Conditional Graph Generation with Graph Principal Flow Network

Tianze Luo [*1] Zhanfeng Mo [*1] Sinno Jialin Pan [1 2]

Abstract

Conditional graph generation is crucial and challenging since the conditional distribution of graph topology and feature is complicated and the semantic feature is hard to be captured by the generative model. In this work, we propose a novel graph conditional generative model, termed Graph Principal Flow Network (GPrinFlowNet), which enables us to progressively generate graphs from low- to high-frequency components. Our GPrinFlowNet effectively captures the subtle yet essential semantic features of graph topology, resulting in high-quality generated graph data.

1. Introduction

The task of conditional graph generation is crucial in various domains such as automatic compound discovery, drug design, and more (Zang & Wang, 2020; Shi et al., 2020; Yang et al., 2020; Wang et al., 2019; Liu et al., 2021). It requires one to generate graph data conditioned on a specific graph label, e.g. graph property, category. In general, a graph data with n node is defined as $\mathbf{G} \triangleq (\mathbf{X}, \mathbf{A}, y)$, where $\mathbf{X} \in \mathcal{X} \subset \mathbb{R}^{n\times d}$ is the node feature matrix, $\mathbf{A} \in \mathcal{A} \subset \mathbb{R}^{n\times n}$ is the graph adjacency matrix, and $y \in \mathcal{Y}$ is the graph label. Suppose the target graph distribution of interest is $\mathcal{G}$, and each graph is sampled from $\mathbf{G} \sim \mathcal{G}$, the goal of conditional generation is to learn a generative model $g(\cdot;\cdot) : (\mathcal{X} \times \mathcal{A}) \times \mathcal{Y} \mapsto (\mathcal{X} \times \mathcal{A})$, such that for each graph label $\forall y \in \mathcal{Y}$, the distribution of $g(\boldsymbol{\epsilon}; y)$ approximates the conditional graph distribution $\mathbf{G}|y$ well, where $\boldsymbol{\epsilon}$ is a noise sampled from a known prior π.

While there is a considerable amount of work and literature dedicated to unconditional graph generation (Zhu et al., 2022), the field of conditional graph data generation is relatively understudied. The main challenge in conditional graph data generation arises from two factors: Firstly, the conditional graph distribution is highly complicated, as the relationship between graph features and topology varies significantly across different graph labels. Secondly, conditional datasets typically consist of fewer data points, leading to a higher demand for the effectiveness of the learning model due to data scarcity.

Admittedly, unconditional models can be transformed into conditional generators by integrating a graph label embedding module, similar to conditional generative vision models (Ho & Salimans, 2021). However, existing unconditional graph generation models have inherent limitations, making them unsuitable for unconditional generation. Likelihood-based models, like (Simonovsky & Komodakis, 2018; Ma et al., 2018; De Cao & Kipf, 2018), estimate the likelihood function of the underlying graph data distribution to generate samples. Yet, these models struggle with complex graph structures and computational burdens, making them less suitable for conditional generation. Another class, diffusion-based models like (Niu et al., 2020b; Jo et al., 2022; Luo et al., 2022), illustrates the state-of-the-art performance in unconditional generation by denoising graph data through reverse diffusion SDE. However, noise insertion in the diffusion process can corrupt semantic information, hindering their ability to capture crucial graph label modes. Hence, diffusion-based models fall short as conditional generation models.

In this work, we leverage graph spectral theory to enhance the learning of subtle yet crucial semantic features. Instead of fitting the likelihood function or recovering the graph from uniform noise, our approach involves progressive graph generation from low to high-frequency components. The low-frequency components correspond to the smallest principal components of the graph Laplacian matrix. This step-by-step approach enables coarse-to-fine learning of the graph. As shown in Figure 1, despite a minor difference in the connection between the clusters, the upper and bottom graphs have distinct graph labels with different topological properties and connectivity. The low-frequency component (on the right) successfully discriminates the connectivity, while the diffusion process results in a completely blurred adjacency matrix, failing to distinguish between them. Thus, the low-frequency component is an ideal starting point for

[*]Co-first authors with equal contributions. [1]Nanyang Technological University, Singapore. [2]Chinese University of Hong Kong. E-mail: {tianze001,zhanfeng001}@ntu.edu.sg; sinnopan@cse.cuhk.hk.

Proceedings of the 40^{th} International Conference on Machine Learning workshop on Structured Probabilistic Inference Generative Modeling, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

unconditional graph generation as it has a smoother distribution, is easier to learn, and captures subtle yet crucial semantic features of the graph label.

