Decoding Cortical Microcircuits: A Generative Model for Latent Space Exploration and Controlled Synthesis

Xingyu Liu[1], **Yubin Li**[1,2], **Guozhang Chen**[1]
[1]National Key Laboratory for Multimedia Information Processing,
School of Computer Science, Peking University, Beijing, China
[2]School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan, China
Corresponding author: guozhang.chen@pku.edu.cn

Abstract

A central idea in understanding brains and building brain neural networks is that structure determines function. However, the brain's connectome is a massively high-dimensional graph, making the direct investigation of its structure-function relationships computationally intractable. Therefore, identifying a compact, low-dimensional representation that captures the connectome's essential structural organization is crucial for elucidating these relationships. We introduce a generative model to learn this underlying representation from detailed connectivity maps of mouse cortical microcircuits. Our model successfully captures the essential structural information of these circuits within a compressed latent space. We then associate specific network structures, as encoded in this latent space, with computational functions using reservoir computing tasks. Building on this, our methodology allows for the controllable generation of novel, synthetic microcircuits with desired structural features by navigating the learned latent space. This research paradigm establishes a computational testbed to systematically investigate the brain's inherent structure-function relationships. The ability to generate diverse, bio-plausible circuits could inform the development of more brain-like artificial neural networks.

1 Introduction

The relationship between structure and function is a core idea in both neuroscience and the development of artificial intelligence (Vázquez-Rodríguez et al. 2019; Clark 2023; White et al. 2023). The brain, with its extraordinary capabilities, inspires us to build better AI systems (Zador et al. 2023; Osegi 2023; Hassabis et al. 2017; Lecun, Bengio, and Hinton 2015). The maps of brain connections are called *connectomes* (Sporns 2013; Sporns, Tononi, and Kötter 2005; Sporns 2011), whose structure is incredibly complex (Craddock, Tungaraza, and Milham 2015; Collin and Whitfield-Gabrieli 2023). Studying connectomes is important because they hold clues about how the brain processes information and learns (Elam et al. 2021; Bargmann and Newsome 2014). However, given that the brain's connectome is a massively high-dimensional graph, it is impractical to directly investigate which structures within the connectome itself yield specific functional performance. Therefore, the objective of this paper is to find a low-dimensional representation that captures the key features of the connectome. This new paradigm leverages this low-dimensional representation as a novel tool to investigate structure-function relationships.

In this work, we focus on *cortical microcircuits*, which are small, repeating patterns of cortical connections that can be thought of as fundamental building blocks and basic computational units of the brain (Douglas and Martin 2004; Miller 2016). We primarily implement this new paradigm for investigating structure-function relationships using data from the MICrONS program (Consortium et al. 2023), which has produced a large and detailed connectome dataset from the visual cortex of a mouse. This dataset provides an unprecedented opportunity to study the organization of these microcircuits. As a supplementary study, we also conduct similar investigations on the FlyWire connectome dataset from the fruit fly (See Appendix L). The source code for this paper is available at https://github.com/Criticality-Cognitive-Computation-Lab/Connectome-Analysis-VAE.

Our primary contributions are as follows:

1. We propose a new paradigm for investigating the structure-function relationships of the brain connectome, which we implement on connectome data from both mice and fruit flies. While the core paradigm remains consistent, the specific datasets and model architectures differ between species. Due to space constraints, this paper primarily details the model for the mouse connectome, with results from the fruit fly experiments presented in the Appendix L.
2. We introduce a Variational Autoencoder (VAE)-based generative model specifically designed to learn a compressed latent representation of microcircuit topology from this mouse visual cortex data.
3. Crucially, we demonstrate that specific aspects of this learned latent space show strong, understandable relationships with key structural properties of the microcircuits, such as how densely connected they are or how they form clusters. This means we can find meaningful ways to describe the core variations in circuit structure.
4. Building on this, we propose a method for the controlled generation of microcircuits. By carefully adjusting these meaningful aspects in the latent space, our method can create new, artificial network structures that have specific desired characteristics.

5. We investigated the influence of connectivity patterns on network function across various tasks. Utilizing controllably generated networks, we constructed reservoir networks and demonstrated that networks generated by the VAE to emulate brain-like connectivity patterns exhibited enhanced task performance compared to randomly connected networks of the same density. Furthermore, we found that alterations in specific structural features had an approximately monotonic effect on the performance of certain tasks.

To our knowledge, the application of VAE for controllable generation of high-resolution structural microcircuits and for the understanding of structural-function relationship has not been done before. Furthermore, understanding the brain's structure could eventually help in designing more efficient and capable artificial neural networks.

2 Problem Definition

Our overarching goal is to develop a computational framework that learns a low-dimensional generative representation of brain microcircuit topology. This representation is intended to serve as a new paradigm for investigating structure-function relationships, providing a basis for controllably generating novel microcircuits with desirable properties. We define two primary objectives:

2.1 Learning Compact Generative Representations of Microcircuit Topology

Given a dataset of N biological microcircuit graphs, denoted as $\mathbf{G}_{\text{data}} = \{\mathcal{G}^{(1)}, \mathcal{G}^{(2)}, \ldots, \mathcal{G}^{(N)}\}$, the first objective is to learn a low-dimensional latent representation $\mathbf{z}^{(i)} \in \mathbb{R}^{d_z}$ for each microcircuit $\mathcal{G}^{(i)}$. Each microcircuit is represented as a graph $\mathcal{G}^{(i)} = (\mathbf{X}^{(i)}, \mathbf{A}^{(i)})$, where $\mathbf{X}^{(i)} \in \mathbb{R}^{n_i \times d_v}$ is a matrix of node features for its n_i neurons (with d_v feature dimensionality), and $\mathbf{A}^{(i)} \in \{0,1\}^{n_i \times n_i}$ is its adjacency matrix indicating synaptic connections. In this work, since the number of nodes varies across the graph data, we pad each adjacency matrix to a fixed size of 100×100. For consistent processing, a canonical node ordering π is assumed for inputs to certain model components, thus we may refer to an ordered graph as $\mathcal{G}_\pi^{(i)}$.

This learned latent representation $\mathbf{z}^{(i)}$ must be *generative*. That is, we aim to learn a probabilistic model $p_\theta(\mathcal{G}_\pi|\mathbf{z})$ capable of generating realistic microcircuit graphs from these latent codes. The quality of this representation will be assessed by its ability to:

1. Faithfully reconstruct observed graph structures and their topological properties from their latent codes.
2. Generate novel, diverse graphs that capture the statistical characteristics of the biological training data.

2.2 Controllable Generation of Microcircuits with Target Properties

Building upon the learned latent space $\mathcal{Z}$ (the space of all $\mathbf{z}$) and the generative model $p_\theta(\mathcal{G}_\pi|\mathbf{z})$ from Section 2.1, our second objective is to enable the *controlled generation* of novel microcircuits.

Specifically, given a target structural, dynamical, or functional property P (e.g., a specific mean degree, clustering coefficient, or level of assortativity), and a desired value t_{target} for this property, the goal is to synthesize new connectome graphs $\mathcal{G}_{\pi,\text{new}}$. These generated graphs should:

1. Optimally satisfy the specified constraint, meaning the property P for $\mathcal{G}_{\pi,\text{new}}$ should be close to t_{target}. We denote this constraint as $\mathcal{T}$.
2. Preserve general topological and dynamical characteristics inherent to biological neural microcircuits, ensuring they remain plausible.

Formally, this requires effectively sampling from, or being guided by, a conditional probability distribution $p(\mathbf{z}|\mathcal{T})$ in the latent space. Latent vectors $\mathbf{z}_{\text{new}}$ drawn from this distribution are then decoded using $p_\theta(\mathcal{G}_\pi|\mathbf{z}_{\text{new}})$ to produce the desired microcircuits.

3 Related Work

Analyzing the structural organization of brain connectomes is fundamental to understanding the intricate relationship between brain structure and function (Xia, Wang, and He 2013; Chen et al. 2018). A number of methods have been used to address this link, including statistical models (Mišić et al. 2016), communication models (Goñi et al. 2014), and biophysical models (Honey et al. 2007; Breakspear 2017; Chen, Scherr, and Maass 2022; Chen and Gong 2019, 2022). By modeling brain networks as graphs, researchers employ graph-theoretic approaches to reveal key topological properties that underlie efficient communication and cognitive processes (Kan et al. 2022; Said et al. 2023). Understanding how these structural features relate to functional dynamics is a central goal in connectomics.

To investigate the principles governing connectome formation and organization, generative models have been developed. Early approaches typically relied on a small set of predefined wiring rules or biological principles, such as cost-efficiency or growth mechanisms, to replicate observed network features in silico (Betzel et al. 2016; Kaiser and Hilgetag 2004; Henriksen, Pang, and Wronkiewicz 2016). While successful in capturing certain global properties, these rule-based methods often lack flexibility and depend on manually specified or constrained generative factors, limiting their ability to explore the full complexity of biological variability.

More recently, deep generative models, such as Variational Autoencoders (VAEs), have emerged as powerful tools for analyzing and synthesizing complex data like brain networks (Yu et al. 2022; Zuo et al. 2023). These models are particularly well-suited for learning a low-dimensional latent representation of the network data, effectively performing dimensionality reduction. Furthermore, by manipulating or sampling from this learned latent space, these models enable the controlled synthesis of novel, biologically plausible network configurations, offering new avenues for exploring connectome variability and its functional implications (Tan et al. 2022; Dong et al. 2023). Prior works have also applied graph VAEs to macro-scale functional connectomes for dis-

criminative tasks or developmental modeling (Behrouzi and Hatzinakos 2022).

4 Method

4.1 MICrONS Dataset and Preprocess

We adopt the IARPA MICrONS dataset (Consortium et al. 2023), which encompasses a 1.4 × 0.87 × 0.84 mm volume of cortex from a P87 mouse, containing neurons within the area and their interconnections, which is the sole source for such high-resolution connectivity in cortex. Given the cortex's division into six distinct layers, each associated with a specific level of information processing (Jones 2000; Felleman and Van Essen 1991; Reid and Alonso 1996), our study concentrates on cortical microcircuits organized as vertically oriented functional columns that traverse all laminar layers (Callaway 1998; Douglas and Martin 2004). To model these microcircuits, we extracted cylindrical subvolumes oriented perpendicularly to the cortical layers. Each cylindrical unit comprises 80–100 neurons (both excitatory and inhibitory neurons), preserving intra-column connectivity while excluding connections extending beyond the columnar boundary. Our VAE was trained on 3,285 circuits from MICrONS dataset. The test set was from a spatially disjoint region to prevent data leakage.

Focusing only on network topology, we treat each circuit as a binary directed graph. We establish a canonical ordering rule, π, sorting neurons by their y-coordinates (reflecting cortical depth). This depth-based ordering is a deliberate choice, as it is both biologically significant (the y-coordinate inherently encodes the cortical layer) and technically crucial. Without this consistent ordering, determining the correct correspondence for element-wise comparison between original and generated adjacency matrices becomes intractable. Neuron type and synapse weight information are temporarily disregarded, as this work focuses solely on the topological structure.

We applied a consistent methodology to the fruit fly connectome. While the overall paradigm of using a generative model to learn a low-dimensional topological representation remains the same, the specific VAE architecture was adapted to accommodate the different structural properties of the fly brain.

4.2 Connectome Graph Variational Autoencoder

The goal is to find latent representations similar to information bottleneck. Adopting a variational autoencoder (VAE) (Kingma and Welling 2022) on the connectome data can be formulated as follows: Assume a set of training connectome graphs $\mathbf{G} = \{\mathcal{G}_\pi^{(i)}\}$ is generated from the distribution of a set of unobserved latent representation $\mathbf{z} = (z_1, ..., z_m)$, where $z^{(i)} \sim p_{\theta^*}(z)$ and training data is sampled from true conditional $p_{\theta^*}(\mathcal{G}_\pi|z^{(i)})$. The data generation process is denoted by $p_\theta(\mathcal{G}_\pi|\mathbf{z})$. Following the standard VAE setting (Kingma and Welling 2022), we approximate the intractable posterior by $q_\phi(\mathbf{z}|\mathcal{G}_\pi) \approx p_\theta(\mathbf{z}|\mathcal{G}_\pi)$ and maximize the evidence lower bound on the marginal likelihood of graph $\mathcal{G}^{(i)}$:

$$\log p_\theta(\mathcal{G}_\pi^{(i)}) \geq \mathcal{L}(\phi, \theta; \mathcal{G}_\pi^{(i)})$$
$$= \mathbb{E}_{q_\phi(\mathbf{z}|\mathcal{G}_\pi^{(i)})}\left[\log p_\theta(\mathcal{G}_\pi^{(i)}|\mathbf{z})\right] - \beta \mathrm{KL}\left[q_\phi(\mathbf{z}|\mathcal{G}_\pi^{(i)})||p_\theta(\mathbf{z})\right],$$

where β is a hyperparameter to balance the reconstruction loss and KL-divergence loss during the training process (Higgins et al. 2016).

The proposed VAE model for connectome graphs comprises four main components. First, a node feature encoder employs a multi-layer, multi-head Graph Attention Network (GAT) (Veličković et al. 2018; Vaswani et al. 2023) to compute an embedding vector for each node. Second, a graph global encoder, which is a transformer encoder augmented with a special token (Devlin et al. 2019; Winter, Noé, and Clevert 2021; Vaswani et al. 2023), extracts an embedding for the entire sequence of node embeddings. This extracted sequence embedding serves as the graph-level embedding, upon which a 32-dimensional mean and variance are computed. The reparameterization of a standard VAE is applied to this 32D latent embedding representing the whole graph. Third, a node feature decoder, which is a transformer decoder, takes the graph-level embedding as input to reconstruct node features necessary for subsequent edge prediction. Finally, the edge predictor utilizes these decoded node features to predict the edges of the graph.

The overall architecture of the VAE model is illustrated in Figure 1, with the following main components:

Node Feature Encoder The node feature encoder, utilizing a three-layer multi-head GAT network, transforms 100-dimensional one-hot node representations into 32-dimensional node embeddings. See Appendix B.2 for GAT architecture details.

Graph Global Encoder Treating the y-ordered nodes as a sequence (analogous to words in a sentence), the graph global encoder transforms node embeddings into a fixed-size latent representation. Following (Devlin et al. 2019; Winter, Noé, and Clevert 2021), a dummy node v_0 is prepended to the sequence to serve as a global embedding. The encoder applies rotational positional encoding (RoPE) (Su et al. 2023) and several transformer encoder layers to this augmented sequence. The final embedding of v_0 is taken as the global graph representation. An MLP is then applied to compute the 32-dimensional mean and variance for sampling the 32-dimensional latent vector $\mathbf{z}$, similar to standard VAEs.

Node Feature Decoder The Node Feature Decoder reconstructs individual node embeddings from the global graph embedding using several transformer decoder layers. The input is the global graph embedding, augmented with rotational positional encoding (RoPE) (Su et al. 2023). The global graph embedding also serves as memory for the decoder's cross-attention, leveraging this global context during node feature reconstruction.

Edge Predictor The edge predictor is a cross-node interaction layer. It takes the node feature decoder output $h \in \mathbb{R}^{n \times d}$, where $n = 100$ is the maximum number of nodes and d is the embedding dimension. Edges are predicted using the

dot product of embeddings transformed by two distinct linear layers with activation.

$$\mathbf{A}_{\text{pred}} = \sigma(\text{LeakyReLU}(\mathbf{h}\mathbf{W}_1)(\text{LeakyReLU}(\mathbf{W}_2\mathbf{h})^\top)), \tag{1}$$

The output $\mathbf{A}_{\text{pred}}$ provides a probabilistic adjacency matrix where each entry, a floating-point number between 0 and 1, denotes the likelihood of an edge. We then perform a Bernoulli sampling process on each entry using this probability to generate a binary adjacency matrix.

The detailed encoder structure is depicted in Appendix Figure 8, and the detailed decoder structure is depicted in Appendix Figure 9.

4.3 Controllable Connectome Generation by Sampling from the Latent Space

Investigating the intricate interplay between structure, dynamics, and function in brain neural microcircuits necessitates the capability to generate novel networks in a controlled fashion. Specifically, given a target network property $\mathcal{T}$ (which can be structural, dynamical, or functional), our objective is to synthesize novel connectomes $\mathcal{G}_\pi$ that optimally satisfy the specified constraint $\mathcal{T}$ while preserving general topological and dynamical characteristics inherent to biological neural microcircuits. Formally, the goal is to generate new samples from the conditional probability distribution $p(\mathcal{G}_\pi|\mathcal{T})$.

Given the computational cost associated with large-scale graph generation via brute-force sampling, we propose leveraging latent space properties to inform our sampling strategy. By employing geometric characteristics of the latent manifold as a sampling heuristic, this approach significantly reduces computational overhead compared to the brute-force generate-then-filter paradigm. Specifically, we aim to find the probability distribution of the latent vector $\mathbf{z}$ conditioned on the target property $\mathcal{T}$, denoted as $p(\mathbf{z}|\mathcal{T})$, which can be represented as an energy model:

$$p(\mathbf{z}|\mathcal{T}) = \frac{1}{Z} p(\mathbf{z})^{1/\tau} \exp(\lambda S(\mathcal{T}, \mathbf{z})), \tag{2}$$

where $S(\mathcal{T}, \mathbf{z})$ is a condition indicator function, defined as $S(\mathcal{T}, \mathbf{z}) = 1$ if $\mathbf{z} \in \Omega_\mathcal{T}$ else 0. $\Omega_\mathcal{T}$ is a subset of latent space where the generated graphs are predicted to be close to the target. $Z = \int p(\mathbf{z})^{1/\tau} \exp(\lambda S(\mathcal{T}, \mathbf{z}))d\mathbf{z}$ is the normalization factor, and λ is a tuning parameter which determine the constraining strength of $\mathcal{T}$ and τ is a temperature parameter which balance exploitation and exploration based on prior latent distribution of $p(\mathbf{z})$.

