Graph Representation Learning with Diffusion Generative Models

Daniel Wesego
Department of Computer Science
University of Illinois Chicago
Chicago, IL 60607
dweseg2@uic.edu

Abstract

Diffusion models have established themselves as state-of-the-art generative models across various data modalities, including images and videos, due to their ability to accurately approximate complex data distributions. Unlike traditional generative approaches such as VAEs and GANs, diffusion models employ a progressive denoising process that transforms noise into meaningful data over multiple iterative steps. This gradual approach enhances their expressiveness and generation quality. Not only that, diffusion models have also been shown to extract meaningful representations from data while learning to generate samples. Despite their success, the application of diffusion models to graph-structured data remains relatively unexplored, primarily due to the discrete nature of graphs, which necessitates discrete diffusion processes distinct from the continuous methods used in other domains. In this work, we leverage the representational capabilities of diffusion models to learn meaningful embeddings for graph data. By training a discrete diffusion model within an autoencoder framework, we enable both effective autoencoding and representation learning tailored to the unique characteristics of graph-structured data. We extract the representation from the combination of the encoder's output and the decoder's first time step hidden embedding. Our approach demonstrates the potential of discrete diffusion models to be used for graph representation learning. The code can be found at https://github.com/DanielMitiku/Graph-Representation-Learning-with-Diffusion-Generative-Models

1 Introduction

Representation learning is a central paradigm in modern machine learning, aiming to transform raw data into informative and compact representations that capture the underlying structure of the domain. Such representations enable a broad range of downstream tasks, including classification, clustering, and generation, by providing features that are more amenable to statistical modeling than the raw inputs [Bengio et al., 2013]. Advances in deep learning have driven rapid progress in different areas: variational autoencoders extract latent codes from images [Kingma and Welling, 2014, Rezende et al., 2014], transformer-based models learn contextualized embeddings of text [Vaswani et al., 2017], and other areas such as robotics [Tereda et al., 2024]. In the context of graphs, different forms of Graph Neural Networks (GNNs) produce representations of nodes, edges, and entire graphs that preserve structural and feature information [Hamilton et al., 2017]. These methods have established representation learning as a unifying principle across domains such as computer vision, natural language processing, multimodal data, and graph analysis [Bengio, 2009, Wesego and Rooshenas, 2024a, Hinton and Salakhutdinov, 2006].

Graphs are powerful structures for modeling relationships between entities, and they are widely used in domains such as social networks, biological networks, transportation systems, and knowledge

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: NPGML.

graphs [Xia et al., 2021]. In these domains, graphs represent complex systems, where nodes typically correspond to entities, and edges represent relationships or interactions between them. The ability to analyze and extract insights from graph-structured data is critical for applications such as recommendation systems, drug discovery, fraud detection, social network analysis [Ying et al., 2018, Velickovic et al., 2018], and others. Graph representation learning has therefore become essential, aiming to map graph-structured data into low-dimensional vector embeddings that preserve both structure and features. These embeddings facilitate efficient analysis and downstream tasks such as node classification, link prediction, and graph classification Kipf and Welling [2017]. Despite significant progress, challenges remain in graph representation learning. Graphs often exhibit highly complex structures, with varying node degrees, long-range dependencies, and hierarchical relationships that are difficult to compress into low-dimensional embeddings [Kipf and Welling, 2017]. Furthermore, real-world graphs are frequently heterogeneous, containing multiple types of nodes, edges, and attributes that must be modeled within a unified framework [Zhang et al., 2019]. Dynamic graphs introduce an additional challenge, where evolving structures require methods that can adapt representations to capture temporal changes [Qin et al., 2023].

