On the Role of Edge Dependency in Graph Generative Models

Sudhanshu Chanpuriya [*1] **Cameron Musco** [*2] **Konstantinos Sotiropoulos** [*3] **Charalampos Tsourakakis** [*4]

Abstract

We investigate the trade-off between the representation power of graph generative models and model *overlap*, i.e., the degree to which the model generates diverse outputs versus regurgitating its training data. In particular, we delineate a nested hierarchy of graph generative models categorized into three levels of complexity: edge independent, node independent, and arbitrarily dependent models. This hierarchy encapsulates a wide range of prevalent methods. We derive theoretical bounds on the number of triangles and other short-length cycles producible by each level of the hierarchy, finding that more complex dependency structure allows an improved trade-off between representation power and overlap. We provide instances demonstrating the asymptotic optimality of our bounds. Furthermore, we introduce new generative models for each of the three hierarchical levels, leveraging dense subgraph discovery. Our evaluation, conducted on real-world datasets, focuses on assessing the output quality and overlap of our proposed models in comparison to other popular models. Our results indicate that our simple, interpretable models provide competitive baselines to popular generative models. Through this investigation, we offer a structured and robust evaluation scheme, thereby facilitating the development of models capable of generating accurate and edge-diverse graphs.

1. Introduction

Mathematicians and physicists have long studied random discrete structures, such as sets, permutations (Kolchin & Chistyakov, 1975), and graphs (Erdös & Rényi, 1960). In recent decades, random graphs (also known as graph generative models) have become increasingly important for modeling and understanding complex networks across a wide range of domains (Easley & Kleinberg, 2010; Frieze & Karoński, 2016; Janson et al., 2011). Random graph models provide a way to capture the uncertainty that is inherent in many real-world networks. For example, in the social sciences, random graphs have been used to study the spread of information and disease (Keliger et al., 2023) and the evolution of social networks (Newman et al., 2002). In economics, random graphs have been used to study the behavior of financial markets and the spread of economic shocks (Jackson et al., 2008). In physics, random graphs have been used to study the behavior of disordered materials, the spread of traffic jams, and the emergence of synchronization in complex systems (Dorogovtsev & Mendes, 2003). In drug discovery, random graphs have been used to study the interaction between drugs and proteins, the design of new drugs, and the prediction of drug toxicity (Pan et al., 2022). In bioinformatics, random graphs have been used to study the structure of DNA and proteins and brain networks (Simpson et al., 2011).

Deep learning (LeCun et al., 2015) has led to the development of new methods for fitting random graph models such as graph autoencoders (Kipf & Welling, 2016; Simonovsky & Komodakis, 2018), NetGAN (Bojchevski et al., 2018), GraphGAN (Wang et al., 2018), GraphRNN (You et al., 2018), among many others. Such methods are typically evaluated based on two key dimensions: efficiency and output quality. Output quality measures how closely the generated graphs resemble real-world graphs that they aim to model. A common way to measure output quality is to measure a model's tendency to produce graphs that have similar statistical properties to a target input graph or set of graphs. Common examples of statistics that are considered include the average degree, maximum degree, degree assortativity, motif counts, and shortest-path measures. In terms of motif counts triangles play a major role. This is due to the fact that triangle density is a hallmark property of several types of networks, especially social networks, and plays an important role in network analysis, e.g., Watts & Strogatz (1998); Yin et al. (2017); Tsourakakis et al. (2017).

Despite the development of the aforementioned sophisti-

[*]Equal contribution [1]University of Illinois Urbana-Champaign, Urbana, USA [2]University of Massachusetts Amherst, Amherst, USA [3]Meta, Menlo Park, United States [4]Boston University, Boston, USA. Correspondence to: Sudhanshu Chanpuriya <schanpuriya@umass.edu>, Konstantinos Sotiropoulos <ksotirop@bu.edu>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

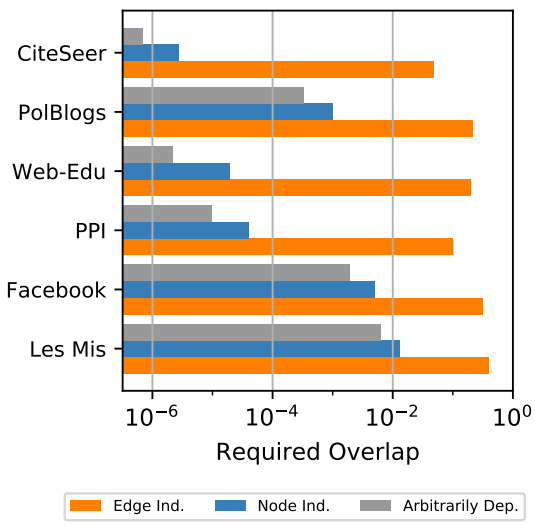

Figure 1. Minimum overlap levels necessary to match the expected triangle counts of certain real-world networks per hierarchy level.

cated algorithms for fitting graph generative models, Chanpuriya et al. (2021) highlight significant limitations of any method that fits an *edge independent* random graph model, where each edge is added independently with some (non-uniform) probability. Informally (see Section 2 for details), *any* edge independent model is limited with respect to how many triangles it can generate, *unless* it memorizes a single target network. The study's findings have implications for the use of neural-network-based graph generative models, such as NetGAN (Bojchevski et al., 2018; Rendsburg et al., 2020), Graphite (Grover et al., 2019), and MolGAN (De Cao & Kipf, 2018), all of which fit edge independent models.

The results of Chanpuriya et al. (2021) introduce the notion of *overlap*, to quantify the diversity of the graphs generated by a given model. Intuitively, useful models that produce diverse output graphs have bounded overlap. Further, without the requirement of bounded overlap, it is trivial to generate graphs that have the same statistical properties as a given input graph, by simply memorizing that graph and outputting it with probability 1. Formally, Chanpuriya et al. (2021) show a trade-off between the number of triangles that an edge independent model can generate and the model overlap.

The notion of overlap will play a central role in our work. In particular, we introduce the following definition, which generalizes the concept of overlap beyond edge independent models. For simplicity, we focus on models that generate unweighted and undirected graphs on a fixed set of n nodes.

Definition 1 (Overlap). *Suppose $\mathcal{A}$ is a distribution over binary adjacency matrices of undirected unweighted graphs on n nodes without self-loops. Let the 'volume' of $\mathcal{A}$ be the expected number of edges in a sample:* $\mathrm{Vol}(\mathcal{A}) := \mathbb{E}_{\mathbf{A}\sim\mathcal{A}}\left[\sum_{i<j}\mathbf{A}_{ij}\right]$. *The overlap of $\mathcal{A}$ is the expected number of shared edges between two independent samples, normalized by volume:*

$$\mathrm{Ov}(\mathcal{A}) := \frac{\mathbb{E}_{\mathbf{A},\mathbf{A}'\sim\mathcal{A}}\left[\sum_{i<j}\mathbf{A}_{ij}\cdot\mathbf{A}'_{ij}\right]}{\mathrm{Vol}(\mathcal{A})}.$$

Contributions. Our key contributions are as follows:

- We present a structured and nested hierarchy of graph generative models organized into three levels of complexity: edge independent, node independent, and arbitrarily dependent models. These levels encompass a wide array of today's prevalent methods as depicted in Table 1.
- We establish theoretical limits on the number of triangles and other short-length cycles that can be generated by each level of our hierarchy, based on the model overlap – see Table 1 for a summary. We also present instances demonstrating that our bounds are (asymptotically) optimal. Our results generalize those of Chanpuriya et al. (2021), which only considered edge independent models: we show that models at higher levels of our hierarchy can achieve higher triangle density given a fixed level of overlap. This finding matches empirical observations that such models can replicate triangle counts of a network with a much lower edge overlap – see Figure 1.
- We introduce new generative models across three complexity levels, fundamentally based on dense subgraph discovery (Gionis & Tsourakakis, 2015) and maximal clique detection (Eppstein et al., 2010). This approach leverages the insight that dense sets indicate higher-order dependencies, simplifying the model design. We evaluate our models against other well-known models using real-world datasets, concentrating on the quality of output and overlap. Our findings reveal that these straightforward models perform comparably to more complex deep learning models in matching graph statistics across different overlap levels, despite the universal challenge across all models of achieving accuracy at lower overlaps.

2. Related Work

Random graph theory has a long history dating back to the seminal works of Gilbert (1959) and Erdös & Rényi (1960). The $G(n,p)$ Erdös-Rényi model is a model of a random graph in which each possible edge on n labeled nodes is present with a given probability p. For convenience, we set the vertex set $V = [n] = \{1,\ldots,n\}$. As in the $G(n,p)$ model, we focus in this work on the set of all $2^{\binom{n}{2}}$ possible *labeled graphs*. In recent decades, random graphs have

Model	Upper Bound on Δ/n^3	Examples
Edge Independent	$\text{Ov}(\mathcal{A})^3$	Erdős–Rényi, SBM, NetGAN
Node Independent	$\text{Ov}(\mathcal{A})^{3/2}$	Variational Graph Auto-Encoder (VGAE)
Arbitrarily Dependent	$\text{Ov}(\mathcal{A})$	GraphVAE, GraphVAE-MM, ERGM

Table 1. Preview of results. The level of edge dependency in graph generative models inherently limits the range of graph statistics, such as triangle counts, that they can produce. Note that $\text{Ov}(\mathcal{A}) \in [0, 1]$, so a higher power on $\text{Ov}(\mathcal{A})$ means a tighter bound on the number of triangles.

become a crucial area of study. A wealth of random graph models that mimic properties of real world networks have been developed, including the Barabási-Albert-Bollobas-Riordan model (Albert & Barabási, 2002; Bollobás et al., 2001), the Watts-Strogatz model (Watts & Strogatz, 1998), Kronecker graphs (Leskovec et al., 2010a), and the Chung-Lu model (Chung et al., 2006).

An important class of random graph models consider nodes with latent embeddings in $\mathbb{R}^d$ and set the probability of an edge between each pair u, v to be some function of the inner product of their embeddings (Young & Scheinerman, 2007; Athreya et al., 2017). Significant work has studied neural-network based embedding methods such as DeepWalk (Perozzi et al., 2014), node2vec (Grover & Leskovec, 2016), VERSE (Tsitsulin et al., 2018), and NetMF (Qiu et al., 2018). Seshadhri et al. (2020) proved that even when the probability function is non-linear, certain types of models based on low-dimensional embeddings cannot simultaneously encode edge sparsity and high triangle density, a hallmark property of real-world networks (Easley & Kleinberg, 2010; Tsourakakis, 2008).

In recent years, there has been a growing interest in deep graph generation methods (Zhu et al., 2022), such as graph autoencoders (Kipf & Welling, 2016), GraphGAN (Wang et al., 2018), GraphRNNs (You et al., 2018), MolGAN (De Cao & Kipf, 2018), GraphVAE (Simonovsky & Komodakis, 2018), NetGAN (Bojchevski et al., 2018), and CELL (Rendsburg et al., 2020).

As discussed in Section 1, Chanpuriya et al. (2021) considered the class of edge independent models, namely all generative models that toss independent coins for each possible edge. This class of models is more widely known in the literature as inhomogeneous random graphs (Bollobás et al., 2007; Bollobás & Riordan, 2011), and it includes both classic models such as the classic Erdős-Rényi $G(n, p)$ model (Erdös & Rényi, 1960; Frieze & Karoński, 2016; Gilbert, 1959) and modern methods such as NetGAN (Bojchevski et al., 2018). These models can be encoded by a symmetric probability matrix $\boldsymbol{P} \in [0, 1]^{n \times n}$, where $\boldsymbol{P}_{ij}$ is the probability that nodes i, j are connected. Whereas prior work has studied the spectral properties of these models (Le et al., 2017; Benaych-Georges et al., 2019), Chanpuriya et al. study them from a new lens of *overlap*, which they define as the expected fraction of common edges between two sampled graphs from $\boldsymbol{P}$. Their key result is that *any* edge independent model with bounded overlap cannot have too many triangles in expectation. However, many generative models are not edge independent and therefore their results do not provide any insights into how many triangles they can generate for a bounded overlap. For example, exponential random graph models (ERGMs) (Frank & Strauss, 1986) go beyond edge independent models as they model the probability of observing a specific network as a function of the network's features, such as the number of edges, triangles, and other network motifs (e.g., stars and cycles). This allows for modeling higher order dependencies at the cost of learning efficiency (Chatterjee & Diaconis, 2013).

