-role boxplots on the same **COLLAB** graph.

Figure 9: Qualitative comparison of ORC, AORC, and FRC curvature signals on a representative **COLLAB** graph. Compared with the **Ego** case, the graph is denser and more locally clustered, leading to weaker role diversity. Nevertheless, FRC still separates structurally distinct edge roles, especially hub-incident and locally supported edges, while producing a coarser conditioning summary.

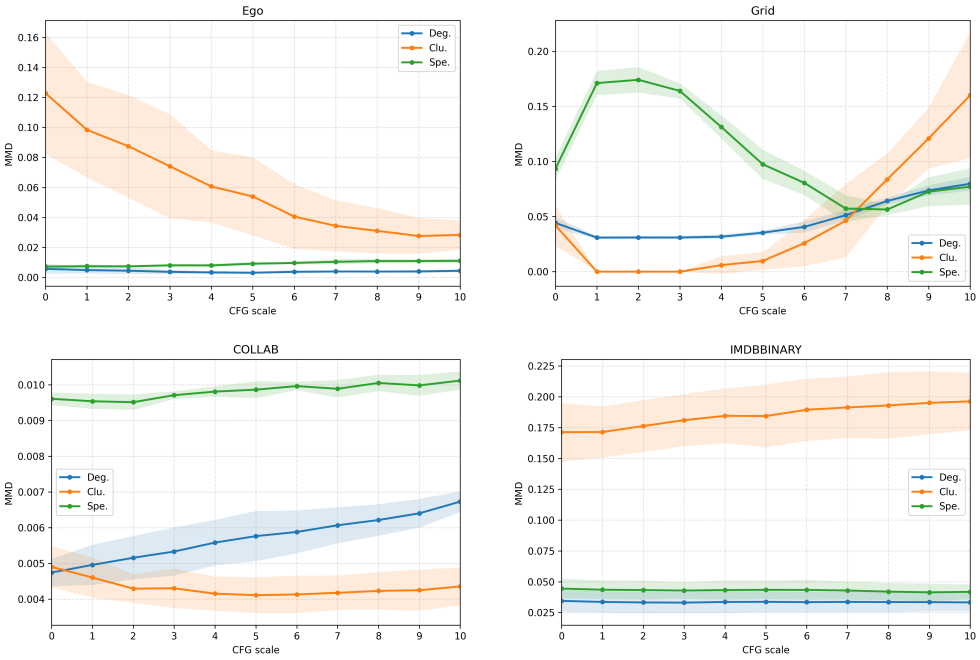
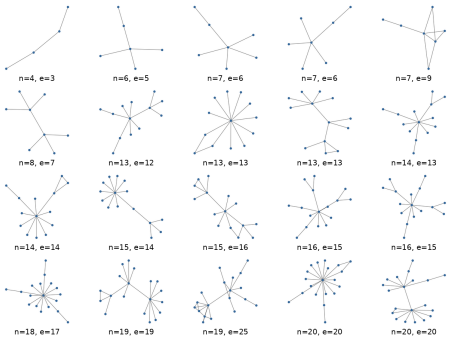
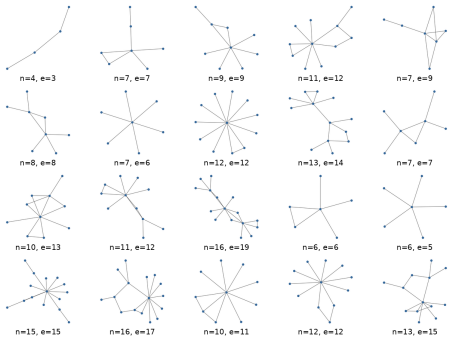


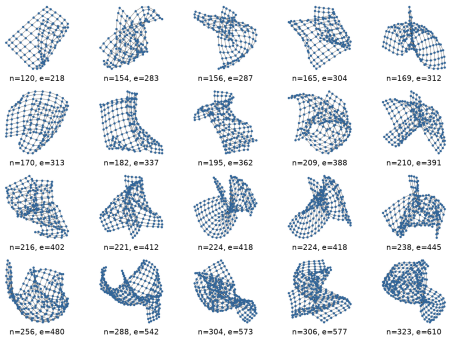
Figure 10: Component-wise guidance sensitivity on **Ego**, **Grid**, **COLLAB**, and **IMDB-BINARY**. Each panel reports mean±std over five seeds for degree, clustering, and spectral MMD. The plots show that guidance affects different graph statistics differently across datasets, so no single guidance scale is uniformly optimal.



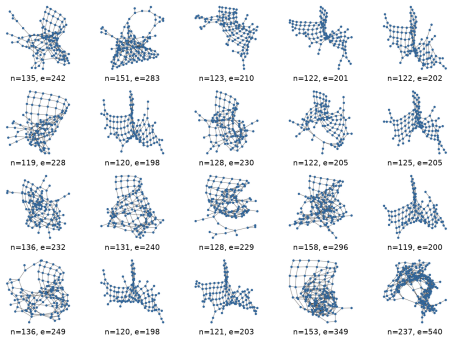
(a) Ego: original graphs.



