GLAD: Improving Latent Graph Generative Modeling with Simple Quantization

Van Khoa Nguyen [1 2] **Yoann Boget** [1 2] **Frantzeska Lavda** [1 2] **Alexandros Kalousis** [1 2]

Abstract

Exploring the graph latent structures has not garnered much attention in the graph generative research field. Yet, exploiting the latent space is as crucial as working on the data space for discrete data such as graphs. However, previous methods either failed to preserve the permutation symmetry of graphs or lacked an effective approaches to model appropriately within the latent space. To mitigate those issues, we propose a simple, yet effective discrete latent graph diffusion generative model. Our model, namely GLAD, not only overcomes the drawbacks of existing latent approaches, but also alleviates inherent issues present in diffusion methods applied on the graph space. We validate our generative model on the molecular benchmark datasets, on which it demonstrates competitive performance compared with the state-of-the-art baselines.

1. Introduction

When it comes to the graph representation space most methods operate over the original data space and learn generative models over node and edge features (Shi et al., 2020; Luo et al., 2021; Kong et al., 2023), showcasing a flexible adaptation to learning over discrete representations. Less attention has been given to methods that operate over graph latent spaces and the appropriate definition and design of such latent spaces. Some initial efforts (et al, 2018; Jin et al., 2018; Liu et al., 2018; Samanta et al., 2020) propose learning the graph distribution over the latent space of variational autoecoders (VAEs) (Kingma & Welling, 2013). Yet, such approaches often suffer from high reconstruction errors, have to solve challenging graph-matching problems, and/or rely on heuristic corrections for the generated graphs.

[1]Geneva School for Business Administration (HES-SO) [2]Department of Computer Science, University of Geneva, 1214 Geneva, Switzerland. Correspondence to: Van Khoa Nguyen <van-khoa.nguyen@etu.unige.ch>.

Accepted by the Structured Probabilistic Inference & Generative Modeling workshop of ICML 2024, Vienna, Austria.

Recently, diffusion-based methods have emerged as a very compelling approach for modeling graph distributions (Niu et al., 2020; Jo et al., 2022; Vignac et al., 2023). They allow for the natural incorporation of the permutation-invariant property as an inductive bias within their denoising component. Obviously they are not without limitations. The initial graph diffusion methods rely on continuous score functions to model the graph distributions. They learn these score functions by denoising graph structures that have been noised with real-valued noise (Niu et al., 2020; Jo et al., 2022). Continuous score functions are a poor match for structures that are inherently discrete. There are some diffusion-based works that treat graphs heads-on as discrete objects (Vignac et al., 2023). In addition, constructing appropriate score functions over the original graph representation is challenging. Common approaches factorise these score functions to node- and adjacency-matrix-specific components which require their own distinct denoising networks. These partial score functions provide a fragmented view of the graph and do not reflect well the true score function making them a poor choice to model graph distributions. Diffusion-based models typically have a denoising component which operates over the edges. The denoising process takes place over densely(fully)-connected graphs (Niu et al., 2020; Jo et al., 2022; Vignac et al., 2023); this poses significant scalability challenges (Qin et al., 2023).

In this work, we propose **G**raph Discrete **La**tent **D**iffusion (GLAD) a generative model that operates on a discrete latent space. We quantize our latent node embeddings to preserve their discrete nature in graph space, which delivers more robust graph reconstruction compared to existing methods operating on continuous latent spaces. Then, we tailor diffusion bridges (Liu et al., 2023) to model latent graph distributions over the proposed quantized space. Our latent diffusion model relies exclusively on a geometric set of latent nodes, and does not resort to component specific decomposition as other diffusion models on the graph space. To our knowledge, GLAD stands for the first equivariant latent graph diffusion model, which ensures GLAD to learn the permutation-invariant distribution of graphs. Through experiments, we demonstrate that GLAD excels in capturing both chemical and structural properties of molecule graphs over the state-of-the-art baselines.

2. Analysing Graph Latent Spaces

In this section, we will categorize existing methods that build graph generative models in latent spaces. Along with each approach, we show their shortcomings that eventually motivate the design of latent space in our model.

Continuous-graph latent space Pioneering work (et al, 2018) explores the latent space of variational autoencoders (VAEs) for graph generative modeling. They establish a global latent representation of graph by pooling its associated node embeddings. The posterior of encoded graph lies in a continuous space, mimicking the vanilla-VAE setting (Kingma & Welling, 2013). However, the use of pooling breaks the permutation symmetry of graphs in the latent space, which thus requires to solve the graph-matching problem when computing graph reconstruction losses. This extra step adds more computational complexity and is inaccurate for large graphs.

Continuous-node latent space To avoid handling the graph-matching problem, consequent works (Li et al., 2018; Samanta et al., 2020) choose to operate instead on a set of latent nodes and use the standard VAE framework, however applied on node levels. By keeping node identity in the latent space, they thus alleviate the ambiguity in the decoding phase, and they can directly use the l_2-pairwise reconstruction loss. These approaches typically define the encoding distribution of graph as the product of its own node posterior distributions, each of which is regularised towards the same fixed prior, a typical example is the uninformative standard Gaussian prior. Due to this regularisation form, we empirically find that these node-encoding distributions are not distinguishable themselves in the latent space. Such issue we call the latent-node-collapsing problem, we observe it drastically hinders graph reconstructions in the downstream decoding process.

Discrete latent space Guaranteeing a distinctive encoding of each local graph structure is necessary (i) to improve the graph reconstruction ability (ii) to ensure generative models capture diverse local graph patterns. These objectives can be simply achieved with an appropriate design of discrete latent space, where the node encoding of itself and its neighborhood (subgraph) can be distinctively embedded in the latent space. In other words, the discrete nature of graphs are meaningfully persevered from data to latent spaces. Notable works (Luo et al., 2021; Boget et al., 2024) first treat the latent graph generative modelling as a discrete problem. The former builds an auto-regressive discrete flow over a discrete latent space defined by a modulo shift transform. The latter relies on the vector quantization (Van Den Oord et al., 2017) to encode latent graph representations into codebooks, and learns the discrete latent graph distribution auto-regressively. While these methods have demonstrated a superiority over their continuous counterpart (Shi et al., 2020), they still belong to auto-regressive methods based on canonical-node ordering, which thus do not learn graph distributions in a permutation-invariant manner.

In this paper, we introduce a novel discrete latent space for graphs, which extends the finite-quantization concept (Mentzer et al., 2023) to the graph generation problem. Our graph latent space is simple in design, yet effective in performance, on top of which we build an equivariant generative model that allows capturing the permutation-invariant distribution of graphs.

3. GLAD: Graph Discrete Latent Diffusion Model

We now will introduce our proposal, namely GLAD, the first equivariant generative model that operates on the discrete latent space of graphs. We present our notations, Section 3.1, define a novel discrete graph latent space, Section 3.2, learn a prior over the defined latent space using diffusion bridges, Section 3.3.

3.1. Notations

In graph space, we denote a graph instance by a tuple $\mathcal{G} = (X, E)$, where $X = \{x_i\}_{i=1}^N$ a set of node features, $E = \{e_{ij}\}_{i,j \in N}$ a set of edge features, N the number of nodes. We rely on the autoencoder framework (AE) to embed graphs, where encoder (ϕ)-, decoder (θ)- architectures are parameterized by equivariant graph neural networks (E-GNNs). In latent space, we denote a set of raw node embeddings $Z_{raw} = \{z_i\}_{i=1}^N$, where $z_i \in \mathbb{R}^f$ corresponds to the embedding of i^{th} node. We denote a quantization operator $\mathcal{F}$ applied on each element of Z_{raw} to obtain a quantized set $Z_{\mathcal{G}} = \{z_i^q\}_{i=1}^N$. We denote $[L_j] = [-\mathcal{R}(L_j/2), \cdots, -1, 0, 1, \cdots, \mathcal{R}(L_j/2)]$ a set of pre-defined possible values, $\mathcal{R}$ is the rounding operator.

3.2. Novel Discrete Graph Latent Space

We drive the design of the latent space by: i) ensuring that we learn graph distributions that are invariant to permutations, ii) encoding nodes and their local structures by ensuring that structural differences are reflected in the encoding, and iii) preserving the discrete nature of the objects we encode, nodes and their neighborhoods, and that of the graph itself.

