# GNNs for Realistic Synthetic Social Networks

Anonymous Author(s)

## ABSTRACT

Social network data sees great amounts of research and is very widely used, but due to financial or privacy concerns, cannot be easily or openly shared without significant alterations. Synthetic network datasets, with realistic topologies, have the potential to allow easier and safer distribution of such data. Widely applied models for synthetic social network generation are currently rule-based and struggle to reproduce structural dynamics. In other domains, for example, molecules and proteins, recent Graph Neural Network (GNN) models have demonstrated highly expressive abilities in generating graph structures. Given the issues with existing methods for synthetic social network generation, we demonstrate the potential of GNNs for synthetic social networks using network datasets from Facebook, Twitch, GitHub and Deezer. With structural and social network specific measurements, we evaluate how realistically synthetic networks behave in typical social network analysis applications. We find that the Gated Recurrent Attention Network (GRAN), a GNN model extends well to social networks, and in comparison to the rule-based methods R-MAT (Recursive-MATrix) and BTER (Block Two-level Erdos-Renyi), is significantly better able to produce a variety of social networks. GRAN out-performs both rule-based methods, specifically on structural and social network metrics.

**ACM Reference Format:**
Anonymous Author(s). 2023. GNNs for Realistic Synthetic Social Networks. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

## 1 INTRODUCTION

Social network data is widely used across a variety of domains such as understanding user behaviour, targeted advertising and law enforcement [4]. Despite the usefulness of studying social networks, researchers often cannot access social network data easily as those who have access cannot share their data. The obstacles to sharing are primarily privacy and security. Social network data also has financial value in several scenarios, primarily in advertising, so organisations may be reticent to share it with others.

There is work on anonymising social network data, where a network is altered to preserve the anonymity or the security of the information it contains, so that it can be shared and used more freely. For instance, Yuan et al. Yuan et al. [29] propose anonymisation through addition of "noise nodes" and Jian et al. Jian et al. [12] propose anonymisation under node and edge differential privacy.

*Conference'17, July 2017, Washington, DC, USA*
© 2023 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

However, anonymisation leads to degradation of utility of data. To this end, methods of creating synthetic networks have been developed, aimed at allowing the sharing of data, while alleviating concerns around privacy, security, or utility. These methods generally use a rule-based process to produce a social network topology, with a separate model to produce attributes, with topology and features then aligned as a final step. Darabi et al. [6] outline a generic framework for this process, which we reproduce in Figure 1.

Our contributions closely relate to work on graph generation, which presents complexities not present in other domains of generative models [31]. Models and algorithms must be resilient to variable node orderings and massively varied size-in-memory with no "standard" sizings as with, for example, image resolutions. Currently employed graph generation methods are generally rule-based [5] and consistently struggle to generate realistic structure. The ability to generate realistic networks would allow easier sharing of data between researchers without the issues that follow using real data and without the compromise of using un-realistic network structures.

There has been the recent development of powerful Graph Neural Network (GNN) models [17]. Having been used for social network analysis tasks, when turned to generation in other fields, they have shown an ability to generate complex and meaningful structure [7]. We argue that these models represent a valuable alternative to the currently employed rule-based models in social network generation where topological structure plays an essential role. This represents an alternative for the Topology Generation component in Figure 1.

Explicitly our contributions are: (i) applying a GNN model, specifically GRAN Liao et al. [16], to the task of generating synthetic social networks; (ii) an algorithm, called Exploration Sampling With Replacement (ESWR), for creating a training dataset by sampling from a larger network; and (iii) an evaluation framework, using social network specific Maximum Mean Discrepancy (MMD)-based metrics to evaluate synthetic social networks, which here we use to compare networks created by GRAN to those from the rule-based method R-MAT [5]. GNNs have not been used to generate social networks. Adapting a GNN model and evaluating its potential to generate realistic social networks is a new contribution.

We choose GRAN, R-MAT and BTER for this study for following reasons. GRAN as it is one of the most efficient graph generation models currently. Among deep-learning methods, GRAN is able to generate networks of sizes similar to those usually found in the analysis of social networks. Given the in-memory cost of alternative GNN models, we surmise that only GRAN extends up to even a few hundred nodes with consistent results. R-MAT, on the other hand, represents an archetypal, simple rule-based model while having been demonstrated to generate networks with a power-law distribution in node degrees, as observed in social networks. BTER is then a more complex rule-based model, targetting social networks specifically, and following the trend of adapting a more simple model (here the Erdos-Renyi random graph model).