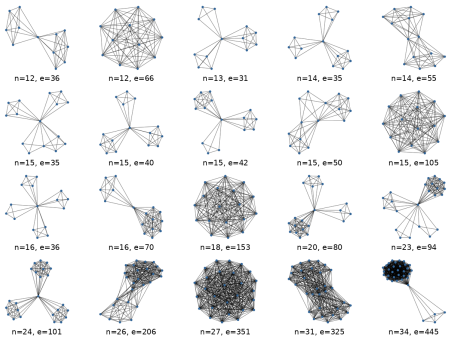
(b) Ego: generated graphs.



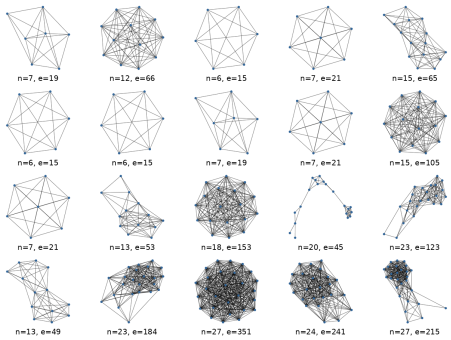
(c) Grid: original graphs.



(d) Grid: generated graphs.



(e) IMDB-BINARY: original graphs.



(f) IMDB-BINARY: generated graphs.

Figure 11: Qualitative visualizations of original and generated graphs.

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 claims in the abstract and introduction accurately reflect the paper's contributions and scope. They focus on mixed-curvature latent graph generation and Ricci-derived topology-aware priors, which correspond directly to the proposed model design and experimental analysis. The paper does not make claims beyond the evaluated setting of unattributed graph generation and standard structural distribution matching metrics.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: The paper discusses several limitations and scope boundaries of the proposed work. In particular, it clarifies that the current experiments focus on unattributed graph generation and standard structural distribution matching metrics, rather than broader settings such as attributed, molecular, or domain-constrained graph generation. It also acknowledges that the framework follows a two-stage latent generation pipeline, where the quality of diffusion generation depends on the learned graph autoencoder representation and decoder. In addition, the paper notes practical constraints in baseline coverage, especially for recent related methods without reproducible official implementations. These discussions help delimit the current contribution and indicate directions for future work.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.

- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

   Justification: For each theoretical result, the paper states the required assumptions before presenting the result and provides the corresponding proof, either in the main text or in the appendix. The theoretical claims are formulated within the stated setting of the proposed latent graph generation framework and are not used to claim guarantees beyond their assumptions. In particular, the results are presented as supporting analysis for the model design rather than as broad generalization guarantees. Thus, the assumptions, proof structure, and scope of each theoretical statement are made explicit.

   Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We provide comprehensive details regarding the model architecture, training hyperparameters, evaluation metrics, and dataset preprocessing to ensure full reproducibility. Detailed mathematical derivations of our method, the computation of the discrete curvature priors, and complete experimental setups (including exact hyperparameters, dataset statistics, and evaluation protocols) are fully disclosed in the Extended Methodology and Implementation Details in Appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

   Answer: [Yes]

   Justification: We have included the complete source code and data processing scripts in the supplementary material (ZIP file).

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: All training and testing details, including data splits, the optimizer used, and the complete hyperparameter settings, are specified in Appendix E.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.

- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: The paper reports experimental results with appropriate statistical information, such as results averaged over multiple runs and corresponding variance/error information where applicable.

   Guidelines:

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: The paper reports the compute resources used for the experiments, including that the experiments were conducted on a workstation equipped with an NVIDIA RTX 4090 GPU. This provides sufficient information about the main hardware setup needed to reproduce the experimental results.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

Justification: The research conforms to the NeurIPS Code of Ethics. It does not involve human subjects, sensitive personal data, deceptive practices, or applications that pose clear ethical risks, and the paper follows standard research integrity expectations such as reporting experimental settings and limitations.

Guidelines:

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

10. **Broader impacts**

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

Answer: [Yes]

Justification: The paper discusses potential positive societal impacts, such as improving graph generation methods for scientific and practical applications, and also acknowledges possible negative impacts, including misuse of generated graph data, biased or unreliable synthetic structures, and computational resource costs.

Guidelines:

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

11. **Safeguards**

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

Answer: [N/A]

Justification: The paper does not release high-risk models or datasets such as large pre-trained language models, image generators, or scraped datasets. Therefore, specific safeguards for responsible release are not applicable to this work.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.

- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: The paper properly credits the original sources of the datasets, baseline methods, and related code/models used in the experiments. The assets are used for academic research purposes, and their original ownership, citations, and applicable usage terms are acknowledged where relevant.

    Guidelines:

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

13. **New assets**

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

    Answer: [Yes]

    Justification: The new assets introduced by the paper, such as the proposed model implementation and experimental resources, are documented with sufficient descriptions of their usage, experimental settings, and reproduction steps. The documentation is provided alongside the released assets where applicable.

    Guidelines:

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

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

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

Answer: [N/A]

Justification: The paper does not involve crowdsourcing experiments or research with human subjects. Therefore, participant instructions, screenshots, and compensation details are not applicable.

Guidelines:

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

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

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

Answer: [N/A]

Justification: The paper does not involve human subjects or study participants. Therefore, participant risks, risk disclosure, and IRB or equivalent approval are not applicable to this work.

Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [N/A]

Justification: The paper does not use LLMs as an important, original, or non-standard component of the core methodology. If LLMs were used only for writing, editing, or formatting assistance, they do not affect the scientific method, rigor, or originality of the research and therefore do not require declaration.

Guidelines:

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