Concretely, we use an E-GNN encoder, ϕ, to map a graph to a set of continuous node embeddings $Z_{raw} = \{z_i\}_{i=1}^N := \phi(\mathcal{G})$. The GNN encoder essentially captures the node encoding information about itself and its local neighborhood. We subsequently apply a quantization operator, $\mathcal{F}$, on the

continuous node embeddings and map them to a quantized space of the same dimension. We follow (Mentzer et al., 2023) to design $\mathcal{F}$, where the quantization operates independently on each latent node dimension, and the quantization of the j^{th} dimension of $z_i \in \mathbb{R}^f$ embedding is given by:

$$z_{ij}^q = \mathcal{F}(z_{ij}, L_j) = \mathcal{R}(\frac{L_j}{2}\tanh z_{ij}),\ 1 \leqslant j \leqslant f \quad (1)$$

where $\mathcal{R}$ is the rounding operator [1], and L_j is the number of quantization levels on the j^{th} latent node dimension. The quantization creates a new set of latent graph representation $Z_\mathcal{G} = \{z_i^q\}_{i=1}^N$ with $z_i^q \in \mathbf{S}$, where the quantized latent space can be defined by a product of subspaces $\mathbf{S} := \prod_{j=1}^f [L_j] = \{l_1 \times \cdots \times l_f : \forall l_j \in [L_j]\}$. We denote K as the total quantization points from $\mathbf{S}$.

The operator $\mathcal{F}$ is permutation equivariant and so is the mapping from the initial graph $\mathcal{G}$ to its quantized latent representation as long as the ϕ graph encoder is permutation equivariant. Thus for any P permutation matrix the following equivariance relation holds:

$$P^T Z_\mathcal{G} = \mathcal{F}(P^T Z) = \mathcal{F}(\phi(P^T E P, P^T X)) \quad (2)$$

We apply an equivariant decoder $\theta(Z_\mathcal{G})$ on the assumption of a fully-connected graph over the quantized latent nodes $Z_\mathcal{G}$, which results in reconstructed graphs that keep the same node labels as the original input graphs. Our graph autoencoder only composes of equivariant components. Therefore, we ensure that decoded distributions are invariant to the node permutations of graphs. We denote Π as the discrete latent distribution of graphs that we learn in the next step with diffusion bridges.

3.3. Learning Quantized Latent Graph Prior with Diffusion Bridges

For a given graph, the quantization maps its set of node embeddings to points that belong to the discrete latent structure $\mathbf{S}$. The latent-graph domain Ω becomes a set structure that we can decompose into a node-wise manner as $\Omega = I_1 \times \cdots \times I_N$, where $I_i = \mathbf{S}$. We extend diffusion bridges (Liu et al., 2023), which learn data distributions on constrained domains, to model our latent-graph structure Ω. In our analysis, we use Brownian motion as a non-conditional diffusion process defined by a SDE:

$$\mathbb{Q} : dZ_t = \sigma(Z_t, t)dW_t \quad (3)$$

where W_t is a Wiener process, $\sigma : [0, T] \times \mathbb{R}^d \mapsto \mathbb{R}$ is a diffusion coefficient.

$Z_\mathcal{G}$-conditioned diffusion bridge We first define a conditional diffusion bridge where the process's endpoint Z_T should be reached to the conditional factor, $Z_T = Z_\mathcal{G}$. We denote the $Z_\mathcal{G}$-bridge as $\mathbb{Q}^{Z_\mathcal{G}}$, which dynamics can be derived from the *h-transform* (Oksendal, 2013):

$$\mathbb{Q}^{Z_\mathcal{G}} : dZ_t = \eta^{Z_\mathcal{G}}(Z_t, t)dt + \sigma(Z_t, t)dW_t, \quad Z_0 \sim \mathbb{P}_0 \quad (4)$$

where we denote the prior distribution of $Z_\mathcal{G}$-bridge, $\mathbb{P}_0$. The $Z_\mathcal{G}$-bridge's drift is defined as $\eta^{Z_\mathcal{G}}(Z_t, t) = \sigma^2(Z_t, t)\nabla_{Z_t} \log q_{T|t}(Z_\mathcal{G}|Z_t)$; $q_{T|t}(Z_\mathcal{G}|Z_t)$ is the probability density of obtaining $Z_\mathcal{G}$ at time T when we have Z_t at time t; $\nabla_{Z_t} \log q_{T|t}(Z_\mathcal{G}|Z_t)$ acts as a steering force, which is central to guiding Z_t towards our specified target, $Z_T = Z_\mathcal{G}$.

Π-conditioned diffusion bridge Given the $Z_\mathcal{G}$-bridge definition, we can now construct a bridge process on the discrete latent-graph distribution Π, which is constrained over the latent-graph domain Ω. We call this bridge process a Π-bridge and denote it by $\mathbb{Q}^\Pi$. The Π-bridge is simply constructed by a mixture of $Z_\mathcal{G}$-bridges; their end-points are conditioned on latent-graph samples $\{Z_\mathcal{G}^i\}$ i.i.d drawn from the latent graph distribution Π. The bridge-mixture's dynamics are governed by the following stochastic differential equation (SDE):

$$\mathbb{Q}^\Pi : dZ_t = \eta^\Pi(Z_t, t)dt + \sigma(Z_t, t)dW_t \quad (5)$$

The Π-bridge's drift is $\eta^\Pi(Z_t, t) = \sigma^2(Z_t, t)\mathbb{E}_{\omega \sim q_{T|t,\Omega}(\omega|Z_t)}[\nabla_{Z_t} \log q_{T|t}(\omega|Z_t)]$, where ω is sampled from a transition probability density given by $q_{T|t,\Omega}(\omega|Z_t) = q(Z_T = \omega|Z_t, Z_T \in \Omega)$. We see that the discrete latent-graph structure Ω is taken into consideration in the expectation of the steering forces. The marginal distribution, $\mathbb{Q}_T^\Pi$, induced by the Π-bridge at $t = T$ will match the latent graph distribution Π by construction, i.e. $\mathbb{Q}_T^\Pi = \Pi$.

Model bridge Our objective is to learn a bridge model, $\mathbb{P}^\psi$, that can generalise from the Π-bridge dynamics and generate new latent-graph samples from the Π distribution. We construct $\mathbb{P}^\psi$ by learning the model-bridge's drift with a neural network, and the model-bridge dynamics are given by a parametric SDE:

$$\mathbb{P}^\psi : dZ_t = \eta^\psi(Z_t, t)dt + \sigma(Z_t, t)dW_t \quad (6)$$

where the model-bridge's drift is $\eta^\psi(Z_t, t) = \sigma(Z_t, t)\psi(Z_t, t) + \eta^\Pi(Z_t, t)$, ψ is a parametric neural network. The Π-bridge imputes noise to latent-graph samples at different time steps which we use to learn the model bridge $\mathbb{P}^\psi$. We train $\mathbb{P}^\psi$ by minimizing the Kullback-Leibler divergence between the two probability path measures $\min_\psi\{\mathcal{L}(\psi) := \mathcal{KL}(\mathbb{Q}^\Pi||\mathbb{P}^\psi)\}$; the Girsanov theorem, (Lejay, 2018), allows us to compute the $\mathcal{KL}$ divergence with the following closed form solution:

$$\mathcal{L} = \mathbb{E}_{Z_\mathcal{G} \sim \Pi, t \sim [0,T], Z_t \sim \mathbb{Q}^{Z_\mathcal{G}}}\left[\|\psi(Z_t, t) - \bar{\eta}(Z_t, t)\|^2\right] \quad (7)$$

[1] To ensure end to end differentiability we implement $\mathcal{R}$ with the straight through estimator.

Table 1. **Generation results on the molecule datasets**. We show the mean values of 3 runs for each experiment. The baselines are sourced from (Jo et al., 2022; Kong et al., 2023). We highlight the most important metrics; the 1st and 2nd best results are bolded and underlined, respectively. We compare with generative models operating on graph space (top row) and on latent-graph space (bottom row). We further provide standard deviations in Appendix D.1.