Building upon recent advancements in generative modeling, specifically the Generative Flow Network (GFlowNet) (Bengio et al., 2021a;b), we propose a novel framework called Graph Principal Flow Net (GPrinFlowNet). This framework facilitates conditional graph generation by employing a step-by-step coarse-to-fine approach. In the language of GFlowNet, this progressive generation can be understood as a Markov chain, where the k-th intermediate state represents the graph adjacency reconstructed from the k lowest principal components of the graph Laplacian. Unlike GFlowNet, which focuses on discrete probabilistic modeling, GPrinFlowNet simultaneously learns the continuous-valued graph feature and eigenvalues of the graph Laplacian.

2. Fundamentals of GFlowNet

Several pivotal developments in generative modeling have led to the emergence of Generative Flow Network (GFlowNet), an advanced model for probabilistic inference (Bengio et al., 2021a; Deleu et al., 2022; Zhang et al., 2022b;a; Pan et al., 2023; Bengio et al., 2021b). GFlowNet employs a reward function $R(\mathbf{x}) \in \mathbb{R}_+$ to efficiently sample data $\mathbf{x}$ from the data space $\mathcal{X}$. Notably, GFlowNet's sampling protocol follows a Markovian trajectory $\tau = (\mathbf{s}_0, \mathbf{s}_1, ..., \mathbf{s}_n)$, where $\mathbf{s}_0$ is the initial state, $\mathbf{s}_i$ is the i-th hidden state, and $\mathbf{s}_n$ is the terminating state with $\mathbf{s}_n = \mathbf{x}$. It's important to note that each hidden state $\mathbf{s}_i$ is derived from the state space $\mathcal{S}$, which may differ from the data space $\mathcal{X}$. Furthermore, the sampling trajectories of GFlowNet, denoted as $\mathcal{T}$, form a Directed Acyclic Graph (DAG), with each node representing a hidden state $\mathbf{s} \in \mathcal{S}$. As highlighted in (Bengio et al., 2021b), the sampling process of GFlowNet is governed by the flow function $F(\cdot)$. This function ensures that the measure of incoming trajectories at each hidden state is equal to the measure of outgoing trajectories. Our primary objective is to learn a flow function $F(\cdot)$ such that the total mass of trajectories terminating at $\mathbf{x}$ is proportional to the reward $R(\mathbf{x})$, mathematically expressed as $\sum_{\tau:\mathbf{s}_n=\mathbf{x}} F(\mathbf{x}) = R(\mathbf{x})$.

In reference to the work of Bengio et al. (Bengio et al., 2021a), three principal supervisons have been introduced for training GFlowNets. These encompass the flow matching condition (Bengio et al., 2021a) as outlined in the same study, the detailed balance condition (Bengio et al., 2021b), and the trajectory balance condition (Malkin et al., 2022). All of these conditions aim to depict the conservation law of flow mass from different levels of granularity. A definition for the flow matching condition is

$$\sum_{\mathbf{s}_{i-1}} F_{\boldsymbol{\theta}}(\mathbf{s}_{i-1}, \mathbf{s}_i) = \sum_{\mathbf{s}_{i+1}} F_{\boldsymbol{\theta}}(\mathbf{s}_i, \mathbf{s}_{i+1}), \tag{1}$$

where $F_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{s}') \triangleq \sum_{(\mathbf{s},\mathbf{s}')\in\tau} F_{\boldsymbol{\theta}}(\tau)$ is the learnable edge flow function. It promotes equality between the masses of incoming and outgoing edge flows. The detailed balanced condition is given by

$$F_{\boldsymbol{\theta}}(\mathbf{s}_i) P_{F,\boldsymbol{\theta}}(\mathbf{s}_{i+1}|\mathbf{s}_i) = F_{\boldsymbol{\theta}}(\mathbf{s}_{i+1}) P_{B,\boldsymbol{\theta}}(\mathbf{s}_i|\mathbf{s}_{i+1}), \tag{2}$$

where $P_{F,\boldsymbol{\phi}}$ and $P_{B,\boldsymbol{\phi}}$ represent the forward and backward transition probabilities. It aims to achieve equal mass for forward and backward transitions between two consecutive states. To further accelerate the convergence and improve the performance of GFlowNet, the trajectory balance objective extends the detailed balance criterion to an entire trajectory by matching the forward and backward trajectory probabilities $P_{F,\boldsymbol{\theta}}(\tau) \triangleq \prod_{i=0}^{n-1} P_{F,\boldsymbol{\theta}}(\mathbf{s}_{i+1}|\mathbf{s}_i)$ and $P_{B,\boldsymbol{\theta}}(\tau) \triangleq \frac{R(x)}{Z_{\boldsymbol{\theta}}} \prod_{i=0}^{n-1} P_{B,\boldsymbol{\theta}}(\mathbf{s}_i|\mathbf{s}_{i+1})$, where $Z_{\boldsymbol{\theta}}$ is a learnable normalization constant.

