A Generative Framework for Exchangeable Graphs with Global and Local Latent Structure

Daniele Micheletti
Department of Statistics
University of California, Irvine
dmichele@uci.edu

Federica Zoe Ricci
Department of Mathematics and Statistics
Swarthmore College, Pennsylvania
fricci1@swarthmore.edu

Erik Sudderth
Department of Computer Science and Statistics
University of California, Irvine
sudderth@uci.edu

Abstract

We introduce a generative framework for exchangeable graphs that combines a Set Transformer-based encoder-decoder architecture with a hierarchical latent space composed of a global variable and node-specific variables. The global latent is modeled via diffusion process and acts as contextual input for node-level Gaussian mixtures. The decoder uses self-attention layers with global context injection to predict edge probabilities, ensuring high expressivity and full permutation invariance. The overall architecture ensures permutation invariance and can operate without node features, using the information in the adjacency matrix, which enables broad applicability beyond feature-rich domains. Through extensive experiments on synthetic benchmarks—including SBM, MMSBM, and the realistic LFR benchmark—we show that our approach accurately reproduces key graph statistics, especially for community-based networks. The global and local latent variables provide meaningful graph and node level context, while the architecture remains scalable to medium-sized dense graphs (e.g. 500-1000 nodes). Overall, our framework balances expressiveness, interpretability, and structural fidelity, offering a versatile tool for modeling complex graph data.

1 Introduction

Graphs are a fundamental and highly flexible data structure, widely used in fields such as biology (e.g., protein–protein interaction networks), social sciences (e.g., social networks), chemistry (e.g., molecular graphs), and transportation (e.g., road networks) [3]. While generative modeling has seen remarkable progress in domains such as text, images, and video [1, 24, 8, 19], developing generative models for graphs remains a crucial and challenging task due to their combinatorial structure and properties such as invariance to node relabeling.

Importantly, exchangeability is not a limitation but a desirable property in many contexts. Classical results from probability theory, such as the Aldous–Hoover representation theorem [2, 13], show that a wide range of well-known statistical models for networks and relational data naturally fall within the exchangeable framework, including Erdős–Rényi graphs [7], stochastic block models [12, 20], and latent space models [11]. Exchangeability ensures that the joint distribution of edges remains consistent under node relabeling, allowing the model to capture complex structural patterns without depending on any arbitrary ordering of the nodes.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: New Perspectives in Advancing Graph Machine Learning.

Building on early approaches such as the Variational Graph Autoencoder (VGAE) [16] and GraphRNN [28], many subsequent models have aimed to improve graph generation performance. Some methods extend the VAE framework, as in GraphVAE [23], while others adapt diffusion-based approaches, such as GDSS [14], EDGE [5], and DiGress [27], or employ flow-matching strategies like CatFlow [6]. These approaches vary in their assumptions and limitations: some are designed primarily for small graphs due to computational constraints, others treat the generation process as a fully black-box diffusion, some do not preserve node exchangeability, and several are specialized for molecular graphs where rich node and edge features are available.

In contrast, our model assumes only knowledge of the graph structure, i.e., the adjacency matrix, making it broadly applicable to graphs without node or edge attributes. It maintains an interpretable latent space, while offering strong generative performance, a flexible decoder architecture, and full permutation invariance. We show that this combination allows the model to capture both local and global graph structure, making it suitable for a wide range of graph types and sizes.

2 Background

In this section, we review the key concepts underlying our model. We denote data by x and latent variables by z.