	QM9					ZINC250K				
	V ↑	U ↑	N ↑	NSPDK ↓	FCD ↓	V ↑	U ↑	N ↑	NSPDK ↓	FCD ↓
GraphAF (Shi et al., 2020)	74.43	88.64	86.59	0.020	5.27	68.47	98.64	100	0.044	16.02
MoFlow (Zang & Wang, 2020)	91.36	98.65	94.72	0.017	4.47	63.11	99.99	100	0.046	20.93
GDSS (Jo et al., 2022)	95.72	98.46	86.27	0.003	2.90	97.01	99.64	100	0.019	14.66
DiGress (Vignac et al., 2023)	99.0	96.66	33.40	<u>0.0005</u>	<u>0.360</u>	91.02	81.23	100	0.082	23.06
GraphArm (Kong et al., 2023)	90.25	95.62	70.39	0.002	1.22	88.23	99.46	100	0.055	16.26
GraphDF (Luo et al., 2021)	93.88	98.58	98.54	0.064	10.93	90.61	99.63	100	0.177	33.55
DGAE (Boget et al., 2024)	92.0	97.61	79.09	0.0015	0.86	77.9	99.94	99.97	<u>0.007</u>	<u>4.4</u>
GLAD	97.12	97.52	38.75	**0.0003**	**0.201**	81.81	100	99.99	**0.002**	**2.54**

with $\bar{\eta}(Z_t, t) = \sigma^{-1}(Z_t, t)(\eta^{Z_\mathcal{G}}(Z_t, t) - \eta^{\Pi}(Z_t, t))$, where $\eta^{Z_\mathcal{G}}$, η^{Π} are introduced in Equation 4 and Equation 5, respectively. Since we use Brownian motion for the non-conditional diffusion $\mathbb{Q}$, we can retrieve its closed-form perturbation kernel and accordingly derive the dynamics of the $Z_\mathcal{G}$-bridge's drift as:

$$\eta^{Z_\mathcal{G}}(Z_t, t) = \sigma_t^2 \frac{Z_\mathcal{G} - Z_t}{\beta_T - \beta_t}, \ \beta_t = \int_0^t \sigma_s^2 ds \tag{8}$$

where we simply define the diffusion coefficient σ_t that depends only on the time variable. As our latent-graph structure Ω can be factorized into the latent-structure of nodes, the expectation over Ω , Equation 5, can simplify to one-dimensional integrals over $\mathbf{S}$. And the Π-bridge's drift can be decomposed into the latent-node structures:

$$\eta^{\Pi}(Z_t, t) = \left[\left[\eta^{\Pi}(Z_t^i, t)\right]^{I_i}\right]_{i=1}^{N}$$
$$\left[\eta^{\Pi}(Z_t^i, t)\right]^{I_i} = \sigma_t^2 \nabla_{Z_t^i} \log \sum_{k=1}^{K} \exp\left(-\frac{\|Z_t^i - s_k\|^2}{2(\beta_T - \beta_t)}\right) \tag{9}$$

where $s_k \in \mathbf{S}$, Z_t^i is the feature of i^{th} latent node of Z_t. We apply the drift terms obtained above to Equation 7 to get the closed-form objective. We provide our mathematical derivations, training and sampling procedures in Appendix.

4. Experiments

Setup We evaluate the capacity of GLAD to capture the complex dependencies between atoms and bonds as they appear in different molecules. We conduct experiments on two standard datasets: QM9 (Ramakrishnan et al., 2014) and ZINC250k (Irwin et al., 2012). Following (Jo et al., 2022), we remove hydrogen atoms and kekulize molecules by RDKit Landrum et al. (2016). We quantitatively evaluate all methods in terms of validity without post-hoc corrections (V ↑), uniqueness (U ↑), and novelty (N ↑) of 10.000 generated molecules. In addition, we compute two more salient metrics that quantify how well the distribution of the generated graphs aligns with the real data distribution, namely Fréchet ChemNet Distance (FCD ↓) and Neighborhood Subgraph Pairwise Distance Kernel (NSPDK ↓). We provide the metric and baseline details in Appendix D.1.

Results Table 1 show the generation results on molecules. GLAD is the first one-shot model that learns to generate molecular structures from a discrete latent space. Even though our model does not learn directly on atom- and bond-type structures, GLAD can still generate molecules with good validity scores without corrections. Importantly, GLAD achieves state of the art performance on the two most salient metrics NSPDK and FCD that capture well both structural and chemical properties of molecule distributions, consistently outperforming by significant margins all baselines. It demonstrates that our quantized latent space offers a suitable discrete structure to encode those properties. We provide generated molecule examples in Appendix D.1.

5. Conclusion

We present the first equivariant graph diffusion model that operates over a discrete latent space. By careful design, our graph-latent structure satisfies certain desiderata that stem from the graph nature, namely it should allow for learning graph-permutation-invariant distributions, being rich representational power, and respecting the inherently discrete nature of graphs. We empirically validate GLAD on molecule datasets and it shows a competitive performance to the state-of-the-art baselines. This paves the way for a more systematic exploration of graph generative modelling in latent spaces, something that until now has received rather limited attention.

References

Bengio, Y., Léonard, N., and Courville, A. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Boget, Y., Gregorova, M., and Kalousis, A. Discrete graph auto-encoder. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=bZ80b0wb9d.

Chen, Z., Chen, L., Villar, S., and Bruna, J. Can graph neural networks count substructures? *Advances in neural information processing systems*, 33:10383–10395, 2020.

Dwivedi, V. P. and Bresson, X. A generalization of transformer networks to graphs. *arXiv preprint arXiv:2012.09699*, 2020.

et al, S. 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.

Irwin, J. J., Sterling, T., Mysinger, M. M., Bolstad, E. S., and Coleman, R. G. Zinc: a free tool to discover chemistry for biology. *Journal of chemical information and modeling*, 52(7):1757–1768, 2012.

Jin, W., Barzilay, R., and Jaakkola, T. Junction tree variational autoencoder for molecular graph generation. In *International conference on machine learning*, pp. 2323–2332. PMLR, 2018.

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.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kong, L., Cui, J., Sun, H., Zhuang, Y., Prakash, B. A., and Zhang, C. Autoregressive diffusion model for graph generation. In *International Conference on Machine Learning*, pp. 17391–17408. PMLR, 2023.

Landrum, G. et al. Rdkit: Open-source cheminformatics software, 2016. *URL http://www. rdkit. org/, https://github. com/rdkit/rdkit*, 149(150):650, 2016.

Lejay, A. The girsanov theorem without (so much) stochastic analysis. *Séminaire de Probabilités XLIX*, pp. 329–361, 2018.

Li, Y., Zhang, L., and Liu, Z. Multi-objective de novo drug design with conditional graph generative model. *Journal of cheminformatics*, 10:1–24, 2018.

Liu, J., Kumar, A., Ba, J., Kiros, J., and Swersky, K. Graph normalizing flows. *Advances in Neural Information Processing Systems*, 32, 2019.

Liu, Q., Allamanis, M., Brockschmidt, M., and Gaunt, A. Constrained graph variational autoencoders for molecule design. *Advances in neural information processing systems*, 31, 2018.

Liu, X., Wu, L., Ye, M., and qiang liu. Learning diffusion bridges on constrained domains. In *The Eleventh International Conference on Learning Representations*, 2023.

Luo, Y., Yan, K., and Ji, S. Graphdf: A discrete flow model for molecular graph generation. In *International Conference on Machine Learning*, pp. 7192–7203. PMLR, 2021.

Mentzer, F., Minnen, D., Agustsson, E., and Tschannen, M. Finite scalar quantization: Vq-vae made simple. *arXiv preprint arXiv:2309.15505*, 2023.

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, 2020.

Oksendal, B. *Stochastic differential equations: an introduction with applications*. Springer Science & Business Media, 2013.

Qin, Y., Vignac, C., and Frossard, P. Sparse training of discrete diffusion models for graph generation. *arXiv preprint arXiv:2311.02142*, 2023.

Ramakrishnan, R., Dral, P. O., Rupp, M., and Von Lilienfeld, O. A. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific data*, 1(1):1–7, 2014.

Samanta, B., De, A., Jana, G., Gómez, V., Chattaraj, P., Ganguly, N., and Gomez-Rodriguez, M. Nevae: A deep generative model for molecular graphs. *Journal of machine learning research*, 21(114):1–33, 2020.

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.

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

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.

Van Den Oord, A., Vinyals, O., et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

Vignac, C., Krawczuk, I., Siraudin, A., Wang, B., Cevher, V., and Frossard, P. Digress: Discrete denoising diffusion for graph generation. In *The Eleventh International Conference on Learning Representations*, 2023.

You, J., Ying, R., Ren, X., Hamilton, W., and Leskovec, J. Graphrnn: Generating realistic graphs with deep autoregressive models. In *International conference on machine learning*, pp. 5708–5717. PMLR, 2018.

Zang, C. and Wang, F. Moflow: an invertible flow model for generating molecular graphs. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 617–626, 2020.