When $\lambda \to \infty$, the energy model degenerates into a strictly constrained version (by $\mathcal{T}$). In this case we have:

$$p(\mathbf{z}|\mathcal{T}) = \frac{p(\mathbf{z})^{1/\tau}}{p(\Omega_\mathcal{T})} \cdot \mathbb{I}(\mathbf{z} \in \Omega_\mathcal{T}), \tag{3}$$

where the new normalization factor becomes $p(\Omega_\mathcal{T}) = \int_{\Omega_\mathcal{T}} p(\mathbf{z})^{1/\tau} d\mathbf{z}$.

4.4 Reservoir Network with Structured Connectivity

We define reservoir network (Jaeger 2001) with a D_r dimensional reservoir $\mathbf{R}$, input projection parameterized by $W_{in} \in \mathbb{R}^{D_r \times D}$ and output projection parameterized by $W_{out} \in \mathbb{R}^{D \times D_r}$. Reservoir $\mathbf{R}$ stores a reservoir state $\mathbf{r}(t)$ with dimension D_r, and the recurrent connection is represented by an adjacency matrix $\mathbf{A} \in \mathbb{R}^{D_r \times D_r}$, whose elements are 0, 1 or -1, corresponding to non-connection, excitatory connection and inhibitory connection. We scale $\mathbf{A}$ to satisfy the pre-specified spectral radius sr. Given input vector $\mathbf{u}(t)$, the reservoir state is updated by $\mathbf{r}(t+1) = (1-\alpha)\mathbf{r}(t) + \alpha \cdot \tanh(\mathbf{A} \cdot \mathbf{r}(t) + W_{in} \cdot \mathbf{u}(t))$, where α is the leaking rate. $\mathbf{r}(t+1)$ is then flowed to the output coupler, where a mapping is done to obtain $\mathbf{v}(t+1) = W_{out} \cdot \mathbf{r}(t+1)$. During the training process, we kept the recurrent connection weights of the reservoir fixed, training only the input and readout layers. This approach allows us to study the computational functions that arise from the connection structure itself.

5 Experiments

5.1 Evaluation Metrics

We selected several graph theory metrics (descriptors) commonly employed in the analysis of connectomes (Rubinov and Sporns 2010), including mean degree, efficiency, transitivity, clustering coefficient, modularity, and assortativity. The detailed definitions of these metrics are provided in the Appendix C.

5.2 Microcircuit Reconstruction and Generation Results

Examples of original and reconstructed graphs obtained through the VAE are provided in the Appendix D. To validate the model's generative capabilities, we evaluated several graph metrics on the generated samples. We employed the maximum mean discrepancy (MMD) (Gretton et al. 2012) to compare the distributions of these graph statistics between an equal number of generated and test graphs. Following the standard protocol established by GraphRNN (You et al. 2018), we specifically measured the distributions of degree, clustering coefficient and spectrum. For MMD computation, we utilized both the Gaussian Earth Mover's Distance (EMD) kernel and the total variation (TV) distance (Liao et al. 2020). Given the absence of existing generative models specifically designed for connectome graphs, we selected three alternative models commonly used for molecule generation or general graph generation (Jo, Lee, and Hwang 2022; Xu et al. 2024; Chen et al. 2023) as baselines. The following Table 1 reports the MMD values for different graph metrics evaluated across these models and our proposed approach. The MMD validation of various structural metrics demonstrates that the graph generated by the VAE is consistent with that of the real brain connectome. To further check the diversity and novelty of the generated results, we verified that 2,000 generated samples are all unique among themselves and not present in the training data.

Figure 2 compares graphs generated by our proposed model and other models with the original data (column Ori.).

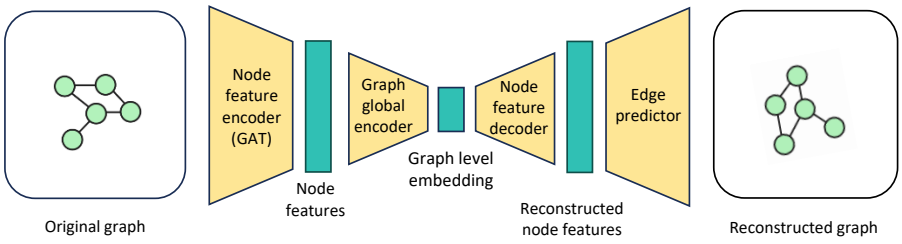
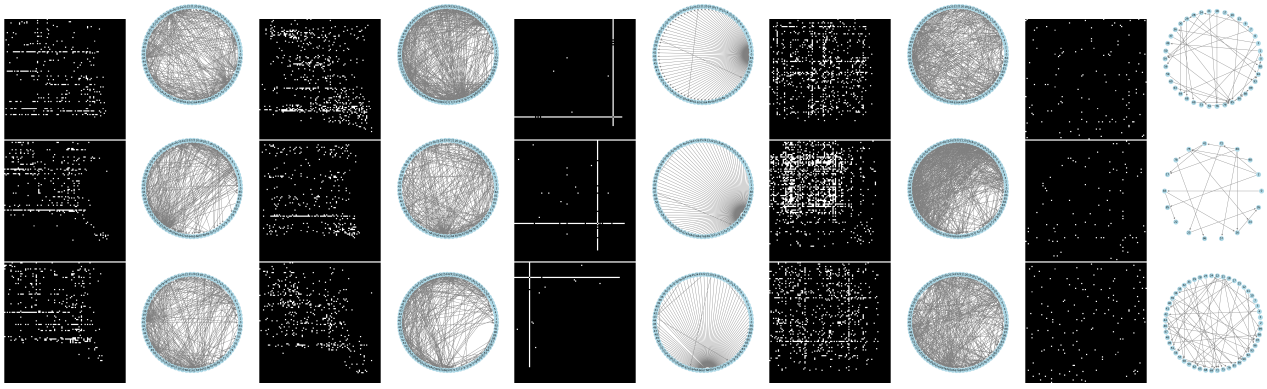


Figure 1: Overall model structure.

Model	Deg.		Clus. Coef.		Spec.	
	EMD ↓	TV ↓	EMD ↓	TV ↓	EMD ↓	TV ↓
GDSS (Jo, Lee, and Hwang 2022)	1.138	0.493	1.381	0.959	0.325	0.515
DisCo (Xu et al. 2024)	1.047	0.304	1.315	0.550	0.094	0.334
EDGE (Chen et al. 2023)	0.660	0.041	0.343	1.016	0.972	0.111
Ours	**0.164**	**0.028**	**0.154**	**0.021**	**0.047**	**0.007**

Table 1: Demonstrating Model Effectiveness: Comparison with Baseline Models



(a) Ori. (b) Ori. (c) Ours (d) Ours (e) GDSS (f) GDSS (g) EDGE (h) EDGE (i) DisCo (j) DisCo

Figure 2: Generated Results Demonstrate the Authenticity of Our Model

5.3 Latent Representation Analysis

SHAP Analysis Using SHAP analysis (Lundberg and Lee 2017) to interpret feature contributions, we analyzed the relationship between the 32 latent dimensions and generated graph properties. Figure 3 illustrates the SHAP results for the clustering coefficient, showing the entangled relationship between the latent dimensions and the graph metrics. Similar analyses were performed for each graph property (see Appendix E). These results motivated our subsequent development of a controllable generation method that operates effectively on such an entangled representation.

Discovering Key Latent Directions Affecting Generation Metrics We first visualized the microcircuit topology with t-SNE (Figure 4). To identify a latent direction for each graph metric, we trained a linear model to predict its value bin (discretized into 20 quantiles from 500 test graphs) from the 32D latent vectors. The resulting constant gradient defines each direction (Appendix F), and the model's high R^2 scores (all around 0.8) confirm a strong fit (Appendix Figure 12, left).

We then verified that these directions correspond to distinct metrics. The pairwise correlations between the metrics themselves (Appendix Figure 12, middle) closely matched the cosine similarities of their corresponding latent directions (Appendix Figure 12, right), confirming their specificity (Appendix H).

Finally, a traversal experiment validated these directions. Moving along a specific gradient from a mean latent vector and then decoding controllably manipulated the corresponding metric in the generated circuits, confirming that the directions directly govern structural features (Appendix G).

5.4 A Pipeline for Controllable Microcircuit Generation With MCMC Sampling

For the investigation of structure-function relationships in brain networks, the ability to generate network samples that closely match specific target descriptor values would be in-

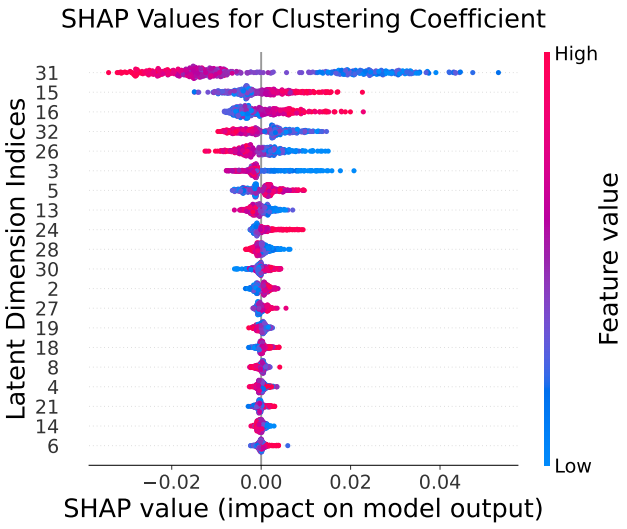


Figure 3: SHAP analysis for clustering coefficient. Latent dimensions are sorted by their importance. The results of SHAP analysis show the entangled relationship between the latent dimensions and the graph metrics.

valuable. Here, we propose a general pipeline to achieve this goal, leveraging our latent VAE connectome generator in conjunction with Markov Chain Monte Carlo (MCMC) sampling (Metropolis et al. 1953).

Given a target metric value t, the linear regression model $y = f(\mathbf{z}) = \mathbf{w}^T\mathbf{z}+b$ previously fitted on the 32-dimensional latent space can be leveraged as a heuristic for search. Our objective is to sample N latent vectors $\mathbf{z} = (z_1, z_2, ..., z_{32})$ that satisfy the condition $\mathcal{T}$:$|f(\mathbf{z}) - t| < \epsilon$, where ϵ defines a tolerance around the target value. This inequality defines a feasible region in the latent space, denoted as $\Omega_\mathcal{T} \subset \mathbb{R}^{32}$. However, since the domain of the linear regression model is unbounded, there are infinitely many solutions to this inequality distributed throughout the space. Many of these solutions might lie far from the true latent space distribution, leading to invalid or unrealistic generated graphs. Therefore, we should impose constraints on the distribution of the sampled latent vectors, aiming for the sampled points to follow the dataset's inherent latent distribution to the maximum extent. We approximate the distribution of the latent space as a 32-dimensional joint distribution by fitting a multivariate Gaussian distribution. Our goal is to sample N latent vectors from the conditioned distribution $p(\mathbf{z}|\mathcal{T})$, where $\mathbf{z} \in \Omega_\mathcal{T}$. According to Equation 3, this conditioned probability distribution can be represented as $p(\mathbf{z}|\mathcal{T}) = \frac{p(\mathbf{z})^{1/\tau}}{p(\Omega_\mathcal{T})} \cdot \mathbb{I}(\mathbf{z} \in \Omega_\mathcal{T})$, where $\mathbb{I}(\mathbf{z} \in \Omega_\mathcal{T})$ is an indicator function that is 1 if $\mathbf{z}$ belongs to the feasible region $\Omega_\mathcal{T}$ and 0 otherwise, and $p(\Omega_\mathcal{T}) = \int_{\Omega_\mathcal{T}} p(\mathbf{z})^{1/\tau} d\mathbf{z}$ is the probability mass of the feasible region.

As direct integration to compute $p(\Omega_\mathcal{T})$ is not feasible, we utilize Markov Chain Monte Carlo (MCMC) (Metropolis et al. 1953) to sample from the target conditional probability distribution despite the unknown denominator. We define a log-target-density function (LTD) to evaluate the likelihood of a proposed latent vector $\mathbf{z}$:

$$\text{LTD}(\mathbf{z}) = \frac{1}{\tau}\log p(\mathbf{z}), \tag{4}$$

Thus, the target conditional distribution for $\mathbf{z}$ should be proportional to $\exp(\text{LTD}(\mathbf{z})) \cdot \mathbb{I}(\mathbf{z} \in \Omega_\mathcal{T})$. We utilize the Metropolis-Hastings acceptance rule to generate a sequence of N sampled latent vectors. The detailed steps and parameters of the sampling algorithm are provided in the Appendix I.

To demonstrate the ability to generate graphs satisfying specific properties, we conducted experiments by specifying target percentile values ranging from 0% to 100%. We set $N = 500$ in the experiment. Figure 5 shows how the mean values of four key graph metrics evolve with the target percentile; the complete set of six curves is deferred to the appendix (Figure 17) for brevity. Generated graph metric values that show a roughly monotonically increasing trend with the target bin index demonstrates the effectiveness of this method. Furthermore, examples of the generated graphs for different target metrics and target values are presented in the AppendixJ.

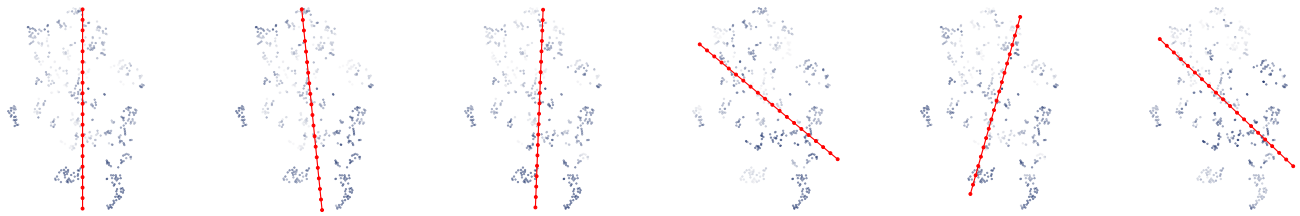
Figure 6 presents examples of controlled graph generation where the mean degree is the targeted property, showing samples generated by setting different target percentile ranges for this metric. Detailed results for controlling other graph properties are provided in the Appendix J.

5.5 Explore the Relationship between Structure and Function

Given that cortical microcircuits inherently operate as recurrent neural networks (RNNs), we employ reservoir computing (RC) to assess their functional capabilities. We use performance on classic tasks, specifically memory (copying) and classification, as a proxy for the network's functional features. To investigate this, we constructed reservoir networks using connectome architectures from the VAE and tested them on copying and classification tasks (Appendix K). We primarily investigated two questions: 1. Do these connectome-based reservoirs outperform randomly connected ones? 2. Are there graph features that, when systematically varied, predictably modulate task-specific performance? All of the hyperparameters of the experiments are stated in Appendix K.

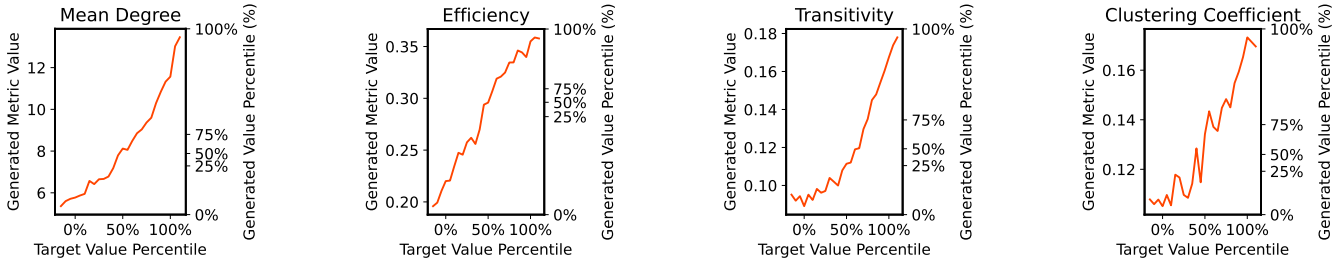
Copy Task Following (Keller et al. 2024), we use a copy task to assess network memory retention (details in Appendix K.1). We specified target values for a set of features at five distinct percentiles (25%, 37.5%, 50%, 62.5%, 75%) of their distributions. For each condition, we constructed Reservoir Networks from three VAE-generated connectome graphs, yielding an average test loss L_{gen}. As a baseline, we repeated this process using density-matched random graphs to obtain a corresponding loss L_{random}. The performance gain of the VAE-generated structures over their random counterparts is quantified by the percentage decrease in test loss, δ, calculated as follows:

$$\delta = \frac{L_{random} - L_{gen}}{L_{random}} \times 100\% \tag{5}$$



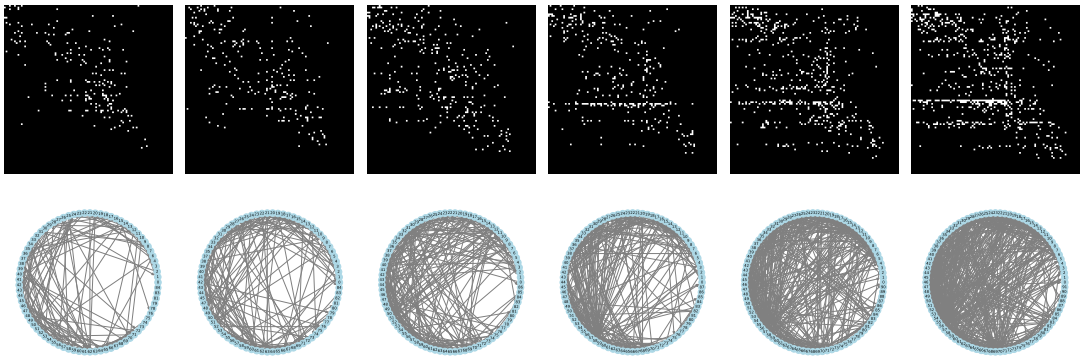
(a) Mean degree (b) Efficiency (c) Transitivity (d) Assortativity (e) Clus. Coef. (f) Modularity

Figure 4: t-SNE visualizations of the latent space reveal that variations in specific topological metrics correspond to distinct gradient directions within the embeddings. The darkness of each point's color corresponds to the magnitude of its associated bin index for the specific metric being considered. To further aid visualization, the red line overlaid on the t-SNE plot represents the direction of metric variation, obtained by fitting an auxiliary 2D linear regression model to the 2D t-SNE embeddings. This 2D regression is solely for visual convenience and does not replace the 32-dimensional analysis described in the main text.



