GraphVAEBM: Graph Generation Combining Variational Autoencoders and Energy-Based Models

Christian Mancini[1][0009−0000−4482−2004] and Daniele Castellana[1][0000−0002−7159−2272]

Dipartimento di Statistica, Informatica, Applicazioni "Giuseppe Parenti"
University of Florence, Italy
{name}.{surname}@unifi.it

Abstract. Controllable graph generation often requires embedding graph structure and target properties within a single generative model, making the incorporation of new constraints costly and limiting modularity. In this *work-in-progress* paper, we introduce GraphVAEBM, a framework that separates graph generation from property modeling. A Graph Variational Autoencoder is first trained to learn a continuous graph manifold and then frozen, while independent Energy-Based Models are trained to represent specific graph properties. This formulation enables the addition of new properties without retraining the generator. In addition, property-guided graph manipulation is performed through Langevin dynamics in the decoder noise space, with energies evaluated on decoded graph samples, resulting in a continuous optimization landscape that is easier to explore than the combinatorial graph space. Experiments on Stochastic Block Models show that the learned energy landscapes induce stable transitions between structural graph classes and support controllable graph transformations[1].

Keywords: Graph Manipulation · Energy-Based Models · Graph Representation Learning

1 Introduction

Despite recent advances in graph generative models, controllable graph generation remains challenging. Existing approaches typically learn graph structure and target properties within a single generative model, making the introduction of new constraints dependent on retraining or additional conditioning mechanisms [10,28,22]. This coupling limits modularity and complicates the composition of multiple graph properties. We propose GraphVAEBM, a framework that explicitly separates graph generation from property modeling. A Graph Variational Autoencoder (GVAE) [13] is first trained to learn a continuous graph manifold capturing the global structure of the data. The GVAE is then frozen, while independent Energy-Based Models (EBMs) [6,5] are trained to represent specific

[1] Code is available at https://github.com/cMancio00/GVAEBM.

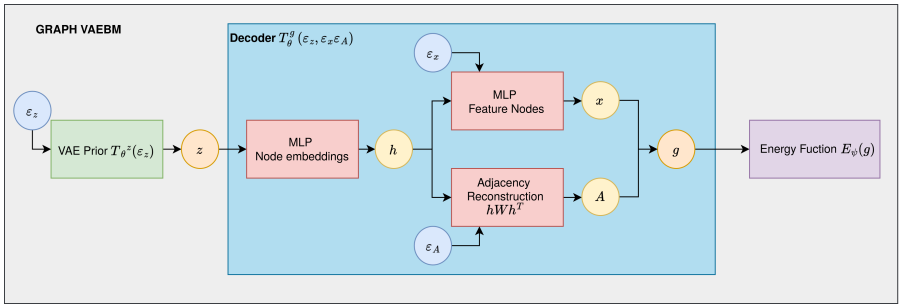


Fig. 1: Our GraphVAEBM is composed of a GraphVAE generator (including the prior and decoder) and swappable energy function that operates on samples g generated by the GraphVAE.

graph properties. Desired properties are enforced through energy-guided transformations on the learned manifold rather than by modifying the generator itself. To avoid sampling directly in the discrete graph domain, we perform Langevin dynamics in the continuous noise space underlying the GVAE decoder while evaluating energies on decoded graph samples. This enables property-guided graph manipulation while exploiting the smooth geometry of the learned graph manifold.

The proposed framework is modular: new properties can be incorporated by training additional lightweight energy models, and used at inference time. We evaluate GraphVAEBM on Stochastic Block Model graphs and show that energy-guided trajectories induce transitions between structural graph classes while remaining stable within the target energy basin. Graph generative modeling has been widely studied through latent-variable models such as VGAE [13] and GraphVAE [23], as well as more recent diffusion-based approaches [9,25,20,21]. While these methods achieve strong generative performance, graph properties are typically encoded within a single model. Energy-Based Models have also been applied to graph generation and property-guided sampling [15,16], often relying on Langevin dynamics [26,17] directly in graph space. Other recent methods refine the generation with an EBM [1,27], but they still operate in the input space. Our approach instead combines a frozen graph VAE with independent property-specific EBMs and performs Langevin dynamics in decoder-noise space. Unlike previous hybrid VAE-EBM frameworks [27], the proposed formulation naturally supports property modularity without retraining the graph generator. The proposed architecture is shown in Figure 1.