Parallel to these developments, recent advances in deep generative models, particularly diffusion models, have opened new directions for representation learning [Preechakul et al., 2022]. Diffusion models have achieved remarkable success in high-quality sample generation across domains such as image, audio, and video synthesis [Ho et al., 2020, Tang et al., 2023, Wesego and Rooshenas, 2024b], and are beginning to be explored in the context of graphs [Vignac et al., 2023]. Their iterative denoising process naturally learns hierarchical and expressive latent structures, suggesting that diffusion models can provide powerful graph representations. However, the use of diffusion models for graph representation learning remains at an early stage, with several open challenges.

In this work, we investigate discrete diffusion autoencoders for graph representation learning. Discrete diffusion models are particularly well-suited for graph data, where node and edge features are often categorical or discrete in nature. By leveraging their generative capabilities, we aim to enhance the quality and expressiveness of graph embeddings, while enabling unsupervised learning in settings where labeled data is scarce or costly. We evaluate our framework on benchmark datasets, including the Protein and IMDB-B dataset from TUDatasets Morris et al. [2020], and demonstrate its effectiveness in downstream tasks. The application of diffusion models to graph learning holds significant promise Yang et al. [2023]. Beyond improved embeddings, their generative nature allows the synthesis of novel graphs, which is especially valuable in applications such as molecular graph generation and drug discovery Jin et al. [2018]. The central hypothesis is that the strong generative capacity of diffusion models requires learning compressed, informative representations that can serve as robust graph embeddings, as proven in other modalities.

To achieve this, we make use of diffusion autoencoders, which extend a standard diffusion model into an autoencoder framework, where an encoder network learns representations and a diffusion decoder reconstructs the input data conditioned on the output of the encoder [Preechakul et al., 2022]. Discrete diffusion autoencoders, in particular, are tailored for discrete data, making them especially suitable for graphs Austin et al. [2023], and in this paper, we explore their potential for graph representation learning. Figure 1 shows the overall framework that is used to get the embeddings after training. Our contributions are summarized as follows:

- **Framework.** We introduce a discrete diffusion graph autoencoder (DDGAE) for graph-structured data. Our model leverages discrete diffusion processes to progressively denoise graph inputs, enabling the capture of complex structural patterns and dependencies.
- **Representation learning.** We show how discrete diffusion models can improve the quality of graph representations by transforming the discrete nature of graph structures into a latent embedding. By integrating diffusion models with an autoencoder architecture, DDGAE learns compact, expressive graph embeddings. The final representation combines the encoder output with the embeddings of the diffusion decoder, enhancing representational richness. Unlike standard diffusion models that rely on repetitive sampling, our approach requires only a single-step sampling from the diffusion decoder during inference, while naturally supporting unsupervised learning and reducing reliance on labeled data.
- **Empirical validation.** We conduct extensive experiments on different graph benchmark datasets from TUDatasets, demonstrating that DDGAE achieves superior performance in downstream graph classification tasks compared to strong baselines.

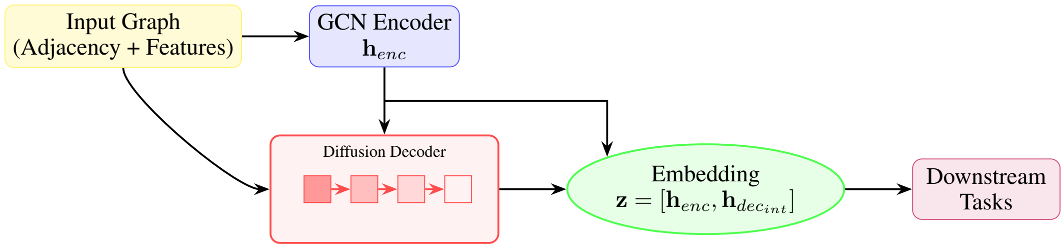

Figure 1: Discrete Diffusion Graph AutoEncoder (DDGAE) embedding extraction: The trained encoder extracts features, which are concatenated with the intermediate output of the trained diffusion decoder as the final embedding $\mathbf{z}$ that will be used for different downstream tasks.