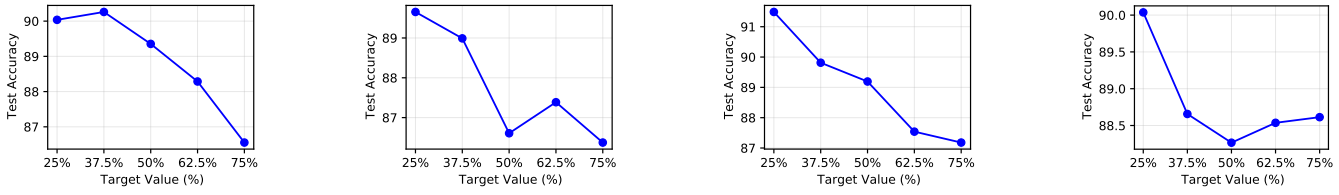
(a) (b) (c) (d)

Figure 5: Metrics of generated graphs when setting different targets (4 of 6 target graph metrics).



(a) target=0% (b) 20% (c) 40% (d) 60% (e) 80% (f) 100%

Figure 6: Generated graph examples targeting different mean degree percentile ranges.



(a) Efficiency (b) Mean Degree (c) Transitivity (d) Clustering Coefficient

Figure 7: 4 of the 6 metrics have a significant impact on task performance.

The results are documented in Table 2:

Target Percentile	25%	37.5%	50%	62.5%	75%	Ave.
Mean Deg.	39.94	54.32	55.23	50.34	55.36	51.04
Clus. Coef.	52.67	52.69	52.45	53.45	49.03	52.06
Efficiency	47.64	59.29	59.31	54.61	52.00	54.57
Transitivity	52.23	45.53	56.05	58.93	51.15	52.78
Assortativity	49.68	57.55	52.01	45.28	53.71	51.64
Modularity	63.98	60.03	58.42	49.82	57.20	57.89

Table 2: Performance Advantage of VAE-Generated Connectomes over Random Baselines in Reservoir Tasks (Measured by % Test Loss Decrease)

Appendix K.1 visually compares the copy task outputs from networks with VAE-generated reservoirs against those with random counterparts.

Classification Task On the sequential MNIST classification task, we found a clear monotonic or near-monotonic relationship between a reservoir network's test accuracy and several of its reservoir's graph features—notably efficiency, mean degree, transitivity, and clustering coefficient (Figure 7). This result was established by training networks built from reservoir graphs systematically generated to have feature values at five distinct percentiles (25-75%). Specifically, for each percentile condition, we randomly selected three generated graphs, constructed an network from each, and averaged their final test accuracies (details in Appendix K.2).

6 Conclusions and Discussions

We introduced a VAE-based generative model that learns a compact, interpretable, low-dimensional latent space for connectome of two species. The existence of such a low-dimensional representation is also supported by the 'genomic bottleneck' theory. An animal's genes guide the development of its nervous system (Arnatkevičiute et al. 2020; Wainberg et al. 2024). However, the amount of information genes can hold is much smaller than what would be needed to explicitly list every single connection in a fully formed brain (Suganuma et al. 2020; Nigam et al. 2024). This observation, known as the 'genomic bottleneck' (Shuvaev et al. 2024), suggests (without negating the combined influence of genetics and environment) that there must be a simpler, more compact set of rules or a low-dimensional representation that guides how brain networks grow and organize themselves, for which the finite information capacity of the genome provides primary evidence.

We demonstrated that directions in this space encode specific network properties, enabling the controlled synthesis of novel circuits with desired features via latent navigation and an MCMC pipeline. This approach provides a powerful tool to investigate neural design principles, explore structure-function relationships, and inform advanced AI development by revealing the low-dimensional generative rules behind complex structures.

Limitations and Future Work Despite promising results, this work has several limitations. Firstly, our choice of a Variational Autoencoder was deliberate, driven by the primary objective of learning an explicit, interpretable, and low-dimensional latent space. This "information bottleneck" is crucial for uncovering the compact generative blueprint hypothesized to underlie brain development and for enabling controlled synthesis. While other modern generative models, such as diffusion models, demonstrate powerful sample generation capabilities, their latent spaces are not always as directly optimized for, or as easily amenable to, the extraction of such a compressed and interpretable code as a VAE's. Future work could, however, investigate adaptations of these models or hybrid approaches to achieve similar goals. Secondly, our model currently focuses on binary topological structure, neglecting crucial biological details such as neuron types, synaptic weights, and activity dynamics. Incorporating these features is a key direction for future work to enhance biological realism. Thirdly, while this study incorporates connectome data from both Drosophila and mouse, the detailed generative modeling and latent analysis are primarily centered on microcircuits from the mouse visual cortex. Consequently, the learned representations and generative rules may require further adaptation and validation for other brain regions, larger-scale connectomes, or a broader range of species. The current fixed-size input representation (100×100 padding) also poses challenges for direct application to circuits of highly variable sizes without architectural modifications. Furthermore, while we identified interpretable linear directions in the latent space, exploring more complex, non-linear relationships and the full extent of encoded biological constraints warrants further investigation. Finally, while preliminary work has linked generated structures to function, more rigorously validating their functional viability is a key next step to bridge structural generation with functional understanding.

Although serving as a simplified paradigm, reservoir computing demonstrates that network function is inherently shaped by structural characteristics. Future research will focus on mapping task-performance gradients directly within the latent space. Crucially, this investigation should extend beyond the central latent regions by performing interpolations across a broader scope of the latent space. This will enable the synthesis of structures from edge-case regions, allowing us to systematically evaluate how structural variations—especially those at the boundaries of the learned distribution—correlate with functional performance.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No. 62576011).

References

Arnatkevičiute, A.; Fulcher, B. D.; Oldham, S.; Tiego, J.; Paquola, C.; Gerring, Z.; Aquino, K.; Hawi, Z.; Johnson, B.; Ball, G.; Klein, M.; Deco, G.; Franke, B.; Bellgrove, M.; and Fornito, A. 2020. Genetic influences on hub connectivity of the human connectome. *bioRxiv*.

Bargmann, C. I.; and Newsome, W. T. 2014. The Brain Research Through Advancing Innovative Neurotechnologies (BRAIN) Initiative and Neurology. *JAMA Neurology*, 71(6): 675–676.

Behrouzi, T.; and Hatzinakos, D. 2022. Graph variational auto-encoder for deriving EEG-based graph embedding. *Pattern Recognition*, 121: 108202.

Betzel, R. F.; Avena-Koenigsberger, A.; Goñi, J.; He, Y.; de Reus, M. A.; Griffa, A.; Vértes, P. E.; Mišic, B.; Thiran, J.-P.; Hagmann, P.; van den Heuvel, M.; Zuo, X.-N.; Bullmore, E. T.; and Sporns, O. 2016. Generative models of the human connectome. *NeuroImage*, 124: 1054–1064.

Breakspear, M. 2017. Dynamic models of large-scale brain activity. *Nature Neuroscience*, 20: 340–352.

Callaway, E. M. 1998. LOCAL CIRCUITS IN PRIMARY VISUAL CORTEX OF THE MACAQUE MONKEY. *Annual Review of Neuroscience*, 21(Volume 21, 1998): 47–74.

Chen, G.; and Gong, P. 2019. Computing by modulating spontaneous cortical activity patterns as a mechanism of active visual processing. *Nature Communications*, 10(1).

Chen, G.; and Gong, P. 2022. A spatiotemporal mechanism of visual attention: Superdiffusive motion and theta oscillations of neural population activity patterns. *Science Advances*, 8(16).

Chen, G.; Scherr, F.; and Maass, W. 2022. A data-based large-scale model for primary visual cortex enables brain-like robust and versatile visual processing. *Science Advances*, 8(44): eabq7592.

Chen, X.; He, J.; Han, X.; and Liu, L.-P. 2023. Efficient and Degree-Guided Graph Generation via Discrete Diffusion Modeling. arXiv:2305.04111.

Chen, X.; Liao, X.; Dai, Z.; Lin, Q.; Wang, Z.; Li, K.; and He, Y. 2018. Topological Analyses of Functional Connectomics: A Crucial Role of Global Signal Removal, Brain Parcellation, and Null Models. *Human Brain Mapping*, 39: 4545 – 4564.

Clark, K. B. 2023. Neural Field Continuum Limits and the Structure–Function Partitioning of Cognitive–Emotional Brain Networks. *Biology*, 12.

Collin, G.; and Whitfield-Gabrieli, S. 2023. Mapping the multimodal connectome: On the architects of brain network science. *PLoS biology*, 21(3): e3002043.

Consortium, T. M.; Bae, J. A.; Baptiste, M.; Bishop, C. A.; Bodor, A. L.; Brittain, D.; Buchanan, J.; Bumbarger, D. J.; Castro, M. A.; Celii, B.; Cobos, E.; Collman, F.; da Costa, N. M.; Dorkenwald, S.; Elabbady, L.; Fahey, P. G.; Fliss, T.; Froudarakis, E.; Gager, J.; Gamlin, C.; Gray-Roncal, W.; Halageri, A.; Hebditch, J.; Jia, Z.; Joyce, E.; Joyce, J.; Jordan, C.; Kapner, D.; Kemnitz, N.; Kinn, S.; Kitchell, L. M.; Koolman, S.; Kuehner, K.; Lee, K.; Li, K.; Lu, R.; Macrina, T.; Mahalingam, G.; Matelsky, J.; McReynolds, S.; Miranda, E.; Mitchell, E.; Mondal, S. S.; Moore, M.; Mu, S.; Muhammad, T.; Nehoran, B.; Ogedengbe, O.; Papadopoulos, C.; Papadopoulos, S.; Patel, S.; Pitkow, X.; Popovych, S.; Ramos, A.; Clay Reid, R.; Reimer, J.; Rivlin, P. K.; Rose, V.; Schneider-Mizell, C. M.; Seung, H. S.; Silverman, B.; Silversmith, W.; Sterling, A.; Sinz, F. H.; Smith, C. L.; Suckow, S.; Takeno, M.; Tan, Z. H.; Tolias, A. S.; Torres, R.; Turner, N. L.; Walker, E. Y.; Wang, T.; Wanner, A.; Wester, B. A.; Williams, G.; Williams, S.; Willie, K.; Willie, R.; Wong, W.; Wu, J.; Xu, C.; Yang, R.; Yatsenko, D.; Ye, F.; Yin, W.; Young, R.; Yu, S.-c.; Xenes, D.; and Zhang, C. 2023. Functional connectomics spanning multiple areas of mouse visual cortex. *bioRxiv*.

Craddock, R. C.; Tungaraza, R. L.; and Milham, M. P. 2015. Connectomics and new approaches for analyzing human brain functional connectivity. *GigaScience*, 4(1): s13742–015–0045–x.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805.

Dong, Z.; Wu, Y.; Xiao, Y.; Chong, J.; Jin, Y.; and Zhou, J. 2023. Beyond the Snapshot: Brain Tokenized Graph Transformer for Longitudinal Brain Functional Connectome Embedding. *International Conference on Medical Image Computing and Computer-Assisted Intervention*, abs/2307.00858.

Dorkenwald, S.; McKellar, C. E.; Macrina, T.; Kemnitz, N.; Lee, K.; Lu, R.; Wu, J.; Popovych, S.; Mitchell, E.; Nehoran, B.; et al. 2022. FlyWire: online community for whole-brain connectomics. *Nature methods*, 19(1): 119–128.

Douglas, R.; and Martin, K. 2004. Neural circuits of the neocortex. *Annual review of neuroscience*, 27: 419–51.

Elam, J. S.; Glasser, M. F.; Harms, M. P.; Sotiropoulos, S. N.; Andersson, J. L.; Burgess, G. C.; Curtiss, S. W.; Oostenveld, R.; Larson-Prior, L. J.; Schoffelen, J.-M.; Hodge, M. R.; Cler, E. A.; Marcus, D. M.; Barch, D. M.; Yacoub, E.; Smith, S. M.; Ugurbil, K.; and Van Essen, D. C. 2021. The Human Connectome Project: A retrospective. *NeuroImage*, 244: 118543.

Felleman, D. J.; and Van Essen, D. C. 1991. Distributed Hierarchical Processing in the Primate Cerebral Cortex. *Cerebral Cortex*, 1(1): 1–47.

Fu, H.; Li, C.; Liu, X.; Gao, J.; Celikyilmaz, A.; and Carin, L. 2019. Cyclical Annealing Schedule: A Simple Approach to Mitigating KL Vanishing. arXiv:1903.10145.

Goñi, J.; van den Heuvel, M. P.; Avena-Koenigsberger, A.; de Mendizabal, N. V.; Betzel, R. F.; Griffa, A.; Hagmann, P.; Corominas-Murtra, B.; Thiran, J.-P.; and Sporns, O. 2014. Resting-brain functional connectivity predicted by analytic measures of network communication. *Proceedings of the National Academy of Sciences*, 111(2): 833–838.

Gretton, A.; Borgwardt, K. M.; Rasch, M. J.; Schölkopf, B.; and Smola, A. 2012. A Kernel Two-Sample Test. *Journal of Machine Learning Research*, 13(25): 723–773.

Hassabis, D.; Kumaran, D.; Summerfield, C.; and Botvinick, M. 2017. Neuroscience-Inspired Artificial Intelligence. *Neuron*, 95(2): 245–258.

Henriksen, S.; Pang, R.; and Wronkiewicz, M. 2016. A simple generative model of the mouse mesoscale connectome. *eLife*, 5: e12366.

Higgins, I.; Matthey, L.; Pal, A.; Burgess, C. P.; Glorot, X.; Botvinick, M. M.; Mohamed, S.; and Lerchner, A. 2016.

beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. In *International Conference on Learning Representations*.

Honey, C. J.; Kötter, R.; Breakspear, M.; and Sporns, O. 2007. Network structure of cerebral cortex shapes functional connectivity on multiple time scales. *Proceedings of the National Academy of Sciences*, 104(24): 10240–10245.

Jaeger, H. 2001. The "Echo State" Approach to Analysing and Training Recurrent Neural Networks. *GMD-Report 148, German National Research Institute for Computer Science*.

Jo, J.; Lee, S.; and Hwang, S. J. 2022. Score-based Generative Modeling of Graphs via the System of Stochastic Differential Equations. arXiv:2202.02514.

Jones, E. G. 2000. Microcolumns in the cerebral cortex. *Proceedings of the National Academy of Sciences*, 97(10): 5019–5021.

Kaiser, M.; and Hilgetag, C. C. 2004. Modelling the development of cortical systems networks. *Neurocomputing*, 58-60: 297–302. Computational Neuroscience: Trends in Research 2004.

Kan, X.; Cui, H.; Lukemire, J.; Guo, Y.; and Yang, C. 2022. FBNETGEN: Task-Aware GNN-based fMRI Analysis via Functional Brain Network Generation. *International Conference on Medical Imaging With Deep Learning*, 172: 618–637.

Keller, T. A.; Muller, L.; Sejnowski, T.; and Welling, M. 2024. Traveling Waves Encode the Recent Past and Enhance Sequence Learning. arXiv:2309.08045.

Kingma, D. P.; and Welling, M. 2022. Auto-Encoding Variational Bayes. arXiv:1312.6114.

Lecun, Y.; Bengio, Y.; and Hinton, G. 2015. Deep learning. *Nature*, 521(7553): 436–444.

Liao, R.; Li, Y.; Song, Y.; Wang, S.; Nash, C.; Hamilton, W. L.; Duvenaud, D.; Urtasun, R.; and Zemel, R. S. 2020. Efficient Graph Generation with Graph Recurrent Attention Networks. arXiv:1910.00760.

Lundberg, S.; and Lee, S.-I. 2017. A Unified Approach to Interpreting Model Predictions. arXiv:1705.07874.

Metropolis, N.; Rosenbluth, A. W.; Rosenbluth, M. N.; Teller, A. H.; and Teller, E. 1953. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6): 1087–1092.

Miller, K. D. 2016. Canonical computations of cerebral cortex. *Current Opinion in Neurobiology*, 37: 75–84. Neurobiology of cognitive behavior.

Mišić, B.; Betzel, R. F.; de Reus, M. A.; van den Heuvel, M. P.; Berman, M. G.; McIntosh, A. R.; and Sporns, O. 2016. Network-Level Structure-Function Relationships in Human Neocortex. *Cerebral Cortex*, 26(7): 3285–3296.

Nigam, A.; Pollice, R.; Friederich, P.; and Aspuru-Guzik, A. 2024. Artificial design of organic emitters via a genetic algorithm enhanced by a deep neural network. *Chem. Sci.*, 15: 2618–2639.