2 Method

2.1 Graph Variational Autoencoder

Let a graph be denoted by $g = (A, X)$, where $A \in \{0, 1\}^{N \times N}$ is the adjacency matrix and $X \in \mathbb{R}^{N \times d}$ represents node attributes. Following the variational

autoencoder framework, we introduce a latent variable $z \in \mathbb{R}^m$ that captures the global structure of the graph. The generative model factorizes as

$$p_\theta(g, z) = p(z)p_{\theta_X}(X|z)p_{\theta_A}(A|z), \tag{1}$$

where $\theta = \{\theta_X, \theta_A\}$ is the set of generative model's parameters. The latent prior $p(z)$ is defined as $\mathcal{N}(0, I)$, and the decoder independently reconstructs node attributes and graph connectivity from the latent graph representation z. In our implementation, we first use a Multi-Layer Perceptron (MLP) to map the graph embedding into a set of node embeddings $h_1, \ldots, h_V$, where V is the number of nodes to generate (see Figure 1). Then, we sample the node features and the adjacency matrix by using:

$$p_{\theta_X}(X|h) = \mathcal{N}\left(\mu_X(h), \Sigma_X(h)\right), \qquad p_{\theta_A}(A|h) = \mathcal{N}\left(f(h), \sigma_A^2 I\right), \tag{2}$$

where $\mu_X(\cdot)$, $\Sigma_X(\cdot)$ are neural networks with parameters θ_X, and $f(h)$ is a bilinear layer with parameters θ_A. The model is trained as usual by maximizing the evidence lower bound. The learned latent space provides a continuous representation of graph structure that will later be exploited to perform energy-guided sampling. The role of the GVAE is not to enforce specific graph properties, but rather to learn a smooth latent manifold capturing the global distribution of graphs.

2.2 Modeling Properties with Energy-Based Models

While the GVAE captures the global graph distribution, specific graph properties are modeled through a collection of independent EBMs. Let c denote a target graph property. For each property, we define an energy function

$$E_{\psi_c}(g) : \mathcal{G} \to \mathbb{R}, \tag{3}$$

parameterized by ψ_c, which assigns low energy to graphs exhibiting property c and higher energy to graphs that do not. Given a dataset $\mathcal{D}$, we partition the graphs according to the target property:

$$\mathcal{D}_c^+ = \{g \in \mathcal{D} \mid c(g) = 1\}, \qquad \mathcal{D}_c^- = \{g \in \mathcal{D} \mid c(g) = 0\}. \tag{4}$$

The energy model is trained using Contrastive Divergence (CD) [24,7,2]. During the positive phase, samples are drawn from $\mathcal{D}_c^+$. For the negative phase, Markov chains are initialized from graphs belonging to the complementary set $\mathcal{D}_c^-$ and refined through a finite number of Langevin dynamics steps. Formally, given an initial graph $g_0^- \sim \mathcal{D}_c^-$, a negative sample g_K^- is obtained after K Markov Chain Monte-Carlo (MCMC) iterations [18,19]. We detail how to perform sampling in Section 2.3. We train the EBM by optimizing the following loss:

$$\mathcal{L} = \underbrace{\mathbb{E}_{g^+}[E_{\psi_c}(g^+)] - \mathbb{E}_{g_K^-}[E_{\psi_c}(g_K^-)]}_{\text{contrastive term}} + \underbrace{\lambda\left(\mathbb{E}_{g^+}\left[E_{\psi_c}(g^+)^2\right] + \mathbb{E}_{g_K^-}\left[E_{\psi_c}(g_K^-)^2\right]\right)}_{\text{energy regularization}} \tag{5}$$

Initializing negative chains from complementary-property graphs encourages the sampler to explore transitions between different regions of the graph space and empirically improves mixing compared to random initialization. As a result, each EBM learns an energy landscape associated with a specific graph property while remaining independent from the underlying generative model.