2 Related Works

2.1 Diffusion Models

Diffusion models have emerged as powerful generative models, achieving strong performance across domains including image synthesis, molecule generation, and representation learning [Ho et al., 2020, Vignac et al., 2023, Yang et al., 2023, Wesego and Rooshenas, 2024a]. These models operate by progressively adding noise to data in the forward diffusion process and learning to reverse this process. This iterative framework enables the generation of high-quality samples that closely resemble the target data distribution. Denoising Diffusion Probabilistic Models (DDPMs) introduced the foundational framework for diffusion-based generation, demonstrating their ability to produce realistic images through iterative denoising [Ho et al., 2020]. Improved DDPMs further enhanced this framework by optimizing noise schedules, learning the variance, and refining architectural designs [Dhariwal and Nichol, 2021]. Other notable advancements include Stable Diffusion, which integrates text conditioning with diffusion models in the latent space to generate high-resolution, text-guided images [Rombach et al., 2021].

2.2 Discrete Diffusion Models for Graphs

Discrete diffusion models extend diffusion principles to discrete data, making them suitable for domains such as text, categorical attributes, and graphs. Unlike continuous diffusion models, which operate in a continuous state space, discrete diffusion models handle data in a way that respects the inherent discreteness of the input [Austin et al., 2023, Hoogeboom et al., 2021]. When applied to graphs, these models leverage diffusion processes to capture the complex relationships and hierarchical structures inherent in graph-structured data. Since graphs are inherently discrete, most diffusion models for graphs operate directly in the discrete space [Vignac et al., 2023, Chen et al., 2023]. DiGress, a discrete denoising diffusion model, generates graphs with categorical node and edge attributes, demonstrating strong effectiveness in handling graph-structured data [Vignac et al., 2023]. Wang et al. [2024] further expanded DiGress by training it on graphs from multiple domains to improve generalization. EDGE [Chen et al., 2023] is another discrete diffusion model that generates adjacency matrices conditioned on degree distributions. By using an absorbing distribution of empty graphs as the terminal state, EDGE reduces the number of diffusion steps, effectively addressing graph sparsity. Together, these approaches highlight the growing potential of discrete diffusion models for structured data generation.

2.3 Graph Representation Learning

Graph representation learning has been a central research area, focusing on transforming graph-structured data into low-dimensional embeddings that preserve structural and semantic information. Several approaches have been proposed to tackle the challenges of learning meaningful graph representations, including contrastive, generative, and autoencoder-based methods. GraphCL (Graph Contrastive Learning) introduced a self-supervised framework that leverages graph augmentations to maximize agreement between representations of the same graph under different transformations [You et al., 2021b]. GraphMAE (Graph Masked Autoencoder) adapts masked autoencoding, widely used in NLP, to graphs by masking portions of the input and training the model to reconstruct them, thereby learning structural patterns [Hou et al., 2022]. GraphVAE (Graph Variational Autoencoder) is a generative framework that models the probabilistic distribution of graph data; it learns latent variables representing graph structures and attributes to compress and reconstruct graphs [Kipf and Welling, 2016]. More recently, Yang et al. [2023] applied diffusion models to graph representation learning by extracting outputs from intermediate layers and injecting directional noise.

In this study, we propose a **Discrete Diffusion Graph Autoencoder (DDGAE)** for graph representation learning. Specifically, we use discrete diffusion models as decoders over adjacency matrices and a GCN encoder to extract latent representations $\mathbf{z}$. Training is performed solely through the discrete diffusion process in the decoder, with gradients propagated back to the encoder. To enrich the learned representations, we combine the encoder output with intermediate embeddings from the diffusion decoder to form the final representation $\mathbf{z}$.

3 Methodology

This section describes the methodology of our **Discrete Diffusion Graph Autoencoder (DDGAE)** for graph-structured data. We first review the general discrete diffusion framework and then detail its application within our graph autoencoder.