Variational Autoencoders (VAEs) [15] learn a probabilistic map from a latent space to the data. They consist of a *prior* $p_\theta(z)$ over z, a *decoder* $p_\theta(x|z)$ defining a likelihood for x, and an *encoder* $q_\phi(z|x)$ that approximates the posterior $p(z|x)$. Typically $p_\theta(z) = \mathcal{N}(0, I)$, and the encoder outputs Gaussian parameters $\mu_\phi(x), \sigma^2_\phi(x)$. As in the original VAE paper [15], we use the *reparameterization trick* to make optimization differentiable, e.g. $z = \mu_\phi(x) + \sigma_\phi(x) \odot \epsilon,\ \epsilon \sim \mathcal{N}(0, I)$, where $\odot$ is elementwise multiplication. The model is then trained by maximizing the ELBO:

$$\mathcal{L}_{\text{ELBO}} = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - D_{\text{KL}}\big(q_\phi(z|x) \,\|\, p(z)\big).$$

Variational Graph Autoencoders (VGAEs) [16] adapt VAEs to graphs by using a Graph Neural Network (GNN) encoder and a decoder that predicts edge probabilities from inner products of node embeddings.

Diffusion Models [24, 25] learn the data distribution $p(x)$ by gradually denoising Gaussian noise. A fixed forward process $q(x_t|x_{t-1})$ corrupts data until it becomes pure noise, and a learned reverse process $p_\theta(x_{t-1}|x_t)$ removes noise to recover samples. Training minimizes

$$\mathcal{L}_{\text{DM}} = \mathbb{E}_{t,x_0,\epsilon}\Big[\big\|\epsilon - \epsilon_\theta(x_t, t)\big\|_2^2\Big],$$

where ϵ_θ predicts the noise in x_t. **Latent Diffusion Models** [21] apply this process in the latent space of a pretrained VAE, enabling sampling from $q_\phi(z|x)$ without needing x.

Set Transformer [18] is an attention-based architecture for sets, enforcing permutation invariance. Its core components are the *Self-Attention Block* (SAB) and *Pooling by Multihead Attention* (PMA), both built from the *Multihead Attention Block* (MAB), namely

$$\text{MAB}(X, Y) = \text{LayerNorm}\big(H + \text{rFF}(H)\big), \quad H = \text{LayerNorm}(X + \text{MHA}(X, Y, Y)),$$

where rFF is a row-wise feedforward network and MHA is Multihead Attention [26]. SAB applies $\text{MAB}(X, X)$ to model interactions within a set, while PMA uses $\text{MAB}(S, X)$ with p learnable seed vectors $S \in \mathbb{R}^{p \times d}$ to produce p pooled outputs (e.g., $p = 1$ for a single graph-level representation). To reduce attention complexity from $\mathcal{O}(n^2)$ to $\mathcal{O}(nm)$, one can use the *Induced Set Attention Block* (ISAB), which introduces m learnable inducing points I and computes $\text{MAB}(X, \text{MAB}(I, X))$.

Graph Normalization (GraphNorm) [4] is a normalization technique for Graph Neural Networks (GNNs) that normalizes node features within each graph using a learnable shift parameter, enhancing convergence speed and generalization performance. Mathematically,

$$\text{GraphNorm}(h_{i,j}) = \gamma_j \frac{h_{i,j} - \alpha_j \cdot \mu_j}{\hat{\sigma}_j} + \beta_j,$$

where $\mu_j = \frac{1}{n}\sum_{i=1}^{n} h_{i,j}, \quad \hat{\sigma}_j^2 = \frac{1}{n}\sum_{i=1}^{n}(h_{i,j} - \alpha_j \mu_j)^2$, β_j and γ_j are learnable affine parameters and α_j controls the fraction of the mean subtracted before scaling.

3 Method

We introduce a model that retains the overall structure of the VGAE while substantially modifying its components. We now describe each element in detail.

3.1 Preprocessing

We compute the normalized Laplacian of each graph and use its first d eigenvectors as node embeddings. No additional node features are required; the model relies solely on the adjacency structure. These embeddings are computed once before training and serve as input to the model.