A. Discrete Latent Graph Diffusion Derivations

We adjust the derivations of Liu et al. (2023) to learning on the graph setting. We start by considering Brownian motion as the non-conditional diffusion process:

$$\mathbb{Q}: \quad dZ_t = \sigma_t dW_t, \quad \beta_t = \int_0^t \sigma_s^2 ds$$

The transition probability between states follows a Gaussian distribution, which density is defined as $q_{T|t}(Z_T|Z_t) = \mathcal{N}(Z_T; Z_t, \beta_T - \beta_t)$. Applying this density to the transition probability Equation 4, we get the $Z_\mathcal{G}$-bridge drift function:

$$\begin{aligned}\eta^{Z_\mathcal{G}}(Z_t, t) &= \sigma_t^2 \nabla_{Z_t} \log q_{T|t}(Z_\mathcal{G}|Z_t) \\ &= \sigma_t^2 \frac{Z_\mathcal{G} - Z_t}{\beta_T - \beta_t}\end{aligned} \tag{10}$$

Thus, we can derive the governing law of the $Z_\mathcal{G}$-bridge:

$$\mathbb{Q}^{Z_\mathcal{G}}: \quad dZ_t = \sigma_t^2 \frac{Z_\mathcal{G} - Z_t}{\beta_T - \beta_t} dt + \sigma_t dW_t$$

And we sample from the $t \in [0, T]$ time step of the $Z_\mathcal{G}$-bridge as follows:

$$Z_t = \frac{\beta_t}{\beta_T}\left(Z_\mathcal{G} + (\beta_T - \beta_t) Z_0\right) + \xi \sqrt{\beta_t \left(1 - \frac{\beta_t}{\beta_T}\right)}, \quad \xi \sim \mathcal{N}(0, I), \quad Z_0 \sim \mathbb{P}_0$$

where $\mathbb{P}_0$ is the bridge prior, we ablate two cases: a fixed prior $\mathbb{P}_0 := \mathbf{0}$, and a standard Gaussian prior $\mathbb{P}_0 := \mathcal{N}(0, I)$. By plugging the probability $q_{T|t}(Z_T|Z_t)$ into Equation 5, we obtain the Π-bridge drift:

$$\begin{aligned}\eta^\Pi(Z_t, t) &= \sigma^2(Z_t, t) \mathbb{E}_{\omega \sim q_{T|t,\Omega}(\omega|Z_t)}[\nabla_{Z_t} \log q_{T|t}(\omega|Z_t)] \\ &= \sigma_t^2 \mathbb{E}_{\omega \sim q_{T|t,\Omega}(\omega|Z_t)}\left[\frac{\omega - Z_t}{\beta_T - \beta_t}\right]\end{aligned}$$

with Z_t samples from the $Z_\mathcal{G}$-bridge, ω is sampled from a transition density given by $q_{T|t,\Omega}(\omega|Z_t) := q(Z_T = \omega|Z_t, Z_T \in \Omega)$. While $\mathbb{Q}$ is Brownian motion, the transition probability can be truncated into a mixture of Gaussian densities over each element $\omega \in \Omega$, which is given by $q_{T|t,\Omega}(\omega|Z_t) \propto \mathbb{I}(\omega \in \Omega) q_{T|t}(Z_T = \omega|Z_t)$.

We observe that the truncated transition density considers the existing elements ω of the latent-graph domain Ω. However, our graph latent representation is a set-based structure, the discrete domain Ω can thus be factorized over latent node dimension, denoted as $\Omega = I_1 \times \cdots \times I_n$, with $I_i = \mathbf{S}$. Intuitively, this factorization represents for independent conditional stochastic processes over each latent node dimension of the given graph latent structure. Hence, the Π- bridge's drift can be decomposed as:

$$\begin{aligned}&\eta^\Pi(Z_t, t) = \left[\eta^{I_i}(Z_t^i, t)\right]_{i=1}^N \\ &\text{where } \eta^{I_i}(Z_t^i, t) = \sigma_t^2 \mathbb{E}_{s_k \sim q_{T|t,\mathbf{S}}(s_k|Z_t^i)}\left[\frac{s_k - Z_t^i}{\beta_T - \beta_t}\right]\end{aligned}$$

The expectation is now taken over the truncated transition probability along each latent node dimension I_i, whose density can be represented as following:

$$\begin{aligned}&q_{T|t,\mathbf{S}}(s_k|Z_t^i) \propto \mathbb{I}(s_k \in \mathbf{S}) \frac{q_{T|t}(s_k|Z_t^i)}{\sum_{j=1}^K q_{T|t}(s_j|Z_t^i)} \\ &\text{where the conditional density } q_{T|t}(s_j|Z_t^i) \text{ is } \mathcal{N}(s_j; Z_t^i, \beta_T - \beta_t)\end{aligned}$$

The Π-bridge's drift on latent node I_i can be further developed as:

$$\begin{aligned}\eta^{I_i}(Z_t^i, t) &= \sigma_t^2 \mathbb{E}_{s_k \sim q_{T|t,\mathbf{S}}(s_k|Z_t^i)} \left[\frac{s_k - Z_t^i}{\beta_T - \beta_t}\right] \\ &= \sigma_t^2 \sum_{k=1}^{K} \frac{q_{T|t}(s_k|Z_t^i)}{\sum_{\sum_{j=1}^{K} q_{T|t}(s_j|Z_t^i)}} \frac{s_k - Z_t^i}{\beta_T - \beta_t} \\ &= \sigma_t^2 \sum_{k=1}^{K} \frac{\exp\left(-\frac{\|Z_t^i - s_k\|^2}{2(\beta_T - \beta_t)}\right)}{\sum_{j=1}^{K} \exp\left(-\frac{\|Z_t^i - s_j\|^2}{2(\beta_T - \beta_t)}\right)} \frac{s_k - Z_t^i}{\beta_T - \beta_t} \\ &= \sigma_t^2 \nabla_{Z_t^{I_i}} \log \sum_{k=1}^{K} \exp\left(-\frac{\|Z_t^{I_i} - s_k\|^2}{2(\beta_T - \beta_t)}\right) \end{aligned} \tag{11}$$

Following (Liu et al., 2023) we train our latent-graph model bridge $\mathbb{P}^\psi$ by minimizing the Kullback-Leibler divergence between the probability-path measures of the model bridge and the Π-bridge, $\min_\psi\{\mathcal{KL}(\mathbb{Q}^\Pi||\mathbb{P}^\psi)\}$, which is equivalent to the minimization of expectation over $\mathcal{KL}\left(\mathbb{Q}^{Z_\mathcal{G}}\|\mathbb{P}^\psi\right)$ up to an additive constant:

$$\mathcal{L} = \mathcal{KL}(\mathbb{Q}^\Pi||\mathbb{P}^\psi) = \mathbb{E}_{Z_\mathcal{G}\sim\Pi}\left[\mathcal{KL}\left(\mathbb{Q}^{Z_\mathcal{G}}\|\mathbb{P}^\psi\right)\right] + const.$$

By Girsanov theorem (Lejay, 2018), the $\mathcal{KL}$ divergence between the probability-path measures of the model-bridge and the $Z_\mathcal{G}$-bridge can be further decomposed:

$$\mathcal{L} = \mathbb{E}_{Z_\mathcal{G}\sim\Pi, Z_t\sim\mathbb{Q}^{Z_\mathcal{G}}}\left[-\log p_0(Z_0) + \frac{1}{2}\int_0^T \left\|\sigma_t^{-1}(\eta^\psi(Z_t,t) - \eta^{Z_\mathcal{G}})\right\|^2\right] + const. \tag{12}$$

where $\eta^\psi(Z_t, t)$ is the parametric model-bridge's drift with a neural network. The prior distribution $p_0(Z_0)$ is a Dirac Δ distribution on the fixed point $\mathbf{0}$, the central mass of $\mathbf{S}$, thus $\log p_0(Z_0)$ is constant. Applying the definition of $\eta^\psi(Z_t, t)$ in Equation 6 to Equation 12, the objective can be obtained with a closed-form solution:

$$\mathcal{L}(\psi) = \mathbb{E}_{t\sim[0,T], Z_\mathcal{G}\sim\Pi, Z_t\sim\mathbb{Q}^{Z_\mathcal{G}}}\left[\left\|\psi(Z_t,t) - \sigma_t^{-1}(\eta^{Z_\mathcal{G}}(Z_t,t) - \eta^\Pi(Z_t,t))\right\|^2\right] + const \tag{13}$$

In sampling new latent-graph samples, we apply the Euler–Maruyama discretization on the model bridge $\mathbb{P}^\psi$:

$$Z_{t+1} = Z_t + \left(\sigma_t\psi(Z_t,t) + \eta^\Pi(Z_t,t)\right)\delta_t + \sigma_t\sqrt{\delta_t}\xi, \quad \xi\sim\mathcal{N}(0,I), \quad Z_0\sim\mathbb{P}_0 \tag{14}$$

where δ_t is the discretization step.

