A. Appendix

A.1. Variance of the Gradient Estimators

We use the score function estimator (Williams, 1992) to obtain the gradients. In some applications, this estimator may suffer from high variance and make the training process unstable. Here, we study the variance of the score function estimator to make sure that it does not cause optimization issues in our application. To show the behavior of the optimizer, we plot the objective (the ELBO) in Figure 4(right) and the variance of the gradient estimator in Figure 4(left); both against training epochs. We can see that the objective decreases smoothly throughout optimization, indicating that the algorithm is stable. The three curves in the left plot show the variance of the gradients for different number of Monte Carlo samples; as expected, the variance decreases as the number of samples increases. Moreover, the variance from a relatively small sample size ($S = 8$) is already decently low. This is because the variational distribution $q_\phi(\pi|G)$ tends to concentrate its probability mass to a small number of node orders, which can be seen from our analysis of the variational distribution (Figure 3 and Figure 5). Considering the tradeoff between computation time and variance, we set $S = 8$ in all our experiments.

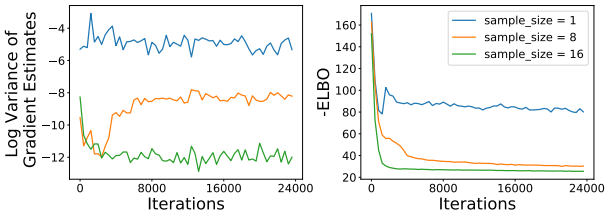

Figure 4. (Left) Training objective (ELBO) against epochs for GraphGEN on the Community-small dataset. The objective decreases smoothly throughout optimization. (Right) Log-variance of the score function gradient estimator for different number of Monte Carlo samples. Using $S = 8$ samples is enough to estimate the gradient.

A.2. Proof of Lemma 1

Lemma 1. *Let $G[V\backslash\{u\}]$ and $G[V\backslash\{v\}]$ respectively denote the subgraphs induced by $V\backslash\{u\}$ and $V\backslash\{v\}$, then u and v are in the same orbit if and only if $G[V\backslash\{u\}]$ and $G[V\backslash\{v\}]$ are isomorphic.*

Proof. Let '$\equiv$' denote the isomorphic relation. Also denote $E\backslash\{u\} = \{(i,j) \in E : i \neq u, j \neq u\}$ as the subset of edges that do not incident u.

We first show the first direction: "u and v being in the same orbit" indicates "$G[V\backslash\{u\}] \equiv G[V\backslash\{v\}]$". If u and v are in the same orbit, then $\exists f \in \text{Auto}(G) : f(v) = u$. Then $\forall i, j \in V\backslash\{u\}$, $(i,j) \in E\backslash\{u\} \iff (f(i), f(j)) \in E\backslash\{v\}$ because f is an automorphism. Then we restrict f to $V\backslash\{u\}$ and get a injection $f' : V\backslash\{u\} \to V\backslash\{v\}$, and $f'(i) = f(i)\ \forall i \in V\backslash\{u\}$. Then $\forall i, j \in V\backslash\{u\}$, $(i,j) \in E\backslash\{u\} \iff (f'(i), f'(j)) \in E\backslash\{v\}$. Therefore, f' is an isomorphism between $G[V\backslash\{u\}]$ and $G[V\backslash\{v\}]$.

We then prove by induction the second direction: "$G[V\backslash\{u\}] \equiv G[V\backslash\{v\}]$" indicates that "$u$ and v being in the same orbit" .

In the base case, we consider graphs with two nodes. Let G be $(V = \{u, v\}, E = \emptyset)$ or $(V = \{u, v\}, E = (u, v))$. In either case, we always have $G\backslash\{u\} \equiv G\backslash\{u\}$. The two nodes u and v are also in the same orbit in both cases. So the second direction holds in the base case.

Then in the induction step, we assume the second direction is true for any graph of size n, then we show that it is also true for a graph of size $n + 1$. Let $f \in \text{Auto}(G)$, and $f(u) = u'$. There are three cases: u' is v, u' is u, or u' is neither of them. If it is the first case, then we have the conclusion directly: u and v are in the same orbit.

Then we check the third case. With the same argument in the proof of the first direction, we restrict f to $V\backslash\{u\}$ and get an isomorphism: $G\backslash\{u\} \equiv G\backslash\{u'\}$. By the condition $G\backslash\{u\} \equiv G\backslash\{v\}$, we also have $G\backslash\{u'\} \equiv G\backslash\{v\}$. We then remove u from both graphs and get $G\backslash\{u, u'\} \equiv G\backslash\{u, v\}$. With the induction rule, we have that u' and v in the same orbit in $G\backslash\{u\}$. Let $g(\cdot) \in \text{Auto}(G\backslash\{u\})$ and $g(u') = v$. We extend $g(\cdot)$ to V and let $g(u) = u$, then $g \circ f$ creates an automorphism on G, and $(g \circ f)(u) = v$. Therefore, u is in the same orbit as v.