3. Hierarchy of Graph Generative Models

Our primary theoretical contribution introduces a hierarchical framework for categorizing graph generative models into three distinct, nested layers: edge independent, node independent, and arbitrarily dependent. Within each layer of the hierarchy, we establish upper limits for the triangle count relative to overlap, with the detailed proofs presented in Section 4, and we extend these findings to offer generalized limits for k-cycles in Appendix A.2. Additionally, for every layer, we identify a specific family of graphs that meets these maximum triangle counts, demonstrating the asymptotic precision of our bounds. Furthermore, we explore previous models that align with each hierarchical level.

3.1. Edge Independent (EI) Model

We begin with the most constrained level of our hierarchy, edge independent models, more widely known as inhomogeneous Erdős-Rényi model This class includes many important models, such as the classical Erdős-Rényi and stochastic block models (SBM) (Abbe, 2017), along with modern deep-learning based models such as NetGAN (Bojchevski et al., 2018; Rendsburg et al., 2020), Graphite (Grover et al., 2019), and MolGAN (De Cao & Kipf, 2018) as they ultimately output a probability matrix based on which edges are sampled independently.

Definition 2 (Edge Independent Graph Model)**.** *An edge*

independent (EI) graph generative model is a distribution $\mathcal{A}$ over symmetric adjacency matrices $\boldsymbol{A} \in \{0,1\}^{n\times n}$, such that, for some symmetric $\boldsymbol{P} \in [0,1]^{n\times n}$, $\boldsymbol{A}_{ij} = \boldsymbol{A}_{ji} = 1$ with probability $\boldsymbol{P}_{ij}$ for all $i, j \in [n]$, otherwise $\boldsymbol{A}_{ij} = \boldsymbol{A}_{ji} = 0$. Furthermore, all entries (i.e., in the upper diagonal) of $\boldsymbol{A}$ are mutually independent.

Any EI graph generative model satisfies the following theorem:

Theorem 1. *Any edge independent graph model $\mathcal{A}$ with overlap $\mathrm{Ov}(\mathcal{A})$ has at most $\frac{1}{6}n^3 \cdot \mathrm{Ov}(\mathcal{A})^3$ triangles in expectation. That is, for $\boldsymbol{A}$ drawn from $\mathcal{A}$, letting $\Delta(\boldsymbol{A})$ be the number of triangles in $\boldsymbol{A}$,*

$$\mathbb{E}\left[\Delta(\boldsymbol{A})\right] \leq \frac{n^3}{6} \cdot \mathrm{Ov}(\mathcal{A})^3. \tag{1}$$

Chanpuriya et al. (2021) give a spectral proof of Theorem 1. We provide an alternative proof based on an elegant generalization of Shearer's Entropy Lemma (Alon & Spencer, 2016; Chung et al., 1986) due to Friedgut (2004). Furthermore, we show that the upper bound is tight. Consider the $G(n,p)$ model. The expected volume is $\mathrm{Vol}(\mathcal{A}) = p \cdot \binom{n}{2}$, whereas the expected number of shared edges between two samples is $p^2 \cdot \binom{n}{2}$, yielding an overlap of p (see Definition 1). Furthermore, since any triangle is materialized with probability p^3, by the linearity of expectation there are $O(n^3 \cdot p^3)$ triangles, which shows that inequality 1 is tight up to constant factors.

3.2. Arbitrarily Dependent (AD) Model

We now move to the other end of the edge dependency spectrum, namely to models that allow for arbitrary edge dependencies. A classic arbitrarily dependent model is the ERGM (Frank & Strauss, 1986; Chatterjee & Diaconis, 2013) as it can model any higher order dependencies. Similarly, auto-regressive approaches, including recent diffusion models like DiGress (Vignac et al., 2022) and flow-based models like GraphAF (Shi et al., 2020), can also be arbitrarily dependent in our hierarchy.

Definition 3 (Arbitrarily Dependent Graph Model)**.** *An arbitrarily dependent graph generative model allows for any possible distribution $\mathcal{A}$ over symmetric adjacency matrices $\boldsymbol{A} \in \{0,1\}^{n\times n}$.*

Allowing for arbitrary edge dependencies enables us to have models with significantly more triangles as a function of the overlap.

Theorem 2. *For any arbitrarily dependent model $\mathcal{A}$ with overlap $\mathrm{Ov}(\mathcal{A})$, the expected number of triangles is at most $\frac{1}{2}n^3 \cdot \mathrm{Ov}(\mathcal{A})$.*

As in the case of EI models, there is a simple instance that shows that the bound is tight. Specifically, consider a model that outputs a complete graph with probability p and an empty graph otherwise. Each edge occurs with probability p, so by the same computation as for EI models, the overlap is p. As for the triangle count, there are no triangles when the graph is empty, but when it is complete, there are all $\binom{n}{3}$ possible triangles. Thus, the expected number of triangles is $O(n^3 \cdot p) = O(n^3 \cdot \mathrm{Ov}(\mathcal{A}))$, again showing that our bound is tight up to constant factors

At a high level, arbitrarily dependent graph generative models often arise when methods sample a graph-level embedding, then produce a graph sample from this embedding, allowing for complex dependencies between edges in the sample. A specific example is the graph variational autoencoder (GraphVAE) (Simonovsky & Komodakis, 2018), in which decoding involves sampling a single graph-level embedding $\boldsymbol{x_G} \in \mathbb{R}^k$, and the presence of each edge is an independent Bernoulli random variable with a parameter that is some function f_{ij} of $\boldsymbol{x}$: $\boldsymbol{A}_{ij} = \mathrm{Bernoulli}(f_{ij}(\boldsymbol{x_G}))$. In particular, these functions are encoded by a fully-connected neural network. Assuming these f_{ij} can closely approximate the sign function, with a 1-dimensional graph embedding ($k = 1$), this model can in fact match the triangle bound of Theorem 2 by simulating the tight instance described above (outputting a complete graph with probability p and the empty graph otherwise).

3.3. Node Independent (NI) Model

We next focus on the middle level of our hierarchy, between the two extremes of EI and AD generative models. This level is built upon the common concept of node embeddings that are stochastic and generated independently for each node.

Definition 4 (Node Independent Graph Model)**.** *A node independent (NI) graph generative model is a distribution $\mathcal{A}$ over symmetric adjacency matrices $\boldsymbol{A} \in \{0,1\}^{n\times n}$, where, for some embedding space $\mathcal{E}$, some mutually independent random variables $\boldsymbol{x_1}, \ldots, \boldsymbol{x_n} \in \mathcal{E}$, and some symmetric function $e : \mathcal{E} \times \mathcal{E} \to [0,1]$, the entries of $\boldsymbol{A}$ are Bernoulli random variables $\boldsymbol{A}_{ij} = \mathit{Bernoulli}\left(e(\boldsymbol{x_i}, \boldsymbol{x_j})\right)$ that are mutually independent conditioned on $\boldsymbol{x_1}, \ldots, \boldsymbol{x_n}$.*

Observe that NI models encompass the important special case of random graph models defined by *graphons* (Diaconis & Janson, 2007; Orbanz & Roy, 2014; Klopp et al., 2017). Such models are essentially a special case of NI in which the embedding space $\mathcal{E}$ is the interval $[0,1]$, and each node's embedding is an independent uniform random variable drawn on this interval. A famous result for these graphon models (also known as *vertex-exchangeable* random graph models) is that, since all of the nodes' embedding distributions are identical, they are unable to produce sparse but non-empty graphs. By contrast, in our definition, nodes can have different embedding distributions, and hence this

limitation does not apply.

Node independent graph models most commonly arise when methods independently sample n node-level embeddings, then determine the presence of edges between two nodes based on some compatibility function of their embeddings. One notable example is the variational graph auto-encoder (VGAE) model (Kipf & Welling, 2016). In the decoding step of this model, a Gaussian-distributed node embedding is sampled independently for each node, and the presence of each possible edge is an independent Bernoulli random variable with a parameter that varies with the dot product of the corresponding embeddings as follows: $\boldsymbol{A}_{ij} = \text{Bernoulli}(\sigma(\boldsymbol{x_i} \cdot \boldsymbol{x_j}))$. Thus, the VGAE model seamlessly fits into our node independent category.

Interestingly, the triangle bound for this class of generative models lies in the middle of the EI and AD triangle bounds.

Theorem 3. *Any node independent graph model $\mathcal{A}$ with overlap $\text{Ov}(\mathcal{A})$ has at most $\frac{1}{6}n^3 \cdot \text{Ov}(\mathcal{A})^{3/2}$ triangles in expectation.*

The proof of Theorem 3 requires expressing the probability of edges and triangles appearing as integrals in the space of node embeddings. Then we apply a continuous version of Shearer's inequality. We show that the bound of Theorem 3 is tight. Generate a random graph by initially starting with an empty graph comprising n nodes. Subsequently, each node independently becomes "active" with a probability of $\sqrt{p}$, and any edges connecting active nodes are subsequently added into the graph. Note that for a given edge to be added, both of its endpoint nodes must be active, which occurs with probability p; this again yields an overlap of p as with the prior tight examples. For a triangle to be added, all three of its endpoint nodes must be active, which occurs with probability $p^{3/2}$. Hence the expected number of triangles is $O(n^3 \cdot p^{3/2}) = O(n^3 \cdot \text{Ov}(\mathcal{A})^{3/2})$.

The random graph can be represented using embeddings based on the provided definition. Consider a 1-dimensional embedding $\boldsymbol{x} \in \mathbb{R}^{n\times 1}$. Let $\boldsymbol{x}_i = 1$ with probability $\sqrt{p}$ and otherwise let $\boldsymbol{x}_i = 0$, independently for each $i \in [n]$. These coin tosses are made independently for each $i \in [n]$. To capture the edges between nodes, we define $e(\boldsymbol{x}_i, \boldsymbol{x}_j)$ as 1 when both arguments are 1, and 0 otherwise. This embedding-based representation precisely implements the described random graph.

4. Proof of Main Results

In this section, we will prove the triangle count bounds stated in Section 3. We begin by introducing certain concepts that will be useful throughout our proofs.

4.1. Theoretical Preliminaries

Inner product space formulation of overlap. We first state equivalent definitions for volume and overlap that will be useful for proofs. We first make the following observation:

Observation 1 (Inner Product Space of Distributions over Vectors)**.** *Suppose $\mathcal{U}, \mathcal{V}$ are distributions over real-valued d-dimensional vectors. Then the following operation defines an inner product:*

$$\langle \mathcal{U}, \mathcal{V} \rangle := \mathop{\mathbb{E}}_{\boldsymbol{u}\sim\mathcal{U}, \boldsymbol{v}\sim\mathcal{V}}[\boldsymbol{u} \cdot \boldsymbol{v}],$$

where $\boldsymbol{u} \cdot \boldsymbol{v}$ is the standard vector dot product.

Throughout, we deal with adjacency matrices of undirected graphs on n nodes without self-loops, and distributions over such matrices. Dot products and inner products of these objects are taken over $\binom{n}{2}$-dimensional vectorizations of entries below the diagonal. Let $\mathcal{F}$ denote the distribution that returns the adjacency matrix $\boldsymbol{F}$ of a graph which is fully connected (i.e., has all possible edges) with probability 1. Then $\text{Vol}(\mathcal{A}) = \langle \mathcal{A}, \mathcal{F} \rangle$ and $\text{Ov}(\mathcal{A}) = \frac{\langle \mathcal{A}, \mathcal{A} \rangle}{\text{Vol}(\mathcal{A})}$.