Osegi, E. N. 2023. Neuronal Auditory Machine Intelligence (NEURO-AMI) In Perspective. arXiv:2401.02421.

Reid, R. C.; and Alonso, J.-M. 1996. The processing and encoding of information in the visual cortex. *Current Opinion in Neurobiology*, 6(4): 475–480.

Rubinov, M.; and Sporns, O. 2010. Complex network measures of brain connectivity: Uses and interpretations. *NeuroImage*, 52(3): 1059–1069. Computational Models of the Brain.

Said, A.; Bayrak, R. G.; Derr, T.; Shabbir, M.; Moyer, D.; Chang, C.; and Koutsoukos, X. 2023. NeuroGraph: Benchmarks for Graph Machine Learning in Brain Connectomics. *Neural Information Processing Systems*.

Shuvaev, S.; Lachi, D.; Koulakov, A.; and Zador, A. 2024. Encoding innate ability through a genomic bottleneck. *Proceedings of the National Academy of Sciences*, 121(38): e2409160121.

Sporns, O. 2011. The human connectome: a complex network. *Annals of the New York Academy of Sciences*, 1224(1): 109–125.

Sporns, O. 2013. The human connectome: Origins and challenges. *NeuroImage*, 80: 53–61. Mapping the Connectome.

Sporns, O.; Tononi, G.; and Kötter, R. 2005. The Human Connectome: A Structural Description of the Human Brain. *PLoS computational biology*, 1(4): e42–e42.

Su, J.; Lu, Y.; Pan, S.; Murtadha, A.; Wen, B.; and Liu, Y. 2023. RoFormer: Enhanced Transformer with Rotary Position Embedding. arXiv:2104.09864.

Suganuma, M.; Kobayashi, M.; Shirakawa, S.; and Nagao, T. 2020. Evolution of Deep Convolutional Neural Networks Using Cartesian Genetic Programming. *Evolutionary Computation*, 28(1): 141–163.

Tan, Y.-F.; Ting, C.; Noman, F. M.; Phan, R.; and Ombao, H. 2022. Graph-Regularized Manifold-Aware Conditional Wasserstein GAN for Brain Functional Connectivity Generation. *arXiv.org*, abs/2212.05316.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2023. Attention Is All You Need. arXiv:1706.03762.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. arXiv:1710.10903.

Vázquez-Rodríguez, B.; Suárez, L. E.; Markello, R. D.; Shafiei, G.; Paquola, C.; Hagmann, P.; van den Heuvel, M. P.; Bernhardt, B. C.; Spreng, R. N.; and Misic, B. 2019. Gradients of structure–function tethering across neocortex. *Proceedings of the National Academy of Sciences*, 116(42): 21219–21227.

Wainberg, M.; Forde, N.; Mansour, S.; Kerrebijn, I.; Medland, S.; Hawco, C.; and Tripathy, S. 2024. Genetic architecture of the structural connectome. *Nature Communications*, 15.

White, C.; Safari, M.; Sukthanker, R.; Ru, B.; Elsken, T.; Zela, A.; Dey, D.; and Hutter, F. 2023. Neural Architecture Search: Insights from 1000 Papers. arXiv:2301.08727.

Winter, R.; Noé, F.; and Clevert, D.-A. 2021. Permutation-Invariant Variational Autoencoder for Graph-Level Representation Learning. arXiv:2104.09856.

Xia, M.; Wang, J.; and He, Y. 2013. BrainNet Viewer: A Network Visualization Tool for Human Brain Connectomics. *PLoS ONE*, 8.

Xu, Z.; Qiu, R.; Chen, Y.; Chen, H.; Fan, X.; Pan, M.; Zeng, Z.; Das, M.; and Tong, H. 2024. Discrete-state Continuous-time Diffusion for Graph Generation. *arXiv preprint arXiv:2405.11416*.

You, J.; Ying, R.; Ren, X.; Hamilton, W. L.; and Leskovec, J. 2018. GraphRNN: Generating Realistic Graphs with Deep Auto-regressive Models. arXiv:1802.08773.

Yu, Y.; Kan, X.; Cui, H.; Xu, R.; Zheng, Y.; Song, X.; Zhu, Y.; Zhang, K.; Nabi, R.; Guo, Y.; Zhang, C.; and Yang, C. 2022. Learning Task-Aware Effective Brain Connectivity for fMRI Analysis With Graph Neural Networks (Extended Abstract). *2022 IEEE International Conference on Big Data (Big Data)*, 4995–4996.

Zador, A.; Escola, S.; Richards, B.; Ölveczky, B.; Bengio, Y.; Boahen, K.; Botvinick, M.; Chklovskii, D.; Churchland, A.; Clopath, C.; DiCarlo, J.; Ganguli, S.; Hawkins, J.; Koerding, K.; Koulakov, A.; LeCun, Y.; Lillicrap, T.; Marblestone, A.; Olshausen, B.; Pouget, A.; Savin, C.; Sejnowski, T.; Simoncelli, E.; Solla, S.; Sussillo, D.; Tolias, A. S.; and Tsao, D. 2023. Toward Next-Generation Artificial Intelligence: Catalyzing the NeuroAI Revolution. arXiv:2210.08340.

Zuo, Q.; Hu, J.; Zhang, Y.; Pan, J.-D.; Jing, C.; Chen, X.; Meng, X.; and Hong, J. 2023. Brain Functional Network Generation Using Distribution-Regularized Adversarial Graph Autoencoder With Transformer for Dementia Diagnosis. *Computer Modeling in Engineering & Sciences : CMES*, 137: 2129 – 2147.

A Training Details

Training was performed with a learning rate (lr) of 0.001. We employed a cyclical beta annealing schedule (Fu et al. 2019) with a cycle length of 600 epochs. In this schedule, β was linearly increased from 0 to $1e-6$ during the first half of each cycle and held constant at $1e-6$ for the second half. The model was trained for a total of 10000 epochs on an RTX4090 GPU.

B Model Structure Details

B.1 Model Components

Node Feature Encoder The node feature encoder, utilizing a three-layer multi-head GAT network, transforms 100-dimensional one-hot node representations into 32-dimensional node embeddings. See Appendix B.2 for GAT architecture details.

Graph Global Encoder Treating the y-ordered nodes as a sequence (analogous to words in a sentence), the graph global encoder transforms node embeddings into a fixed-size latent representation. Following (Devlin et al. 2019; Winter, Noé, and Clevert 2021), a dummy node v_0 is prepended to the sequence to serve as a global embedding. The encoder applies rotational positional encoding (RoPE) (Su et al. 2023) and several transformer encoder layers to this augmented sequence. The final embedding of v_0 is taken as the global graph representation. An MLP is then applied to compute the 32-dimensional mean and variance for sampling the 32-dimensional latent vector $\mathbf{z}$, similar to standard VAEs.

Node Feature Decoder The Node Feature Decoder reconstructs individual node embeddings from the global graph embedding using several transformer decoder layers. The input is the global graph embedding, augmented with rotational positional encoding (RoPE) (Su et al. 2023). The global graph embedding also serves as memory for the decoder's cross-attention, leveraging this global context during node feature reconstruction.

Edge Predictor The edge predictor is a cross-node interaction layer. It takes the node feature decoder output $h \in \mathbb{R}^{n\times d}$, where $n = 100$ is the maximum number of nodes and d is the embedding dimension. Edges are predicted using the dot product of embeddings transformed by two distinct linear layers with activation.

$$\mathbf{A}_{\text{pred}} = \sigma(\text{LeakyReLU}(\mathbf{h}\mathbf{W}_1)(\text{LeakyReLU}(\mathbf{W}_2\mathbf{h})^\top)), \tag{6}$$

The output $\mathbf{A}_{\text{pred}}$ provides a probabilistic adjacency matrix where each entry, a floating-point number between 0 and 1, denotes the likelihood of an edge. We then perform a Bernoulli sampling process on each entry using this probability to generate a binary adjacency matrix.

B.2 GAT Mechanism

In GAT network, the representation of each node is iteratively refined by aggregating information from its neighboring nodes through a message-passing mechanism. Specifically, for each node v_i, an attention score e_{ij} is computed with respect to its neighboring nodes v_j by $e_{ij} = a\left(\mathbf{W}\vec{h}_i, \mathbf{W}\vec{h}_j\right)$, where $\vec{h}_i$ represents the current feature vector of node v_i, $\vec{h}_i$ represents the current feature vector of a neighboring node v_j, and a denotes the attention mechanism, parameterized by the weight matrix $\mathbf{W}$. These attention scores are then normalized across the neighbors of v_i using the softmax function to obtain the attention coefficients α_{ij}. Finally, the updated representation $\vec{h}'_i$ of node v_i is obtained by aggregating the feature vectors of its neighbors, weighted by the calculated attention coefficients, followed by a non-linear activation function σ by $\vec{h}'_i = \sigma(\sum_{j\in N_i} \alpha_{ij} W\vec{h}_j)$

C Definition of the Graph Metrics

Mean Degree: The mean degree of a graph is the mean total degree of each node i:

$$k_i = \sum_{j\in N} a_{ij}. \tag{7}$$

Efficiency:

$$E = \frac{1}{n}\sum_{i\in N} E_i = \frac{1}{n}\sum_{i\in N} \frac{\sum_{j\in N, j\neq i} d_{ij}^{-1}}{n-1}, \tag{8}$$

where E_i is the efficiency of node i. Directed global efficiency is defined as:

$$E^{\rightarrow} = \frac{1}{n}\sum_{i\in N} \frac{\sum_{j\in N, j\neq i} (d_{ij})^{-1}}{n-1}. \tag{9}$$

Clustering coefficient:

$$C = \frac{1}{n}\sum_{i\in N} C_i = \frac{1}{n}\sum_{i\in N} \frac{2t_i}{k_i(k_i-1)}, \tag{10}$$

where C_i is the clustering coefficient of node $i (C_i = 0$ for $k_i < 2)$. Directed clustering coefficient is defined as:

$$C^{\rightarrow} = \frac{1}{n}\sum_{i\in N} \frac{t_i}{\left(k_i^{\text{out}} + k_i^{\text{in}}\right)\left(k_i^{\text{out}} + k_i^{\text{in}} - 1\right) - 2\sum_{j\in N} a_{ij}a_{ji}}. \tag{11}$$

Transitivity of the network:

$$T = \frac{\sum_{i\in N} 2t_i}{\sum_{i\in N} k_i(k_i-1)}, \tag{12}$$

Directed transitivity is defined as:

$$T^{\rightarrow} = \frac{\sum_{i\in N} t_i^{\rightarrow}}{\sum_{i\in N}[(k_i^{\text{out}} + k_i^{\text{in}})(k_i^{\text{out}} + k_i^{\text{in}} - 1) - 2\sum_{j\in N} a_{ij}a_{ji}]}. \tag{13}$$

Modularity:

$$Q = \sum_{u\in M}\left[e_{uu} - \left(\sum_{v\in M} e_{uv}\right)^2\right], \tag{14}$$

where the network is fully subdivided into a set of non-overlapping modules M, and e_{uv} is the proportion of all links that connect nodes in module u with nodes in module v.

Directed modularity is defined as:

$$Q^{\rightarrow} = \frac{1}{l}\sum_{i,j\in N}\left[a_{ij} - \frac{k_i^{\text{out}}k_i^{\text{in}}}{l}\right]\delta_{m_i,m_j}. \tag{15}$$

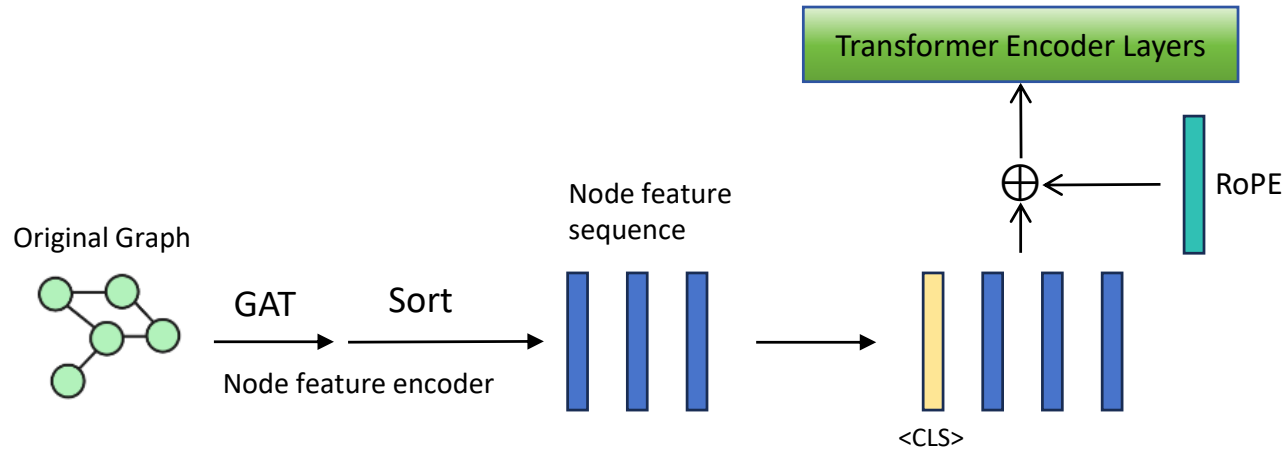


Figure 8: Encoder Structure

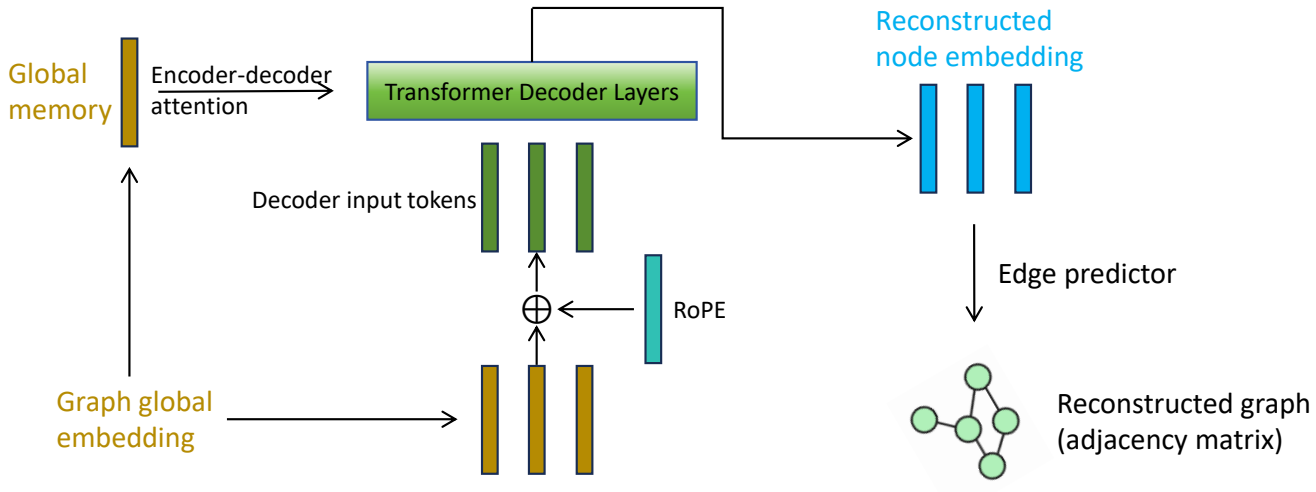


Figure 9: Decoder Structure

D Reconstruction Results

The reconstruction results are shown in Figure 10.

E Details of SHAP Analysis

To understand the relationship between the latent space and the generated graph properties, we trained a random forest regression model to predict each graph metric of the generated graphs based on the 32-dimensional latent vectors. Subsequently, we performed standard SHAP analysis and visualized the SHAP values for each latent dimension. The SHAP analysis details for different metrics is shown in Figure 11.

For instance, the SHAP analysis for assortativity indicates that a few dimensions significantly influence the assortativity.

F Details of the Linear Regression Model

Formally, let $y \in \{0, 1, 2, ..., 19\}$ represent the bin index for a given metric, corresponding to the 20 bins of the metric's values. Given N graphs in our test set, we first compute their latent codes: $\mathbf{z}^{(i)*} = E(\mathcal{G}_\pi^{(i)})$, where $\mathbf{z}^{(i)*}$ is a 32-dimensional vector representing the latent code for the i-th graph $\mathcal{G}_\pi^{(i)}$ obtained from the encoder E. We normalize the latent codes to have zero mean and unit variance dimension-wise by $\mathbf{z}^{(i)} = \dfrac{\mathbf{z}^{(i)*} - \mu_\mathbf{z}}{\sigma_\mathbf{z}}$. The linear regression model uses these normalized latent codes $\mathbf{z}^{(i)}$ to predict the corresponding bin index $\hat{y}$:

$$\hat{y} = f(\mathbf{z}) = \mathbf{w}^T\mathbf{z} + b. \tag{16}$$

We fit this linear equation using Ridge regression. The gradient of the predicted bin index $\hat{y}$ with respect to the normalized latent vector $\mathbf{z}$ is then given by: $\nabla f(\mathbf{z}) = \mathbf{w}$. Finally, we consider the direction of the gradient by normalization: $\mathbf{w}_0 = \dfrac{\mathbf{w}}{|\mathbf{w}|}$.

The R^2 scores of the linear regression, Spearman's rank correlation coefficient matrix between the six graph metric descriptors and cosine distance matrix between the gradient directions in the 32-dimensional latent space are shown in Fig 12

G Details of Gradient Direction Moving

To validate whether these identified directions truly reflect changes in the corresponding metrics, we performed a traversal experiment. Starting from the mean latent vector of the test set, we moved along each metric's identified gradient direction (both positive and negative shifts) and decoded the resulting latent vectors to observe the change in the metric's value (Figure 13). The moving region was carefully chosen to ensure that the shifted latent vector with the maximum offset remained within the range of $[\mu_\mathbf{z} - 2\sigma_\mathbf{z}, \mu_\mathbf{z} + 2\sigma_\mathbf{z}]$ across all dimensions, where μ_z and σ_z are the mean and standard deviation of the latent vectors in the test set, respectively.