B. Algorithms

We train GLAD in two stages.

In the first stage, we train the graph-autoencoder, which learns a structural mapping from the graph space to the designed latent space and reconstructs original graphs from their latent representations. As a technical detail, we note that the rounding operator $\mathcal{R}$ of the quantization is non-differentiable. To ensure end-to-end training of our discrete graph latent space mapping, we use the straight-through estimator (STE) (Bengio et al., 2013), implemented with a stop-gradient operator $\mathcal{S}$ as $Z_\mathcal{G} \mapsto Z + \mathcal{S}(Z_\mathcal{G} - Z)$. In addition, we use mean-squared loss as it offers more stable training experiments.

In the second stage, we freeze the graph-autoencoder parameters while training the model bridge. We learn the graph latent diffusion bridge over the quantized space defined on the graph-encoder output and the quantization operator. We show the training procedure in Algorithm 1.

Algorithm 1 Graph Discrete Latent Diffusion

Input encoder ϕ, decoder θ, model bridge ψ

Stage 1:

Objective: learn graph-autoencoder (ϕ, θ)

for batch $\mathcal{G} = (X, E)$:

$Z_{raw} \leftarrow \phi(\mathcal{G})$

$Z_{\mathcal{G}} \leftarrow \mathcal{F}(Z_{raw}, L)$ {quantization}

$Z_{\mathcal{G}} \leftarrow \textit{ste}(Z_{\mathcal{G}}, Z_{raw})$ {straight-through estimator}

$\mathcal{L}_1 \leftarrow \|X - \theta_X(Z_{\mathcal{G}})\|^2 + \|E - \theta_E(Z_{\mathcal{G}})\|^2$

Update ϕ, θ by gradient descent $\nabla_{\phi,\theta}(\mathcal{L}_1)$

Stage 2:

Objective: learn model bridge (f_ψ)

for batch ($Z_{\mathcal{G}}, t \sim [0, T]$):

$Z_0 \sim \mathbb{P}_0$

$Z_t \leftarrow \frac{\beta_t}{\beta_T}(Z_{\mathcal{G}} + (\beta_T - \beta_t)Z_0) + \xi\sqrt{\beta_t\left(1 - \frac{\beta_t}{\beta_T}\right)}$ {$Z_{\mathcal{G}}$-bridge}

$\left[\eta^{Z_{\mathcal{G}}}\right]^{I_i} \leftarrow \sigma_t^2 \frac{Z_{\mathcal{G}}^{I_i} - Z_t^{I_i}}{\beta_T - \beta_t}, \forall I_i \in \Omega$ {$Z_{\mathcal{G}}$-bridge drift}

$\left[\eta^{\Pi}\right]^{I_i} \leftarrow \sigma_t^2 \nabla_{Z_t^{I_i}} \log \sum_{s_k \in \mathbf{S}} \exp\left(-\frac{\|Z_t^{I_i} - s_k\|^2}{2(\beta_T - \beta_t)}\right), \forall I_i \in \Omega$ {Π-bridge drift}

$\mathcal{L}_2 \leftarrow \sum_{I_i \in \Omega} \left\| [\psi(Z_t, t)]^{I_i} - \sigma_t^{-1}\left(\left[\eta^{Z_{\mathcal{G}}}\right]^{I_i} - \left[\eta^{\Pi}\right]^{I_i}\right)\right\|^2$ {drift loss}

Update ψ by gradient descent $\nabla_\psi(\mathcal{L}_2)$

Sampling:

$t \leftarrow 0$, $\delta_t \leftarrow \frac{1}{T}$, $Z_0 \sim \mathbb{P}_0$

while ($t < T$):

$\xi \sim \mathcal{N}(0, I)$

$Z_{t+1} \leftarrow Z_t + \left(\sigma_t \psi(Z_t, t) + \eta^{\Pi}(Z_t, t)\right)\delta_t + \sigma_t\sqrt{\delta_t}\xi$ {EM}

$\mathcal{G}_{new} \leftarrow \theta(Z_T)$

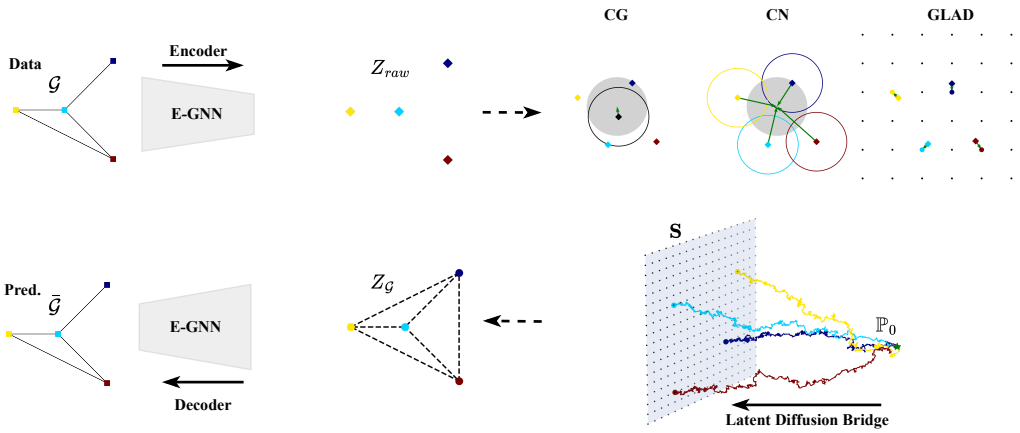

Figure 1. (**Top**) Graph encoded to a set of node embeddings, colored diamonds. We show three studied graph-latent structures including continuous graph (CG), continuous node (CN), and quantisation (GLAD). Grey disks denote the normal prior, colored circles denote posterior distributions. GLAD distinctively maps the raw-node embeddings to points on a uniformly quantized space, black dots. (**Bottom**) Graph decoded from a fully-connected pseudo graph formed by a set of quantized latent nodes $Z_{\mathcal{G}}$, dashed-edge graph. We leverage diffusion bridges to learn a graph-latent distribution Π constrained over the quantized-latent structure $\mathbf{S}$. We denote a bridge's prior $\mathbb{P}_0$, and the colored paths represent an example of $Z_{\mathcal{G}}$-bridge.

C. Model Details

C.1. Pipeline and architectures

We show the GLAD's pipeline in Figure 2.

We learn data distributions that lie on both graph- and set-based structures in GLAD. As sets are more or less considered as fully-connected graphs. We opt to employ a general-purpose architecture such as graph transformers (Dwivedi & Bresson, 2020). Future works could consider a more specific choice of architecture designed to graph- or set-related domains. We adapt the graph transformer (Vignac et al., 2023) and further customize the architecture for our graph encoder ϕ, graph decoder θ, and drift model ψ. We visualize our architectures in Figure 2.

The graph encoder takes a node feature matrix X, an edge feature matrix E, and a spectral feature vector Y as inputs. Due to the inherent limitations of most message-passing neural networks (Chen et al., 2020), we compute structural features like cycle counts and spectral features of graph-Laplacian decomposition as hand-crafted input features Y. The encoder outputs a set of raw node embeddings that is quantized to obtain a set of quantized latent nodes $Z_{\mathcal{G}}$. The graph decoder takes a pseudo-latent graph, which is formed by the latent nodes $Z_{\mathcal{G}}$, a pseudo-adjacency matrix $Z_{\mathcal{G}} Z_{\mathcal{G}}^T$; in addition, we incorporate a global representation feature to the decoder by pooling the set of latent nodes $\oplus Z_{\mathcal{G}}$. The drift model is similar in the design of graph decoder, however it takes a noisy set of latent nodes Z_t sampled at different time step t from the $Z_{\mathcal{G}}$-bridge. The model learns the dynamic of Π-bridge drift by η_t^{ψ}, which structure is identical to $Z_{\mathcal{G}}$.