These expressions allow us to derive the following upper bound on volume in terms of overlap, which applies to any graph generative model:

Lemma 1. *For a graph generative model $\mathcal{A}$ with overlap $\text{Ov}(\mathcal{A})$, the expected number of edges $\text{Vol}(\mathcal{A})$ is at most $\binom{n}{2} \cdot \text{Ov}(\mathcal{A})$.*

Proof. By the definition of overlap and Cauchy-Schwarz,

$$\text{Ov}(\mathcal{A}) = \frac{\langle \mathcal{A}, \mathcal{A} \rangle}{\langle \mathcal{A}, \mathcal{F} \rangle} \geq \frac{\langle \mathcal{A}, \mathcal{F} \rangle^2}{\langle \mathcal{A}, \mathcal{F} \rangle \cdot \langle \mathcal{F}, \mathcal{F} \rangle} = \frac{\langle \mathcal{A}, \mathcal{F} \rangle}{\langle \mathcal{F}, \mathcal{F} \rangle} = \frac{\text{Vol}(\mathcal{A})}{\langle \mathcal{F}, \mathcal{F} \rangle}.$$

Since $\langle \mathcal{F}, \mathcal{F} \rangle = \langle \boldsymbol{F}, \boldsymbol{F} \rangle = \binom{n}{2}$, rearranging yields the desired inequality, $\text{Vol}(\mathcal{A}) \leq \binom{n}{2} \cdot \text{Ov}(\mathcal{A})$. □

4.2. Proofs of Triangle Count Bounds

We now prove the upper bounds on expected triangle count for edge independent, node independent, and arbitrarily dependent models appearing in Theorems 1, 2, and 3.

Edge independent. A proof for Theorem 1 based on expressing triangle count in terms of eigenvalues of $\boldsymbol{P}$ follows directly from Lemma 1 and Chanpuriya et al. (2021), but we present a different proof based on the following variant of Cauchy-Schwarz from Friedgut (2004):

$$\sum\nolimits_{ijk} a_{ij} b_{jk} c_{ki} \leq \sqrt{\sum\nolimits_{ij} a_{ij}^2 \sum\nolimits_{ij} b_{ij}^2 \sum\nolimits_{ij} c_{ij}^2}. \quad (2)$$

Proof of Theorem 1. Let $\boldsymbol{P}$ be the edge probability matrix of the edge independent model $\mathcal{A}$. Then, by the above

inequality, for a sample $\boldsymbol{A}$ from $\mathcal{A}$,

$$\begin{aligned}\mathbb{E}[\Delta(\boldsymbol{A})] &= \tfrac{1}{6}\sum\nolimits_{ijk}\boldsymbol{P}_{ij}\boldsymbol{P}_{jk}\boldsymbol{P}_{ki} \le \tfrac{1}{6}\sqrt{\left(\sum\nolimits_{ij}\boldsymbol{P}_{ij}^2\right)^3} \\ &= \tfrac{1}{6}\left(2\sum\nolimits_{i<j}\boldsymbol{P}_{ij}^2\right)^{3/2} = \tfrac{1}{6}\left(2\cdot \mathrm{Ov}(\mathcal{A})\mathrm{Vol}(\mathcal{A})\right)^{3/2} \\ &= \tfrac{\sqrt{2}}{3}\cdot \mathrm{Ov}(\mathcal{A})^{3/2}\mathrm{Vol}(\mathcal{A})^{3/2}.\end{aligned}$$

Applying Lemma 1 yields

$$\mathbb{E}[\Delta(\boldsymbol{A})] \le \tfrac{\sqrt{2}}{3}\tbinom{n}{2}^{3/2}\cdot \mathrm{Ov}(\mathcal{A})^3 \le \tfrac{1}{6}n^3\cdot \mathrm{Ov}(\mathcal{A})^3. \quad \square$$

Arbitrarily dependent. We now prove the triangle bound for arbitrary graph generative models, which follows from Lemma 1. Note that, given a random adjacency matrix $\boldsymbol{A} \in \{0,1\}^{n\times n}$, the product $\boldsymbol{A}_{ij}\boldsymbol{A}_{jk}\boldsymbol{A}_{ik}$ is an indicator random variable for the existence of a triangle between nodes $i, j, k \in [n]$.

Proof of Theorem 2. Let $\boldsymbol{A}$ be a sample from $\mathcal{A}$. From Lemma 1, we have

$$\sum\nolimits_{i<j}\mathbb{E}[\boldsymbol{A}_{ij}] = \mathrm{Vol}(\mathcal{A}) \le \tbinom{n}{2}\cdot \mathrm{Ov}(\mathcal{A}).$$

So, for the expected number of triangles in a sample, we have:

$$\begin{aligned}\mathbb{E}[\Delta(\boldsymbol{A})] &= \mathbb{E}\left[\sum\nolimits_{i<j<k}\boldsymbol{A}_{ij}\boldsymbol{A}_{jk}\boldsymbol{A}_{ik}\right] \\ &= \sum\nolimits_{i<j<k}\mathbb{E}\left[\boldsymbol{A}_{ij}\boldsymbol{A}_{jk}\boldsymbol{A}_{ik}\right] \\ &\le \sum\nolimits_{i<j<k}\mathbb{E}[\boldsymbol{A}_{ij}] \le n\cdot\sum\nolimits_{i<j}\mathbb{E}[\boldsymbol{A}_{ij}] \\ &\le n\cdot\tbinom{n}{2}\cdot \mathrm{Ov}(\mathcal{A}) \le \tfrac{1}{2}n^3\cdot \mathrm{Ov}(\mathcal{A}). \quad \square\end{aligned}$$

Node independent. Proving Theorem 3 is more involved than the prior proofs, and requires first establishing the following lemma:

Lemma 2. *For a sample $\boldsymbol{A}$ from any node independent model on n nodes and any three nodes $i, j, k \in [n]$, the probability that i, j, k form a triangle is upper-bounded as follows:*

$$\mathbb{E}[\boldsymbol{A}_{ij}\boldsymbol{A}_{jk}\boldsymbol{A}_{ik}] \le \sqrt{\mathbb{E}[\boldsymbol{A}_{ij}]\mathbb{E}[\boldsymbol{A}_{jk}]\mathbb{E}[\boldsymbol{A}_{ik}]}.$$

We prove Lemma 2 in Appendix A.1 by expressing the probability of edges and triangles appearing as integrals in the space of node embeddings, after which we can apply a theorem of Friedgut (2004) (also proved by Finner (1992)), which can be seen as a continuous version of Inequality 2. With Lemma 2 in place, it is straightforward to prove Theorem 3:

Proof of Theorem 3. Let $\boldsymbol{A}$ be a sample from a node independent model. For the expected number of triangles $\mathbb{E}[\Delta(\boldsymbol{A})]$ in the sample, we have that

$$\begin{aligned}\mathbb{E}[\Delta(\boldsymbol{A})] &= \sum\nolimits_{i<j<k}\mathbb{E}[\boldsymbol{A}_{ij}\boldsymbol{A}_{ik}\boldsymbol{A}_{jk}] \\ &\le \sum\nolimits_{i<j<k}\sqrt{\mathbb{E}[\boldsymbol{A}_{ij}]\cdot\mathbb{E}[\boldsymbol{A}_{ik}]\cdot\mathbb{E}[\boldsymbol{A}_{jk}]} \\ &\le \sqrt{\tbinom{n}{3}\cdot\sum\nolimits_{i<j<k}\mathbb{E}[\boldsymbol{A}_{ij}]\cdot\mathbb{E}[\boldsymbol{A}_{ik}]\cdot\mathbb{E}[\boldsymbol{A}_{jk}]} \\ &\le \sqrt{\tbinom{n}{3}\cdot\tfrac{1}{6}n^3\cdot \mathrm{Ov}(\mathcal{A})^3} \le \tfrac{1}{6}n^3\cdot \mathrm{Ov}(\mathcal{A})^{3/2},\end{aligned}$$

where second line follows from Cauchy-Schwarz and the third from Theorem 1 for EI models. $\square$

Note that we favor simplicity and interpretability of the final result over tightness. In particular, as our proofs reveal, the bound for EI models, and hence the one for NI models, is tighter when expressed in terms of both overlap and volume.

5. Generative Models Leveraging Maximal-Clique Detection

We shift our focus towards empirically evaluating the real-world trade-off between overlap and performance across several specific models on real-world networks. In this endeavor, we concentrate on graph generative models that, given an input adjacency matrix $\mathbf{A} \in \{0,1\}^{n\times n}$, yield a distribution $\mathcal{A}$ over adjacency matrices in $\{0,1\}^{n\times n}$. Typically, a trade-off is observed between two desired characteristics in these distributions:

1. Samples from $\mathcal{A}$ should closely align with various graph statistics of the input $\mathbf{A}$, such as the degree distribution and triangle count.
2. $\mathcal{A}$ should not exhibit very high overlap, to deter the model from merely memorizing and reproducing $\mathbf{A}$.

Due to the inherent conflict between these two objectives for $\mathcal{A}$, it is advantageous for the model to allow for *easy tuning of overlap*. Note that this is a different goal from simply seeking low overlap. While our theoretical results focus on limitations of graph generative models as they tend away from very high overlap (which would correspond to memorization and regurgitation), it is not necessarily true that a very low overlap is ideal. We expect the acceptable range of overlap to be application-dependent, so it is desirable to be able to easily adjust overlap and to achieve reasonable fidelity in matching graph statistics across a wide range of overlap values.

Recent graph generative models, especially those that incorporate edge dependency, often involve complex deep

architectures with a large number of parameters that are trained with gradient descent. The abundance of parameters implies that these models may have the capacity to simply memorize the input graph. At the same time, the complexity of the approaches obscures the roles of each component in yielding performance (in particular, the role of edge dependency is unclear), and specifying a desired overlap with the input graph is generally not possible, short of heuristics like early stopping. Altogether, the preceding discussion inspires the following research questions:

1. Do the relaxed theoretical constraints on triangle counts and other graph statistics, as applied in graph generative models with edge dependency, manifest in the modeling of real-world networks?
2. Can straightforward baselines be employed to achieve edge dependency while also allowing for easy adjustment of overlap?
3. Can these simple models attain graph statistics comparable to deep models across different degrees of overlap?

As a glimpse into our findings, we ascertain overall affirmative responses to all three of these inquiries.

5.1. Proposed Models

We introduce three graph generative models, one for each of the categories of our framework: EI, NI, and AD. These baseline models mainly exploit the dense subgraph structure of the input graph, specifically the set of maximal cliques (MCs) (Eppstein et al., 2010). We refer to our models as MCEI, MCNI, and MCAD, respectively. The high-level concept of these models is to start with an empty graph and plant edges from each of the target (input) graph G_t's max cliques with some fixed probability hyperparameter p. How the edges are planted depends on the desired type of edge dependency and reflects the characteristic 'tight' examples for each category of the hierarchy from Section 3 – see Algorithm 1 for details.

Observe that, the lower p, the fewer the expected edges in the final sampled graph G_p. To compensate for this 'residual' with respect to the target graph G_t, we also sample a second graph G_r and return the union of G_p and G_r. Specifically, we produce G_r by sampling from a simple edge independent model that is configured so that the node degrees of the union of G_p and G_r match those of G_t in expectation. We defer details of this auxiliary EI model to Section B. The end result is that, in each of our models, ranging the hyperparameter p from 0 to 1 ranges the overlap of the resulting distribution from very low to 1. In particular, the distribution $\mathcal{A}$ goes from being nearly agnostic to the input $\mathbf{A}$ (depending only on its degree sequence), to exactly returning $\mathbf{A}$ with probability 1.

We note that these models are not intended to be suitable for all kinds of graphs. The focus on max cliques is meant to cover the common case of graphs with strong community structure and a tendency towards triadic closure. Given this focus on local structure, we do not expect the models to be strong at capturing global connectivity patterns. We introduce these models not to assert that they should be used for real-world applications, but rather to show two things: first, that there is room for improvement in modern generative models in the low-to-mid overlap regime; and second, that versions of our otherwise similar models are stronger at capturing local structure when granted stronger edge dependency, which aligns with our theoretical results.