3. Methodology

3.1. Graph Principal Flow Network

Assume that each conditional graph instance $\mathbf{G}|y$ is generated by a Markov sample path $\tau \triangleq (\mathbf{s}_0|y, ..., \mathbf{s}_{n-1}|y, \mathbf{G}|y)$, and our ultimate goal is to sample in proportion to the fidelity of $\mathbf{G}|y$. To facilitate the learning process of the transition policies between each successive hidden states $(\mathbf{s}_i|y, \mathbf{s}_{i+1}|y)$, we introduce the following Graph Principal Flow Network. We define the graph Laplacian associated to the adjacency matrix $\mathbf{A}$ as $\mathbf{L} \triangleq \mathbf{D} - \mathbf{A}$, where $\mathbf{D}$ is the diagonal degree matrix defined by $\mathbf{D}[i,i] \triangleq \sum_{j=1}^n \mathbf{A}[i,j]$. We denote the eigen decomposition of $\mathbf{L}$ as $\mathbf{L} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\top$, where $\lambda_i \triangleq \boldsymbol{\Lambda}[i,i]$ is the i-th smallest eigenvalue, and $\mathbf{U}[:,i]$ is the corresponding eigenvector.

Definition 3.1 (Graph Principal Flow Network)**.** Suppose $d(\cdot,\cdot) : (\mathcal{X} \times \mathcal{A}) \times (\mathcal{X} \times \mathcal{A}) \mapsto \mathbb{R}_+$ is a graph discrepancy score, $\mathcal{T}$ is the Graph Principal trajectory space, where each trajectory $\tau \in \mathcal{T}$ is defined by

$$\tau \triangleq (\mathbf{s}_0|y, ..., \mathbf{s}_{n-1}|y, \mathbf{G}|y),\ \forall y \in \mathcal{Y}, \tag{3}$$

$$\mathbf{G}|y \triangleq (\mathbf{X}, \mathbf{A})|y,\ \mathbf{s}_{n-1}|y \triangleq (\mathbf{X}_{n-1}, \mathbf{A}_{n-1})|y. \tag{4}$$

We assume that the transition between each $(\mathbf{s}_i|y, \mathbf{s}_{i+1}|y)$ follows

$$(\mathbf{s}_{i+1}|\mathbf{s}_i, y) \sim P_{F,\boldsymbol{\theta}}(\mathbf{s}_{i+1}|\mathbf{s}_i; y), \tag{5}$$

$$(\mathbf{s}_i|\mathbf{s}_{i+1}, y) \sim P_{B,\boldsymbol{\theta}}(\mathbf{s}_i|\mathbf{s}_{i+1}; y). \tag{6}$$

Here, $P_{F,\boldsymbol{\theta}}$ and $P_{B,\boldsymbol{\theta}}$ represent the learnable forward and backward transition kernels, which are parameterized by

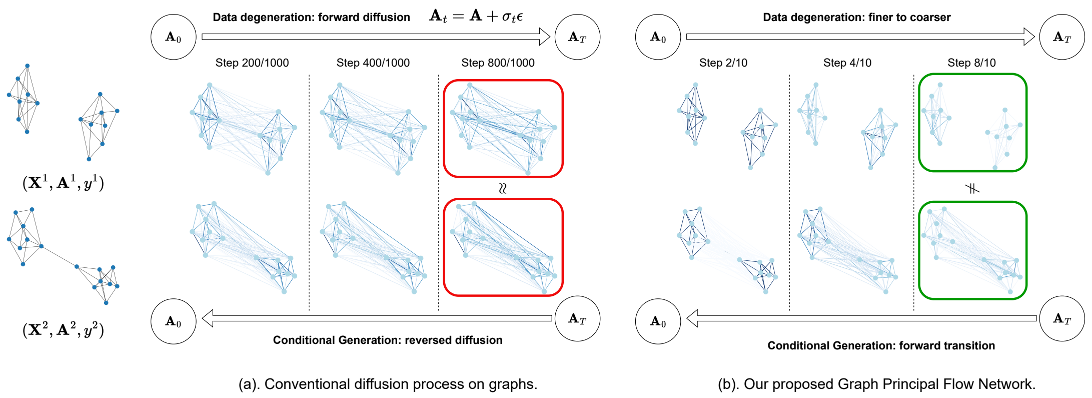
Data degeneration: forward diffusion $\mathbf{A}_t = \mathbf{A} + \sigma_t \epsilon$
$\mathbf{A}_0$
$\mathbf{A}_T$
Step 200/1000
Step 400/1000
Step 800/1000
$(\mathbf{X}^1, \mathbf{A}^1, y^1)$
$(\mathbf{X}^2, \mathbf{A}^2, y^2)$
Conditional Generation: reversed diffusion
Data degeneration: finer to coarser
Step 2/10
Step 4/10
Step 8/10
Conditional Generation: forward transition
(a). Conventional diffusion process on graphs.
(b). Our proposed Graph Principal Flow Network.