We show the generated graphs when we move along the direction of gradient of different metrics in Figure 14 and Figure 15.

H Details of Correlations Between Graph Metrics

The correlations between pairs of graph metrics are shown in Figure 16.

I MCMC Sampling Details

For features $\mathbf{z}^* = (z_1, ..., z_{32})$, we first normalize it to zero-mean and unit-variance $\mathbf{z}$. Then we fit a linear regression model from the 32-dimensional feature to the percentile of certain property, for example, mean degree. Denote the linear regression model as:

$$y = f(\mathbf{z}) = \mathbf{w}^T\mathbf{z} + b \tag{17}$$

We rewrite the LTD function as:

$$LTD(\mathbf{z}) = \begin{cases} w\log p(\mathbf{z}) & \text{if } \mathbf{z} \in \Omega_\mathcal{T} \\ -\infty & \text{else} \end{cases} \tag{18}$$

where $p(\mathbf{z})$ is estimated by fitting a multivariate Gaussian distribution according to the correlations between dimensions of the normalized latent vectors:

$$f(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^D|\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu})\right) \tag{19}$$

and $w = \dfrac{1}{\tau}$ is the weight to control the degree how the sampled points concentrated around the high density area of the latent space. Let the target value to be t We find the initial feasible point $\mathbf{z_0}$ by equation:

$$k = \frac{t - \bar{y}}{|\mathbf{w}|^2}, \tag{20}$$

$$\mathbf{z_0} = k\mathbf{w}. \tag{21}$$

This point must satisfy the condition $|f(\mathbf{z}) - t| < \epsilon$ because the linear regression plane pass the center point of the dataset, where $\mathbf{z} = \mathbf{0}$ and $y = \bar{y}$.

From this initial feasible solution, we run the Metropolis-Hastings Algorithm:

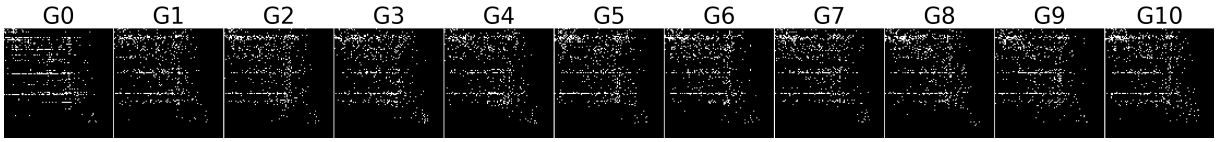
In the specific setting, we adopt $burn = 0$, $w = 10$, $\epsilon = 0.1$, $\sigma = 0.01$ and $thin = 1$.

J Generated Graphs for Different Targets

Generated graph examples targeting different metric percentile ranges are shown in Figure 18 (mean degree), Figure 19 (efficiency), Figure 20 (transitivity), Figure 21 (clustering coefficient), Figure 22 (assortativity) and Figure 23 (modularity).

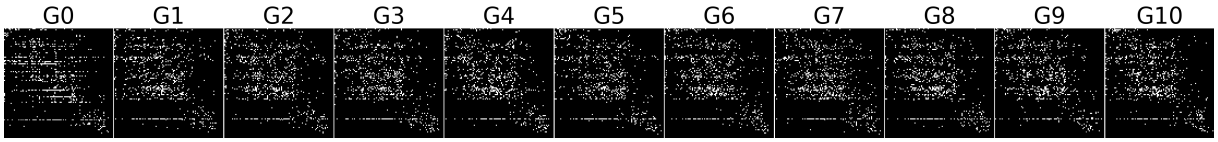
K Details of Reservoir Network Experiments

We employed a reservoir network architecture with a reservoir of $N = 100$ units. We trained both the input weights (W_{in}) and output weights (W_{out}), while only the recurrent weights (W) within the reservoir were held fixed post-initialization. The neuron types were set to excitatory or inhibitory according to a fixed configuration sampled from the MICrONS dataset distributions, which remained constant across all experiments. The reservoir's connectivity matrix



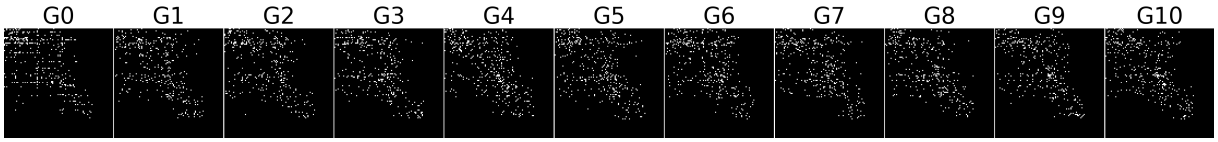
(a) Sample1, Adjacency Matrix

(b) Sample1, Graph Network



(c) Sample2, Adjacency Matrix

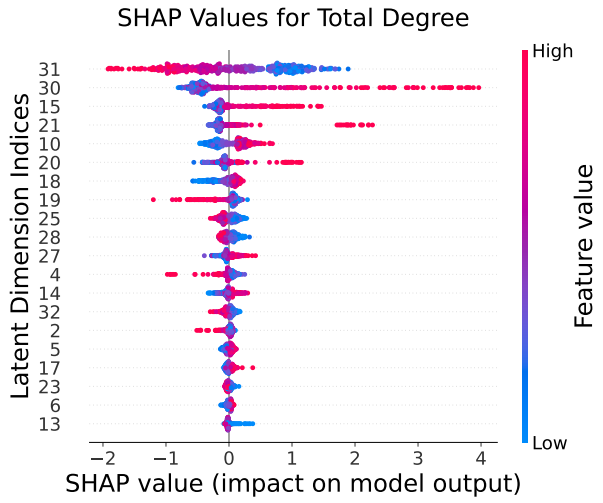
(d) Sample2, Graph Network



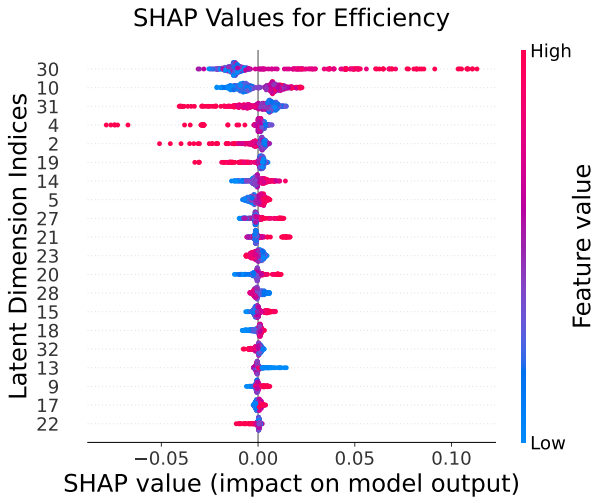
(e) Sample3, Adjacency Matrix

(f) Sample3, Graph Network

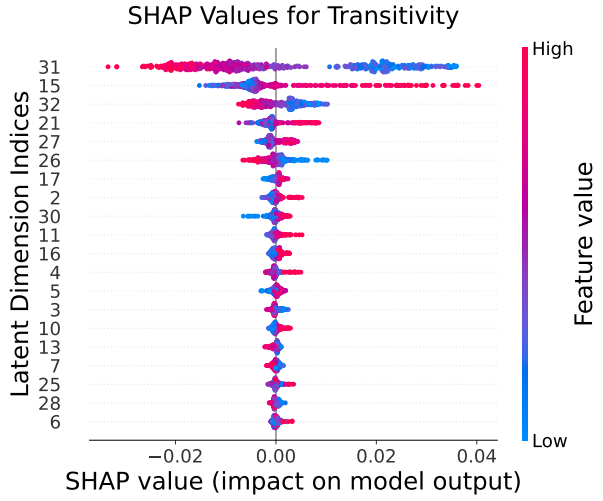
Figure 10: Reconstruction Examples: Original Graphs (G0) and Multiple Decoded Samples (G1-G10). Each row displays one original graph followed by ten distinct reconstructions, resulting from the probabilistic nature of the VAE and the binarization process.



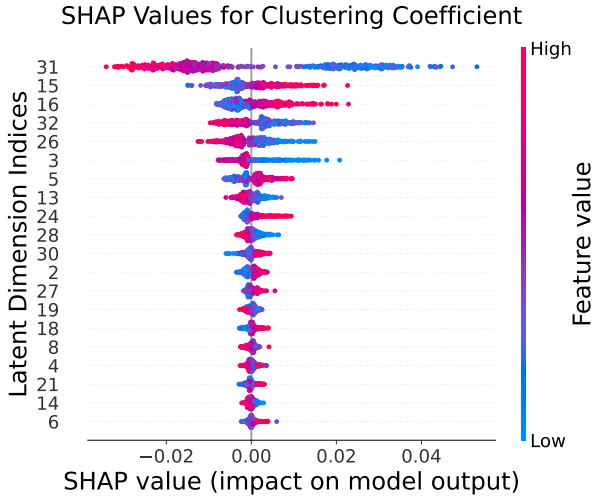
(a)



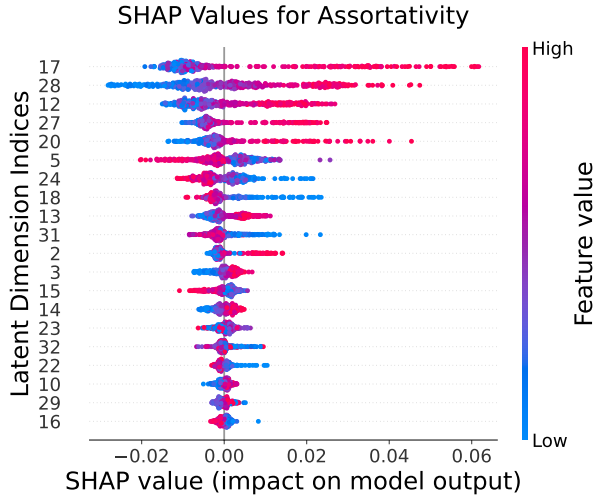
(b)



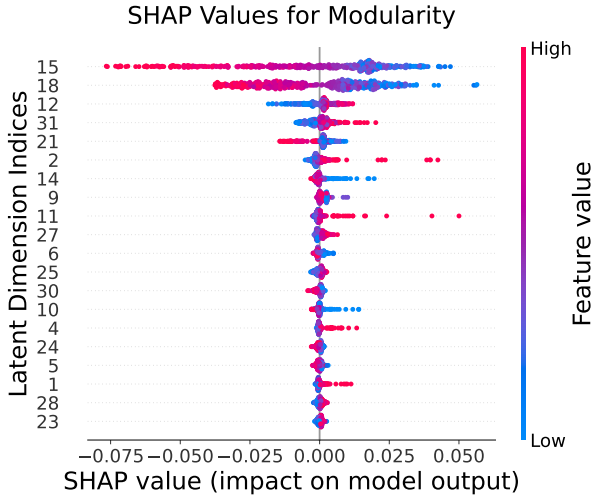
(c)



(d)



(e)



(f)

Figure 11: SHAP analysis for different metrics. The dimensions are sorted by their importance of contribution and only the top 20 are displayed due to limited space.

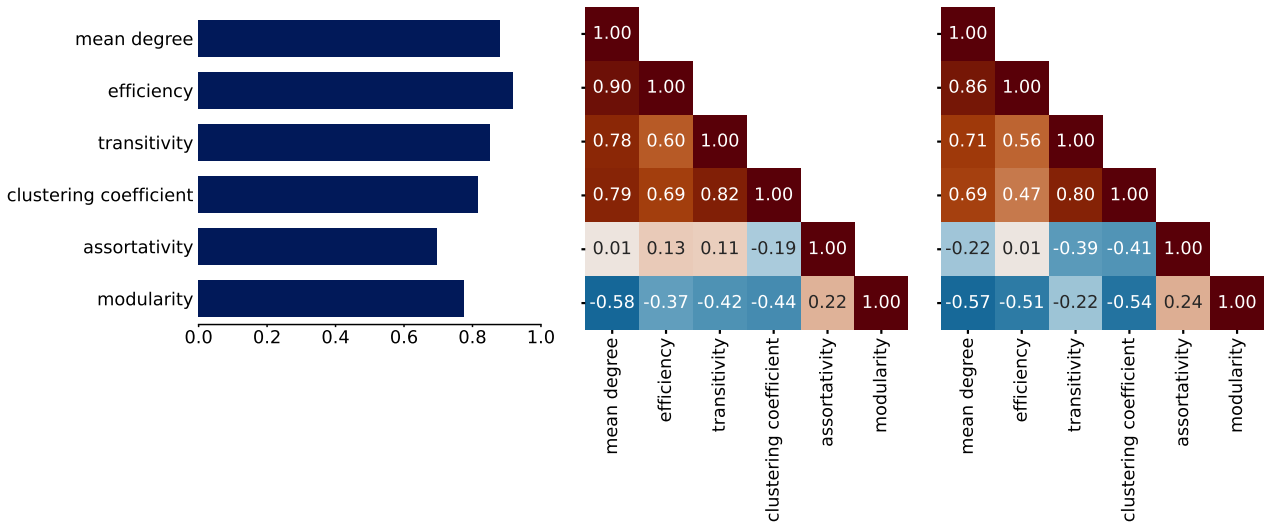


Figure 12: Left: R^2 scores of the linear regression models predicting each of the six graph metrics from the 32-dimensional latent embeddings, indicating the predictive strength of the latent space. Middle: Spearman's rank correlation coefficient matrix between the six graph metric descriptors, calculated directly from the test dataset. Right: Cosine distance matrix between the gradient directions in the 32-dimensional latent space for each of the six graph metrics, illustrating the similarity in how different metrics are encoded in the latent space.

Algorithm 1: Metropolis-Hastings Algorithm

Input:
  t: Target value;
  $\mathbf{z_0}$: Initial feasible point;
  Samples: The list of sampled points, initialized as an empty list;
  $\mathbf{\Sigma}, \mu$: The covariance matrix and mean vector of the fitted multivariate Gaussian distribution;
  $f(\mathbf{z})$: Linear regression function;
  w: Prior probability weight;
  σ: Proposal standard deviation;
  N: Total sample number;
  $burn$: burn in round;
  $thin$: Sampling interval.
Output: A list of sampled latent vectors: $\mathbf{z}_0, ..., \mathbf{z}_N$.

$c = \mathbf{x}_0$
Samples.$append(c)$
total iteration $= burn + N * thin$
for $i = 1$ **to** total iteration **do**
  Propose a new point c' by disturbing c a small step: $c' \sim \mathcal{N}(c, \sigma^2)$
  log acceptance ratio $=$ LTD(c') $-$ LTD(c)
  $r \sim \mathcal{U}(0, 1)$
  if log$r <$ log acceptance ratio **then** $c = c'$
  end if
  if $i \geq burn$ and $(i - burn)$ mod $thin == 0$ **then**
    Samples.$append(c)$
  end if
end for
return Samples

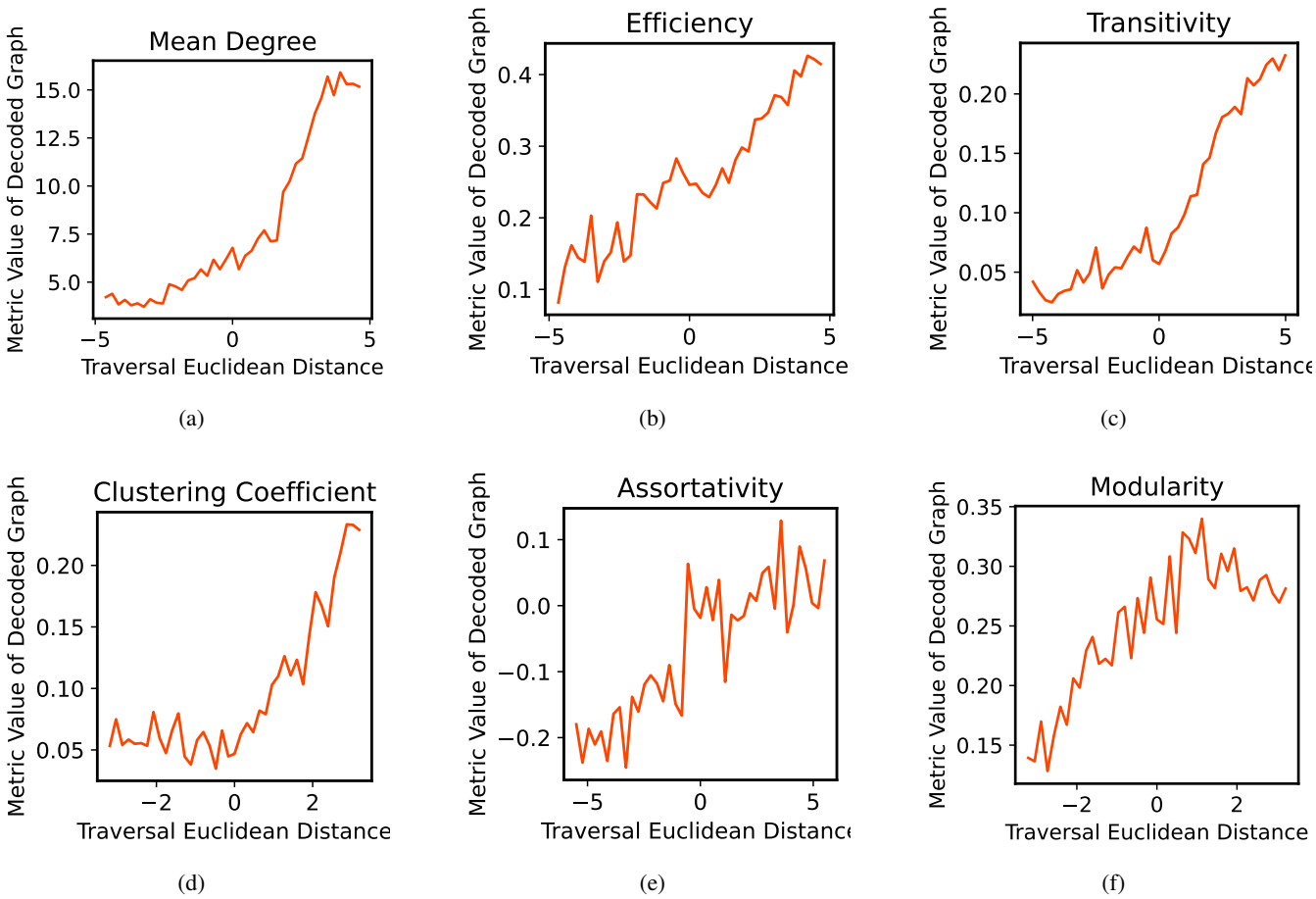


Figure 13: Metric variation of decoded graphs along the directions of different gradient vectors in the latent space.

was either generated by our Variational Autoencoder (VAE) to produce structured patterns or formulated as a density-matched random graph for control comparisons. The initial recurrent weights were assigned values of $+1$ (from an excitatory neuron), -1 (from an inhibitory neuron), or 0 (no connection), and the final matrix W was scaled to a specified spectral radius. Both W_{in} and W_{out} were initialized with weights from a uniform distribution $U[-1, 1)$ and subsequently trained end-to-end using the Adam optimizer. This design allowed us to investigate the influence of fixed, structured connectivity while permitting the network to adapt its input and output mappings for each task.