Algorithm 1 Sampling G_p based on max cliques

Input : target graph G_t, planting probability p, model type (MCEI, MCAD, or MCNI)
Output : sampled graph G_p

initialize the sampled graph G_p to be empty
foreach max clique $M \in$ target graph G_t **do**
  if model type is MCEI **then**
    foreach edge $e \in M$ **do**
      with probability p, add e to G_p
  else if model type is MCAD **then**
    with probability p, add all edges in M to G_p
  else if model type is MCNI **then**
    foreach node $v \in M$ **do**
      with probability $\sqrt{p}$, set v to be 'active' in M
    add edges between all pairs of nodes active in M to G_p
return G_p

5.2. Experimental Setup and Results

Setup. We perform an evaluation of different graph generative models under the perspective of the *overlap* criterion. Specifically, we see how these models match different statistics of an input graph as overlap is varied. We choose 8 statistics that capture both the connectivity of a graph, as well local patterns within it. We evaluate six models in total: the three models we introduce in Section 5.1, as well as three deep learning-based models, CELL (Rendsburg et al., 2020), VGAE (Kipf & Welling, 2016), and GraphVAE (Simonovsky & Komodakis, 2018), which are EI, NI, and AD, respectively. For our methods, we vary overlap by simply varying the p hyperparameter; for the other methods, we use early stopping or vary the representation dimensionality. Further details of the experimental setup are included in Appendix C. We evaluate on 8 real-world networks and 1 synthetic network. In Figure 2, we include results for LES MISERABLES (Knuth, 1993), a co-appearance network of characters in the eponymous novel; the remaining results are deferred to Appendix D.

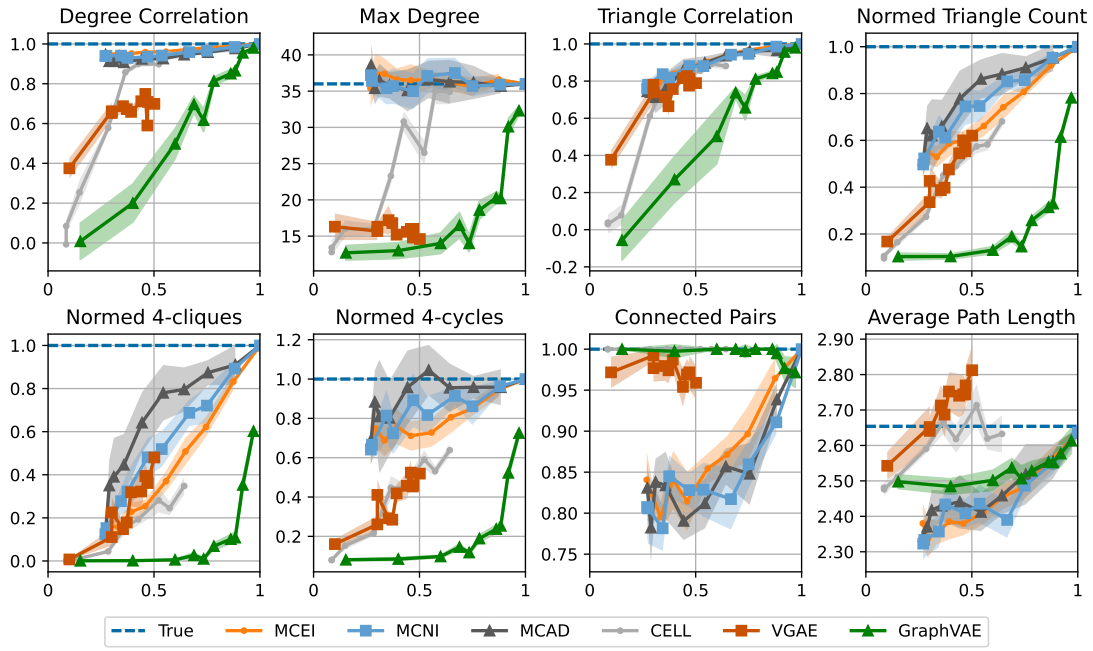

Figure 2. Statistics for LES MISERABLES as a function of the *overlap*: max degree; Pearson correlation coefficient of degree and triangle sequences; triangle count (normalized by true count), 4-clique, and 4-cycle counts; fraction of connected pairs, $\sum_i \binom{|C_i|}{2} / \binom{n}{2}$ where C_i is the ith connected component; and characteristic (average) path length. We plot the mean and 90% confidence interval across 10 trials.

Results. Our findings highlight the importance of *overlap* as a third dimension in the evaluation of graph generative models. We see that deep-learning based models, like GraphVAE and CELL, can almost fit the input graph as we allow the training to be performed for a sufficient number of epochs. However, when one wants to generate a diverse set of graphs, these models fail to match certain statistics of the input graph. For example, we see in Figure 2 that the CELL method generates graphs with a low number of triangles for the LES MISERABLES dataset when the overlap between the generated graphs is small. We observe similar results for almost all datasets and other statistics, especially those pertaining to small dense subgraphs (like 4-cliques and 4-cycles). Similarly, the GraphVAE method almost always fits the input graph in a high overlap regime. Although GraphVAE possesses greater theoretical capacity as an AD generative model, we observe that instances with low overlap deviate significantly from the statistical characteristics of the input graph, as illustrated in Figure 2. Consequently, this approach fails to achieve generalization when trained on a single graph instance. The VGAE model encounters additional limitations due to its dot-product kernel, which has been demonstrated to have certain constraints in generating graphs with sparsity and triangle density (Seshadhri et al., 2020; Chanpuriya et al., 2020). As depicted in Figures 2, we observe that this model fails to adequately capture the characteristics of the input graph, even when trained for many epochs.

Surprisingly, despite their simplicity, the three introduced models exhibit desirable characteristics. Firstly, the overlap can be easily adjusted by increasing the probability hyperparameter p, providing more predictability compared to other models (refer to the results for the POLBLOGS dataset in the Appendix). Secondly, these models generally generate a higher number of triangles, which closely aligns with the input graph, in comparison to methods such as CELL. For instance, in Figure 2, these baselines produce graphs with a greater number of triangles, 4-cliques, and 4-cycles than CELL and GraphVAE, particularly when the overlap is low. Furthermore, as the level of dependency rises (from edge independent to arbitrarily dependent), we observe a higher triangle count for a fixed overlap. This finding supports our theoretical assertions from Section 3 regarding the efficacy of different models within the introduced hierarchy. However, a drawback of these two-stage methods is their inability to capture the connectivity patterns of the input graph, as evident from the fraction of connected pairs and the characteristic path length statistics.

6. Conclusion

We have proved tight trade-offs for graph generative models between their ability to produce networks that match the high triangle densities of real-world graphs, and their ability to achieve low *overlap* and generate a diverse set of networks. We show that as the models are allowed higher

levels of edge dependency, they are able to achieve higher triangle counts with lower overlap, and we formalize this finding by introducing a three-level hierarchy of edge dependency. An interesting future direction is to refine this hierarchy to be finer-grained, and also to investigate the roles of embedding length and complexity of the embedding distributions. We also emphasize our introduction of *overlap* as a third dimension along which to evaluate graph generative models, together with output quality and efficiency. We believe these directions can provide a solid groundwork for the systematic theoretical and empirical study of graph generative models.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Abbe, E. Community detection and stochastic block models: recent developments. *The Journal of Machine Learning Research*, 18(1):6446–6531, 2017.

Adamic, L. A. and Glance, N. The political blogosphere and the 2004 us election: divided they blog. In *Proceedings of the 3rd International Workshop on Link Discovery*, pp. 36–43, 2005.

Aiello, W., Chung, F., and Lu, L. A random graph model for power law graphs. *Experimental mathematics*, 10(1): 53–66, 2001.

Albert, R. and Barabási, A.-L. Statistical mechanics of complex networks. *Reviews of modern physics*, 74(1):47, 2002.

Alon, N. and Spencer, J. H. *The probabilistic method*. John Wiley & Sons, 2016.

Athreya, A., Fishkind, D. E., Tang, M., Priebe, C. E., Park, Y., Vogelstein, J. T., Levin, K., Lyzinski, V., and Qin, Y. Statistical inference on random dot product graphs: a survey. *The Journal of Machine Learning Research*, 18 (1):8393–8484, 2017.

Benaych-Georges, F., Bordenave, C., and Knowles, A. Largest eigenvalues of sparse inhomogeneous erdős–rényi graphs. *The Annals of Probability*, 47(3):1653–1676, 2019.

Bojchevski, A., Shchur, O., Zügner, D., and Günnemann, S. NetGAN: Generating graphs via random walks. *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018.

Bollobás, B. and Riordan, O. Sparse graphs: metrics and random models. *Random Structures & Algorithms*, 39(1): 1–38, 2011.

Bollobás, B., Riordan, O., Spencer, J., and Tusnády, G. The degree sequence of a scale-free random graph process. *Random Structures & Algorithms*, 18(3):279–290, 2001.

Bollobás, B., Janson, S., and Riordan, O. The phase transition in inhomogeneous random graphs. *Random Structures & Algorithms*, 31(1):3–122, 2007.

Chanpuriya, S., Musco, C., Sotiropoulos, K., and Tsourakakis, C. Node embeddings and exact low-rank representations of complex networks. *Advances in neural information processing systems*, 33:13185–13198, 2020.

Chanpuriya, S., Musco, C., Sotiropoulos, K., and Tsourakakis, C. On the power of edge independent graph models. *Advances in Neural Information Processing Systems*, 34:24418–24429, 2021.

Chatterjee, S. and Diaconis, P. Estimating and understanding exponential random graph models. 2013.

Chung, F. and Lu, L. The average distances in random graphs with given expected degrees. *Proceedings of the National Academy of Sciences*, 99(25):15879–15882, 2002a.

Chung, F. and Lu, L. Connected components in random graphs with given expected degree sequences. *Annals of combinatorics*, 6(2):125–145, 2002b.

Chung, F., Chung, F. R., Graham, F. C., Lu, L., et al. *Complex graphs and networks*. Number 107. American Mathematical Soc., 2006.

Chung, F. R., Graham, R. L., Frankl, P., and Shearer, J. B. Some intersection theorems for ordered sets and graphs. *Journal of Combinatorial Theory, Series A*, 43(1):23–37, 1986.

De Bie, T. Maximum entropy models and subjective interestingness: an application to tiles in binary databases. *Data Mining and Knowledge Discovery*, 23(3):407–446, 2011.

De Cao, N. and Kipf, T. MolGAN: An implicit generative model for small molecular graphs. *ICML Deep Generative Models Workshop*, 2018.

Diaconis, P. and Janson, S. Graph limits and exchangeable random graphs. *arXiv preprint arXiv:0712.2749*, 2007.

Dorogovtsev, S. N. and Mendes, J. F. *Evolution of networks: From biological nets to the Internet and WWW*. Oxford university press, 2003.

Easley, D. and Kleinberg, J. *Networks, crowds, and markets: Reasoning about a highly connected world*. Cambridge university press, 2010.

Eppstein, D., Löffler, M., and Strash, D. Listing all maximal cliques in sparse graphs in near-optimal time. *arXiv preprint arXiv:1006.5440*, 2010.

Erdös, P. and Rényi, A. On the evolution of random graphs. *Publications of the Mathematical Institute of the Hungarian Academy of Sciences*, 5(1):17–60, 1960.

Finner, H. A generalization of holder's inequality and some probability inequalities. *The Annals of Probability*, pp. 1893–1901, 1992.

Frank, O. and Strauss, D. Markov graphs. *Journal of the american Statistical association*, 81(395):832–842, 1986.

Friedgut, E. Hypergraphs, entropy, and inequalities. *The American Mathematical Monthly*, 111(9):749–760, 2004.

Frieze, A. and Karoński, M. *Introduction to random graphs*. Cambridge University Press, 2016.

Gilbert, E. N. Random graphs. *The Annals of Mathematical Statistics*, 30(4):1141–1144, 1959.

Gionis, A. and Tsourakakis, C. E. Dense subgraph discovery: Kdd 2015 tutorial. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2313–2314, 2015.

Gleich, D., Zhukov, L., and Berkhin, P. Fast parallel pagerank: A linear system approach. *Yahoo! Research Technical Report*, 13:22, 2004.

Grover, A. and Leskovec, J. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 855–864, 2016.

Grover, A., Zweig, A., and Ermon, S. Graphite: Iterative generative modeling of graphs. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 2019.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2.