Figure 1: A comparison between the degeneration of graph adjacency across different data degeneration states of diffusion-based models and our GPrinFlowNet. The low-frequency component (green box) proficiently captures the subtle yet crucial patterns that are discriminative to graph labels.

$\boldsymbol{\theta}$. Then $(d, \mathcal{T}, P_{F,\boldsymbol{\theta}}, P_{B,\boldsymbol{\theta}})$ forms a Graph Principal Flow Network (GPrinFlowNet), if there exists a sequence of normalizers $\{Z_i\}_{i=1}^n$ such that

$$P_{F,\boldsymbol{\theta}}(\mathbf{s}_{i+1}|\mathbf{s}_i; y)\frac{R(\mathbf{s}_i|y)}{Z_i} = P_{B,\boldsymbol{\theta}}(\mathbf{s}_i|\mathbf{s}_{i+1}; y)\frac{R(\mathbf{s}_{i+1}|y)}{Z_{i+1}},$$
$$R(\mathbf{s}_i|y) \triangleq \exp(-d((\mathbf{X}_i, \mathbf{A}_i), (\mathbf{X}, \mathbf{A}_{(i)}))) \tag{7}$$

Here, $\mathbf{A}_{(i)} \in \mathcal{A}$ is the i-th level granularity reconstruction of $\mathbf{A}$, that is

$$\mathbf{A}_{(i)}[i,j] \triangleq (\mathbf{U}\boldsymbol{\Lambda}_i\mathbf{U}^\top)[i,j] \cdot \delta_{ij}, \tag{8}$$
$$\boldsymbol{\Lambda}_i \triangleq \mathrm{diag}(\lambda_1, ..., \lambda_i, 0, ..., 0),$$

where the self-loop in $\mathbf{A}$ is omitted.

GPrinFlowNet stands apart from the standard GFlowNet in two crucial ways: Firstly, each hidden state of GPrinFlowNet resides within a continuous-valued space. Secondly, as demonstrated in (7), at the i-th intermediate step, the distribution of the generated graph adjacency aligns with the distribution of $\mathbf{A}_{(i)}$ at the corresponding granularity level. This alignment effectively steers GPrinFlowNet to generate conditional graph data incrementally, from lower to higher frequency components.

We propose an effective parameterization and training objective to train a GPrinFlowNet. Specifically, we parameterize the forward and the backward transition kernels as learnable Gaussian distributions, i.e.

$$P_{F,\boldsymbol{\theta}}(\mathbf{s}|\mathbf{s}'; y) \triangleq N(\mathbf{s}; \mu_{F,\boldsymbol{\theta}}(\mathbf{s}', y), \Sigma_{F,\boldsymbol{\theta}}(\mathbf{s}', y)), \tag{9}$$
$$P_{B,\boldsymbol{\theta}}(\mathbf{s}|\mathbf{s}'; y) \triangleq N(\mathbf{s}; \mu_{B,\boldsymbol{\theta}}(\mathbf{s}', y), \Sigma_{B,\boldsymbol{\theta}}(\mathbf{s}', y)), \tag{10}$$

where $\mathbf{s} \triangleq (\mathbf{X}, \boldsymbol{\Lambda}, \mathbf{U})$, and the mean and covariance are learned by a multi-layer Graph Convolutional Network (GCN) (Defferrard et al., 2016). Now that the transition probabilities have explicit expression, we can train the GPrinFlowNet by minimizing the following Graph Principal Trajectory Balance objective, which is defined as

$$\mathcal{L}(\boldsymbol{\theta}; \tau) \triangleq \sum_{i=0}^{n-1}\left(\log \frac{Z_{i,\boldsymbol{\theta}} \prod_{j=0}^{i-1} P_{F,\boldsymbol{\theta}}(\mathbf{s}_{j+1}|\mathbf{s}_j; y)}{R(\mathbf{s}_i|y)\prod_{j=0}^{i-1} P_{B,\boldsymbol{\theta}}(\mathbf{s}_j|\mathbf{s}_{j+1}; y)}\right)^2, \tag{11}$$

where the normalizers $\{Z_{i,\boldsymbol{\theta}}\}_{i=1}^n$ are trainable scalars.

3.2. Conditional Generation with GPrinFlowNet

With a well-trained GPrinFlowNet, we can efficiently generate high-quality conditional graph data in a maximum of n steps, substantially fewer than the steps required by diffusion-based models. The conditional generation process is detailed in Algorithm 1.