3.2 Generative model

Prior. We define the latent space with a global variable z_0 for each graph and local variables z_i for each node $i \in [1, \dots, n]$. Specifically, we assume

$$p_\theta(z_0, z) = p_\theta(z_0) \prod_{i=1}^{n} p_\theta(z_i \mid z_0),$$

The global latent variable z_0 is initialized as $z_0 \sim \mathcal{N}(0, I)$ at the start of a Latent Diffusion Model. It is then evolved through a diffusion process, guided by a simple MLP score network, to generate samples from $q_\phi(z_0 \mid x)$. Conditioned on the global variable, the local variables are assumed a priori to be distributed according to a K-component Gaussian mixture (denoted GM_K), e.g

$$z_i \mid z_0 \sim \mathrm{GM}_K \left(\{\pi_k(z_0)\}_{k=1}^K,\ \{\mu_k(z_0)\}_{k=1}^K,\ \{\sigma_k^2(z_0)\}_{k=1}^K \right).$$

By conditioning the parameters of the node-level Gaussian mixture on z_0, the prior explicitly defines z_0 as a global context for all the nodes, expecting it to capture high-level graph properties.

Decoder. We propose an expressive, learnable decoder that leverages Self-Attention Blocks (SABs) from the Set Transformer, while injecting information from the global latent z_0. For each node i, an intermediate representation h_i is computed after L SAB layers. These representations incorporate both local node information and aggregated information from other nodes via attention, as well as global context derived from z_0. Specifically, the global latent is injected into each SAB layer by concatenating the node embedding h with a linearly transformed version of z_0 through a small MLP. Moreover, we alternate attention blocks with Graph Normalization [4] layers.

Edge probabilities between nodes i and j are then computed using a bilinear layer:

$$P(A_{ij} = 1 | z, z_0) = \sigma\Big(h_i^\top W(z_0)\, h_j + b(z_0)\Big),$$

where $W(z_0)$ and $b(z_0)$ are functions of the global latent, allowing the model to assign graph-specific parameters to the bilinear interaction. This design enables each edge prediction to depend on local node features, attention-weighted information from other nodes, and graph-level context provided by z_0, making the decoder highly expressive and adaptable to the structure of each graph. The whole architecture satisfies permutation invariance as it involves only row-wise or set operations. This means that permuting the nodes in the input of the decoder will result in the same predicted probabilities.

3.3 Encoder

We replace the GNN encoder of the VGAE with a Set Transformer-inspired encoder $q(z_0, z \mid x)$. The encoder first computes node embeddings from the graph Laplacian eigenvectors. These embeddings are processed through alternating Self-Attention Blocks (SABs) and Graph Normalization layers to produce intermediate node representations.

Node-level latent variables z_i are then obtained by applying ResNets [10] to the final node representations from the last SAB and GraphNorm layer, predicting the mean and variance of a Gaussian distribution for each node. The global latent z_0 is computed by pooling the node representations with a Pooling by Multihead Attention (PMA) block, followed by ResNets that predict its Gaussian parameters.

Permutation invariance is maintained throughout, as all operations are either applied row-wise to individual nodes or are inherently set-based.

3.4 Training and Loss

Training occurs in two stages. First, the VAE is optimized using the ELBO, extended to account for the global latent, i.e., $q(z, z_0 \mid x)$ instead of the standard $q(z \mid x)$. Given that the prior over the node-level latents involves a Gaussian Mixture, part of the KL term (the cross term with the prior) requires Monte Carlo estimation, as it has no closed form. Second, the latent diffusion score network is trained using the encoded z_0 as target data, minimizing the corresponding diffusion loss.

4 Experiments and results

This section presents experiments on both well-known and synthetic datasets. In all experiments, we set $d = 20$, i.e., we use the first 20 eigenvectors of the normalized Laplacian as input for the model. Moreover, we split the Community, MMSBM and LFR datasets into 80% for training and 20% for testing.

Community First, we conduct a preliminary evaluation of our model on the *Community* dataset [28] (see Appendix A.1), comparing it against GraphRNN using the MMD [9] between degree, clustering coefficient, and average orbit count statistics. Table 1 shows that our model consistently improves all metrics on *Community*.

Table 1: Comparison of our model with GraphRNN on the Community dataset using MMD. The maximum number of nodes and edges is indicated in brackets. The results for the baseline models are taken from the original GraphRNN paper [28].