Hoff, P. D. Random effects models for network data. In *Dynamic Social Network Modeling and Analysis: Workshop Summary and Papers*. Citeseer, 2003.

Jackson, M. O. et al. *Social and economic networks*, volume 3. Princeton university press Princeton, 2008.

Jain, S. and Seshadhri, C. The power of pivoting for exact clique counting. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pp. 268–276, 2020.

Janson, S., Luczak, T., and Rucinski, A. *Random graphs*. John Wiley & Sons, 2011.

Keliger, D., Lovász, L., Móri, T., and Ódor, G. Switchover phenomenon for general graphs. *arXiv preprint arXiv:2304.11971*, 2023.

Kipf, T. N. and Welling, M. Variational graph auto-encoders. *NeurIPS Bayesian Deep Learning Workshop*, 2016.

Klopp, O., Tsybakov, A. B., and Verzelen, N. Oracle inequalities for network models and sparse graphon estimation. 2017.

Knuth, D. E. *The Stanford GraphBase: a platform for combinatorial computing*, volume 1. AcM Press New York, 1993.

Kolchin, V. F. and Chistyakov, V. P. The cyclic structure of random permutations. *Mathematical notes of the Academy of Sciences of the USSR*, 18(6):1139–1144, 1975.

Le, C. M., Levina, E., and Vershynin, R. Concentration and regularization of random graphs. *Random Structures & Algorithms*, 51(3):538–561, 2017.

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

Leskovec, J. and Mcauley, J. J. Learning to discover social circles in ego networks. In *Advances in Neural Information Processing Systems 25 (NeurIPS)*, pp. 539–547, 2012.

Leskovec, J., Chakrabarti, D., Kleinberg, J., Faloutsos, C., and Ghahramani, Z. Kronecker graphs: an approach to modeling networks. *Journal of Machine Learning Research*, 11(2), 2010a.

Leskovec, J., Huttenlocher, D., and Kleinberg, J. Signed networks in social media. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pp. 1361–1370, 2010b.

Newman, M. E., Watts, D. J., and Strogatz, S. H. Random graph models of social networks. *Proceedings of the national academy of sciences*, 99(suppl_1):2566–2572, 2002.

Orbanz, P. and Roy, D. M. Bayesian models of graphs, arrays and other exchangeable random structures. *IEEE transactions on pattern analysis and machine intelligence*, 37(2):437–461, 2014.

Pan, X., Lin, X., Cao, D., Zeng, X., Yu, P. S., He, L., Nussinov, R., and Cheng, F. Deep learning for drug repurposing: Methods, databases, and applications. *Wiley interdisciplinary reviews: Computational molecular science*, 12(4):e1597, 2022.

Perozzi, B., Al-Rfou, R., and Skiena, S. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 701–710, 2014.

Qiu, J., Dong, Y., Ma, H., Li, J., Wang, K., and Tang, J. Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In *Proceedings of the eleventh ACM international conference on web search and data mining*, pp. 459–467, 2018.

Rendsburg, L., Heidrich, H., and von Luxburg, U. NetGAN without GAN: From random walks to low-rank approximations. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, 2020.

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

Seshadhri, C., Sharma, A., Stolman, A., and Goel, A. The impossibility of low-rank representations for triangle-rich complex networks. *Proceedings of the National Academy of Sciences*, 117(11):5631–5637, 2020.

Shi, C., Xu, M., Zhu, Z., Zhang, W., Zhang, M., and Tang, J. Graphaf: a flow-based autoregressive model for molecular graph generation. *arXiv preprint arXiv:2001.09382*, 2020.

Simonovsky, M. and Komodakis, N. Graphvae: Towards generation of small graphs using variational autoencoders. In *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part I 27*, pp. 412–422. Springer, 2018.

Simpson, S. L., Hayasaka, S., and Laurienti, P. J. Exponential random graph modeling for complex brain networks. *PloS one*, 6(5):e20039, 2011.

Stark, C., Breitkreutz, B.-J., Chatr-Aryamontri, A., Boucher, L., Oughtred, R., Livstone, M. S., Nixon, J., Van Auken, K., Wang, X., Shi, X., et al. The BioGRID interaction database: 2011 update. *Nucleic Acids Research*, 39:D698–D704, 2010.

Takeaki, U. Implementation issues of clique enumeration algorithm. *Special issue: Theoretical computer science and discrete mathematics, Progress in Informatics*, 9: 25–30, 2012.

Tsitsulin, A., Mottin, D., Karras, P., and Müller, E. Verse: Versatile graph embeddings from similarity measures. In *Proceedings of the 2018 world wide web conference*, pp. 539–548, 2018.

Tsourakakis, C. E. Fast counting of triangles in large real networks without counting: Algorithms and laws. In *2008 Eighth IEEE International Conference on Data Mining*, pp. 608–617. IEEE, 2008.

Tsourakakis, C. E., Pachocki, J., and Mitzenmacher, M. Scalable motif-aware graph clustering. In *Proceedings of the 26th International Conference on World Wide Web*, pp. 1451–1460, 2017.

Vignac, C., Krawczuk, I., Siraudin, A., Wang, B., Cevher, V., and Frossard, P. Digress: Discrete denoising diffusion for graph generation. *arXiv preprint arXiv:2209.14734*, 2022.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

Wang, H., Wang, J., Wang, J., Zhao, M., Zhang, W., Zhang, F., Xie, X., and Guo, M. Graphgan: Graph representation learning with generative adversarial nets. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Watts, D. J. and Strogatz, S. H. Collective dynamics of 'small-world'networks. *nature*, 393(6684):440–442, 1998.

Yin, H., Benson, A. R., Leskovec, J., and Gleich, D. F. Local higher-order graph clustering. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 555–564, 2017.

You, J., Ying, R., Ren, X., Hamilton, W., and Leskovec, J. GraphRNN: Generating realistic graphs with deep auto-regressive models. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pp. 5708–5717, 2018.

Young, S. J. and Scheinerman, E. R. Random dot product graph models for social networks. In *Algorithms and Models for the Web-Graph: 5th International Workshop, WAW 2007, San Diego, CA, USA, December 11-12, 2007. Proceedings 5*, pp. 138–149. Springer, 2007.

Zhu, Y., Du, Y., Wang, Y., Xu, Y., Zhang, J., Liu, Q., and Wu, S. A survey on deep graph generation: Methods and applications. *arXiv preprint arXiv:2203.06714*, 2022.

A. Deferred Proofs and Theoretical Results

A.1. Proof of Lemma 2

We prove Lemma 2 using the following theorem from Friedgut (2004) (see also Finner (1992)), which can be seen as a continuous version of Inequality 2:

Theorem 4 (Friedgut (2004))**.** *Let X, Y, Z be three independent probability spaces and let*

$$f : X \times Y \to \mathbb{R}, \quad g : Y \times Z \to \mathbb{R}, \quad \textit{and} \quad h : Z \times X \to \mathbb{R}$$

be functions that are square-integrable with respect to the relevant product measures. Then

$$\int f(x,y)g(y,z)h(z,x)\,dx\,dy\,dz \leq \sqrt{\int f^2(x,y)\,dx\,dy \int g^2(y,z)\,dy\,dz \int h^2(z,x)\,dz\,dx}.$$

Proof of Lemma 2. Consider any set of three distinct nodes (i, j, k). Then, by assumption, the embeddings of these nodes Z_i, Z_j, Z_k are independent random variables. Let $\rho_{Z_i}, \rho_{Z_j}, \rho_{Z_k}$ be the PDFs of the respective nodes' embeddings. Recall that there exists a symmetric function $e : \mathbb{R}^k \times \mathbb{R}^k \to [0, 1]$ of two nodes' embeddings that determines the probability of an edge between them. Based on this, the probability that nodes i, j, k form a triangle is:

$$\mathbb{E}[\boldsymbol{A}_{ij}\boldsymbol{A}_{jk}\boldsymbol{A}_{ik}] = \int \rho_{Z_i}(z_i)\rho_{Z_j}(z_j)\rho_{Z_k}(z_k)e(z_i,z_j)e(z_j,z_k)e(z_i,z_k)\,dz_i\,dz_j\,dz_k$$

Now, define $f(z_i, z_j) = \sqrt{\rho_{Z_i}(z_i) \cdot \rho_{Z_j}(z_j)} \cdot e(z_i, z_j)$, $g(z_j, z_k) = \sqrt{\rho_{Z_j}(z_j) \cdot \rho_{Z_k}(z_k)} \cdot e(z_j, z_k)$, and $h(z_i, z_k) = \sqrt{\rho_{Z_i}(z_i) \cdot \rho_{Z_k}(z_k)} \cdot e(z_i, z_k)$, so that

$$\mathbb{E}[\boldsymbol{A}_{ij}\boldsymbol{A}_{jk}\boldsymbol{A}_{ik}] = \int f(z_i,z_j)g(z_j,z_k)h(z_i,z_k)\,dz_i\,dz_j\,dz_k$$

Now, we can apply Theorem 4, yielding:

$$\begin{aligned}
&\mathbb{E}[\boldsymbol{A}_{ij}\boldsymbol{A}_{jk}\boldsymbol{A}_{ik}] \leq \sqrt{\int f^2(z_i,z_j)\,dz_i\,dz_j \int g^2(z_j,z_k)\,dz_j\,dz_k \int h^2(z_i,z_k)\,dz_i\,dz_k} \\
&= \sqrt{\int \rho_{Z_i}(z_i)\rho_{Z_j}(z_j)e^2(z_i,z_j)\,dz_i\,dz_j \int \rho_{Z_j}(z_j)\rho_{Z_k}(z_k)e^2(z_j,z_k)\,dz_j\,dz_k \int \rho_{Z_i}(z_i)\rho_{Z_k}(z_k)e^2(z_i,z_k)\,dz_i\,dz_k} \\
&\leq \sqrt{\int \rho_{Z_i}(z_i)\rho_{Z_j}(z_j)e(z_i,z_j)\,dz_i\,z_j \int \rho_{Z_j}(z_j)\rho_{Z_k}(z_k)e(z_j,z_k)\,dz_j\,dz_k \int \rho_{Z_i}(z_i)\rho_{Z_k}(z_k)e(z_i,z_k)\,dz_i\,dz_k}.
\end{aligned}$$

where the last inequality simply uses the fact that since the image of the function e is $[0, 1]$, it must be that $e^2(x, y) \leq e(x, y)$. Finally, the lemma follows from observing that

$$\int \rho_{Z_i}(z_i)\rho_{Z_j}(z_j)e(z_i,z_j)\,dz_i\,dz_j = \mathbb{E}[\boldsymbol{A}_{ij}]. \qquad \square$$

A.2. Generalized Bounds for Squares and Other k-cycles

We can extend the prior bounds on triangles to bounds on the expected number of k-cycles in graphs sampled from the generative model $\mathcal{A}$ in terms of $\mathrm{Ov}(\mathcal{A})$. For the adjacency matrix $\boldsymbol{A}$ of a graph G, let $C_k(\boldsymbol{A})$ denote the number of k-cycles in G. We can prove the following bounds, where the random graph models from Section 3 also provide tight examples for any generalized k-cycles.