3.1 Discrete Diffusion Framework

Discrete diffusion models define a Markov chain of T diffusion steps that progressively corrupt the input data into noise. Let $\mathbf{x}_0$ denote the original data and $\mathbf{x}_T$ denote the fully noised state. The forward process is governed by transition probabilities $q(\mathbf{x}_t|\mathbf{x}_{t-1})$, designed so that the distribution of $\mathbf{x}_t$ approaches a tractable noise distribution as t increases. A generative model $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ is then trained to reverse this process, iteratively denoising from $\mathbf{x}_T$ to reconstruct the original data $\mathbf{x}_0$.

The marginal probability of the data is defined as $p_\theta(\mathbf{x}_0) := \int p_\theta(\mathbf{x}_{0:T}) d\mathbf{x}_{1:T}$, where the joint distribution over the diffusion trajectory is $p_\theta(\mathbf{x}_{0:T}) := p_\theta(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$, and the forward process is $q(\mathbf{x}_{1:T}|\mathbf{x}_0) := \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})$. Training maximizes the evidence lower bound (ELBO) of the log-likelihood [Austin et al., 2023]:

$$\mathbb{E}[-\log p_\theta(\mathbf{x}_0)] \leq \mathbb{E}_q\Big[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\Big]. \tag{1}$$

This can be expressed as a sum of KL divergences over diffusion steps:

$$\begin{aligned} L_{VB} =& \mathbb{E}_{q(\mathbf{x}_0)}\Big[\underbrace{D_{KL}[q(\mathbf{x}_T|\mathbf{x}_0)||p_\theta(\mathbf{x}_T)]}_{L_T}\Big] + \\ & \sum_{t=2}^{T} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)}\Big[\underbrace{D_{KL}[q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)||p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)]}_{L_{t-1}}\Big] - \mathbb{E}_{q(\mathbf{x}_1|\mathbf{x}_0)}\Big[\underbrace{\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}\Big]. \end{aligned} \tag{2}$$

Here, L_T contains no trainable parameters and is zero by design. The final training objective is a linear combination of the ELBO and a term directly predicting $\mathbf{x}_0$, controlled by a hyperparameter λ [Austin et al., 2023]:

$$L_\lambda = L_{VB} + \lambda \mathbb{E}_{q(\mathbf{x}_0)} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} \left[-\log \tilde{p}_\theta(\mathbf{x}_0|\mathbf{x}_t)\right]. \tag{3}$$

3.2 Discrete Diffusion Graph Autoencoder

We apply the discrete diffusion framework to graph-structured data. The input graph is denoted $\mathbf{G}(\mathbf{X}, \mathbf{A})$, where $\mathbf{X}$ contains node features and $\mathbf{A}$ is the adjacency matrix. An encoder network $E_\phi(\mathbf{G})$ maps the input graph to a latent representation $\mathbf{z}_{enc}$. While any graph neural network can be used, we adopt a GCN to capture both structural and feature information effectively.

The decoder is a discrete diffusion model operating on the adjacency matrix $\mathbf{A}$, conditioned on the encoder representation $\mathbf{z}_{enc}$. During training, the adjacency matrix undergoes a forward diffusion process, producing a noisy version $\mathbf{A}_t$ according to a predefined schedule that transforms $\mathbf{A}$ into an absorbing state of zeros. The decoder, $p_\theta(\mathbf{A}_{t-1}|\mathbf{A}_t, \mathbf{z}_{enc})$, iteratively reconstructs $\mathbf{A}$ from the noisy state, step-by-step, conditioned on $\mathbf{z}_{enc}$. The reconstructed adjacency matrix is denoted $\hat{\mathbf{A}}$.

The final graph representation $\mathbf{z}$ is obtained by concatenating the encoder embedding $\mathbf{z}_{enc}$ with an intermediate embedding from the UNet-based diffusion decoder. After training, a single pass through the model suffices. First, we obtain $\mathbf{z}_{enc}$ forwarding the data through the encoder; this is passed to the decoder along with $\mathbf{A}_0$ to obtain the intermediate embedding, which is concatenated with $\mathbf{z}_{enc}$ to produce the final embedding $\mathbf{z}$ used for downstream tasks.