	Community (160,1945)		
	Degree	Clustering	Orbit
GraphRNN-S	0.02	0.15	0.01
GraphRNN	0.03	0.03	0.01
Our Model	**0.003**	**0.015**	**0.00001**

MMSBM We evaluate our model on the more challenging Mixed Membership Stochastic Block-models (MMSBM) after confirming good performance on standard SBMs. We simulated 1,000 graphs with node counts drawn from a Poisson distribution (mean 500), each with a 50% chance of having 3 or 5 clusters. Cluster sizes and inter-/intra-community connection probabilities vary across graphs (details in Appendix A.2). This high variability makes MMSBM a challenging test of model flexibility. Figure 1 shows that the model accurately approximates the degree distribution and that the global variable is able to differentiate between the 3-cluster and the 5-cluster graphs

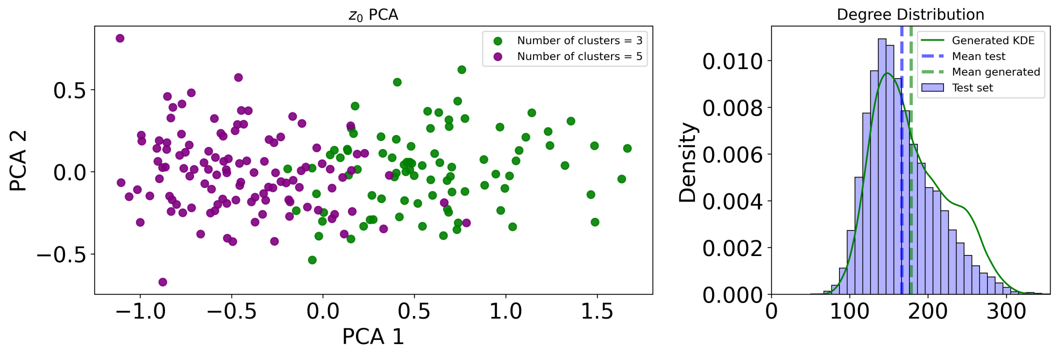

Figure 1: Latent representation and degree distribution for MMSBM graphs. *Left*: 2D PCA projection of the global latent variable z_0, where each point represents a graph and colors indicate whether it has 3 or 5 clusters. *Right*: degree distributions for test graphs (histogram) and generated graphs (KDE), with dashed vertical lines showing their mean degrees.

LFR. We demonstrate that our model can generate realistic graphs and that the global latent variable captures meaningful graph-level information. In order to do that, we use the Lancichinetti–Fortunato–Radicchi (LFR) benchmark [17]. The LFR is commonly regarded as a realistic and challenging benchmark, reproducing heavy-tailed degree and community distributions commonly observed in real-world networks [22]. We generate 3,000 graphs with 500 nodes each. Specifically, we generate 500 graphs for each of six settings varying based on the fraction of inter-community edges $\mu \in \{0.1, 0.3, 0.5\}$ and the maximum degree $k_{\max} \in \{100, 200\}$. The choice of μ allows us to explore graphs with different levels of community strength, ranging from well-separated to highly mixed communities, while different combinations of μ and $k_{\max}$ induce variability in the number of communities in the graphs. For generating all graphs, we set the power-law degree exponent to $\tau_1 = 2$, the power-law exponent of the community size distribution to $\tau_2 = 1$ and the average degree to $\bar{d} = 50$. Figure 2 shows that our model is able to correctly approximate the degree distribution of the test data and that the global variable is able to capture different graph-level information, as the number of clusters and the maximum degree. For additional results on this experiment, see Appendix A.3.