Theorem 5 (Bound on Expected k-cycles)**.** *Let $\boldsymbol{A}$ be an adjacency matrix sampled from a graph generative model $\mathcal{A}$, and let $C_k(\boldsymbol{A})$ denote the number of k-cycles in the graph corresponding to $\boldsymbol{A}$. If $\mathcal{A}$ is edge independent, node independent, or arbitrarily dependent then $\mathbb{E}[C_k(\boldsymbol{A})]$ is bounded above asymptotically by $n^k \cdot \mathrm{Ov}(\mathcal{A})^k$, $n^k \cdot \mathrm{Ov}(\mathcal{A})^{k/2}$, and $n^k \cdot \mathrm{Ov}(\mathcal{A})$, respectively.*

Proof. For notational simplicity, we focus on $k = 4$. The proof directly extends to general k. Let $C_4(G)$ be the number of non-backtracking 4-cycles in G (i.e. squares), which can be written as

$$\mathbb{E}_{\boldsymbol{A}\sim\mathcal{A}}\left[C_4(\boldsymbol{A})\right] = \frac{1}{8}\cdot\sum_{i=1}^{n}\sum_{j\in[n]\setminus\{i\}}\sum_{k\in[n]\setminus\{i,j\}}\sum_{\ell\in[n]\setminus\{i,j,k\}}\boldsymbol{A}_{ij}\boldsymbol{A}_{jk}\boldsymbol{A}_{k\ell}\boldsymbol{A}_{\ell i}.$$

The $1/8$ factor accounts for the fact that in the sum, each square is counted 8 times – once for each potential starting vector i and once for each direction it may be traversed. For general k-cycles this factor would be $\frac{1}{2k}$. We then can bound

$$\mathbb{E}_{\boldsymbol{A}\sim\mathcal{A}}\left[C_4(\boldsymbol{A})\right] \leq \frac{1}{8}\cdot\sum_{i\in[n]}\sum_{j\in[n]}\sum_{k\in[n]}\sum_{\ell\in[n]}\boldsymbol{A}_{ij}\boldsymbol{A}_{jk}\boldsymbol{A}_{k\ell}\boldsymbol{A}_{\ell i}.$$

If $\mathcal{A}$ is an edge independent model, the bound on the expected number of 4-cycles proceeds like the one for triangles, except using the following variant of Cauchy-Schwarz:

$$\sum\nolimits_{ijkl} a_{ij}b_{jk}c_{kl}d_{li} \leq \sqrt{\sum\nolimits_{ij} a_{ij}^2 \sum\nolimits_{ij} b_{ij}^2 \sum\nolimits_{ij} c_{ij}^2 \sum\nolimits_{ij} d_{ij}^2},$$

which, letting $\boldsymbol{P} = \mathbb{E}[\boldsymbol{A}]$ be the edge probability matrix for $\boldsymbol{A}\sim\mathcal{A}$, yields

$$\begin{aligned}\mathbb{E}[C_4(\boldsymbol{A})] &= \tfrac{1}{8}\sum\nolimits_{ijkl}\boldsymbol{P}_{ij}\boldsymbol{P}_{jk}\boldsymbol{P}_{kl}\boldsymbol{P}_{li} \leq \tfrac{1}{8}\sqrt{\left(\sum\nolimits_{ij}\boldsymbol{P}_{ij}^2\right)^4}\\ &= \tfrac{1}{8}\left(2\cdot\mathrm{Ov}(\mathcal{A})\mathrm{Vol}(\mathcal{A})\right)^{4/2} = O\left(n^4\mathrm{Ov}(\mathcal{A})^4\right),\end{aligned}$$

where again the last step is applying Lemma 1.

If $\mathcal{A}$ is an arbitrarily dependent model, the proof of the bound carries through almost exactly:

$$\begin{aligned}\mathbb{E}[C_4(\boldsymbol{A})] &= \sum_{i<j<k<l}\mathbb{E}\left[\boldsymbol{A}_{ij}\boldsymbol{A}_{jk}\boldsymbol{A}_{kl}\boldsymbol{A}_{li}\right] \leq \sum_{i<j<k<l}\mathbb{E}[\boldsymbol{A}_{ij}]\\ &= O(n^2)\sum\nolimits_{i<j}\mathbb{E}[\boldsymbol{A}_{ij}] = O(n^2)\cdot\mathrm{Vol}(\mathcal{A}) \leq O(n^4)\cdot\mathrm{Ov}(\mathcal{A}).\end{aligned}$$

□

If $\mathcal{A}$ is a node independent model, the bound on the expected number of 4-cycles follows as before, except using the following extended version of Theorem 4 (that we prove in Sec. A.3):

$$\begin{aligned}&\int f(x,y)g(y,z)h(z,w)l(w,x)\,dx\,dy\,dz\,dw\\ &\leq \sqrt{\int f^2(x,y)\,dx\,dy\int g^2(y,z)\,dy\,dz\int h^2(z,w)\,dz\,dw\int l^2(w,x)\,dw\,dx}.\end{aligned}$$

Applying this equation as before yields

$$\mathbb{E}[\boldsymbol{A}_{ij}\boldsymbol{A}_{jk}\boldsymbol{A}_{kl}\boldsymbol{A}_{li}] \leq \sqrt{\mathbb{E}[\boldsymbol{A}_{ij}]\mathbb{E}[\boldsymbol{A}_{jk}]\mathbb{E}[\boldsymbol{A}_{kl}]\mathbb{E}[\boldsymbol{A}_{li}]},$$

which allows us to apply the edge independent bound for squares:

$$\begin{aligned}\mathbb{E}[C_4(\boldsymbol{A})] &= \sum_{i<j<k<l}\mathbb{E}[\boldsymbol{A}_{ij}\boldsymbol{A}_{jk}\boldsymbol{A}_{kl}\boldsymbol{A}_{li}] \leq \sum_{i<j<k<l}\sqrt{\mathbb{E}[\boldsymbol{A}_{ij}]\mathbb{E}[\boldsymbol{A}_{jk}]\mathbb{E}[\boldsymbol{A}_{kl}]\mathbb{E}[\boldsymbol{A}_{li}]}\\ &\leq \sqrt{\tbinom{n}{4}\sum\nolimits_{i<j<k<l}\mathbb{E}[\boldsymbol{A}_{ij}]\mathbb{E}[\boldsymbol{A}_{jk}]\mathbb{E}[\boldsymbol{A}_{kl}]\mathbb{E}[\boldsymbol{A}_{li}]}\\ &\leq \sqrt{O\left((n^4)^2\cdot\mathrm{Ov}(\mathcal{A})^4\right)} = O(n^4\mathrm{Ov}(\mathcal{A})^{4/2}).\end{aligned}$$

A.3. Integral Inequality and Proof

Here we prove the following theorem that was referenced in Section 4.2:

Theorem 6. *For square integrable functions $f_1, \dots f_k$, we have:*

$$\int_{x_1}\int_{x_2}\dots\int_{x_k} f_1(x_1,x_2)\cdot f_2(x_2,x_3)\cdot\ldots\cdot f_k(x_k,x_1)\,dx_1\dots dx_k$$
$$\leq\sqrt{\int_{x_1}\int_{x_2} f_1^2(x_1,x_2)dx_1dx_2\cdot\ldots\cdot\int_{x_k}\int_{x_1} f_k^2(x_k,x_1)dx_kdx_1}$$

Before we prove this theorem, we first prove an intermediate result that will be useful:

Lemma 3. *For $k\geq 3$, we have:*

$$\int_{x_1}\int_{x_2}\left(\int_{x_3}\dots\int_{x_k} f_2(x_2,x_3)\cdot\ldots\cdot f_k(x_k,x_1)dx_3\dots dx_k\right)^2 dx_1dx_2$$
$$\leq\int_{x_2}\int_{x_3} f_2^2(x_2,x_3)dx_2dx_3\cdot\ldots\cdot\int_{x_k}\int_{x_1} f_k^2(x_k,x_1)dx_kdx_1$$

Proof. We proceed using induction. For $k=3$, we have

$$\int_{x_1}\int_{x_2}\left(\int_{x_3} f_2(x_2,x_3)\cdot f_3(x_3,x_1)dx_3\right)^2 dx_1dx_2$$
$$\leq\int_{x_1}\int_{x_2}\int_{x_3} f_2^2(x_2,x_3)dx_3\int_{x_3} f_3^2(x_3,x_1)dx_3dx_2dx_1$$
$$=\int_{x_2}\int_{x_3} f_2^2(x_2,x_3)dx_2dx_3\cdot\int_{x_1}\int_{x_3} f_3^2(x_3,x_1)dx_3dx_1,$$

where the first inequality follows from Cauchy-Schwarz.
Now assume the statement is true for some $k>3$. We will prove the induction step for $k+1$:

$$\int_{x_1}\int_{x_2}\left(\int_{x_3}\dots\int_{x_{k+1}} f_2(x_2,x_3)\cdot\ldots\cdot f_{k+1}(x_{k+1},x_1)dx_3\dots dx_{k+1}\right)^2 dx_1dx_2$$
$$=\int_{x_1}\int_{x_2}\left(\int_{x_3} f_2(x_2,x_3)\left(\int_{x_4}\int_{x_{k+1}} f_2(x_2,x_3)\cdot\ldots\cdot f_{k+1}(x_{k+1},x_1)\dots dx_{k+1}\right)dx_3\right)^2 dx_1dx_2$$
$$\leq\int_{x_2}\int_{x_3} f_2^2(x_2,x_3)dx_3dx_2\int_{x_1}\int_{x_3}\left(\int_{x_4}\dots\int_{x_{k+1}} f_3(x_3,x_4)\dots f_{k+1}(x_{k+1},x_1)dx_4\dots dx_{k+1}\right)^2 dx_3dx_1$$
$$\leq\int_{x_2}\int_{x_3} f_2^2(x_2,x_3)dx_2dx_3\cdot\ldots\cdot\int_{x_{k+1}}\int_{x_1} f_{k+1}^2(x_{k+1},x_1)dx_{k+1}dx_1,$$

where the first inequality follows from Cauchy-Schwarz, and the last one from the inductive hypothesis. □

Now, we are ready to prove Theorem 6.

Proof.

$$
\begin{aligned}
&\int_{x_1}\int_{x_2}\dots\int_{x_k} f_1(x_1,x_2)\cdot f_2(x_2,x_3)\cdot\ldots\cdot f_k(x_k,x_1)\,dx_1\dots dx_k\\
&=\int_{x_1}\int_{x_2} f_1(x_1,x_2)\left(\int_{x_3}\dots\int_{x_k} f_2(x_2,x_3)\cdot\ldots\cdot f_k(x_k,x_1)dx_3\dots dx_k\right)dx_2dx_1\\
&\le\sqrt{\int_{x_1}\int_{x_2} f_1^2(x_1,x_2)dx_1dx_2\int_{x_1}\int_{x_2}\left(\int_{x_3}\dots\int_{x_k} f_2(x_2,x_3)\cdot\ldots\cdot f_k(x_k,x_1)dx_3\dots dx_k\right)^2 dx_1dx_2}\\
&\le\sqrt{\int_{x_1}\int_{x_2} f_1^2(x_1,x_2)dx_1dx_2\cdot\ldots\cdot\int_{x_k}\int_{x_1} f_k^2(x_k,x_1)dx_kdx_1},
\end{aligned}
$$

where the first inequality follows from Cauchy-Schwarz and the second from Lemma 3. □

B. Further Details on Our Baseline Graph Generative Models

We provide additional details on our new graph generative models that were introduced in Section 5.1. Recall that our models are based on sampling a primary graph G_p based on the maximal cliques of the target (input) graph G_t, then a 'residual' graph G_r based on the node degrees of G_t, and finally returning the union of G_p and G_r. Specifically, we produce G_r by sampling from a simple edge independent model, the *odds product model* (Hoff, 2003; De Bie, 2011; Chanpuriya et al., 2021). This is as a variant of the well-known Chung-Lu configuration model (Aiello et al., 2001; Chung & Lu, 2002a;b), which produces graphs that match an input degree sequence in expectation. The odds product model does the same, except without constraints on the input degree sequence. In this work, we generalize this model to produce a sampled graph G_r such that its union with a G_p matches the input graph's degree distribution in expectation. Here we discuss some more details about fitting and sampling from the odds product model to produce G_r.

In the odds product model, there is a logit $\ell \in \mathbb{R}$ assigned to each node, and the probability of an edge occurring between two nodes i and j rises with the sum of their logits, and is given by $\sigma(\ell_i + \ell_j)$, where σ is the logistic function. To fit this model, we find a vector $\boldsymbol{\ell} \in \mathbb{R}^n$ of n logits, one for each node, such that a sample from the model has the same expected node degrees (i.e., row and column sums) as the original graph. This model is edge independent, so to use it, we simply find the expected adjacency matrix of edge probabilities, then sample entries independently to produce a graph.