K.1 Copy Task

The copy memory task uses sequences of categorical inputs. Each input sequence has a total length of $T+2t$ and is structured as follows:

- The first t tokens are one-hot vectors randomly chosen from a set of N categories, representing the information to be memorized.
- These are followed by T "blank" tokens (category 0), which create a delay period where the network must hold the information in memory.
- A delimiter token (category $N + 1$) is then presented to signal the start of the recall phase.
- The sequence concludes with $t - 1$ final blank tokens.

The corresponding target sequence is of the same length, consisting of blank tokens except for the last t positions, which must reproduce the initial random sequence from the input. For our experiments with a 100-neuron reservoir network, we set the task parameters to $t = 5$, $N = 3$, and $T = 10$.

The networks were trained for 60,000 epochs with a learning rate of 0.001. For the reservoir parameters, we configured the leaking rate to 0.3 and the spectral radius to 0.999.

Figures 24 and 25 compare the copy task performance of reservoir connectivity built from VAE-generated graphs (left column) versus density-matched random graphs (right column). The VAE graphs were generated by targeting the 50th percentile for various structural metrics.

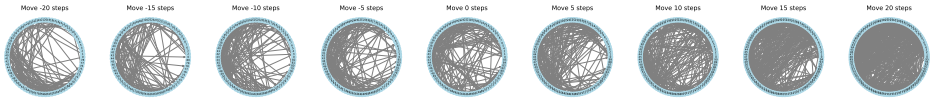
K.2 Classification Task

In the sequential MNIST (sMNIST) task, each image from the MNIST dataset is presented to the reservoir network row-by-row or column-by-column. The objective is to classify the digit based on the resulting hidden states of the reser-

(a) Adjacency matrix along gradient of mean degree

(b) Network structure along gradient of mean degree

(c) Adjacency matrix along gradient of efficiency

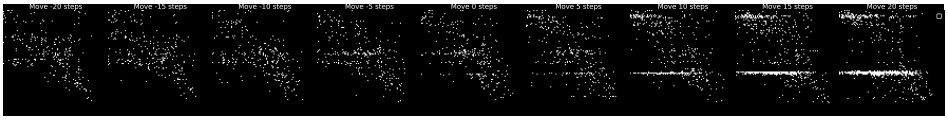


(d) Network structure along gradient of efficiency

(e) Adjacency matrix along gradient of transitivity

(f) Network structure along gradient of transitivity

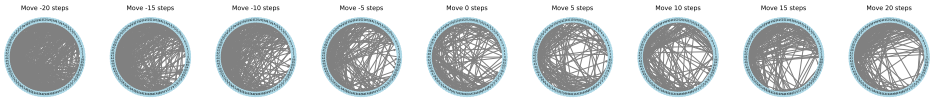
Figure 14: Generated samples by traversing the latent space along the gradient direction of different metrics (Part 1). Images show decoded graphs as the latent vector moves from a center point (middle) towards the positive (right) and negative (left) gradient directions. Each number indicates the Euclidean distance from the center point of the dataset.



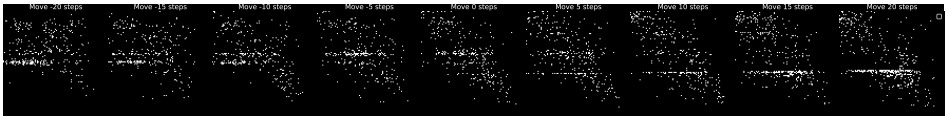
(a) Adjacency matrix along gradient of clustering coefficient

(b) Network structure along gradient of clustering coefficient

(c) Adjacency matrix along gradient of assortativity



(d) Network structure along gradient of assortativity



(e) Adjacency matrix along gradient of modularity

(f) Network structure along gradient of modularity

Figure 15: Generated samples by traversing the latent space along the gradient direction of different metrics (Part 2). Images show decoded graphs as the latent vector moves from a center point (middle) towards the positive (right) and negative (left) gradient directions. Each number indicates the Euclidean distance from the center point of the dataset.

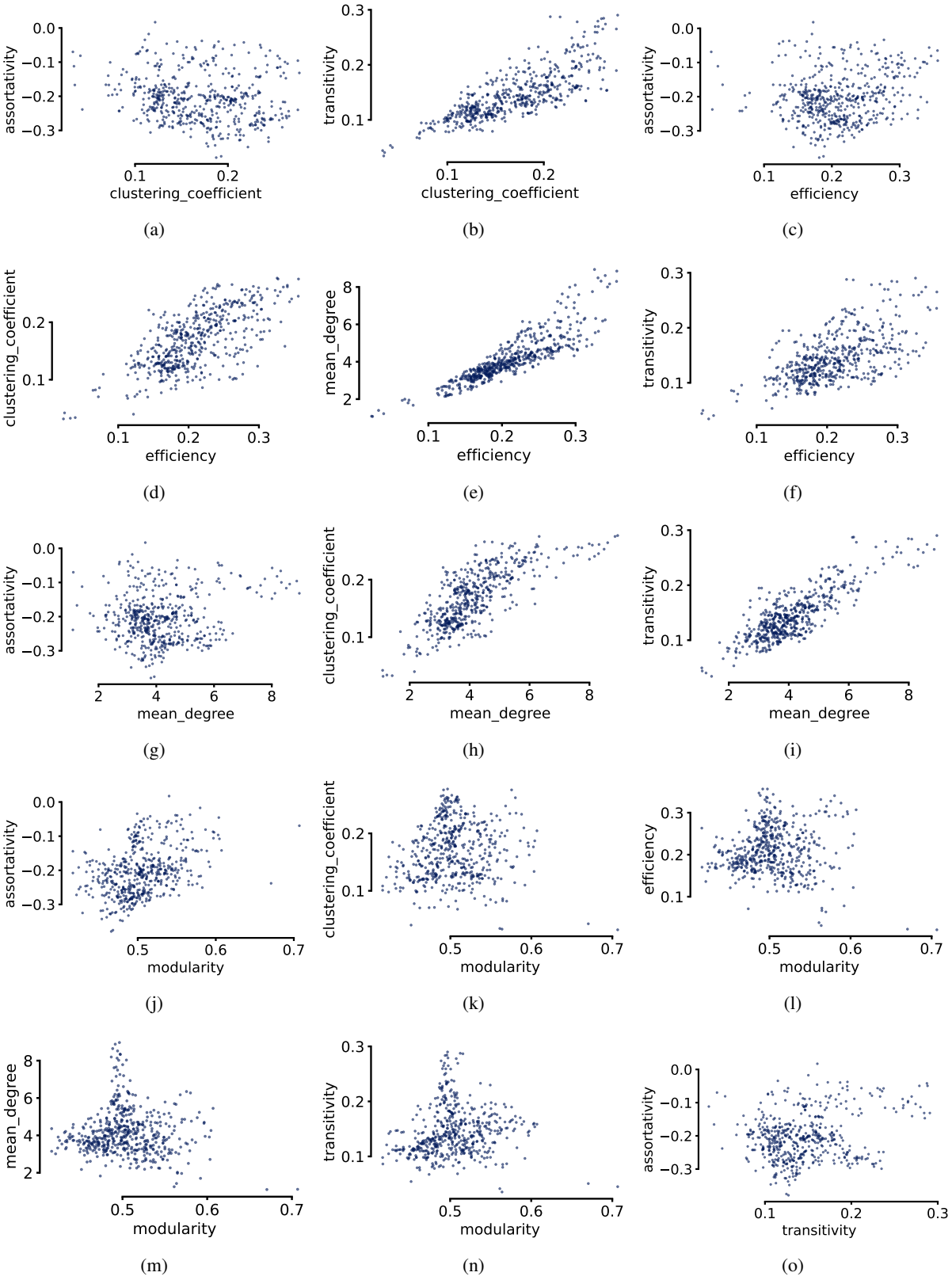


Figure 16: Correlation between Pairs of Graph Metrics

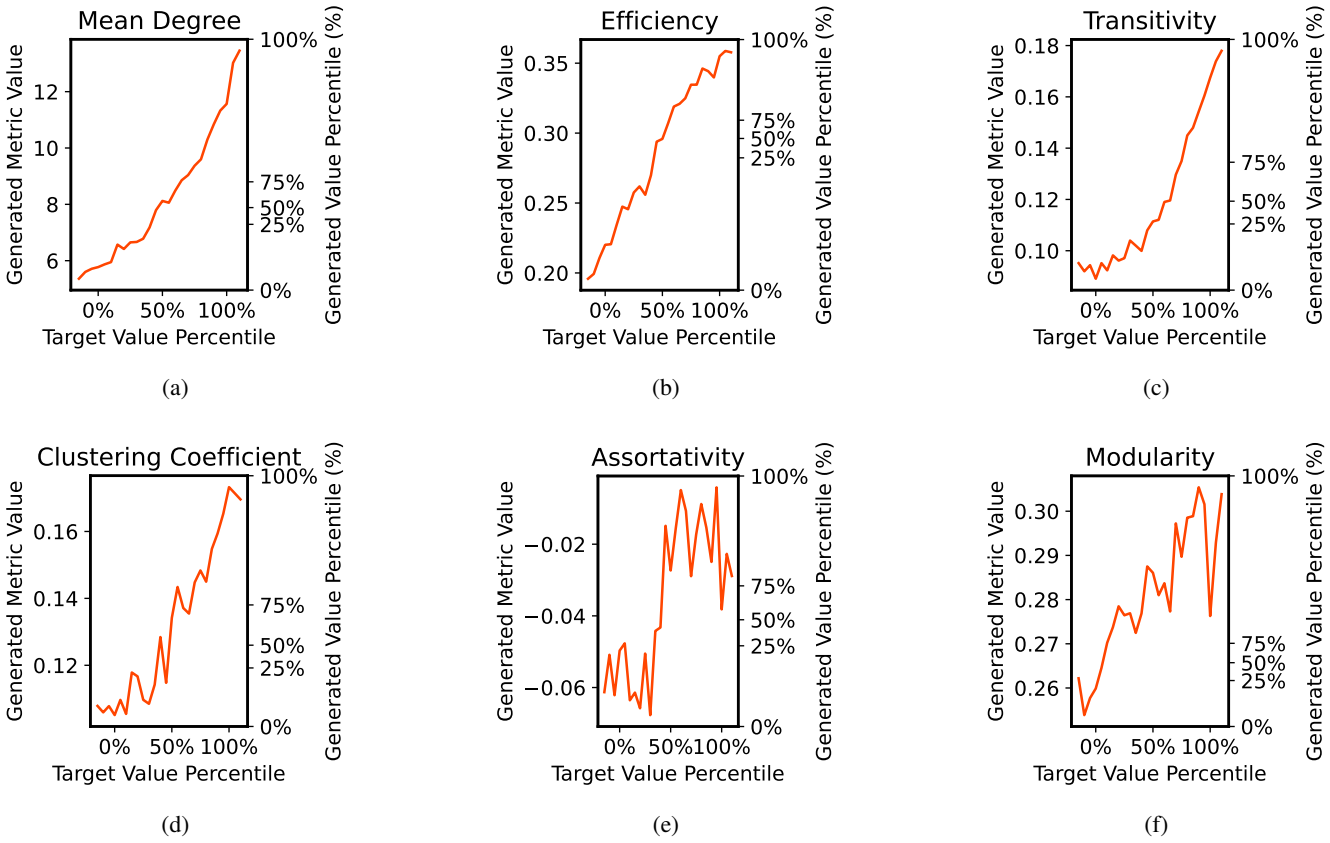
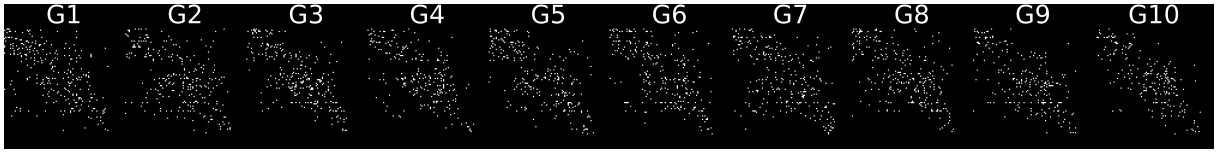
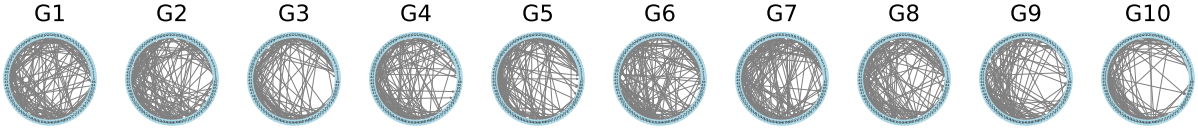


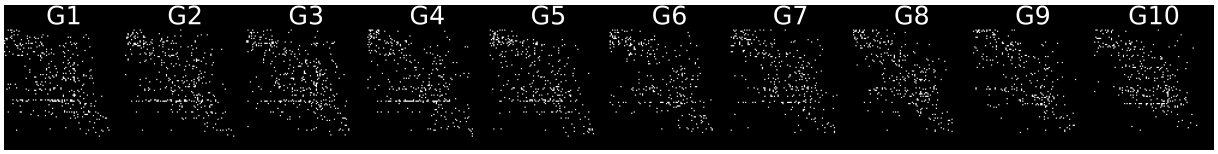
Figure 17: Metrics of generated graphs when setting different targets.



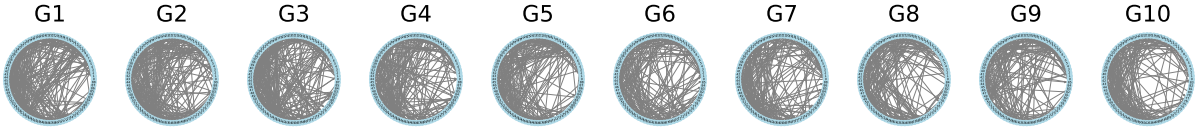
(a) Target mean degree: 0%-5% percentile, Adjacency Matrix



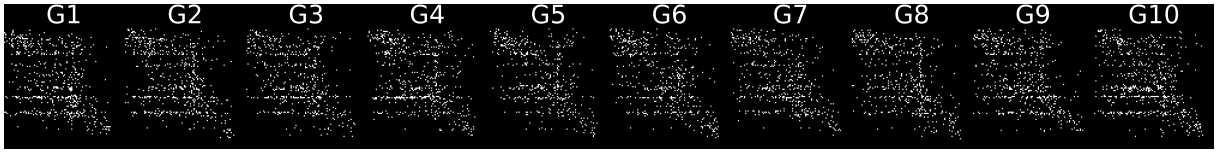
(b) Target mean degree: 0%-5% percentile, Network Visualization



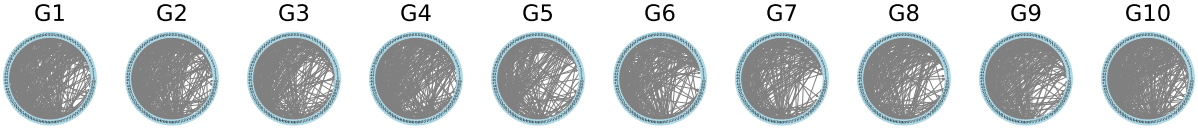
(c) Target mean degree: 50%-55% percentile, Adjacency Matrix



(d) Target mean degree: 50%-55% percentile, Network Visualization

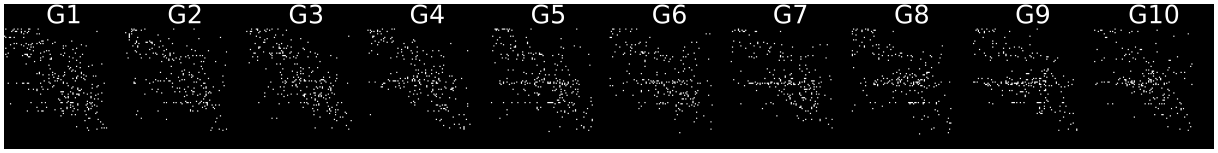


(e) Target mean degree: 95%-100% percentile, Adjacency Matrix

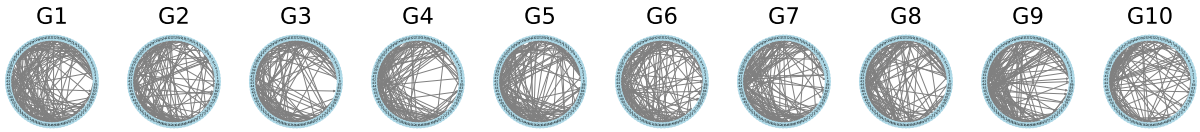


(f) Target mean degree: 95%-100% percentile, Network Visualization

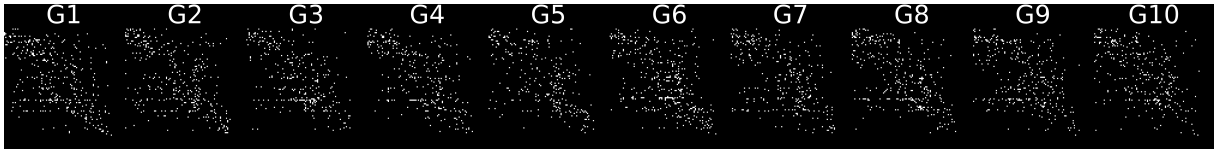
Figure 18: Generated graph examples targeting different mean degree percentile ranges. For each target range, 10 graphs were randomly selected from 1000 graphs sampled and decoded. Pairs of adjacency matrices and corresponding network visualizations are shown for target mean degrees in the 0%-5% (a, b), 50%-55% (c, d), and 95%-100% (e, f) percentile ranges of the dataset.