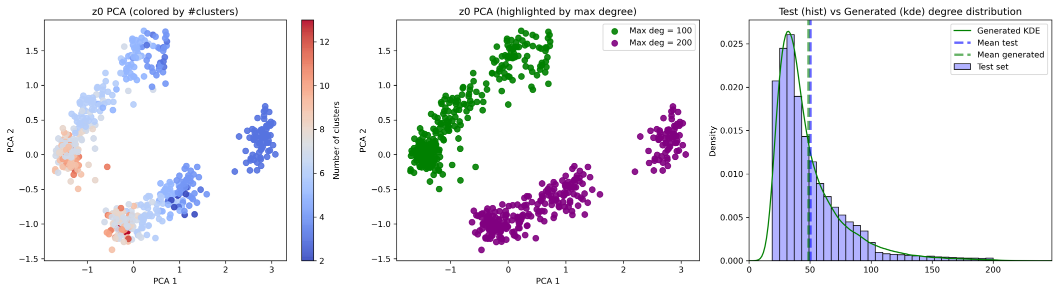

Figure 2: Latent representation analysis and degree distribution for LFR graphs. *Left*: 2D PCA projection of the global latent variable z_0, with each point representing a graph and colored by the number of clusters. *Center*: same PCA projection, now highlighting graphs with maximum degree equal to 100 (green) or 200 (purple). *Right*: comparison between the degree distribution of test graphs (histogram) and generated graphs (KDE), with dashed lines indicating their means.

5 Conclusion

We presented a set transformer-based generative framework for exchangeable graphs, combining a Gaussian mixture prior with a global latent variable evolved through latent diffusion. This design allows the model to capture both local and global structure while maintaining permutation invariance and interpretability. Our experiments on synthetic but challenging and realistic benchmarks, including MMSBM and LFR, show that the approach accurately reproduces key graph statistics, such as degree distributions, and that the global latent encodes meaningful graph-level information.

While promising, our method has limitations: scalability to very large graphs remains challenging, and performance may degrade on strongly non-exchangeable graph families. Future work will focus on scaling up using inducing-point variants of the Set Transformer, exploring sparse attention mechanisms to handle large, sparse graphs, and extending the framework to partially exchangeable or more structured graph settings.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] David J Aldous. Representations for partially exchangeable arrays of random variables. *Journal of Multivariate Analysis*, 11(4):581–598, 1981.

[3] Markus Brede. Networks—an introduction. mark ej newman.(2010, oxford university press.) (hardcover), 772 pages. isbn-978-0-19-920665-0., 2012.

[4] Tianle Cai, Shengjie Luo, Keyulu Xu, Di He, Tie-yan Liu, and Liwei Wang. Graphnorm: A principled approach to accelerating graph neural network training. In *International Conference on Machine Learning*, pages 1204–1215. PMLR, 2021.

[5] Xiaohui Chen, Jiaxing He, Xu Han, and Li-Ping Liu. Efficient and degree-guided graph generation via discrete diffusion modeling. *arXiv preprint arXiv:2305.04111*, 2023.

[6] Floor Eijkelboom, Grigory Bartosh, Christian Andersson Naesseth, Max Welling, and Jan-Willem van de Meent. Variational flow matching for graph generation. *Advances in Neural Information Processing Systems*, 37:11735–11764, 2024.

[7] P ERDdS and A R&wi. On random graphs i. *Publ. math. debrecen*, 6(290-297):18, 1959.

[8] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[9] Arthur Gretton, Karsten Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alex Smola. A kernel method for the two-sample-problem. *Advances in neural information processing systems*, 19, 2006.

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

[11] Peter D Hoff, Adrian E Raftery, and Mark S Handcock. Latent space approaches to social network analysis. *Journal of the american Statistical association*, 97(460):1090–1098, 2002.

[12] Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983.

[13] Douglas N Hoover. Relations on probability spaces and arrays of. *Preprint, Institute for Advanced Study*, 1979.

[14] 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*, pages 10362–10383. PMLR, 2022.

[15] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[16] Thomas N Kipf and Max Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.

[17] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics*, 78(4):046110, 2008.

[18] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International conference on machine learning*, pages 3744–3753. PMLR, 2019.

[19] Yixin Liu, Kai Zhang, Yuan Li, Zhiling Yan, Chujie Gao, Ruoxi Chen, Zhengqing Yuan, Yue Huang, Hanchi Sun, Jianfeng Gao, et al. Sora: A review on background, technology, limitations, and opportunities of large vision models. *arXiv preprint arXiv:2402.17177*, 2024.