Finally, we show how to construct an $f(\cdot)$ such that $f(u) = u' \neq u$. Since $G[V\backslash\{v\}] \equiv G[V\backslash\{u\}]$, there is a isomorphism $h : V\backslash\{v\} \to V\backslash\{u\}$, and $h(u) = u'$. Note that u' cannot be u because u is not in the range of h. We extend h to the domain V and let $h(v) = u$, so h is a permutation of V. For any $i, j \in E\backslash\{v\}$, $(i,j) \in E\backslash\{v\} \iff (h(i), h(j)) \in E\backslash\{u\}$ because h is an isomorphism. It is also true that $(i,j) \in E \iff (h(i), h(j)) \in E$ because (i,j) does not incident v, and $(f(i), f(j))$ does not incident u. Since h is a permutation, the composition of h forms a group: $\{h^0, h^1, \ldots, h^K\}$. The inverse h^{-1} is the same as h^K. Let $j \in V\backslash\{v\}$, and $j = h^{-1}(i)$, $i \in V\backslash\{u\}$, then $(h(v), h(j)) = (u, i)$. With the previous argument, $(u,i) \in E \iff (h(u), h(i)) \in E$. By the compisition rule, we further have $(h(u), h(i)) \in E \iff \ldots \iff (h^K(u), h^K(i)) = (v, j) \in E$. This works for any $j \in V\backslash v$, that is, $\forall j \in V\backslash\{v\}$, $(v,j) \in E \implies (h(v), h(j)) \in E$, then h is a non-trivial automorphism on G and $h(u) \neq u'$. □

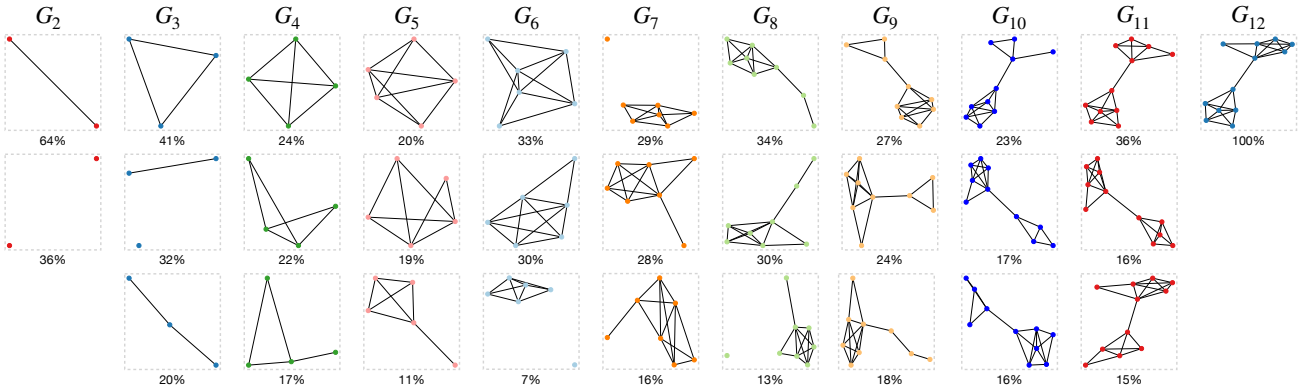

Figure 5. Graph generative sequences sampled from $q_\phi(\pi|G)$ for a graph from Community-small. The variational distribution prefers sequences of connected graphs. Similary to the distribution indicated in Figure 3 (bottom left), it first generates a comunity and then adds another one.

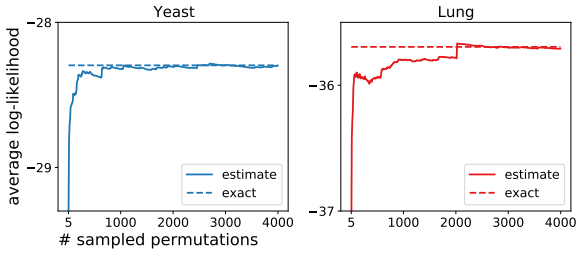

Figure 6. Comparison of estimated log-likelihood and exact log-likelihood for small graphs (fewer than 10 nodes) in the Lung and Yeast test sets. The estimation from $S = 2200$ importance samples is very accurate.