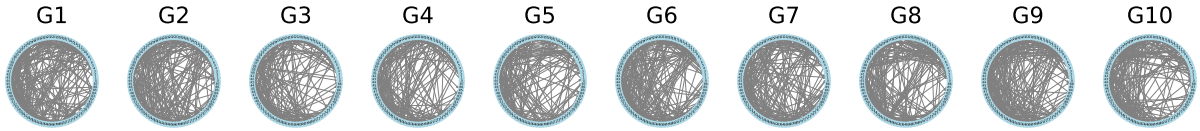
(a) Target efficiency: 0%-5% percentile, Adjacency Matrix



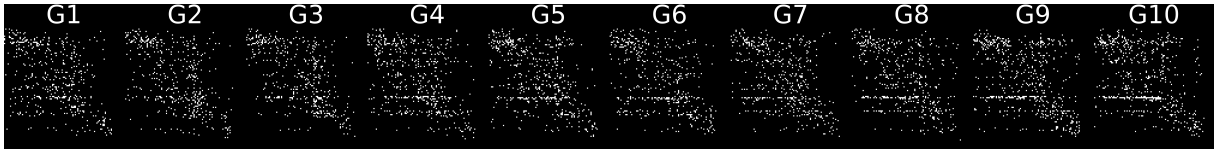
(b) Target efficiency: 0%-5% percentile, Network Visualization



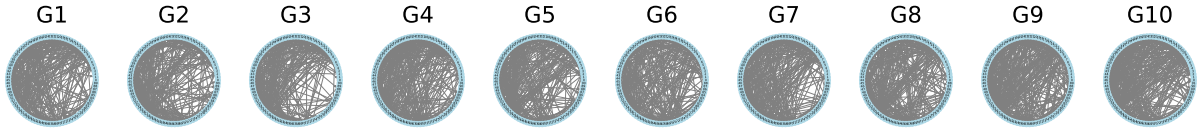
(c) Target efficiency: 50%-55% percentile, Adjacency Matrix



(d) Target efficiency 50%-55% percentile, Network Visualization

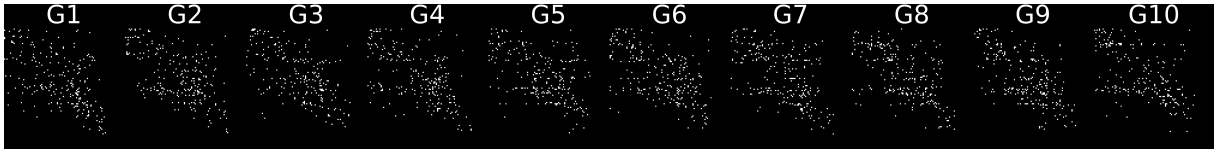


(e) Target efficiency: 95%-100% percentile, Adjacency Matrix

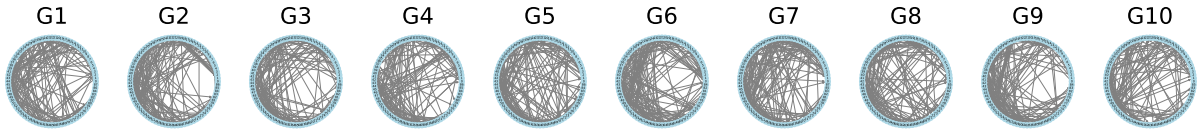


(f) Target efficiency: 95%-100% percentile, Network Visualization

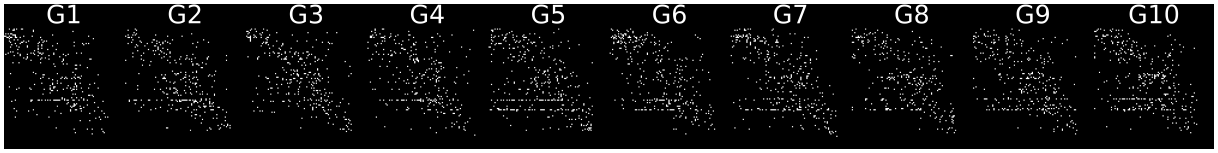
Figure 19: Generated graph examples targeting different efficiency percentile ranges. For each target range, 10 graphs were randomly selected from 1000 graphs sampled and decoded. Pairs of adjacency matrices and corresponding network visualizations are shown for target efficiency in the 0%-5% (a, b), 50%-55% (c, d), and 95%-100% (e, f) percentile ranges of the dataset.



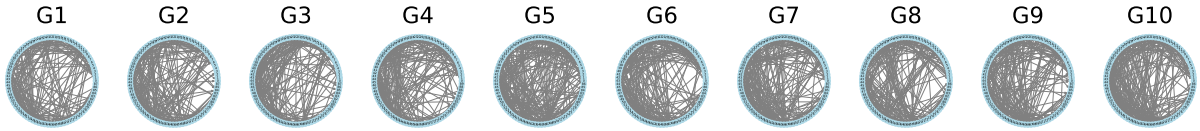
(a) Target transitivity: 0%-5% percentile, Adjacency Matrix



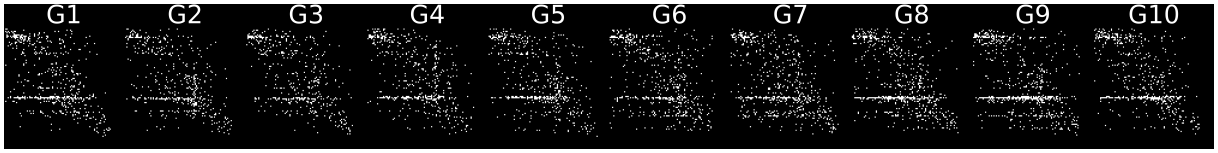
(b) Target transitivity: 0%-5% percentile, Network Visualization



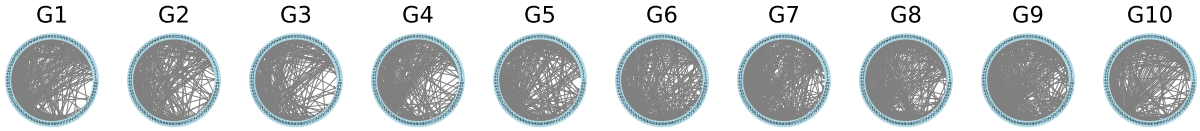
(c) Target transitivity: 50%-55% percentile, Adjacency Matrix



(d) Target transitivity 50%-55% percentile, Network Visualization

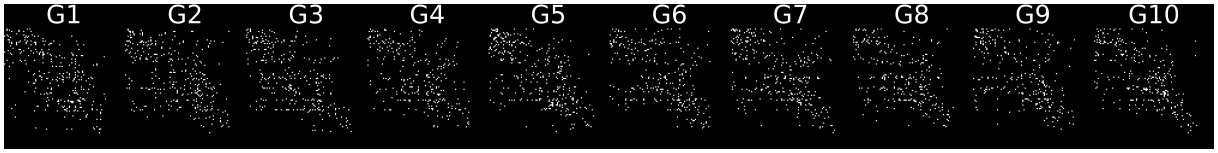


(e) Target transitivity: 95%-100% percentile, Adjacency Matrix

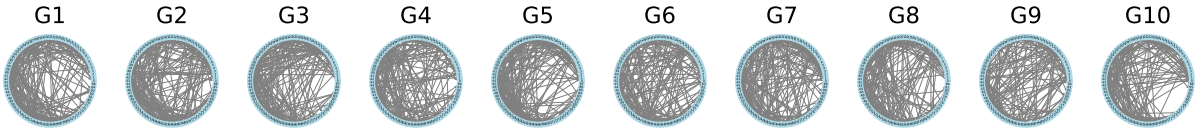


(f) Target transitivity: 95%-100% percentile, Network Visualization

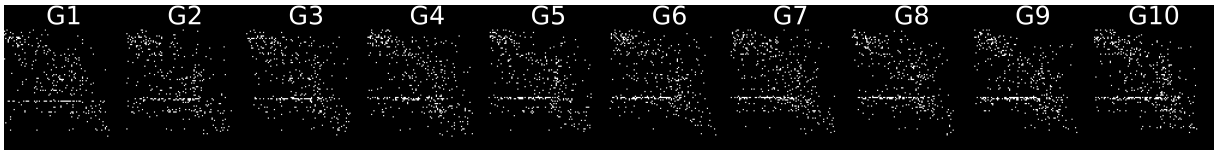
Figure 20: Generated graph examples targeting different transitivity percentile ranges. For each target range, 10 graphs were randomly selected from 1000 graphs sampled and decoded. Pairs of adjacency matrices and corresponding network visualizations are shown for target transitivity in the 0%-5% (a, b), 50%-55% (c, d), and 95%-100% (e, f) percentile ranges of the dataset.



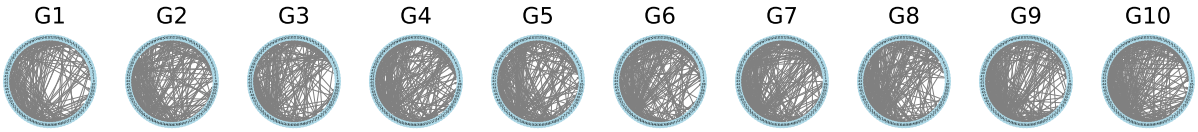
(a) Target clustering coefficient: 0%-5% percentile, Adjacency Matrix



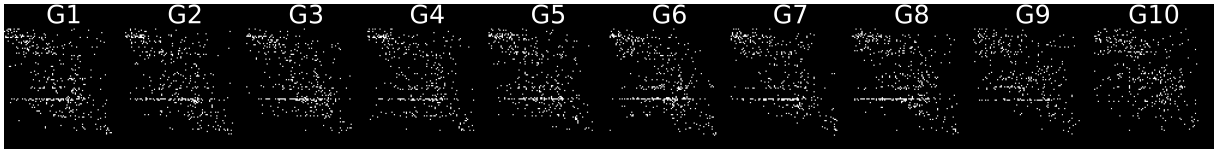
(b) Target clustering coefficient: 0%-5% percentile, Network Visualization



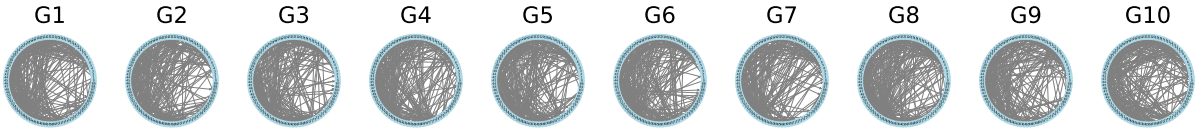
(c) Target clustering coefficient: 50%-55% percentile, Adjacency Matrix



(d) Target clustering coefficient 50%-55% percentile, Network Visualization

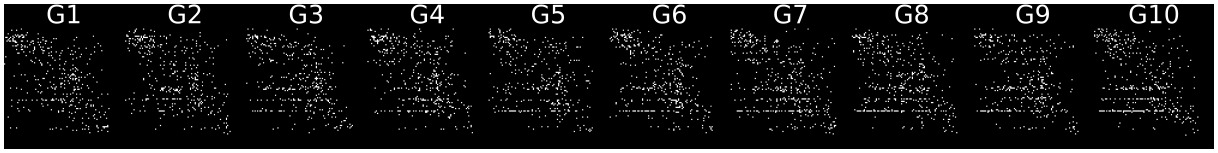


(e) Target clustering coefficient: 95%-100% percentile, Adjacency Matrix

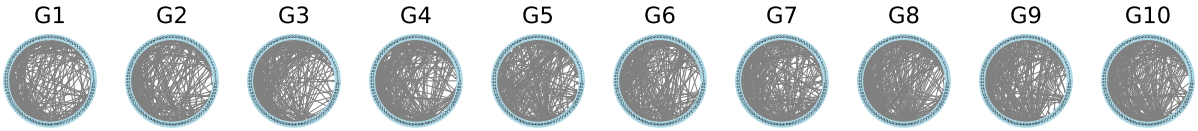


(f) Target clustering coefficient: 95%-100% percentile, Network Visualization

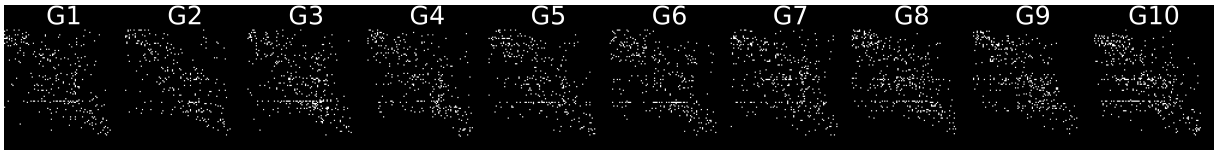
Figure 21: Generated graph examples targeting different clustering coefficient percentile ranges. For each target range, 10 graphs were randomly selected from 1000 graphs sampled and decoded. Pairs of adjacency matrices and corresponding network visualizations are shown for target clustering coefficient in the 0%-5% (a, b), 50%-55% (c, d), and 95%-100% (e, f) percentile ranges of the dataset.



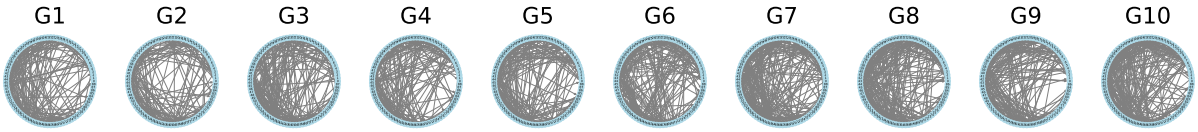
(a) Target assortativity: 0%-5% percentile, Adjacency Matrix



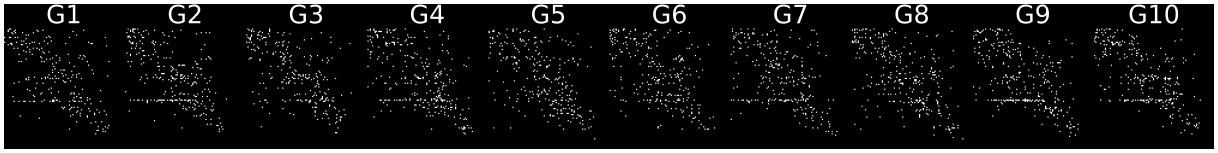
(b) Target assortativity: 0%-5% percentile, Network Visualization



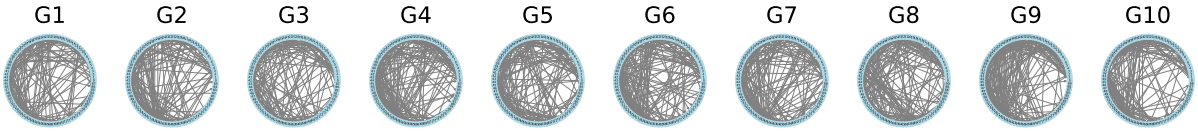
(c) Target assortativity: 50%-55% percentile, Adjacency Matrix



(d) Target assortativity 50%-55% percentile, Network Visualization

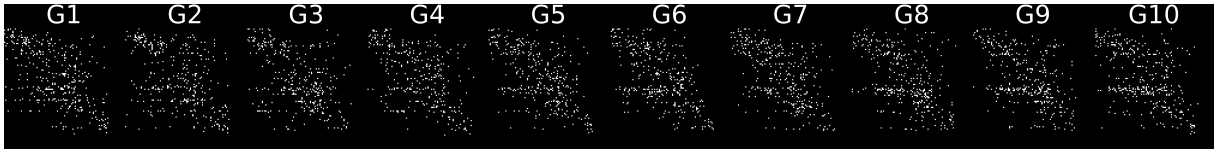


(e) Target assortativity: 95%-100% percentile, Adjacency Matrix

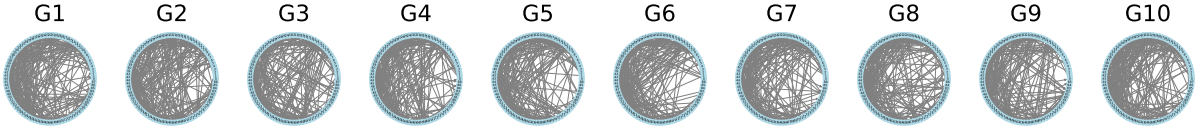


(f) Target assortativity: 95%-100% percentile, Network Visualization

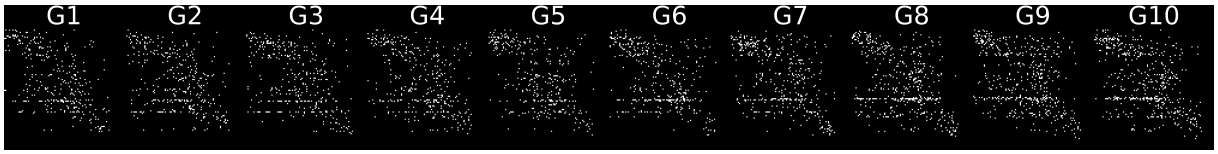
Figure 22: Generated graph examples targeting different assortativity percentile ranges. For each target range, 10 graphs were randomly selected from 1000 graphs sampled and decoded. Pairs of adjacency matrices and corresponding network visualizations are shown for target assortativity in the 0%-5% (a, b), 50%-55% (c, d), and 95%-100% (e, f) percentile ranges of the dataset.