We use largely the same algorithm as Chanpuriya et al. (2021) to fit the model, but with some modifications, since our goal is slightly more complex. Recall that in the context of this work, we actually sample from this model to produce a 'residual' graph with adjacency matrix $\boldsymbol{A_r}$; we then return the union $\boldsymbol{A_u}$ of this graph with the primary sampled graph $\boldsymbol{A_p}$, with the goal that, in expectation, $\boldsymbol{A_u}$ has the same node degrees as the input graph $\boldsymbol{A_i}$. (Note that this is a strict generalization of the same problem without the primary sampled graph, which is apparent from setting $\boldsymbol{A_p}$ to be all zeros.) Denote the degree sequence of the input graph and the expected degree sequence of the union graph by $\boldsymbol{d} = \boldsymbol{A_i}\mathbf{1} \in \mathbb{R}^n$ and $\hat{\boldsymbol{d}} = \mathbb{E}[\boldsymbol{A_u}]\mathbf{1} \in \mathbb{R}^n$, respectively. Fitting the model is a root-finding problem: we seek $\boldsymbol{\ell} \in \mathbb{R}^n$ such that the degree errors are zero, that is, $\hat{\boldsymbol{d}} - \boldsymbol{d} = \mathbf{0}$. We employ the multivariate Newton-Raphson method to find the root, for which we must calculate the Jacobian matrix $\boldsymbol{J}$ of derivatives of the degree errors with respect to the entries of $\boldsymbol{\ell}$. The following calculation largely mirrors that of Chanpuriya et al. (2021), though we must account for the union of graphs. If an edge is planted in the primary sample with probability p and in the residual sample with probability q, then it exists in the union with probability $1-(1-p)(1-q)$. In our notation,

$$
\begin{aligned}
\mathbb{E}[\boldsymbol{A_u}_{ij}] &= 1 - \left(1 - \mathbb{E}[\boldsymbol{A_p}_{ij}]\right)\cdot\left(1 - \mathbb{E}[\boldsymbol{A_r}_{ij}]\right)\\
&= 1 - \left(1 - \mathbb{E}[\boldsymbol{A_p}_{ij}]\right)\cdot\left(1 - \sigma(\boldsymbol{\ell}_i + \boldsymbol{\ell}_j)\right).
\end{aligned}
$$

Letting δ_{ij} be 1 if $i = j$ and 0 otherwise (i.e. the Kronecker delta), we have

$$
\begin{aligned}
\frac{\partial \hat{\boldsymbol{d}}_i}{\partial \boldsymbol{\ell}_j} &= \frac{\partial}{\partial \boldsymbol{\ell}_j} \sum\nolimits_{k\in[n]} \mathbb{E}[\boldsymbol{A}_{\boldsymbol{u}ik}] \\
&= \frac{\partial}{\partial \boldsymbol{\ell}_j} \sum\nolimits_{k\in[n]} \left(1 - \left(1 - \mathbb{E}[\boldsymbol{A}_{\boldsymbol{p}ik}]\right) \cdot \left(1 - \sigma(\boldsymbol{\ell}_i + \boldsymbol{\ell}_k)\right)\right) \\
&= \frac{\partial}{\partial \boldsymbol{\ell}_j} \left(1 - \left(1 - \mathbb{E}[\boldsymbol{A}_{\boldsymbol{p}ij}]\right) \cdot \left(1 - \sigma(\boldsymbol{\ell}_i + \boldsymbol{\ell}_j)\right)\right) \\
&\quad + \delta_{ij} \sum\nolimits_{k\in[n]} \frac{\partial}{\partial \boldsymbol{\ell}_i} \left(1 - \left(1 - \mathbb{E}[\boldsymbol{A}_{\boldsymbol{p}ik}]\right) \cdot \left(1 - \sigma(\boldsymbol{\ell}_i + \boldsymbol{\ell}_k)\right)\right) \\
&= \left(1 - \mathbb{E}[\boldsymbol{A}_{\boldsymbol{p}ij}]\right) \cdot \sigma(\boldsymbol{\ell}_i + \boldsymbol{\ell}_j) \cdot \left(1 - \sigma(\boldsymbol{\ell}_i + \boldsymbol{\ell}_j)\right) \\
&\quad + \delta_{ij} \sum\nolimits_{k\in[n]} \left(1 - \mathbb{E}[\boldsymbol{A}_{\boldsymbol{p}ik}]\right) \cdot \sigma(\boldsymbol{\ell}_i + \boldsymbol{\ell}_k) \cdot \left(1 - \sigma(\boldsymbol{\ell}_i + \boldsymbol{\ell}_k)\right) \\
&= \left(1 - \mathbb{E}[\boldsymbol{A}_{\boldsymbol{u}ij}]\right) \cdot \sigma(\boldsymbol{\ell}_i + \boldsymbol{\ell}_j) + \delta_{ij} \sum\nolimits_{k\in[n]} \left(1 - \mathbb{E}[\boldsymbol{A}_{\boldsymbol{u}ik}]\right) \cdot \sigma(\boldsymbol{\ell}_i + \boldsymbol{\ell}_k) \\
&= \mathbb{E}[\boldsymbol{A}_{\boldsymbol{r}ij}] \cdot \left(1 - \mathbb{E}[\boldsymbol{A}_{\boldsymbol{u}ij}]\right) + \delta_{ij} \sum\nolimits_{k\in[n]} \mathbb{E}[\boldsymbol{A}_{\boldsymbol{r}ij}] \cdot \left(1 - \mathbb{E}[\boldsymbol{A}_{\boldsymbol{u}ik}]\right).
\end{aligned}
$$

The expression for the gradient in this context closely resembles the one presented in Chanpuriya et al. (2021), with the only variation being the substitution of $\mathbb{E}[\boldsymbol{A}_{\boldsymbol{u}}]$ matrices with additional $\mathbb{E}[\boldsymbol{A}_{\boldsymbol{r}}]$ matrices. This substitution is reasonable since the referenced work does not involve a primary sampled graph, and therefore the union graph corresponds precisely to the residual graph. This modified gradient translates directly into a modified fitting algorithm, pseudocode for which is given in Algorithm 2.

Algorithm 2 Fitting the modified odds-product model

input : target degrees $\boldsymbol{d} \in \mathbb{R}^n$, primary expected adjacency $\mathbb{E}[\boldsymbol{A}_{\boldsymbol{p}}] \in [0,1]^{n\times n}$, error threshold ϵ
output : symmetric probability matrix $\mathbb{E}[\boldsymbol{A}_{\boldsymbol{r}}] \in [0,1]^{n\times n}$

$\boldsymbol{\ell} \leftarrow \boldsymbol{0}$ ▷ $\boldsymbol{\ell} \in \mathbb{R}^n$ is the vector of logits, initialized to all zeros
$\mathbb{E}[\boldsymbol{A}_{\boldsymbol{r}}] \leftarrow \sigma\left(\boldsymbol{\ell}\boldsymbol{1}^\top + \boldsymbol{1}\boldsymbol{\ell}^\top\right)$ ▷ σ is an entrywise logistic function, $\boldsymbol{1}$ is a length-n all-ones column vector
$\mathbb{E}[\boldsymbol{A}_{\boldsymbol{u}}] \leftarrow \boldsymbol{1}\boldsymbol{1}^\top - \left(\boldsymbol{1}\boldsymbol{1}^\top - \mathbb{E}[\boldsymbol{A}_{\boldsymbol{p}}]\right) \circ \left(\boldsymbol{1}\boldsymbol{1}^\top - \mathbb{E}[\boldsymbol{A}_{\boldsymbol{r}}]\right)$ ▷ $\circ$ is an entrywise product
$\hat{\boldsymbol{d}} \leftarrow \mathbb{E}[\boldsymbol{A}_{\boldsymbol{u}}]\boldsymbol{1}$ ▷ expected degree sequence of $\mathbb{E}[\boldsymbol{A}_u]$
while $\|\hat{\boldsymbol{d}} - \boldsymbol{d}\|_2 > \epsilon$ **do**
  $\boldsymbol{B} \leftarrow \mathbb{E}[\boldsymbol{A}_{\boldsymbol{r}}] \circ \left(\boldsymbol{1}\boldsymbol{1}^\top - \mathbb{E}[\boldsymbol{A}_{\boldsymbol{u}}]\right)$ $\boldsymbol{J} \leftarrow \boldsymbol{B} + \mathrm{diag}\left(\boldsymbol{B}\boldsymbol{1}\right)$ ▷ diag is the diagonal matrix with the input vector along its diagonal
  $\boldsymbol{\ell} \leftarrow \boldsymbol{\ell} - \boldsymbol{J}^{-1}\left(\hat{\boldsymbol{d}} - \boldsymbol{d}\right)$ ▷ rather than inverting $\boldsymbol{J}$, we solve this linear system
  $\mathbb{E}[\boldsymbol{A}_{\boldsymbol{r}}] \leftarrow \sigma\left(\boldsymbol{\ell}\boldsymbol{1}^\top + \boldsymbol{1}\boldsymbol{\ell}^\top\right)$ $\mathbb{E}[\boldsymbol{A}_{\boldsymbol{u}}] \leftarrow \boldsymbol{1}\boldsymbol{1}^\top - \left(\boldsymbol{1}\boldsymbol{1}^\top - \mathbb{E}[\boldsymbol{A}_{\boldsymbol{p}}]\right) \circ \left(\boldsymbol{1}\boldsymbol{1}^\top - \mathbb{E}[\boldsymbol{A}_{\boldsymbol{r}}]\right)$ $\hat{\boldsymbol{d}} \leftarrow \mathbb{E}[\boldsymbol{A}_{\boldsymbol{u}}]\boldsymbol{1}$
return $\mathbb{E}[\boldsymbol{A}_{\boldsymbol{r}}]$

C. Further Details on Experimental Setting

We now expand on the discussion of our experimental setting from Section 5.

Statistics. We choose 8 statistics that capture both the connectivity of a graph, as well local patterns within it. We look at the following network statistics:

- Pearson correlation coefficient (PCC) of the input and generated degree sequences (number of nodes incident to each node), as well as max degree.
- PCC of the input and generated triangle sequences (number of triangles each node belongs to).
- Triangle, 4-clique, and 4-cycle counts, each normalized by the respective counts in the input graph.
- Characteristic (average) path length and fraction of node pairs which are connected. Letting $|C_i|$ be the size of i-th connected component, the latter quantity is $\sum_i \binom{|C_i|}{2} / \binom{n}{2}$).

Datasets. In our experiments we use the following eight publicly available datasets (together with one synthetic dataset) that we describe next. Table 2 also provides a summary of them.

- CITESEER: A graph of papers from six scientific categories and the citations among them
- CORA: A collection of scientific publications and the citations among them.
- PPI: A subgraph of the PPI network for Homo Sapiens. Vertices represent proteins and edges represent interactions.
- POLBLOGS: A collection of political blogs and the links between them.
- WEB-EDU: A web-graph from educational institutions.

We also include the following smaller sized datasets where we additional evaluate the GraphVAE method, as it is designed for datasets of that scale.

- LES MISERABLES: Co-appearance network of characters in the novel Les Miserables.
- FACEBOOK-EGO: A small-sized ego-network of Facebook users.
- WIKI-ELECT: A (temporal) network of Wikipedia users voting in administrator elections for or against other users to be promoted to administrator. We use the first 15% of edges.
- RING OF CLIQUES: A union of 10 cliques of size 10 connected in a ring.