Algorithm 1 Conditional Generation with GPrinFlowNet

```
Input: training data $\mathbf{S}$, a target label $y$, the forward and the backward transition policy networks $P_{F,\boldsymbol{\theta}}$ and $P_{B,\boldsymbol{\theta}}$, the conditional and normalized average eigenvector matrix $\bar{\mathbf{U}}|y$ and feature matrix $\bar{\mathbf{X}}|y$, a temperature hyperparameter $\sigma > 0$.
Output: a plausible conditional graph data $(\widehat{\mathbf{X}}, \widehat{\mathbf{A}})$
Sample $\mathbf{U} \sim N(\bar{\mathbf{U}}|y, \sigma)$, $\mathbf{X}_0 \sim N(\bar{\mathbf{X}}|y, \sigma)$
Initialize $\mathbf{s}_0 \leftarrow (\mathbf{X}_0, \mathbf{0}, \mathbf{U})$
for $i = 0$ to $n-1$ do
  $\mathbf{s}_{i+1} \sim P_{F,\boldsymbol{\theta}}(\cdot|\mathbf{s}_i, y)$ {Forward transition}
end for
$(\widehat{\mathbf{X}}, \widehat{\mathbf{A}}) \leftarrow (\mathbf{X}_n, \mathbf{U}\boldsymbol{\Lambda}_n\mathbf{U}^\top)$
return $(\widehat{\mathbf{X}}, \widehat{\mathbf{A}})$
```

4. Experiments

4.1. Conditional Graph Generation

Baselines and datasets. We compare our method with the state-of-the-art graph generation method, including graph diffusion methods such as GDSS (Jo et al., 2022), EDP-GNN (Niu et al., 2020a); VAE-based methods such as GraphVAE (Simonovsky & Komodakis, 2018); autoregressive models such as GraphAF (Shi et al., 2020), GraphDF (Luo et al., 2021), and GraphRNN (You et al., 2018). Although these existing methods focus on unconditional generation, we effectively modify and extend them for conditional generation by integrating a graph label embedding module, mirroring the approach we employed in GPrinFlow. We adopt the AIDS (Morris et al., 2020), Enzymes (Schomburg et al., 2004), and Synthie datasets (Feragen et al., 2013) for graph conditional generation. More details of the datasets and the evaluation metric are included in Appendix A.

Results and analysis. Following the graph generation evaluation setting (Jo et al., 2022), for each category, we adopt the same train versus test split ratio as (Jo et al., 2022). We measure the maximum mean discrepancy (MMD) to compare the distributions of graph statistics between the same number of generated and test graphs under each category, and report the mean MMD as our overall evaluation score. As shown in Figure 4, our proposed method turns out to be the best performance among the state-of-the-art graph generation baselines.

	AIDS Avg. MMD↓	Enzymes Avg. MMD↓	Synthie Avg. MMD↓
GraphRNN (You et al., 2018)	0.139	0.307	0.317
GraphAF (Shi et al., 2020)	0.105	1.068	0.205
GraphDF (Luo et al., 2021)	0.101	0.953	0.253
GraphVAE (Simonovsky & Komodakis, 2018)	0.256	0.772	0.344
GNF (Liu et al., 2019)	0.134	-	-
EDP-GNN (Niu et al., 2020b)	0.078	0.177	0.226
GDSS (Jo et al., 2022)	0.044	0.109	0.170
Ours	**0.029**	**0.045**	**0.056**

Table 1: Generation results on the **conditional** graph datasets. We report the MMD distances between the test datasets and generated graphs. The best results are highlighted in bold (the smaller the better). Hyphen (-) denotes out-of-resources that take more than 10 days or are not applicable due to memory issues.

4.2. Unconditional Graph Generation

We also conduct unconditional graph generation experiments with the aforementioned state-of-the-art graph generation methods on synthetic datasets: Community-small and Grid from (You et al., 2018); and a real-world dataset: Enzymes (Schomburg et al., 2004). As shown in Table 2, our GPrinFlowNet still achieves state-of-the-art generation results on the unconditional generation task.

Furthermore, we compare the graph generation efficiency of some representative aforementioned methods in Table 3. We record and report the graph generation time (in seconds) for generating 100 samples. Our GPrinFlowNet achieves the highest generation speed among all existing methods.