C.2. Quantization

We empirically found that a quantized latent space of dimension $f = 6$ works well to encode graphs within low reconstruction errors. In all experiments, we fix the quantization vector $L = [5, 5, 5, 5, 5, 5]$ where at each latent node dimension $L_j,\ 1 \leqslant j \leqslant 6$, we quantize to one of five possible values $[-2, -1, 0, 1, 2]$. In total, we have $K = 5^6$ quantization points that are uniformly distributed in the discrete latent space. These points serve as the referent points to which latent nodes are assigned. For illustration, given a raw latent node vector Z^i, its quantized version $Z_{\mathcal{G}}^i$ is obtained as following:

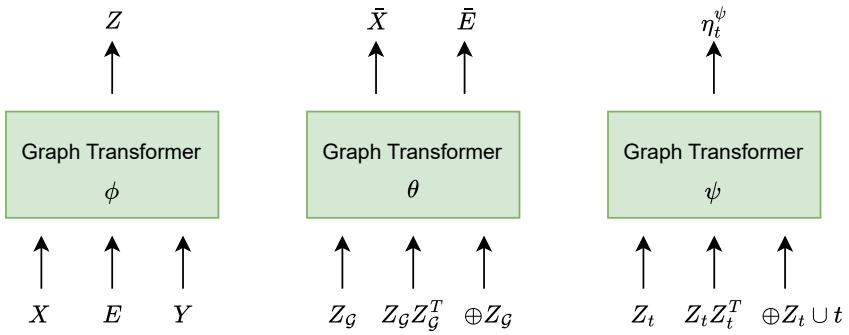

Figure 2. (**Left**) graph-encoder architecture, (**Middle**) graph-decoder architecture, (**Right**) model-bridge-drift architecture. We denote $\oplus$ as pooling operator on a set of latent nodes, and $\cup$ as concatenation operator.

Table 2. Statistics of molecule graphs on QM9 and ZINC250k.

	Number of Graphs	Number of Nodes	Number of Node Types	Number of Edge Types
QM9	133885	$1 \leqslant \lvert X \rvert \leqslant 9$	4	3
ZINC250K	249455	$6 \leqslant \lvert X \rvert \leqslant 38$	9	3

$$Z^i = \begin{bmatrix} -1.1 \\ -1.7 \\ -0.01 \\ 0.1 \\ 3.2 \\ 0.6 \end{bmatrix} \longrightarrow \frac{L_j}{2} \tanh Z^{ij} \longrightarrow \begin{bmatrix} -2.00 \\ -2.34 \\ -0.03 \\ 0.25 \\ 2.49 \\ 1.34 \end{bmatrix} \longrightarrow \text{Rounding} \longrightarrow Z_\mathcal{G}^i = \begin{bmatrix} -2 \\ -2 \\ 0 \\ 0 \\ 2 \\ 1 \end{bmatrix}$$

The same technique is applied on all latent nodes of the set.

D. Experimental Details

D.1. Molecule generation

Metrics We quantitatively evaluate all methods in terms of validity without post-hoc corrections (V ↑), uniqueness (U ↑), and novelty (N ↑) of 10.000 generated molecules. Validity without post-hoc corrections is the percentage of valid molecules without valence correction and edge resampling, uniqueness is as the percentage of valid molecules that are unique, and novelty is the percentage of valid molecules not in the training set. Notably, all graph generative methods can easily achieve 100% valid molecules with simple ad-hoc valency correction rules on generated molecules.

In addition, we compute two more salient metrics that quantify how well the distribution of the generated graphs aligns with the real data distribution, namely Fréchet ChemNet Distance (FCD ↓) and Neighborhood Subgraph Pairwise Distance Kernel (NSPDK ↓). FCD evaluates the distance in the chemical space between generated and training graphs by using the activation of the ChemNet's penultimate layer, while NSPDK measures the MMD distance, showing the similarity of underlying structures between generated and test molecules, which typically considers both the atom and bond types in computation. These last two metrics show the most relevant comparisons as they directly take into account the distribution of molecular properties, e.g chemical and structural, rather than with only molecule-graph statistics.

Datasets We show the details of molecule datasets in Table 2. We kekulize the molecules by the RDKit library (Landrum et al., 2016) and remove hydrogen atoms. We use the same train / test split with the baselines for a fair comparison.

Table 3. Statistics of generic graphs on community small, ego small, and enzymes.

	Number of Graphs	Number of Nodes	Real / Synthetic
COMMUNITY-SMALL	100	$12 \leqslant \lvert X\rvert \leqslant 20$	Synthetic
EGO-SMALL	200	$4 \leqslant \lvert X\rvert \leqslant 18$	Real
ENZYMES	587	$10 \leqslant \lvert X\rvert \leqslant 125$	Real

Baselines We compare GLAD with generative models that operate on graph spaces, including continuous flow GraphAF (Shi et al., 2020), conditional flow MoFlow (Zang & Wang, 2020), continuous diffusion GDSS (Jo et al., 2022), discrete diffusion DiGress (Vignac et al., 2023), and auto-regressive diffusion GraphArm (Kong et al., 2023). We exclusively compare with other generative models that work with discrete latent structures, including auto-regressive discrete flow GraphDF (Luo et al., 2021), and auto-regressive discrete autoencoder DGAE (Boget et al., 2024).

Results We did ablation study on different hyper-parameters and chose the ones that exhibit the highest score on the product between uniqueness and validation. Our best hyperparameters are described in Table 8. We present the detailed results in Table 7. Note that, we do not show the validity with corrections as this score is typically 100% for all methods.

QM9's novelty issues The dataset comprises all enumerated small molecules that compose from only four atoms C, N, O, F. Hence, the generation of novel graphs beyond this dataset might not accurately reflect the model's ability to capture the underlying data distribution. Typically, models that capture better the chemical property (FCD) distribution of QM9 tend to exhibit lower novelty scores. However, such problem can be alleviated when train models on large-scale datasets such as ZINC250k.

Visualisation of generated molecules We provide non-cherry-picked molecule samples generated from GLAD in Figure 4 and Figure 5.

D.2. Generic graph generation

In this section, we provide some experimental results on generic-graph datasets.

Metrics We evaluate model performance by computing the maximum-mean discrepancy (MMD) between the generated-set and test-set distributions of the following graph statistics: degree (D $\downarrow$), clustering coefficient (C $\downarrow$), and orbit 4-node occurrences (O $\downarrow$). At each evaluation, we generate an equal number of graphs to the current test set. We adopt the Gaussian Earth Mover's Distance (EMD) kernel to calculate MMD thanks to its computational stability.

Datasets We measure GLAD's ability to capture the underlying structures of generic graphs on three datasets: (a) ego-small (Sen et al., 2008), (b) community-small, and (c) enzymes (Schomburg et al., 2004). Table 3 shows their detailed description. On these graphs, there are no explicit information about nodes available for training. We therefore use node degrees as augmented-node features to the graph-encoder's inputs. We use the same train- and test- splits as the baselines for a fair comparison.

Baselines Here under the approach working directly on graph space, we compare to auto-regressive GraphRNN (You et al., 2018), auto-regressive flow GraphAF (Shi et al., 2020), auto-regressive diffusion GraphArm (Kong et al., 2023), continuous diffusion GDSS (Jo et al., 2022), and discrete diffusion DiGress (Vignac et al., 2023). Under the latent-space direction, we have continuous flow GraphNF(Liu et al., 2019), discrete flow GraphDF (Luo et al., 2021), and discrete autoencoder DGAE (Boget et al., 2024). In all baselines, we use the continuous- and discrete- terms with an implicit indication on how data represented or treated by individual approach.

Results We report the means and standard deviations of 15 runs for each experiment. We do not include the GNF's enzyme baseline due to its computation constraint on large graphs. We did an ablation study on GLAD's hyper-parameters and chose the checkpoints yielding the lowest-average MMD distance. Our main hyper-parameters are described in Table 8. We compare between the baselines in Table 4 and provide the detailed results in Table 6.

Table 4. **Generation results on the generic graph datasets**. We show the mean values of 15 runs for each experiment. The baselines are sourced from (Jo et al., 2022; Kong et al., 2023). We compare with generative models operating on graph space (top row) and on latent-graph space (bottom row). Hyphen (-) denotes unreproducible results. The 1st and 2nd best results are bolded and underlined, respectively.