Table 2. Dataset summaries

Dataset	**Nodes**	**Edges**	**Triangles**
CITESEER (Sen et al., 2008)	2,110	3,668	1,083
CORA (Sen et al., 2008)	2,485	5,069	1,558
PPI (Stark et al., 2010)	3,852	37,841	91,461
POLBLOGS (Adamic & Glance, 2005)	1,222	16,714	101,043
WEB-EDU (Gleich et al., 2004)	3,031	6,474	10,058
LES MISERABLES (Knuth, 1993)	77	254	467
WIKI-ELECT (Leskovec et al., 2010b)	367	1,475	2,249
FACEBOOK-EGO (Leskovec & Mcauley, 2012)	324	2,514	10,740
RING OF CLIQUES (synthetic dataset)	100	4,600	1,200

Hyperparameters and ranging overlap. We range overlap in the following way for each method:

- **MCEI**, **MCNI**, **MCAD**: We range, respectively, the probability p of including an edge of a clique, or node of a clique, or a clique in the sampled graph. We typically use evenly spaced numbers for p over the interval between 0 and 1. For graphs with larger number of maximal cliques (PolBlogs and PPI), we additionally take the square of these values to decrease the probability.
- **CELL**: We set the dimension of the embeddings at 32 for the larger datasets and 8 for the smaller ones. In order to range overlap, we follow the same early-stopping approach as in the original paper. Namely, we stop training at every iteration and sample 10 graphs every time the overlap between the generated graphs and the input graph exceeds some value (we set 10 equally spaced thresholds between 0.05 and 0.75).
- **GraphVAE**: We use 512 dimensions for the graph-level embedding that precedes the MLP and 32 dimensions for the hidden layers of the graph autoencoder. Again, we follow the early-stopping approach during training. We use the following implementation: `https://github.com/kiarashza/graphvae-mm`.
- **VGAE**: As this method relies on a dot-product kernel that has been shown to be unable to generate sparse and triangle-dense graphs, instead of fixing the embedding dimensions at a small number, we increase the dimensions of the hidden layers from 4 to 1024 and train for $5,000$ epochs. We use the publicly available implementation by the authors of this method: `https://github.com/tkipf/gae`.

Implementation. Our implementation of the methods we introduce is written in Python and uses the NumPy (Harris et al., 2020) and SciPy (Virtanen et al., 2020) packages. Additionally, to calculate the various graph metrics, we use the following packages: MACE (MAximal Clique Enumerator) (Takeaki, 2012) for maximal clique enumeration (it takes $O(|V||E|)$ time for each maximal clique) and Pivoter (Jain & Seshadhri, 2020) for k-clique counting.

D. Additional Statistics Plots

We also include plots for the statistics of the remaining networks: CITESEER, CORA, POLBLOGS, WEB-EDU, WIKIELECT, FACEBOOK-EGO, and the synthetic RING OF CLIQUES graph. The results largely follow the same pattern as results for the dataset presented in Section 5. Note that our baseline methods typically outperform other methods in generating a graphs with a higher number of triangles in a low overlap regime. This is especially more evident in sparse networks, e.g., compare CORA with POLBLOGS.

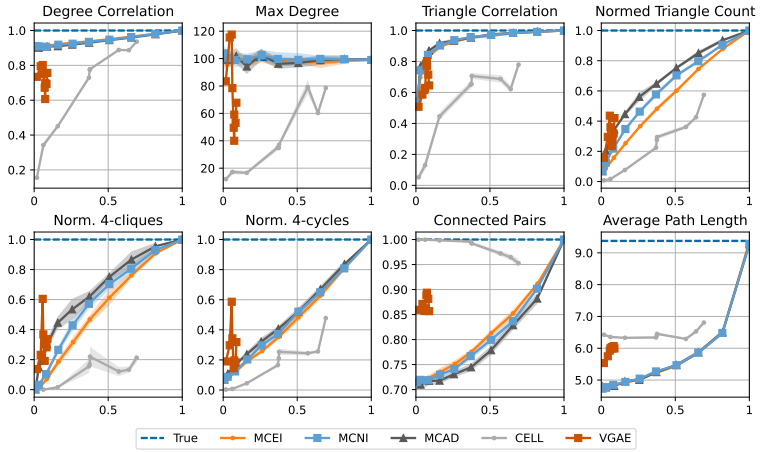

Figure 3. Statistics for CITESEER as a function of the *overlap*.

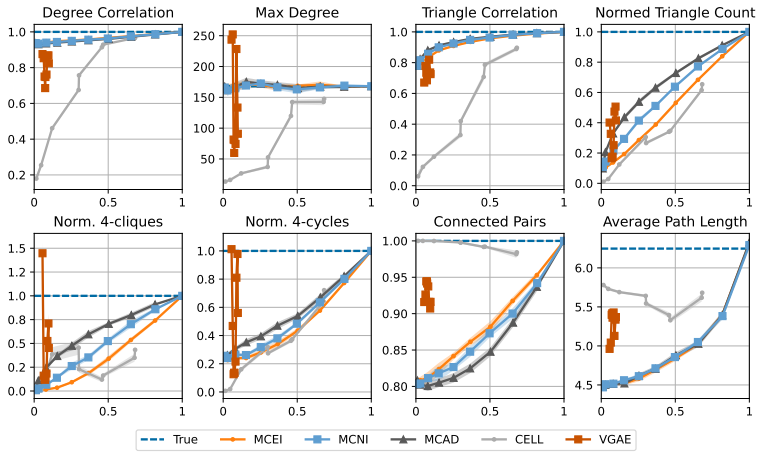

Figure 4. Statistics for CORA as a function of the *overlap*.

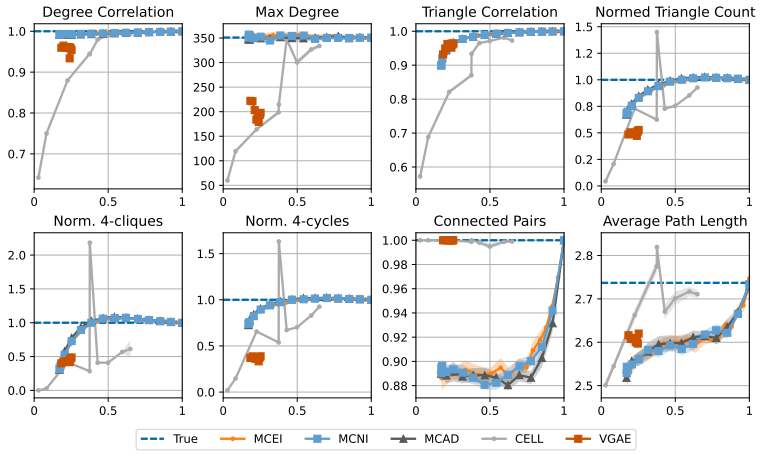

Figure 5. Statistics for POLBLOGS as a function of the *overlap*.

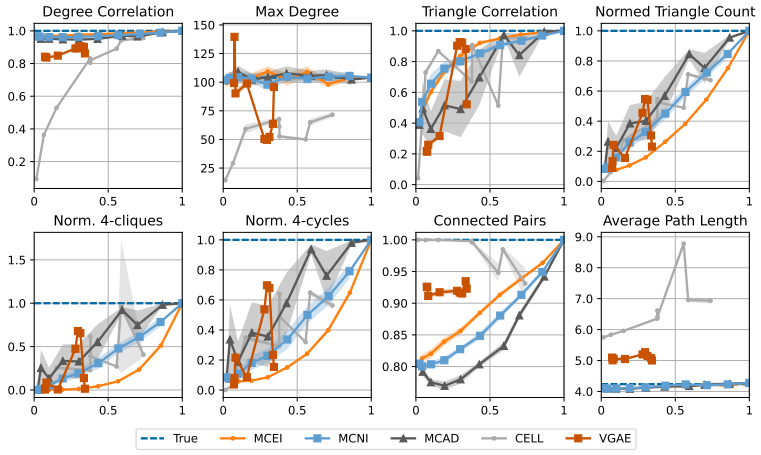

Figure 6. Statistics for WEB-EDU as a function of the *overlap*.

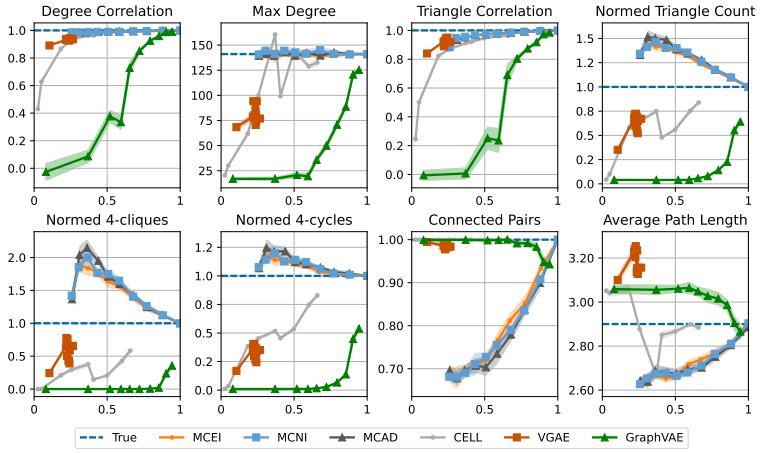

Figure 7. Statistics for WIKIELECT as a function of the *overlap*.

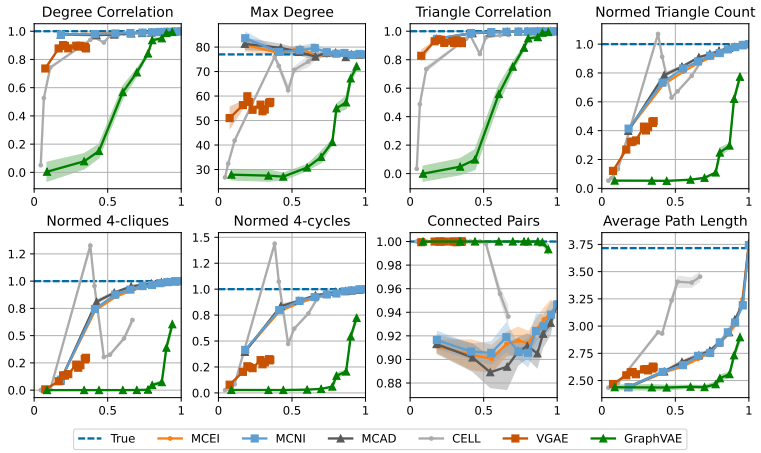

Figure 8. Statistics for FACEBOOK-EGO as a function of the *overlap*.

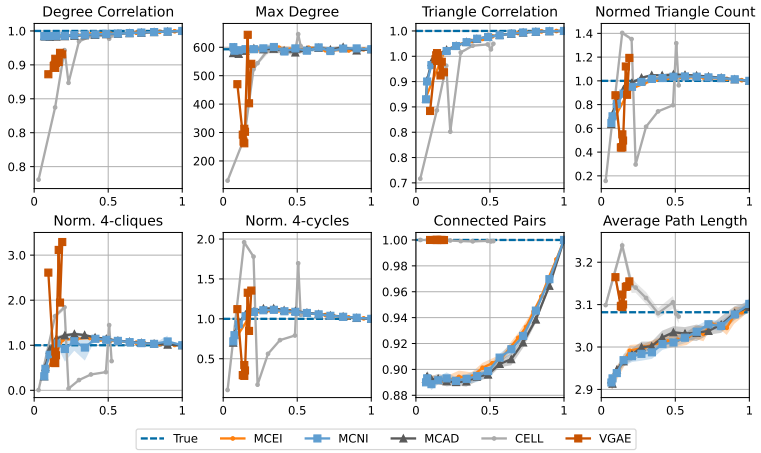

Figure 9. Statistics for PPI as a function of the *overlap*.

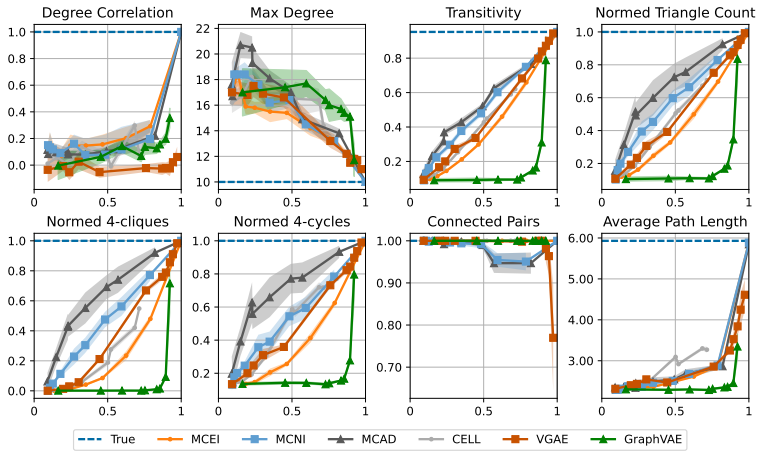

Figure 10. Statistics for the RING OF CLIQUES graph. All nodes are incident to the same number of triangles, so correlation is not defined for this graph. We replace it with transitivity.