	Community-small Avg. MMD↓	Enzymes Avg. MMD↓	Grid Avg. MMD↓
DeepGMG (Li et al., 2018)	0.523	-	-
GraphRNN (You et al., 2018)	0.080	0.043	-
GraphAF (Shi et al., 2020)	0.133	1.073	-
GraphDF (Luo et al., 2021)	0.070	0.922	-
GNF (Liu et al., 2019)	0.170	-	-
GraphVAE (Simonovsky & Komodakis, 2018)	0.623	0.730	0.846
EDP-GNN (Niu et al., 2020b)	0.074	0.124	0.340
SubspaceDiff (Jing et al., 2022)	0.056	0.051	0.076
WSGM (Guth et al., 2022)	0.044	0.048	0.051
GDSS (Jo et al., 2022)	0.046	0.046	0.062
Ours	**0.037**	**0.039**	**0.038**

Table 2: **Generation results on the unconditional graph datasets.** We report the MMD distances between the test datasets and generated graphs. The best results are highlighted in bold (the smaller the better). Hyphen (-) denotes out-of-resources that take more than 10 days or are not applicable due to memory issues.

Dataset	GraphAF	GraphDF	EDP-GNN	GDSS	Ours
Community-small	357	$2.47e^3$	368	72	**2.7**
Enzymes	596	$7.58e^3$	665	128	**10.2**
Grid	$5.83e^3$	$6.42e^4$	$7.58e^3$	$1.75e^3$	**30.89**

Table 3: Graph generation time comparison (in seconds) for generating 100 graphs under the methods' default setting.

4.3. Ablation Studies

We further conduct ablation studies on how the intermediate supervision (e.g. aligning the distribution of the i-th intermediate output to the distribution of the reconstruction version of the adjacency matrix at the i-th granularity level) affects the performance of GPrinFlowNet. The results presented in Table 4 underscore the importance of imposing supervision to enable GPrinFlowNet to effectively learn the distribution of the reconstructed graph adjacency matrix at various granularity levels.

Supervision scheme	AIDS	Enzymes	Synthie
No supervision	0.037	0.075	0.086
Supervision per 10 steps	0.035	0.061	0.072
Supervision per 5 steps	0.032	0.054	0.066
Supervision per 2 steps	0.030	0.049	0.058
Supervision in every step	**0.029**	**0.045**	**0.056**

Table 4: **Ablation studies.** We report the mean MMD over distributions of degree, clustering coefficient, and the number of orbits, for conditional graph generation.

5. Discussion

In this paper, we address the challenge of conditional graph generation using the Graph Principal Flow Network (GPrinFlowNet). Through its progressive coarse-to-fine graph generation process, GPrinFlowNet excels at capturing the subtle yet crucial semantic features, making it the state-of-the-art conditional generation model.

References

Bengio, E., Jain, M., Korablyov, M., Precup, D., and Bengio, Y. Flow network based generative models for non-iterative diverse candidate generation. *Advances in Neural Information Processing Systems*, 34:27381–27394, 2021a.

Bengio, Y., Deleu, T., Hu, E. J., Lahlou, S., Tiwari, M., and Bengio, E. Gflownet foundations. *CoRR*, abs/2111.09266, 2021b. URL https://arxiv.org/abs/2111.09266.

De Cao, N. and Kipf, T. Molgan: An implicit generative model for small molecular graphs. *arXiv preprint arXiv:1805.11973*, 2018.

Defferrard, M., Bresson, X., and Vandergheynst, P. Convolutional neural networks on graphs with fast localized spectral filtering. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pp. 3844–3852, Red Hook, NY, USA, 2016. Curran Associates Inc. ISBN 9781510838819.

Deleu, T., Góis, A., Emezue, C. C., Rankawat, M., Lacoste-Julien, S., Bauer, S., and Bengio, Y. Bayesian structure learning with generative flow networks. In *The 38th Conference on Uncertainty in Artificial Intelligence*, 2022. URL https://openreview.net/forum?id=HElfed8j9g9.

Feragen, A., Kasenburg, N., Petersen, J., de Bruijne, M., and Borgwardt, K. Scalable kernels for graphs with continuous attributes. *Advances in neural information processing systems*, 26, 2013.

Guth, F., Coste, S., Bortoli, V. D., and Mallat, S. Wavelet score-based generative modeling. *ArXiv*, abs/2208.05003, 2022.

Ho, J. and Salimans, T. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021. URL https://openreview.net/forum?id=qw8AKxfYbI.

Jing, B., Corso, G., Berlinghieri, R., and Jaakkola, T. Subspace diffusion generative models. In *ECCV*, 2022.

Jo, J., Lee, S., and Hwang, S. J. Score-based generative modeling of graphs via the system of stochastic differential equations. In *International Conference on Machine Learning*, pp. 10362–10383. PMLR, 2022.

Li, Y., Vinyals, O., Dyer, C., Pascanu, R., and Battaglia, P. Learning deep generative models of graphs. *arXiv preprint arXiv:1803.03324*, 2018.