2.3 Noise-Space Langevin Dynamics

Unlike previous graph EBMs that perform MCMC directly in graph space [15,16], we perform Langevin dynamics in the continuous noise variables underlying the decoder. Training and sampling directly in graph space is challenging due to the discrete nature of graphs and the poor mixing behavior of Markov chains based on local graph modifications [15,18]. To address this issue, we perform Langevin dynamics in the continuous noise space underlying the GVAE decoder while keeping the energy function defined over graphs. The generative model can be reparameterized through a collection of independent Gaussian noise variables

$$\boldsymbol{\varepsilon} = (\varepsilon_z, \varepsilon_x, \varepsilon_A), \qquad \boldsymbol{\varepsilon} \sim \mathcal{N}(0, I), \tag{6}$$

from which latent variables, node attributes and adjacency matrices are generated through deterministic transformations (see Figure 1):

$$z = T^z(\varepsilon_z), \qquad g = T_\theta^g\left(z, \varepsilon_x, \varepsilon_A\right), \tag{7}$$

where z is the sampled latent graph embedding, and $g = (A, X)$ is the sampled graph. In our case, T^z is the identity function since we assume that $z \sim \mathcal{N}(0, I)$, while T_θ^g applies the reparameterization trick [12] to sample from normal distributions in Equation (2). Combining both mappings yields a deterministic transformation:

$$g = T_\theta(\boldsymbol{\varepsilon}) = T_\theta^g\left(T^z(\varepsilon_z), \varepsilon_x, \varepsilon_A\right), \tag{8}$$

thus, the induced energy on the noise space is:

$$\tilde{E}_\psi(\boldsymbol{\varepsilon}) = E_\psi\big(T_\theta(\boldsymbol{\varepsilon})\big) = E_\psi(g). \tag{9}$$

The gradient of the induced energy can be computed through automatic differentiation by propagating gradients through the decoder. Langevin dynamics is then performed directly in the noise space according to:

$$\boldsymbol{\varepsilon}_{t+1} = \boldsymbol{\varepsilon}_t - \frac{\eta}{2}\nabla_{\boldsymbol{\varepsilon}}\tilde{E}_\psi(\boldsymbol{\varepsilon}_t) + \sqrt{\eta}\omega_t, \quad \omega_t \sim \mathcal{N}(0, I), \tag{10}$$

where η denotes the step size. The updated noise variables are subsequently decoded to obtain a new graph sample,

$$g_{t+1} = T_\theta(\boldsymbol{\varepsilon}_{t+1}). \tag{11}$$

To start the negative phase of the Contrastive Divergence in Eq. 5, we can encode the complementary-property graphs to obtain ε_z and then sample ε_x and

ε_A from $\mathcal{N}(0, I)$, obtaining the initial noise vector ε to use in Eq. 8. Operating in noise space provides two key advantages. First, the Markov chain evolves in a continuous Euclidean space, avoiding the difficulties associated with discrete graph modifications. Second, all variables share a common Gaussian prior and therefore a comparable scale, allowing a single Langevin dynamics scheme to jointly refine latent variables, node attributes and graph connectivity.

3 Experiment

3.1 Experimental Setup

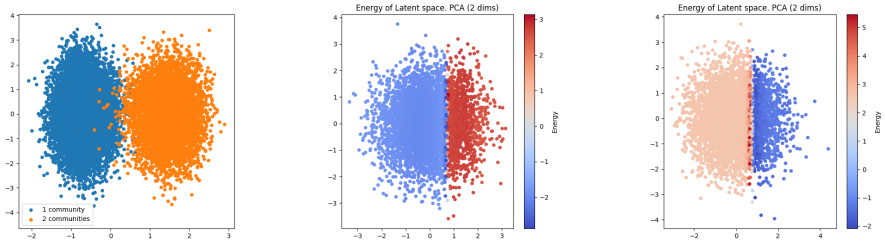
We evaluate GraphVAEBM on synthetic graphs generated through Stochastic Block Models (SBMs). Using this controlled setting, we assess whether the proposed framework can successfully separate global graph generation from property modeling. The dataset consists of $40K$ graphs with at most 20 nodes and two structural classes corresponding to one-community and two-community structures which are the property of interest. Graphs are generated with an intra-community connection probability of $p_{in} = 0.9$ and an extra-community probability of $p_{extra} = 0.01$, producing clearly distinguishable community structures. The single node feature of X is a scalar $x_f \sim \mathcal{N}(0, 1)$ for the first community and $x_f \sim \mathcal{N}(5, 1)$ for the second community. In this setting, we consider two graph distributions:

- EBM-1 models the distribution of graphs with only one community;
- EBM-2 models the distribution of graphs with two communities.

We evaluate each EBM by its ability to transform graphs with complementary properties into graphs with the desired property. To this end, we initialize the sampling chain from graphs drawn from the complementary community and guide it toward the target distribution via Langevin dynamics in the decoder noise space. The objective of these experiments is not to benchmark graph generation performance, but rather to verify that the proposed modular framework can learn property-specific energy landscapes and induce controlled transitions between graph classes. Further information of model architecture is given in Appendix A.

3.2 Latent Manifold and Energy Landscape

To analyze the learned graph manifold, we train the GVAE with a two-dimensional latent space and visualize the latent representations associated with different community structures. Figure 2 shows the latent embedding together with the energy landscape induced by the corresponding property-specific EBM. The learned latent space organizes graphs according to their structural properties, while the energy function assigns low-energy regions to graphs exhibiting the target property. This suggests that the GVAE captures a meaningful global graph manifold and that the EBM successfully identifies regions associated with the desired structural configuration. The resulting landscape provides an interpretable



(a) Latent space learned by GVAE.

(b) Energy landscape learned by EBM-1.

(c) Energy landscape learned by EBM-2.

Fig. 2: Visualization of the learned graph manifold and the energy landscape induced by the EBM. The energy function identifies regions corresponding to the target graph property. GVAE constructs a latent space with one-community graphs on the left and two-community graphs on the right.

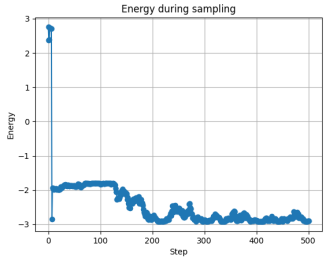
Table 1: Success rate of property-guided graph manipulation. A manipulation is considered successful when the final graph is assigned to the target community structure by the Louvain community detection algorithm.

	Manipulation Success	Intra-density	Inter-density
EBM-1	78.00	0.938 ± 0.094	-
EBM-2	69.00	0.950 ± 0.055	0.016 ± 0.040

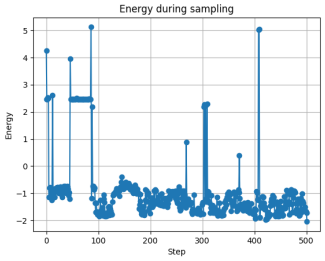
view of the optimization process performed during Langevin sampling, where graph transformations correspond to trajectories moving toward low-energy regions associated with the target property.

3.3 Energy-Guided Graph Transitions

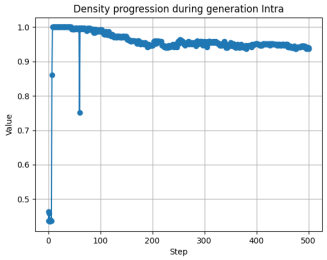
To evaluate the ability of GraphVAEBM to manipulate graph properties, we initialize Langevin chains from graphs belonging to the complementary property class and iteratively apply energy-guided updates in the decoder noise space. A manipulation is considered successful if the final graph is assigned to the target community structure by the Louvain community detection algorithm. As reported in Table 1, GraphVAEBM achieves a success rate of 78.0% for EBM-1 and 69.0% for EBM-2. These results indicate that the proposed framework is capable of inducing controlled structural changes under a fixed pretrained GVAE decoder. Importantly, the GVAE contains 914 parameters, while the property-specific EBMs contain 177 parameters, highlighting that the manipulation capability is achieved with a lightweight energy model built on top of a fixed generative backbone. We further characterize the generated graphs using intra and inter-community densities. Across runs, graphs classified as the



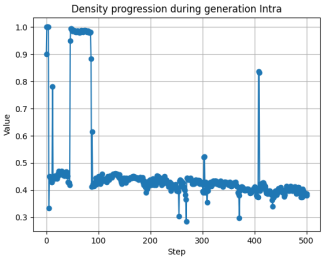
(a) Transition toward the one-community regime.