4 Experiments

This section outlines the experimental setup of our proposed model and the baselines. We compare our model against relevant baselines on a graph classification task using the PROTEINS and IMDB-BINARY datasets.

4.1 Dataset

The PROTEINS dataset comprises 1113 graphs representing proteins, each classified as either an enzyme (class 1) or non-enzyme (class 0). The graphs have an average of 39 nodes, with each node representing an amino acid and edges representing interactions between them. The IMDB-BINARY dataset consists of 1000 ego-networks extracted from the Internet Movie Database (IMDB), where each graph represents the collaboration network of actors in a movie. The graphs are classified into two categories based on the genre of the movie. Nodes correspond to actors, and edges indicate co-appearances in the same movie. Each graph has an average of 19 nodes and 193 edges. These datasets provide a suitable benchmark for evaluating our model's ability to learn meaningful representations from graph-structured data [Morris et al., 2020].

4.2 Baselines

We compare our discrete diffusion graph autoencoder (DDGAE) against multiple baseline models used in [Yang et al., 2023], including Infograph [Sun et al., 2019], GraphCL [You et al., 2021b], JOAO [You et al., 2021a], GCC [Qiu et al., 2020], MVGRL [Hassani and Khasahmadi, 2020], 2020), GraphMAE [Hou et al., 2022], and DDM [Yang et al., 2023]. From supervised learning methods, the comparisons include GIN [Xu et al., 2018].

4.3 Model Architecture and Training

Our model utilizes a Graph Convolutional Network (GCN) encoder to extract the latent representation $\mathbf{z}_{enc}$ from the input graph's features and adjacency matrix. The node embeddings from the GCN are aggregated using mean pooling to obtain a graph-level embedding. The decoder employs a UNET architecture, commonly used in diffusion models [Preechakul et al., 2022], to reconstruct the adjacency matrix from the latent representation and noise. A diffusion timestep of 32 was used for the discrete diffusion process, and a latent size of 64 + 64 = 128 dimensions is used across our model. The models are trained on a single Nvidia GPU T4. The training takes approximately 1 hour for each dataset.

4.4 Evaluation

To evaluate the quality of the learned graph representations, we adopt the procedure by Yang et al. [2023] where we first extract the representations from the models and train an SVM classifier using 10-fold cross-validation on the extracted representations $\mathbf{z}$. We use classification accuracy as an evaluation metric on how good the learned representations are across the different models. This evaluation scheme allows us to directly assess the effectiveness of the learned representations in capturing discriminative information for graph classification from the graph properties.

4.5 Results

Table 1 presents the main results reporting the accuracy of the models trained on the representations learned by each model. The baselines used are similar to Yang et al. [2023], and we evaluated our model similarly to have a fair comparison.

Table 1: Results of supervised (top 2) and unsupervised representation learning for graph classification datasets

Dataset	IMDB-B	PROTEINS
GIN	75.1±5.1	76.2±2.8
Infograph	73.03±0.87	74.44±0.31
GraphCL	71.14±0.44	74.39±0.45
JOAO	70.21±3.08	74.55±0.41
GCC	72	-
MVGRL	74.20±0.70	-
GraphMAE	75.52±0.66	75.30±0.39
DDM	76.40±0.22	75.47±0.50
DDGAE	**76.90±0.03**	**76.28±0.05**

Our DDGAE model achieves the highest test accuracy on both datasets, demonstrating the superiority of the learned representations compared to the baseline models. This result highlights the effectiveness of our approach in capturing the complex structural information within graph data, leading to more discriminative and informative representations, opening a new research path towards using discrete diffusion autoencoder models for graph representation learning.