[20] Krzysztof Nowicki and Tom A B Snijders. Estimation and prediction for stochastic blockstructures. *Journal of the American statistical association*, 96(455):1077–1087, 2001.

[21] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[22] Hua-Wei Shen. *Community structure of complex networks*. Springer Science & Business Media, 2013.

[23] Martin Simonovsky and Nikos Komodakis. Graphvae: Towards generation of small graphs using variational autoencoders. In *International conference on artificial neural networks*, pages 412–422. Springer, 2018.

[24] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. pmlr, 2015.

[25] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[27] Clement Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. Digress: Discrete denoising diffusion for graph generation. *arXiv preprint arXiv:2209.14734*, 2022.

[28] Jiaxuan You, Rex Ying, Xiang Ren, William Hamilton, and Jure Leskovec. Graphrnn: Generating realistic graphs with deep auto-regressive models. In *International conference on machine learning*, pages 5708–5717. PMLR, 2018.

A Experiments details

A.1 Community

Data description. The dataset consists of 500 two-community graphs from 60 to 160 nodes, with each community formed as an Erdős–Rényi graph ($p = 0.3$) and $0.05|V|$ inter-community edges added uniformly at random.

Latent space and reconstruction. Since all the graphs belong to the same type (2 communities and same inter- and intra-community probabilities) the global variable is not useful in this case. The local variable is able to cluster the communities in the latent space. Moreover, the model is able to reconstruct the edge probabilities accurately as shown in Figure 3.

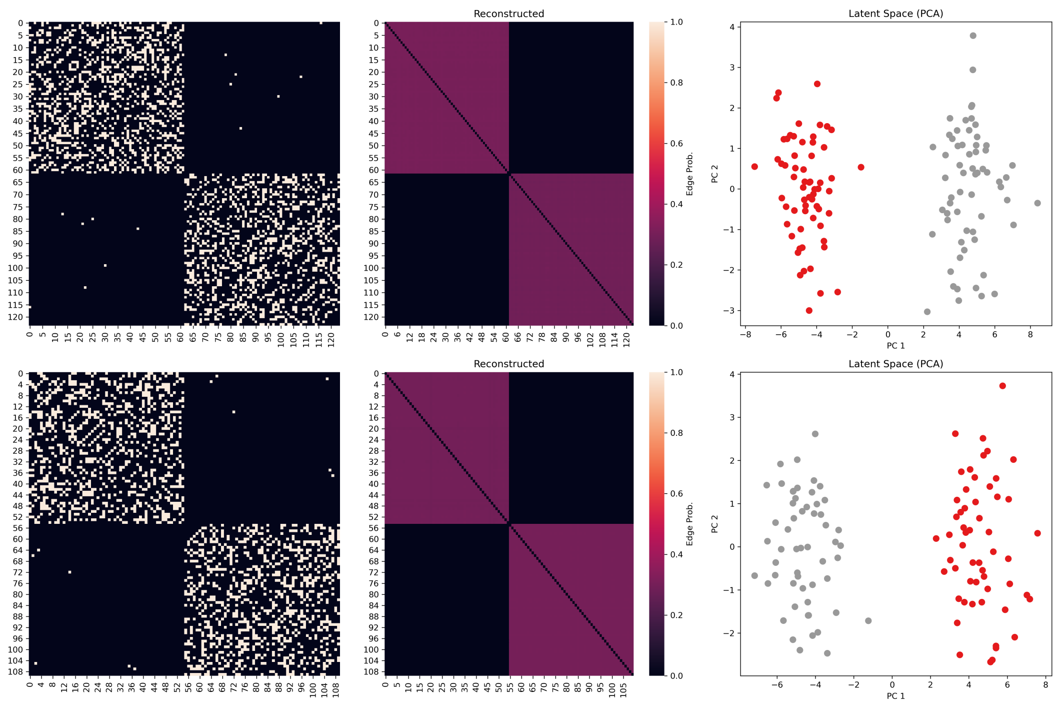

Figure 3: Visualization of two graphs from the Community test data. *Left*: true adjacency matrices ordered by community membership. *Center*: reconstructed edge probabilities from the model. Notice the true value for the intra-community probability is 0.3. *Right*: 2D PCA projection of the latent node representations, where each point corresponds to a node and colors indicate its true community.