(b) Transition toward the two-community regime.



(c) Density transition toward the one-community regime.



(d) Density transition toward the two-community regime.

Fig. 3: Energy trajectories during property-guided Langevin sampling and the relative density change.

target structure exhibit high intra-community density (approximately 0.95) and low inter-community density (approximately 0.016 in EBM-2), which is consistent with the ground-truth SBM parameters (0.9 and 0.01, respectively). Figure 3 reports the evolution of the energy during sampling and how the intra-class density, calculated on the decoded graph, changes. Starting from a graph that does not exhibit the target property, the chain progressively decreases the energy until reaching a low-energy region associated with the desired structural configuration. Minor oscillations are expected due to the stochastic Langevin updates and do not affect convergence toward the target energy basin. The resulting trajectory reveals a transformation from the initial community configuration toward the target one. These observations suggest that the learned energy landscape induces meaningful transitions between graph property classes rather than isolated local modifications. Importantly, the transition is obtained while performing Langevin dynamics exclusively in the continuous noise space. The energy model never directly modifies graph structures; instead, it guides the latent noise variables whose decoded graphs progressively move toward lower-energy configurations. Figure 4 in Appendix A illustrates representative graph samples extracted from different stages of the same chain.

4 Conclusion

We introduced GraphVAEBM, a hybrid graph generative framework that separates global graph modeling from graph property modeling. A frozen GVAE is used to learn a continuous graph manifold, while lightweight EBMs independently capture desired graph properties. By performing Langevin dynamics in the decoder noise space, the proposed approach enables property-guided graph transformations without directly exploring the discrete graph domain. Experiments on synthetic data demonstrate that the learned energy functions induce stable transitions between structural graph classes and define meaningful energy landscapes associated with different graph properties. The resulting trajectories reveal the ability of the model to cross energy barriers and remain within target structural regimes. New graph properties can be incorporated through independently trained energy models without retraining the underlying generator. We believe this separation between global graph structure and property-specific modeling provides a promising direction for controllable graph generation and graph manipulation. Future work will investigate larger graph domains, molecular applications, and the integration of pretrained graph generators with modular EBMs for compositional property control. The exploration of more efficient ways to move in the energy landscapes such as annealing, which smooths the energy landscape and facilitates exploration across regions, will also be taken into consideration. We will also verify the possibility of optimizing multiple graph properties through energy composition as shown in [8,4].

References

1. Balcerak, M., Amiranashvili, T., Terpin, A., Shit, S., Bogensperger, L., Kaltenbach, S., Koumoutsakos, P., Menze, B.: Energy matching: Unifying flow matching and energy-based models for generative modeling. In: Advances in Neural Information Processing Systems (NeurIPS) (2025)
2. Carreira-Perpiñán, M.A., Hinton, G.: On contrastive divergence learning. In: Cowell, R.G., Ghahramani, Z. (eds.) Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics. Proceedings of Machine Learning Research, vol. R5, pp. 33–40. PMLR (06–08 Jan 2005)
3. Clevert, D.A., Unterthiner, T., Hochreiter, S.: Fast and accurate deep network learning by exponential linear units (elus). In: International Conference on Learning Representations (2015)
4. Du, Y., Li, S., Mordatch, I.: Compositional visual generation with energy based models. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. NIPS '20, Curran Associates Inc., Red Hook, NY, USA (2020)
5. Du, Y., Mordatch, I.: Implicit generation and modeling with energy based models. In: Advances in Neural Information Processing Systems (NeurIPS) (2019)
6. Grathwohl, W., Wang, K.C., Jacobsen, J.H., Duvenaud, D., Norouzi, M., Swersky, K.: Your classifier is secretly an energy based model and you should treat it like one. In: International Conference on Learning Representations (2020)
7. Hinton, G.E.: Training products of experts by minimizing contrastive divergence. Neural Computation **14**(8) (08 2002)