Liu, H., Jing, L., Wen, J., Xu, P., Wang, J., Yu, J., and Ng, M. K. Interpretable deep generative recommendation models. *J. Mach. Learn. Res.*, 22:202–1, 2021.

Liu, J., Kumar, A., Ba, J., Kiros, J., and Swersky, K. Graph normalizing flows. In *NeurIPS*, 2019.

Luo, T., Mo, Z., and Pan, S. J. Fast graph generative model via spectral diffusion. *arXiv preprint arXiv:2211.08892*, 2022.

Luo, Y., Yan, K., and Ji, S. Graphdf: A discrete flow model for molecular graph generation. In *ICML*, pp. 7192–7203, 2021.

Ma, T., Chen, J., and Xiao, C. Constrained generation of semantically valid graphs via regularizing variational autoencoders. In *NeurIPS*, 2018.

Malkin, N., Jain, M., Bengio, E., Sun, C., and Bengio, Y. Trajectory balance: Improved credit assignment in GFlownets. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=5btWTw1vcw1.

Morris, C., Kriege, N. M., Bause, F., Kersting, K., Mutzel, P., and Neumann, M. Tudataset: A collection of benchmark datasets for learning with graphs. *arXiv preprint arXiv:2007.08663*, 2020.

Niu, C., Song, Y., Song, J., Zhao, S., Grover, A., and Ermon, S. Permutation invariant graph generation via score-based generative modeling. In *International Conference on Artificial Intelligence and Statistics*, pp. 4474–4484. PMLR, 2020a.

Niu, C., Song, Y., Song, J., Zhao, S., Grover, A., and Ermon, S. Permutation invariant graph generation via score-based generative modeling. In *AISTATS*, pp. 4474–4484, 2020b.

Pan, L., Zhang, D., Courville, A., Huang, L., and Bengio, Y. Generative augmented flow networks. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=urF_CBK5XC0.

Schomburg, I., Chang, A., Ebeling, C., Gremse, M., Heldt, C., Huhn, G., and Schomburg, D. Brenda, the enzyme database: updates and major new developments. *Nucleic acids research*, 32(suppl_1):D431–D433, 2004.

Shi, C., Xu, M., Zhu, Z., Zhang, W., Zhang, M., and Tang, J. Graphaf: a flow-based autoregressive model for molecular graph generation. *arXiv preprint arXiv:2001.09382*, 2020.

Simonovsky, M. and Komodakis, N. Graphvae: Towards generation of small graphs using variational autoencoders. In *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part I 27*, pp. 412–422. Springer, 2018.

Wang, H., Xu, T., Liu, Q., Lian, D., Chen, E., Du, D., Wu, H., and Su, W. Mcne: an end-to-end framework for learning multiple conditional network representations of social network. In *KDD*, pp. 1064–1072, 2019.

Yang, Y., Feng, Z., Song, M., and Wang, X. Factorizable graph convolutional networks. In *NeurIPS*, pp. 20286–20296, 2020.

You, J., Ying, R., Ren, X., Hamilton, W., and Leskovec, J. Graphrnn: Generating realistic graphs with deep auto-regressive models. In *ICML*, pp. 5708–5717, 2018.

Zang, C. and Wang, F. Moflow: an invertible flow model for generating molecular graphs. In *KDD*, pp. 617–626, 2020.

Zhang, D., Chen, R. T. Q., Malkin, N., and Bengio, Y. Unifying generative models with gflownets. *ArXiv*, abs/2209.02606, 2022a.

Zhang, D., Malkin, N., Liu, Z., Volokhova, A., Courville, A., and Bengio, Y. Generative flow networks for discrete probabilistic modeling. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S. (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 26412–26428. PMLR, 17–23 Jul 2022b. URL https://proceedings.mlr.press/v162/zhang22v.html.

Zhu, Y., Du, Y., Wang, Y., Xu, Y., Zhang, J., Liu, Q., and Wu, S. A survey on deep graph generation: Methods and applications. *arXiv preprint arXiv:2203.06714*, 2022.

A. Complete Experiment Results and experiment details

We follow the evaluation setting of (Jo et al., 2022; You et al., 2018): we split the data into train/test set according to (Jo et al., 2022; You et al., 2018), and sample the same number of graphs as in the test set. Then, we use the maximum mean discrepancy (MMD) to compare the distributions of graph statistics between the same number of generated and test graphs. We follow (Jo et al., 2022; You et al., 2018) to measure the distribution difference of degree, clustering coefficient, and the number of occurrences of orbits with 4 nodes. We further average the MMDs and present them in the fourth column under each dataset. We present the complete experiment results for conditional graph generation in Table 5, and unconditional graph generation in Table 6.