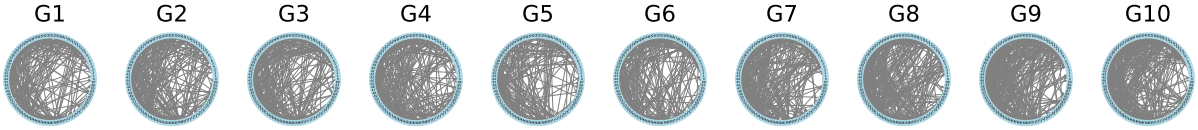
(a) Target modularity: 0%-5% percentile, Adjacency Matrix



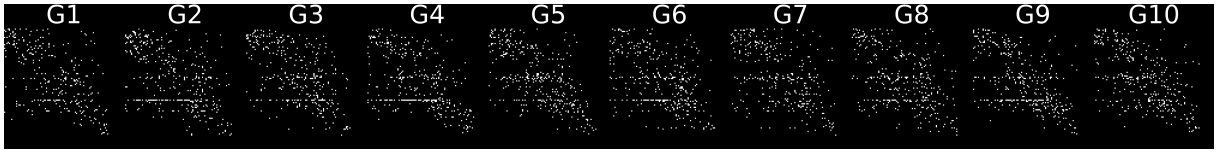
(b) Target modularity: 0%-5% percentile, Network Visualization



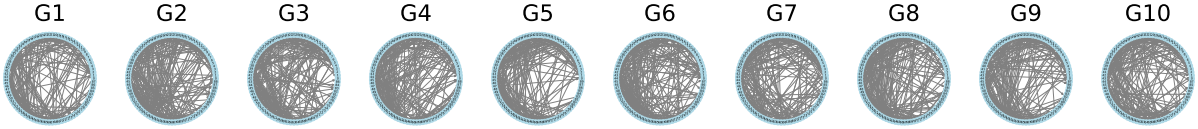
(c) Target modularity: 50%-55% percentile, Adjacency Matrix



(d) Target modularity 50%-55% percentile, Network Visualization

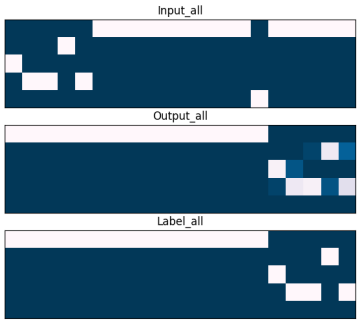


(e) Target modularity: 95%-100% percentile, Adjacency Matrix

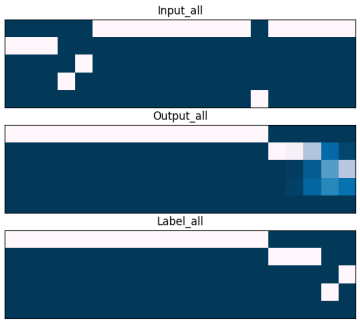


(f) Target modularity: 95%-100% percentile, Network Visualization

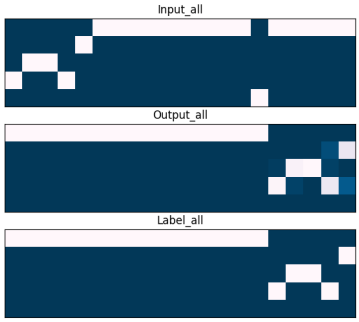
Figure 23: Generated graph examples targeting different modularity percentile ranges. For each target range, 10 graphs were randomly selected from 1000 graphs sampled and decoded. Pairs of adjacency matrices and corresponding network visualizations are shown for target modularity in the 0%-5% (a, b), 50%-55% (c, d), and 95%-100% (e, f) percentile ranges of the dataset.



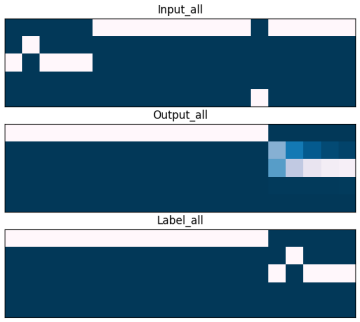
(a) VAE-Generated Network (Target: Mean Degree)



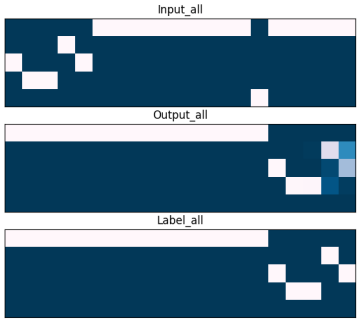
(b) Random Network (Target: Mean Degree)



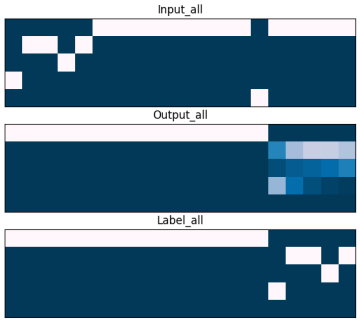
(c) VAE-Generated Network (Target: Efficiency)



(d) Random Network (Target: Efficiency)

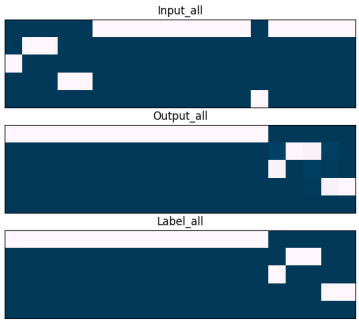


(e) VAE-Generated Network (Target: Transitivity)

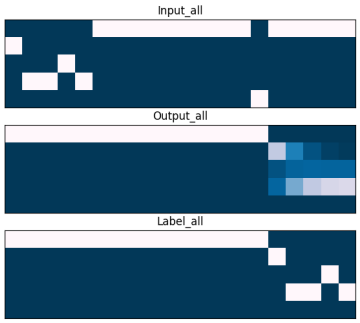


(f) Random Network (Target: Transitivity)

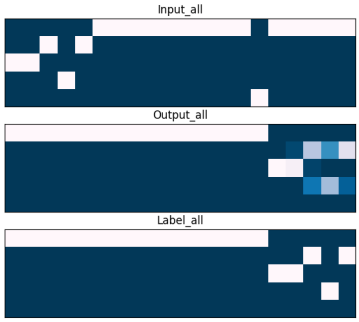
Figure 24: Performance comparison of reservoir networks on the copy task using VAE-generated graphs (left column) versus density-matched random graphs (right column) for three target metrics: Mean Degree, Efficiency, and Transitivity. The comparison is continued in Figure 25. Each subplot visualizes the input pattern to be memorized (top), the model's output (middle), and the ground truth (bottom).



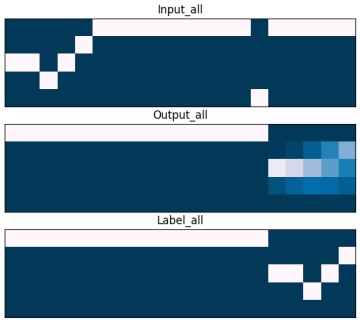
(a) VAE-Generated Network (Target: Clustering Coefficient)



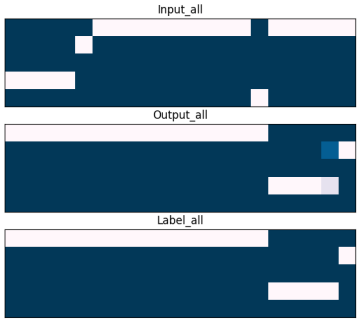
(b) Random Network (Target: Clustering Coefficient)



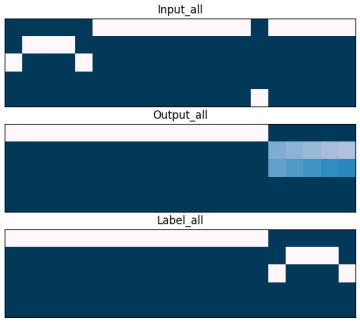
(c) VAE-Generated Network (Target: Assortativity)



(d) Random Network (Target: Assortativity)



(e) VAE-Generated Network (Target: Modularity)



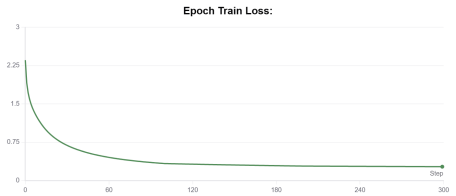
(f) Random Network (Target: Modularity)

Figure 25: Performance comparison of reservoir networks on the copy task using VAE-generated graphs (left column) versus density-matched random graphs (right column). This figure shows the results for the last three target metrics: Clustering Coefficient, Assortativity, and Modularity. Each subplot visualizes the input pattern to be memorized (top), the model's output (middle), and the ground truth (bottom).

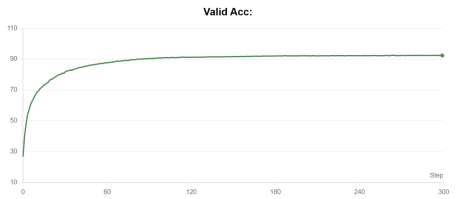
voir network.

The networks were trained for 300 epochs with an initial learning rate of 0.003, which decayed by a factor of 1/3 every 100 epochs. For the reservoir parameters, we configured the leaking rate to 0.3 and the spectral radius to 0.999.

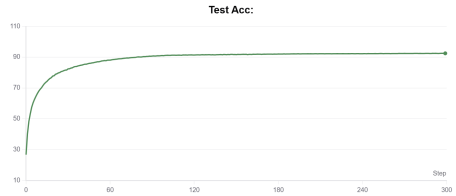
Figure 26 shows example training and testing curves for the sMNIST task, where the objective is the 25th percentile of the transitivity metric.



(a) Train Loss



(b) Valid Accuracy



(c) Test Accuracy

Figure 26: Example training and testing curves for the sMNIST task (Target metric: Transitivity, Value: 25th percentile).

L Another Example of Our Research Paradigm: Studies on FlyWire Dataset

To validate our approach on more diverse data, we conducted analogous experiments on the FlyWire (Dorkenwald et al. 2022) dataset, a Drosophila (fruit fly) connectome. In this configuration, the graph global encoder and the node feature decoder within the VAE were substituted with Multilayer Perceptrons (MLPs). The node types are considered by grouping neurons by their neurotransmitter types. The evaluation of the generation results is presented in Table 3.

The model shows particularly strong performance in clustering coefficient, surpassing all comparative methods. This metric reflects the tightness of local circuit connections, and its high fidelity demonstrates our framework's effectiveness in capturing the modular properties of columnar functional units in biological neural networks. In contrast, the reconstruction accuracy for orbit counts is relatively lower, which aligns with expectations: global topological features, being influenced by more stochastic factors, inevitably suffer greater information loss during low-dimensional compression.

Method	FlyWire			
	Deg.↓	Clus.↓	Orbit↓	Avg.↓
EDGE	0.009	0.099	0.038	**0.048**
DisCo	0.141	0.552	0.377	0.356
GDSS	0.054	0.633	0.702	0.463
GruM	**0.002**	0.165	**0.035**	0.067
Ours	**0.002**	**0.056**	0.129	0.062

Table 3: Generation results on FlyWire. We compute the Maximum Mean Discrepancy (MMD) between generated and test graphs for three graph features, with the best results shown in bold.

In Figure 27, we present the generation results of our model alongside comparative models on the FlyWire dataset. The results demonstrate that our model exhibits superior performance in generation diversity compared to baseline methods.

This performance differentiation reveals intrinsic characteristics of our method: while the information bottleneck causes slight distortion in global topology, it effectively filters noise and makes key biological patterns (e.g., local clustering) more salient in the latent space. Notably, although EDGE achieves the lowest average MMD, its high-dimensional representation lacks explicit structural constraints, making subsequent causal analysis substantially more challenging to implement. Our method, through careful balancing of reconstruction accuracy and dimensional compression, establishes interpretable mappings between latent variables and generated graph structural features, providing a solid foundation for analyzing brain connectomes.

In addition to this, we also observe the reconstruction performance of our model on FlyWire.

The reconstruction results in Table 4 demonstrate our model's ability to reasonably recover both node categories and edge connectivity, though with some expected information loss. We achieve 0.982 accuracy in node classification and 0.978 edge reconstruction accuracy, with slightly lower performance in edge AUC (0.959), suggesting the model captures major structural patterns while missing some finer connection details. These metrics reflect the inherent trade-offs of our approach - while the information bottleneck causes some reconstruction imperfection, it enables the low-dimensional analysis that is central to our framework. The performance is encouraging given this constraint, though we recognize these results would benefit from comparison with standard reconstruction baselines in future work. The modest differences between node and edge metrics may represent the necessary compromise between reconstruction fidelity and analytical tractability that defines our method's design philosophy.

Figure 28 demonstrates the model's graph reconstruction

(a) Origin (b) Ours (c) EDGE (d) GruM (e) GDSS (f) DisCo

Figure 27: Generation results on FlyWire in Adjacency Matrix Format. (a) Original: Shows the sampled graphs from the Drosophila brain connectome with three distinct divergence levels. (b) Ours: Our model's generation results, also exhibiting three divergence levels. (c)-(d) EDGE and GruM results: While performing well on high-divergence graphs, they lack medium and low-divergence samples, indicating limited diversity. (e)-(f) GDSS and DisCo results: Their main limitations are insufficient edge generation and similarly constrained diversity.

Method	**FlyWire**			
	Node Acc.↑	**Node F1↑**	**Edge Acc.↑**	**Edge AUC↑**
Ours	0.982	0.979	0.978	0.959

Table 4: Reconstruction performanc on FlyWire. We evaluate (1) Edge reconstruction performance using AUC and accuracy between original and reconstructed adjacency matrices, and (2) Node category recovery using classification accuracy and F1-score comparing original versus predicted node categories.

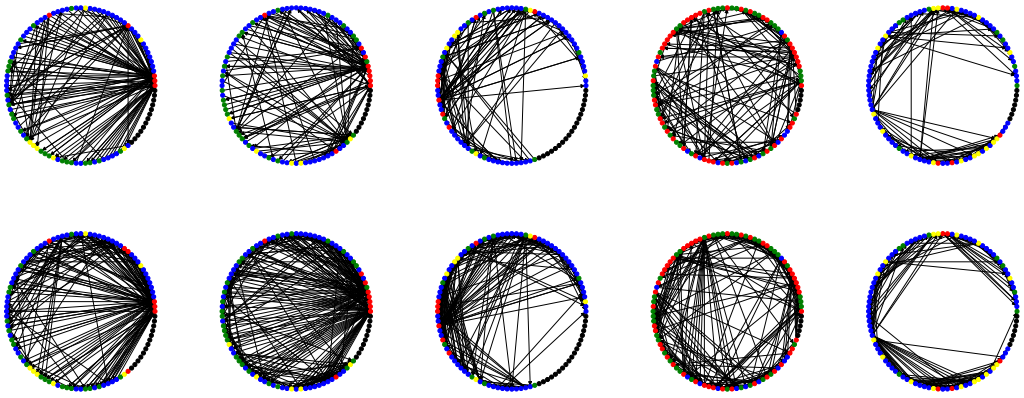


Figure 28: Graph reconstruction results: Original samples (top) and their reconstructed counterparts (bottom), with node colors representing type categories.

capability by comparing original connectome samples (top row) with their reconstructed counterparts (bottom row). The visual comparison shows that the model successfully preserves the overall topological structure and node type distributions (represented by consistent color patterns), while some subtle differences in local connectivity can be observed upon closer inspection. The reconstructions maintain the characteristic clustering patterns of neural circuits, though with minor variations in edge density and connection specificity that reflect the expected information loss through our model's bottleneck architecture. These results align quantitatively with the reconstruction metrics reported in Table 4, providing visual confirmation that while the model captures essential connectome features, perfect reconstruction remains challenging due to our framework's emphasis on dimensional reduction and interpretability over absolute fidelity. The preserved node category assignments (colors) are particularly noteworthy, suggesting the model reliably maintains neuron type information during encoding-decoding.