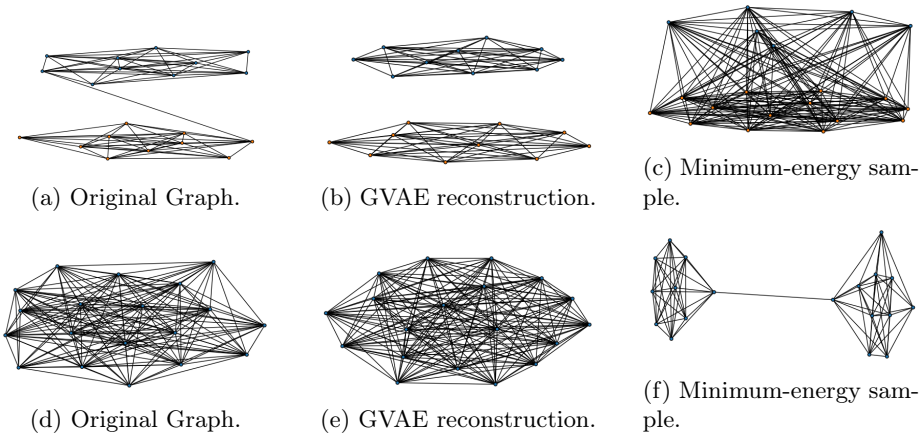
8. Hinton, G.E.: Training products of experts by minimizing contrastive divergence. Neural Comput. **14**(8), 1771–1800 (2002)
9. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. In: Advances in Neural Information Processing Systems (NeurIPS) (2020)
10. Jin, W., Barzilay, R., Jaakkola, T.: Junction tree variational autoencoder for molecular graph generation. In: Proceedings of the 35th International Conference on Machine Learning (ICML) (2018)
11. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. International Conference on Learning Representations (12 2014)
12. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. In: International Conference on Learning Representations (ICLR) (2014)
13. Kipf, T.N., Welling, M.: Variational graph auto-encoders (2016)
14. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations (2017)
15. Liu, M., Yan, K., Oztekin, B., Ji, S.: GraphEBM: Molecular graph generation with energy-based models. In: Energy Based Models Workshop - ICLR (2021)
16. Miglior, L., Simone, L., Podda, M., Bacciu, D.: Towards efficient molecular property optimization with graph energy based models. In: ESANN 2025 proceesdings. ESANN 2025 (2025)
17. Neal, R.M.: Mcmc using hamiltonian dynamics. Handbook of Markov Chain Monte Carlo (2011)
18. Nijkamp, E., Hill, M., Han, T., Zhu, S.C., Wu, Y.N.: On the anatomy of mcmc-based maximum likelihood learning of energy-based models. In: Conference on Artificial Intelligence (AAAI) (2020)
19. Nijkamp, E., Hill, M., Zhu, S.C., Wu, Y.N.: Learning non-convergent non-persistent short-run mcmc toward energy-based model. In: Advances in Neural Information Processing Systems (NeurIPS) (2019)
20. Ninniri, M., Podda, M., Bacciu, D.: Classifier-free graph diffusion for molecular property targeting. In: Bifet, A., Davis, J., Krilavičius, T., Kull, M., Ntoutsi, E., Žliobaitė, I. (eds.) Machine Learning and Knowledge Discovery in Databases. Research Track. pp. 318–335. Springer Nature Switzerland, Cham (2024)
21. Ninniri, M., Podda, M., Bacciu, D.: Graph Diffusion That Can Insert and Delete. In: Advances in Neural Information Processing Systems (NeurIPS) (2025)
22. Shi, C., Xu*, M., Zhu, Z., Zhang, W., Zhang, M., Tang, J.: Graphaf: a flow-based autoregressive model for molecular graph generation. In: International Conference on Learning Representations (2020)
23. Simonovsky, M., Komodakis, N.: Graphvae: Towards generation of small graphs using variational autoencoders. In: Artificial Neural Networks and Machine Learning – ICANN 2018. Cham (2018)
24. Song, Y., Kingma, D.P.: How to train your energy-based models (2021)
25. Vignac, C., Krawczuk, I., Siraudin, A., Wang, B., Cevher, V., Frossard, P.: Digress: Discrete denoising diffusion for graph generation. In: International Conference on Learning Representations (ICLR) (2023)
26. Welling, M., Teh, Y.W.: Bayesian learning via stochastic gradient langevin dynamics. ICML (2011)
27. Xiao, Z., Kreis, K., Kautz, J., Vahdat, A.: Vaebm: A symbiosis between variational autoencoders and energy-based models. In: International Conference on Learning Representations (ICLR) (2021)
28. Zang, C., 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. p. 617–626. KDD '20, ACM (Aug 2020)