	COMMUNITY-SMALL				EGO-SMALL				ENZYMES			
	D ↓	C ↓	O ↓	AVG	D ↓	C ↓	O ↓	AVG	D ↓	C ↓	O ↓	AVG
GraphRNN (You et al., 2018)	0.080	0.120	0.040	0.080	0.090	0.220	<u>0.003</u>	0.104	0.017	0.062	0.046	0.042
GraphAF (Shi et al., 2020)	0.180	0.200	0.020	0.133	0.030	0.110	**0.001**	0.047	1.669	1.283	0.266	1.073
GDSS (Jo et al., 2022)	0.045	0.086	0.007	0.046	0.021	0.024	0.007	0.017	0.026	0.061	0.009	0.032
DiGress (Vignac et al., 2023)	0.047	**0.041**	0.026	0.038	<u>0.015</u>	0.029	0.005	0.016	**0.004**	0.083	<u>0.002</u>	0.030
GraphArm (Kong et al., 2023)	0.034	0.082	**0.004**	0.040	0.019	<u>0.017</u>	0.010	<u>0.015</u>	0.029	0.054	0.015	0.033
GNF (Liu et al., 2019)	0.200	0.200	0.110	0.170	0.030	0.100	**0.001**	0.044	-	-	-	-
GraphDF (Luo et al., 2021)	0.060	0.120	0.030	0.070	0.040	0.130	0.010	0.060	1.503	1.061	0.202	0.922
DGAE (Boget et al., 2024)	<u>0.032</u>	0.062	<u>0.005</u>	<u>0.033</u>	0.021	0.041	0.007	0.023	0.020	<u>0.051</u>	0.003	<u>0.025</u>
GLAD	**0.029**	<u>0.047</u>	0.008	**0.028**	**0.012**	**0.013**	0.004	**0.010**	<u>0.012</u>	**0.014**	**0.001**	**0.009**

We observe that our model achieves competitive performance compared to the state-of-the-art baselines; it consistently maintains the lowest-average MMD distance across three datasets. Moreover, GLAD significantly outperforms all baselines that make explicit use of latent-space structures, namely GNF, GraphDF, and DGAE. These accomplishments would highlight two pivotal strengths of GLAD: first, our quantized graph-latent space can encode rich local graph sub-structures, serving as a cornerstone for any latent-driven generative models. Second, the diffusion bridges work seamlessly within the proposed latent structure by design, which is underscored by its capability to learn the set-based latent representation of graphs. While GLAD does not operate directly on the graph space like GDSS and DiGress, which are also diffusion-based frameworks, GLAD demonstrates superior performance to capture the underlying topologies of graphs in a holistic manner.

D.3. Ablation study on latent spaces

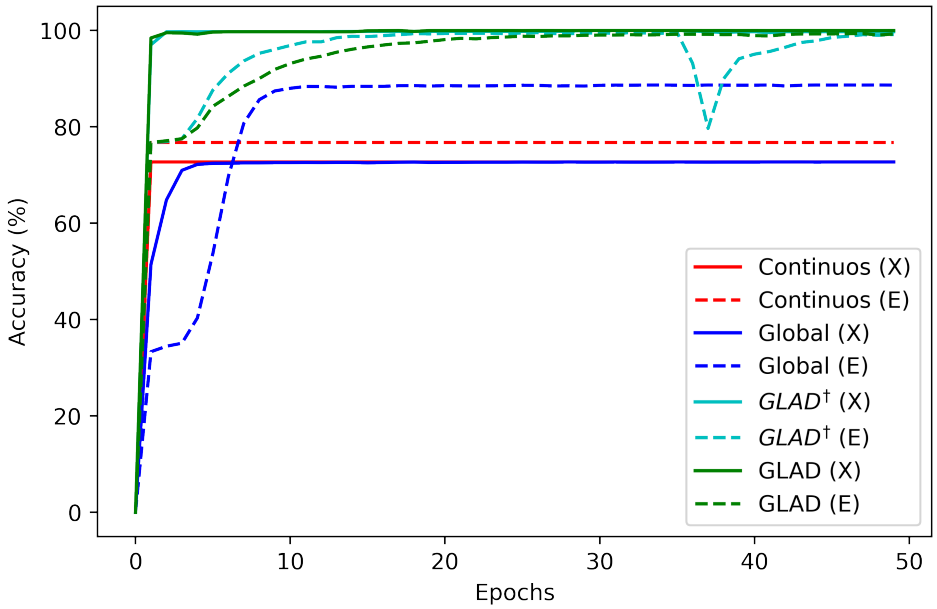

Figure 3. Test reconstructions of different latent spaces on QM9. Node (X) and edge (E) reconstruction accuracies: i) continuous-graph latent representation (C-G). ii) continuous-node latent representation (C-N) iii) unquantized discrete-node latent space GLAD†, iv) quantised discrete-node latent space GLAD.

Prior	QM9				
	Val. ↑	Uni. ↑	Nov. ↑	NSPDK ↓	FCD ↓
$\mathcal{N}(\mathbf{0}, \mathbf{1})$	95.38	97.38	41.54	0.0004	0.330
$\mathbf{0}$	97.12	97.52	38.75	0.0003	0.201
$\mathbf{0}^{\dagger}$	83.12	97.54	62.44	0.0009	1.207

Table 5. **Ablations on prior $\mathbb{P}_0$ and quantization $\mathcal{F}$.** Generation results when the prior distribution is initialized as a standard normal distribution and a Dirac Δ distribution on the fixed point $\mathbf{0}$; the latter is ablated with and without quantization, denoted by $\mathbf{0}$ and $\mathbf{0}^{\dagger}$ respectively.

Graph reconstruction We argued the quantized-graph latent space is pivotal to the good performance of GLAD. We now verify our claim by empirically comparing the reconstruction ability of GLAD to the following latent spaces: i) continuous-graph latent space; we train a vanilla VAE on a global structure, pooled latent nodes, to get a continuous-latent representation ii) continuous-node latent space; similar to i) we also train a vanilla VAE, however applied on a set of latent nodes. Finally in iii) we simply use a set of raw node embeddings without any constraints imposed to the graph-latent space, akin to a standard autoencoder for graphs. We evaluate the quality of the underlying graph latent spaces using their bond-type (X) and atom-type (E) reconstruction accuracies on QM9, Figure 3.

The two continuous latent spaces i) and ii) have considerably lower accuracies compared to the two GLAD variants, which preserve to a different extent the original discrete topology. Both GLAD variants converge very fast to nearly 100% reconstructions. GLAD has a small advantage when it comes to the bond type (E), for which its training is more stable than that of its unquantized sibling GLAD†. The latent spaces of i) and ii) suffer from the so-called node-collapse problem. They are not able to well describe different atom local structures. As a result, most reconstructed nodes are assigned to the dominant Carbon atom, roughly corresponding the 70% of atoms in QM9.

Quantization effects We study the effect of quantisation by removing the quantization step. Node embeddings remain discrete-, but non-quantised- structures, and they are embedded into a continuous domain, which has the same dimensionality to the quantized one ($f = 6$), denoted as $\Omega_c = [L_{\min}, L_{\max}]^{(N \times f)}$. We retrain our graph autoencoder, model bridges, and obtain $L_{\min}, L_{\max}$ that correspond to the min- and max- values of learned latent nodes for the entire training set. The Π-bridge's drift on a continuous domain is computed as:

$$\left[\eta^{\Pi}\right]^h = \sigma_t^2 \nabla_{Z_t^h} \log \left(F(\frac{Z_t^h - L_{\min}}{\sqrt{\beta_T - \beta_t}}) - F(\frac{Z_t^h - L_{\max}}{\sqrt{\beta_T - \beta_t}}) \right), \; 1 \leqslant h \leqslant (N \times f) \tag{15}$$

where $Z_t \in [L_{\min}, L_{\max}]^{(N \times f)}$, F is the standard Gaussian Cumulative Distribution Function (CDF).

As per Figure 3, the two GLAD variants have a rather similar reconstruction performance, but this is not the case when we compare their generation performance, described in Table 5. There we see that the diffusion bridges can not capture well the graph distribution when they operate on the unquantized space, with a performance drop for most measures (importantly on the two measures capturing molecular distributions, NSPDK and FCD), showing the benefits of the quantised discrete latent space. Quantization acts as a spatial regulariser over the non-quantised structures by constraining latent nodes on high-dimensional discretized grid instead of letting them localized in a infinite-continuous space. We hypothesize this spatial regularisation is non-trivial to construct effective latent structures for graphs. We further provide the detailed results in Table 7.

D.4. Graph Latent Diffusion Priors

Here we evaluate model performance on different priors; a fixed point prior $\mathbb{P}_0 = \mathbf{0}$, which is the central mass of our quantized latent space, and a standard normal distribution prior, $\mathbb{P}_0 = \mathcal{N}(\mathbf{0}, \mathbf{I})$. We conduct the experiments on QM9 and give the results in Table 5. As a result of the steering forces of the bridge processes, there are only negligible differences between the two priors making prior selection less of a nuance. With the fixed prior, we obtain generated molecules with higher validity scores, closer to the training distribution in both chemical space (FCD score), structural space (NSPDK score). Using the normal prior we have slightly better performance in terms of uniqueness and novelty scores.