5 Conclusion and Discussion

In this paper, we introduce a discrete diffusion graph autoencoder model (DDGAE) for learning representations of graph-structured data. Our approach leverages the power of discrete diffusion models to capture the complex dependencies within the graph nodes and edges. By combining this generative framework with an encoder network, we learn a latent representation that effectively captures the underlying structure of the graph data. This representation can be used for various downstream tasks, such as graph generation, classification, and other use cases.

Despite the promising results, our approach has some limitations. First, the computational cost of training diffusion-based decoders on large graphs can be significant. Second, our current model primarily focuses on static graphs with categorical node and edge features, and does not directly handle dynamic graphs. For future work, we plan to extend DDGAE to address these limitations by exploring more efficient diffusion schedules, incorporating heterogeneous and temporal graph data, and evaluating the model across a wider variety of datasets and tasks. Additionally, we aim to investigate the integration of contrastive or self-supervised objectives to further enhance the quality of learned representations. Overall, we believe that discrete diffusion autoencoders offer a promising new direction for graph representation learning, and our work lays the foundation for further exploration in this area.

References

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces, 2023. URL `https://arxiv.org/abs/2107.03006`.

Yoshua Bengio. Learning deep architectures for AI. *Found. Trends Mach. Learn.*, 2(1):1–127, 2009. doi: 10.1561/2200000006. URL `https://doi.org/10.1561/2200000006`.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: a review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798—1828, August 2013. ISSN 0162-8828. doi: 10.1109/tpami.

2013.50. URL http://www.cs.princeton.edu/courses/archive/spring13/cos598C/RepresentationLearning-AReviewandNewPerspectives.pdf.

Xiaohui Chen, Jiaxing He, Xu Han, and Li-Ping Liu. Efficient and degree-guided graph generation via discrete diffusion modeling, 2023. URL https://arxiv.org/abs/2305.04111.

Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat gans on image synthesis. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 8780–8794, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/49ad23d1ec9fa4bd8d77d02681df5cfa-Abstract.html.

William L. Hamilton, Rex Ying, and Jure Leskovec. Representation learning on graphs: Methods and applications. *IEEE Data Eng. Bull.*, 40(3):52–74, 2017. URL http://sites.computer.org/debull/A17sept/p52.pdf.

Kaveh Hassani and Amir Hosein Khasahmadi. Contrastive multi-view representation learning on graphs. In *International conference on machine learning*, pages 4116–4126. PMLR, 2020.

G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006. doi: 10.1126/science.1127647. URL https://www.science.org/doi/abs/10.1126/science.1127647.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 12454–12465, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/67d96d458abdef21792e6d8e590244e7-Abstract.html.

Zhenyu Hou, Xiao Liu, Yukuo Cen, Yuxiao Dong, Hongxia Yang, Chunjie Wang, and Jie Tang. Graphmae: Self-supervised masked graph autoencoders, 2022. URL https://arxiv.org/abs/2205.10803.

Wengong Jin, Regina Barzilay, and Tommi S. Jaakkola. Junction tree variational autoencoder for molecular graph generation. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 2328–2337. PMLR, 2018. URL http://proceedings.mlr.press/v80/jin18a.html.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6114.

Thomas N. Kipf and Max Welling. Variational graph auto-encoders, 2016. URL https://arxiv.org/abs/1611.07308.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

Christopher Morris, Nils M. Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. Tudataset: A collection of benchmark datasets for learning with graphs, 2020. URL https://arxiv.org/abs/2007.08663.

Konpat Preechakul, Nattanat Chatthee, Suttisak Wizadwongsa, and Supasorn Suwajanakorn. Diffusion autoencoders: Toward a meaningful and decodable representation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Tiexin Qin, Benjamin Walker, Terry Lyons, Hong Yan, and Haoliang Li. Learning dynamic graph embeddings with neural controlled differential equations, 2023. URL `https://arxiv.org/abs/2302.11354`.

Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. Gcc: Graph contrastive coding for graph neural network pre-training. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1150–1160, 2020.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1278–1286, Bejing, China, 22–24 Jun 2014. PMLR. URL `https://proceedings.mlr.press/v32/rezende14.html`.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021.

Fan-Yun Sun, Jordan Hoffmann, Vikas Verma, and Jian Tang. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. *arXiv preprint arXiv:1908.01000*, 2019.

Zineng Tang, Ziyi Yang, Chenguang Zhu, Michael Zeng, and Mohit Bansal. Any-to-any generation via composable diffusion. *CoRR*, abs/2305.11846, 2023. doi: 10.48550/ARXIV.2305.11846. URL `https://doi.org/10.48550/arXiv.2305.11846`.

Amanuel Tereda, Sun Yi, and Xingguang Li. Efficient autonomous navigation for gps-free mobile robots: A vfh-based approach integrated with ros-based slam. In *ASME International Mechanical Engineering Congress and Exposition*, volume 88636, page V005T07A054. American Society of Mechanical Engineers, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017. URL `https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html`.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL `https://openreview.net/forum?id=rJXMpikCZ`.

Clément Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. Digress: Discrete denoising diffusion for graph generation. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL `https://openreview.net/forum?id=UaAD-Nu86WX`.

Yu Wang, Ryan A. Rossi, Namyong Park, Huiyuan Chen, Nesreen K. Ahmed, Puja Trivedi, Franck Dernoncourt, Danai Koutra, and Tyler Derr. Large generative graph models, 2024. URL `https://arxiv.org/abs/2406.05109`.

Daniel Wesego and Amirmohammad Rooshenas. Revising multimodal vaes with diffusion decoders. *arXiv preprint arXiv:2408.16883*, 2024a.

Daniel Wesego and Pedram Rooshenas. Score-based multimodal autoencoder. *Transactions on Machine Learning Research*, 2024b. ISSN 2835-8856. URL `https://openreview.net/forum?id=JbuP6UV3Fk`.

Feng Xia, Ke Sun, Shuo Yu, Abdul Aziz, Liangtian Wan, Shirui Pan, and Huan Liu. Graph learning: A survey. *CoRR*, abs/2105.00696, 2021. URL https://arxiv.org/abs/2105.00696.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

Run Yang, Yuling Yang, Fan Zhou, and Qiang Sun. Directional diffusion models for graph representation learning, 2023. URL https://arxiv.org/abs/2306.13210.

Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In Yike Guo and Faisal Farooq, editors, *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 974–983. ACM, 2018. doi: 10.1145/3219819.3219890. URL https://doi.org/10.1145/3219819.3219890.

Yuning You, Tianlong Chen, Yang Shen, and Zhangyang Wang. Graph contrastive learning automated. In *International conference on machine learning*, pages 12121–12132. PMLR, 2021a.

Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations, 2021b. URL https://arxiv.org/abs/2010.13902.

Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V. Chawla. Heterogeneous graph neural network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 793–803, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450362016. doi: 10.1145/3292500.3330961. URL https://doi.org/10.1145/3292500.3330961.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our claims match the results of our experiments

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: Limitation discussed in the conclusion section

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

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

   Answer: [NA]

Justification: Our equations are based on previous papers, prrofs can be found on those papers.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: Information is available to reproduce the results, plus see attached code

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

Answer: [Yes]

Justification: Attached code has everything to reproduce the results.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Yes, described in the experiment section; other details can be found in the attached code.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: Experiment has standard deviation for the different k-fold values. Couldn't retrain the models again due to computational resources, but have attached the code.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Yes, please look at the experiment section

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We respect and follow the Neurips Code of Ethics

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [NA]

    Justification: Not required for our work

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [NA]

    Justification: Not required for our work

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: Datasets used are cited correctly.

    Guidelines:

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

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [NA]

    Justification: No new assets introduced

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: Not used in our work

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: Not required for our work

    Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [NA]

Justification: Doesn't involve LLMs

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.