Table 2: Model architecture and training hyperparameters.

Module	**Parameter**	**Value**
	Graph VAE Architecture	
Encoder	Layers	DenseGCN(1,16) $\rightarrow$ DenseGCN(16,2)
Encoder	Pooling	Mean pooling
Encoder	Activation	LeakyReLU
Latent Space	Dimension	2
Decoder ($z \rightarrow h$)	MLP	Linear(2,16) $\rightarrow$ Linear(16, num-nodes$\cdot$ hidden-dim)
Decoder ($h \rightarrow X$)	MLP	Linear(2,16) $\rightarrow$ Linear(16, node-features $\cdot$ 2)
Adjacency Decoder	Structure	Bilinear $hWh^\top$
	Energy-Based Model	
EBM Encoder	Layers	DenseGCN(1,16) $\rightarrow$ DenseGCN(16,8)
EBM Energy Head	Layer	Linear(8,1)
EBM Activation	Function	ELU [3]
	Training Hyperparameters	
Optimizer	Learning rate (GVAE)	10^{-3}
Optimizer	Learning rate (EBM)	10^{-3}
GVAE	β	2
Training	Batch size	128
Training	Epochs	150
Contrastive Divergence	Penalty weight	0.5
Contrastive Divergence	Persistent chain	No
Langevin Dynamics	Steps	5
Langevin Dynamics	Step size	10^{-3}

A Plots and Model Architecture

This appendix provides additional implementation details to facilitate reproducibility of the proposed GraphVAEBM framework. Table 2 summarizes the architectures of both the Graph Variational Autoencoder and the Energy-Based Model, together with the training hyperparameters used throughout all experiments. The GVAE encoder consists of two graph convolutional layers [14] followed by mean pooling to obtain a graph-level latent representation. The latent space dimension was fixed to 2 in order to facilitate visualization and analysis of the learned energy landscape. The decoder reconstructs both node features and graph connectivity from the latent representation. The Energy-Based Model shares a similar graph-convolutional architecture and outputs a scalar energy score. During sampling, Langevin dynamics is performed in latent space, while the decoder remains fixed. This allows the EBM to guide generated samples toward regions associated with the desired graph property. Both models use ADAM [11] with default parameters. Figure 4 illustrates representative graph transitions generated by GraphVAEBM. In both cases, the GVAE reconstruction remains close to the original graph structure, while the energy-guided refinement step moves the sample toward a configuration exhibiting the target property. In



(a) Original Graph. (b) GVAE reconstruction. (c) Minimum-energy sample.

(d) Original Graph. (e) GVAE reconstruction. (f) Minimum-energy sample.

Fig. 4: Transformation induced by GraphVAEBM. The frozen GVAE reconstruction preserves the original structural configuration, while energy-guided Langevin dynamics moves the sample toward a graph exhibiting the target property. Top row shows transition from two communities to one community. Bottom row shows transition from one community to two communities.

the top row, the model transforms a graph containing two communities into a graph characterized by a single community structure. Conversely, the bottom row shows the reverse transition. These are the corresponding graphs whose energy and edge density was described in Figure 3. These examples qualitatively illustrate that the EBM modifies the latent representation in a targeted manner, preserving overall graph coherence while inducing the desired structural change.