Table 6. **Detailed results on generic graphs**. We show the means and standard deviations of 15 runs for each experiment.

	D ↓	C ↓	O ↓
COMMUNITY-SMALL	0.029 ± 0.020	0.047 ± 0.025	0.008 ± 0.006
EGO-SMALL	0.012 ± 0.007	0.013 ± 0.006	0.004 ± 0.002
ENZYMES	0.012 ± 0.004	0.014 ± 0.003	0.001 ± 0.000

Table 7. **Detailed results on molecule graphs, bridge-prior- and quantisation- ablations**. We show the means and standard deviations of 3 runs on each dataset. ZINC250k and QM9 are GLAD, i.e. fixed prior and quantised discrete latent space. QM9* ablates the use of standard normal distribution for the bridge prior. QM9† ablates the effect of non-quantisation in the latent space.

	V ↑	U ↑	N ↑	NSPDK ↓	FCD ↓
ZINC250K	81.81 ± 0.29	100 ± 0.00	99.99 ± 0.01	0.0021 ± 0.0001	2.54 ± 0.05
QM9	97.12 ± 0.03	97.52 ± 0.06	38.75 ± 0.25	0.0003 ± 0.0000	0.201 ± 0.003
QM9*	95.38 ± 0.00	97.38 ± 0.00	41.54 ± 0.01	0.0004 ± 0.0000	0.330 ± 0.013
QM9†	83.12 ± 0.01	97.54 ± 0.01	62.44 ± 0.53	0.0009 ± 0.0000	1.207 ± 0.003

	Hyperparameter	Community-small	Ego-small	Enzymes	QM9	ZINC250k
ϕ	Number of heads	8	8	8	8	8
	Number of layers	8	8	8	8	8
	Hidden dimension X	256	256	128	256	256
	Hidden dimension E	128	128	32	128	128
	Hidden dimension Y	64	64	64	64	64
θ	Number of heads	8	8	8	8	8
	Number of layers	4	4	4	4	4
	Hidden dimension X	256	256	128	256	256
	Hidden dimension E	128	128	32	128	128
	Hidden dimension Y	64	64	64	64	64
f_ψ	Number of heads	8	8	8	8	8
	Number of layers	4	4	8	8	8
	Hidden dimension X	256	256	256	256	256
	Hidden dimension E	128	128	128	128	128
	Hidden dimension Y	64	64	64	64	64
SDE	$\sigma_{\min}$	1.0	1.0	1.0	1.0	1.0
	$\sigma_{\max}$	3.0	3.0	3.0	3.0	3.0
	Number of diffusion steps	1000	1000	1000	1000	1000
AE	Optimizer	Adam	Adam	Adam	Adam	Adam
	Learning rate	$5e^{-4}$	$5e^{-4}$	$5e^{-4}$	$5e^{-4}$	$5e^{-4}$
	Batch size	32	32	32	5240	512
	Number of epochs	3000	3000	3000	1000	1000
Model Bridge	Optimizer	Adam	Adam	Adam	Adam	Adam
	Learning rate	$[1e^{-4}, 5e^{-4}]$	$[1e^{-4}, 5e^{-4}]$	$[1e^{-4}, 5e^{-4}]$	$[1e^{-4}, 2e^{-3}]$	$[3e^{-4}, 2e^{-3}]$
	Batch size	64	64	64	1280	512
	Number of epochs	5000	5000	5000	2000	2000

Table 8. **GLAD hyperparameters** . We show the main hyperparameters for the generic and molecule generation tasks. We provide those hyperparameters to encoder ϕ, decoder θ, drift model ψ, graph autoencoder (AE), model bridge (Model Bridge), and latent diffusion bridge processes (SDE).

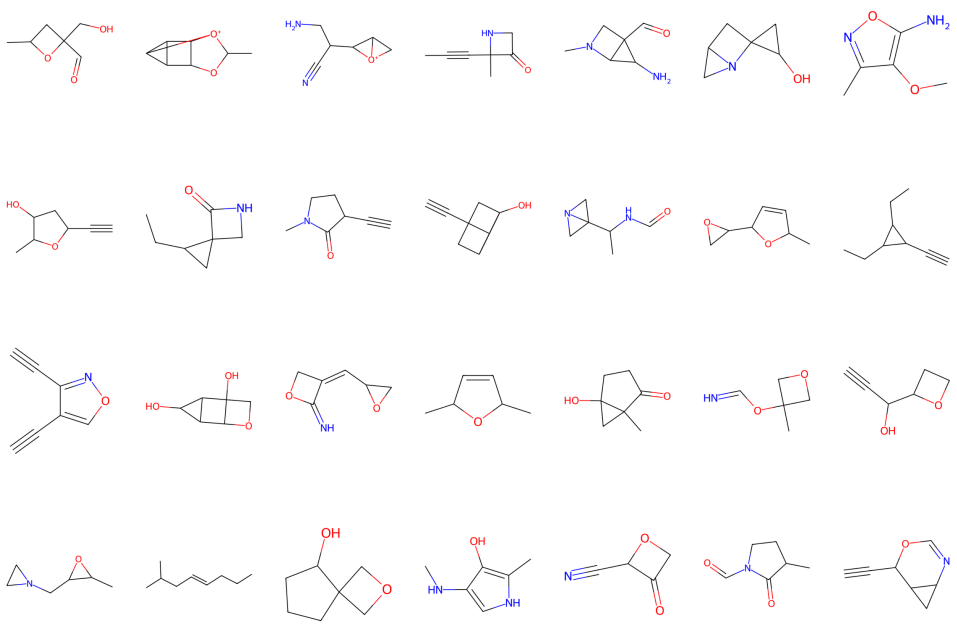

Figure 4. **Visualization of generated samples on QM9 from GLAD.**

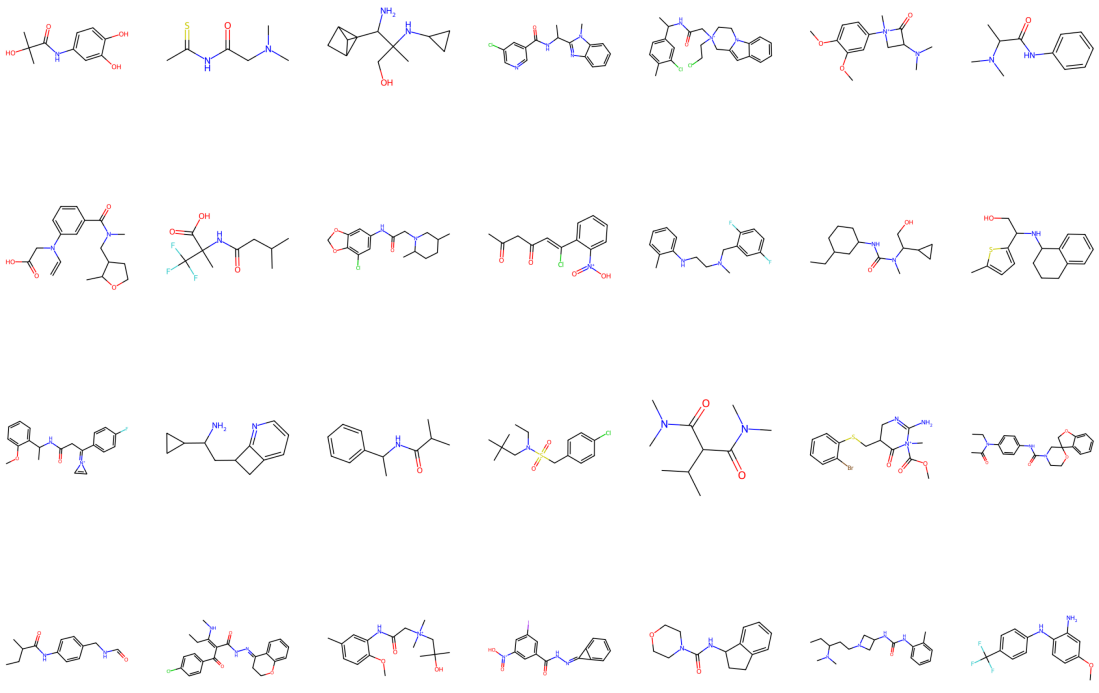

Figure 5. **Visualization of generated samples on ZINC250k from GLAD.**