	AIDS				Enzymes				Synthie			
	Real, $\lvert V\rvert \leq 95$, $\lvert C\rvert = 2$				Real, $\lvert V\rvert \leq 125$, $\lvert C\rvert = 6$				Synthetic, $\lvert V\rvert \leq 100$, $\lvert C\rvert = 4$			
	Deg.↓	Clus.↓	Orbit↓	Avg.↓	Deg.↓	Clus.↓	Orbit↓	Avg.↓	Deg.↓	Clus.↓	Orbit↓	Avg.↓
GraphRNN (You et al., 2018)	0.241	0.143	0.034	0.139	0.086	0.294	0.307	0.229	0.247	0.285	0.419	0.317
GraphAF (Shi et al., 2020)	0.197	0.093	0.026	0.105	0.058	0.174	0.156	0.129	0.137	0.176	0.302	0.205
GraphDF (Luo et al., 2021)	0.184	0.085	0.031	0.101	0.062	0.196	0.204	0.154	1.681	1.265	0.258	1.068
GraphVAE (Simonovsky & Komodakis, 2018)	0.358	0.284	0.127	0.256	1.249	0.687	0.381	0.772	1.554	1.074	0.232	0.953
GNF (Liu et al., 2019)	0.224	0.159	0.018	0.133	-	-	-	-	-	-	-	-
EDP-GNN (Niu et al., 2020b)	0.127	0.082	0.024	0.077	0.067	0.241	0.225	0.177	0.148	0.185	0.347	0.226
GDSS[1] (Jo et al., 2022)	0.062	0.049	0.022	0.044	0.038	0.158	0.132	0.109	0.114	0.126	0.269	0.169
Ours	**0.046**	**0.031**	**0.012**	**0.029**	**0.027**	**0.062**	**0.046**	**0.045**	**0.048**	**0.042**	**0.079**	**0.056**

Table 5: **Generation results on the conditional graph generation.** We report the MMD distances between the test datasets and generated graphs. The best results are highlighted in bold (the smaller the better). Hyphen (-) denotes out-of-resources that take more than 10 days or are not applicable due to memory issues.

	Community-small				Enzymes				Grid			
	Synthetic, $12 \leq \lvert V\rvert \leq 20$				Real, $10 \leq \lvert V\rvert \leq 125$				Synthetic, $100 \leq \lvert V\rvert \leq 400$			
	Deg.↓	Clus.↓	Orbit↓	Avg.↓	Deg.↓	Clus.↓	Orbit↓	Avg.↓	Deg.↓	Clus.↓	Orbit↓	Avg.↓
DeepGMG (Li et al., 2018)	0.220	0.950	0.400	0.523	-	-	-	-	-	-	-	-
GraphRNN (You et al., 2018)	0.080	0.120	0.040	0.080	0.017	**0.043**	0.021	0.043				
GraphAF (Shi et al., 2020)	0.18	0.200	0.020	0.133	1.669	1.283	0.266	1.073	-	-	-	-
GraphDF (Luo et al., 2021)	0.060	0.120	0.030	0.070	1.503	1.061	0.202	0.922	-	-	-	-
GraphVAE (Simonovsky & Komodakis, 2018)	0.350	0.980	0.540	0.623	1.369	0.629	0.191	0.730	1.619	**0.0**	0.919	0.846
GNF (Liu et al., 2019)	0.200	0.200	0.110	0.170	-	-	-	-	-	-	-	-
EDP-GNN (Niu et al., 2020b)	0.053	0.144	0.026	0.074	0.023	0.268	0.082	0.124	0.455	0.238	0.328	0.340
SubspaceDiff (Jing et al., 2022)	0.057	0.098	0.012	0.056	0.037	0.099	0.018	0.051	0.124	0.013	0.090	0.076
WSGM (Guth et al., 2022)	0.039	0.084	0.009	0.044	0.034	0.097	0.013	0.048	0.083	0.006	0.065	0.051
GDSS[1] (Jo et al., 2022)	0.045	0.086	0.007	0.046	0.026	0.102	0.009	0.046	0.111	0.005	0.070	0.062
Ours	**0.021**	**0.068**	**0.021**	**0.037**	**0.021**	0.088	**0.009**	**0.039**	**0.056**	0.042	**0.015**	**0.038**

Table 6: **Generation results on the unconditional graph datasets.** We report the MMD distances between the test datasets and generated graphs. The best results are highlighted in bold (the smaller the better). Hyphen (-) denotes out-of-resources that take more than 10 days or are not applicable due to memory issues.

The average results of the Enzymes dataset reported in the GDSS original paper is 0.032. However, the best result we can obtain using the author's released code and checkpoint with careful fine-tuning is 0.046.