A.2 MMSBM

Data generating process. The Mixed Membership Stochastic Block Model (MMSBM) follows a hierarchical generative process. Each graph has K clusters. Cluster membership proportions for each node are drawn from a Dirichlet distribution:

$$\alpha \sim \text{Dirichlet}(B, \dots, B), \quad \pi_i \sim \text{Dirichlet}(\alpha),$$

where we set $B = 5$. The edge probabilities between nodes depend on cluster assignments: for clusters l and k, the intra- and inter-cluster connection probabilities θ_{lk} are drawn from Beta distributions:

$$\theta_{lk} \sim \begin{cases} \text{Beta}(60, 10), & \text{if } l = k \text{ (within-cluster)} \\ \text{Beta}(5, 40), & \text{if } l \neq k \text{ (between-cluster)} \end{cases}.$$

For each graph, we choose $K \in \{3, 5\}$ with probability 0.5.

Latent space and reconstruction (MMSBM). As Figure 4 shows, the model successfully recovers the community structure: the reconstructed adjacency shows clear intra- and inter-community patterns. In the latent space of z, the model is able to distinguish the different clusters. However, in this case, the clusters are not sharply separated, reflecting the soft memberships of the MMSBM, which is an expected and desirable behavior.

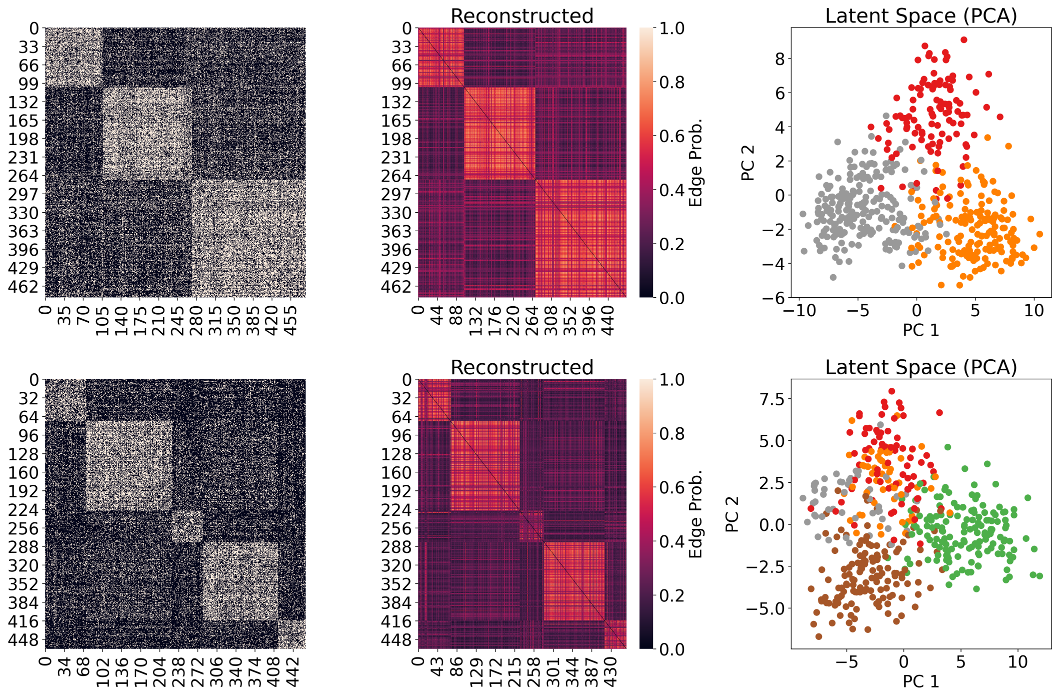

Figure 4: Visualization of two MMSBM graphs from the test data. *Left*: true adjacency matrices ordered by community membership. *Center*: reconstructed edge probabilities from the model. *Right*: 2D PCA projection of the latent node representations, where each point corresponds to a node and colors indicate its true community.