A.3. The Accuracy of the Log-Likelihood Estimation

To make sure we give an accurate estimation of the log-likelihood, we compare the estimated log-likelihood using different number of importance samples against the true log-likelihood. We compute the true log-likelihood of a graph by enumerating all possible permutations. We conduct the experiment on two datasets, Yeast and Lung. Since the calculation of the true log-likelihood is only feasible on small graphs, we keep graphs with fewer than 10 nodes in each of the two datasets. We use GraphRNN trained by variational inference as the model here, and the proposal distribution is the learned $q_\phi(\pi|G)$. Figure 6 shows the results on the two datasets. We see that when the number of samples is over 1000, the gap between the true log-likelihood and the estimated log-likelihood becomes very small (less than 0.1). When we increase the number of samples to 2200, the estimation is very accurate for both datasets. We conclude that the importance sampling estimator can be reliably used for model selection and model comparison.

Algorithm 1 VI algorithm for training a graph model based on the adjacency matrix $\mathbf{A}$

Input: Dataset of graphs $\mathbb{G} = \{G_1, \ldots, G_n\}$, model p_θ, variational distribution q_ϕ, sample size S, transition function $\gamma_a(\cdot)$
Output: Learned parameters θ and ϕ
repeat
  for $G \in \mathbb{G}$ **do**
    Sample $\pi^{(1)}, \ldots, \pi^{(S)} \overset{\text{iid}}{\sim} q_\phi(\pi|G)$
    Obtain $\mathbf{A}^{(s)} = \gamma_a(G, \pi^{(s)})$
    Set $p_\theta(G, \pi^{(s)}) = \frac{1}{|\Pi[\mathbf{A}^{(s)}]|} p_\theta(\mathbf{A}^{(s)})$
    Compute $\nabla_\phi \leftarrow \nabla_\phi L(\theta, \phi, G)$
    Compute $\nabla_\theta \leftarrow \nabla_\theta L(\theta, \phi, G)$
    Update ϕ, θ using the gradients ∇_ϕ, ∇_θ
  end for
until convergence of the parameters (θ, ϕ)

A.4. Training Algorithm

We present the training procedure in Algorithm 1. We optimize the parameters using the score function estimator, which is a standard choice in VI for non-conjugate models. Algorithm 1 can be applied to many autoregressive models operating with the adjacency matrix $\mathbf{A}$, such as GraphRNN and GraphGEN. For models that operate with the graph sequence instead, such as DeepGMG, we only need to replace the surjection γ_a with γ_s to obtain $G^{(s)}_{1:n}$ from each $(G, \pi^{(s)})$. In this case, $p_\theta(G, \pi^{(s)}) = \frac{1}{|\Pi[G^{(s)}_{1:n}]|} p_\theta(G^{(s)}_{1:n})$.

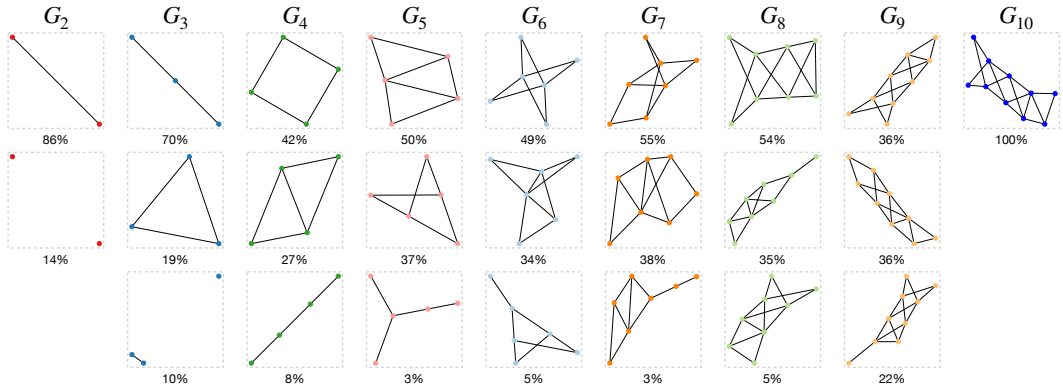

Figure 7. Graph generative sequences sampled from $q_\phi(\pi|G)$ for a graph from Enzymes. The variational distribution has a strong preference for sequences of connected graphs.

A.5. Graph Sequence Pattern in DeepGMG

In Section 4, we have investigated the variational distribution when training GraphRNNs. Here we study the variational distribution when training DeepGMG. For this experiment, we also consider the Community-small and the Enzymes dataset in order to show how our model learns a set of preferred orders. We choose the smallest graph from each dataset (a graph with 12 nodes for Community-small and a graph with 10 nodes for Enzymes). For each graph, we sample 720 graph sequences from the trained variational distribution. We show the sampled graph sequences in Figure 5 and Figure 7. Without any prior knowledge, the variational distribution has strong preference for sequences of connected graphs. In addition, in Community-small, just like GraphRNN, the model prefers to generate communities one by one.