A.3 LFR

Data description. The LFR benchmark produces heterogeneous networks with heavy-tailed degree and community size distributions. Full details of the generation process are provided in Section 4.

Latent space and reconstruction (LFR). Figure 5 illustrates the model's reconstruction and the latent space of nodes for LFR graphs. The reconstructed adjacency matrices reveal clearly connected communities with sparser inter-community links. In the latent space of z, nodes from the same community tend to cluster together, emphasizing the community structure. Nonetheless, due to the more complex and heterogeneous nature of LFR graphs, further interpretation of the latent geometry is less straightforward.

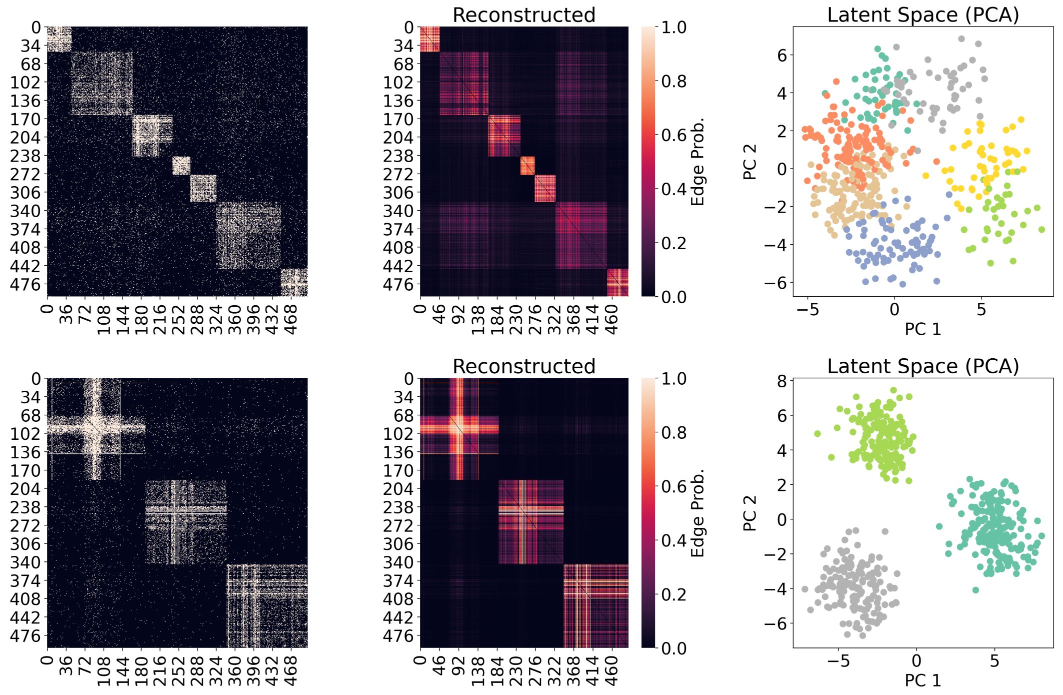

Figure 5: Visualization of two LFR graphs from the test data. *Left*: true adjacency matrices ordered by community membership. *Center*: reconstructed edge probabilities from the model. *Right*: 2D PCA projection of the latent node representations, where each point corresponds to a node and colors indicate its true community.

Table 2: Model architecture and training settings across experiments.

Setting	**Community**	**MMSBM**	**LFR**
Hidden dimension	24	32	24
Node latent dimension (z)	10	10	10
Global latent dimension (z_0)	10	10	10
SAB layers (Encoder / Decoder)	4 / 4	4 / 4	3 / 3
Batch size	10	10	5
Epochs	20	10	5
Attention heads	4	4	4
Gaussian mixture components (K)	10	10	10
Score network hidden dimension	128	128	128
Training time	$\sim$5 min	$\sim$27 min	$\sim$53 min