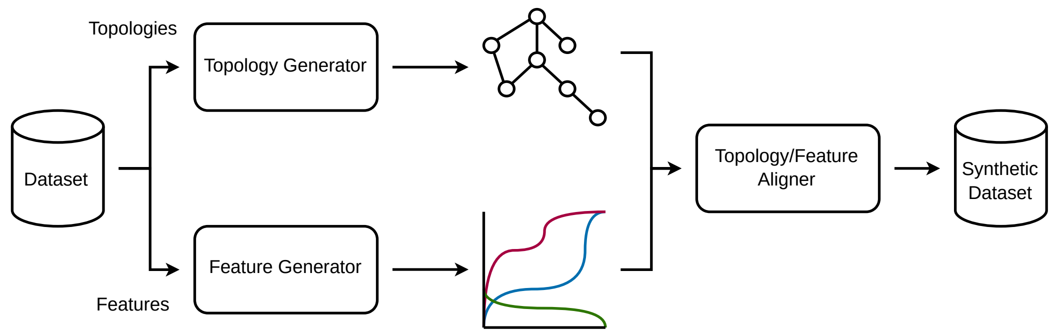

**Figure 1: A schematic of a generic synthetic social network generator like that proposed by Darabi et al. [6]. The dataset is broken down into features and topologies, which are then generated seperately, then finally aligned into a synthetic dataset. We propose that the topology generation stage could be improved using a GNN model.**

We focus on static networks as evaluating the quality of synthetic topologies is less opaque without accounting for temporal changes. We place emphasis on domain-specific applications via shortest path lengths, SIR simulation and Louvain community detection [8]. Our aim is not to reach the limits of performance with GRAN, but to evaluate whether a GNN model such as GRAN can be rapidly deployed by users to create synthetic social network topologies of a higher quality than those produced by rule-based methods.

The rest of this paper of organised as follows: Section 2 details the relevant work on synthetic social network generation, including rule-based and deep learning methods; Section 3 details our methodology; Section 4 covers our datasets and experiments, analysing our key findings; Section 5 presents our assumptions and any threats to validity; and finally Section 6 brings our final remarks.

## 2 BACKGROUND AND RELATED WORK

Graph generation is often used in domains with greatly varied graph sizes and characteristics, for example, in molecule generation [7]. Our work relates to rule-based methods and deep-learning methods for graph or network generation.

### 2.1 Rule-Based Methods for Graph Generation

One of the simplest rule-based models is that of Erdos and Renyi [9], who propose the Erdos-Renyi random graph model, where a given pair of nodes has a set probability of being connected. Watts and Strogatz [28] propose a random generator that exhibits common properties in real-world networks: high clustering coefficients and small diameters. Each node has a set number of neighbours and connections between nodes are randomly sampled. This presents issues in that each node retains the same number of neighbours resulting in networks that are over-simple or non-realistic.

Real-world networks present a power-law distribution in node degrees. A more recent generative method, R-MAT [5], generates via a recursive division of the original network's adjacency matrix, and results in networks with this power-law distribution. R-MAT remains commonly used in recent works, often adapted with iterative changes [11, 18, 19]. Similarly the model BTER [25] adapts the Erdos-Renyi model into hierarchies to target degree, clustering and to some extent community structure. Notably all of these methods may miss complex dynamics of social networks, in particular in generating meaningful large-scale structure, though the currently employed metrics for topology generation do not necessarily express this.

### 2.2 Deep-Learning Methods for Graph Generation

Deep-learning models have not widely been applied to real social network data. However, the recent development of Graph Neural Networks (GNNs) [17] has allowed development of models for network data structures, including network generation, with much recent work in chemistry.

Deep-learning approaches to network generation vary considerably. Zhao [32] provides an overview of network generation. Zhang et al. Zhang et al. [31] and Zhu et al. Zhu et al. [33] provide alternative reviews, including example applications. Some approaches model network construction as biased random walks, such as NetGAN [3]. Others are "one-shot" methods, for example GraphGAN [27]. Often models build on architectures from other forms of data with papers such as De Cao and Kipf [7] and Simonovsky and Komodakis [26] adapting GANs and VAEs. These models are limited to small networks due to issues presented with node orderings and the rapidly scaling memory requirements of networks with increasing size.

Recently models such as GRAN [16] and TG-GAN [30] are able to generate networks of the sizes found in social network analysis. The lack of application of models such as GRAN and TG-GAN on social network data specifically is what we seek to address. Additionally, other research has not considered metrics specific to social networks, as in this work.

## 3 METHODOLOGY

We denote a network as $G$. We consider un-directed networks to be a set of nodes $V : \{v_1, v_2, v_3, \ldots\}$ with edges between nodes $E : \{(v_i, v_j), (v_k, v_l), (v_m, v_o), \ldots\}$. In social network, a node could be a user and an edge between nodes could refer to a relationship between users or an interaction between users in a network.

### 3.1 Dataset Construction via ESWR

We propose Exploration Sampling With Replacement (ESWR) to construct a network dataset for experimentation with any graph-generation method. Here, we use it to generate datasets four our experiments, including training GRAN. ESWR takes as input a single large network, and outputs a dataset of sampled sub-networks. In other machine learning applications, exploration sampling with replacement is not advised, as models tend to overfit if they are presented the same data more than once. In graph generation, there are many degeneracies in how networks are presented to a deep-learning model, primary being node ordering. This means that while overlapping sections of the overall network may be presented to the model as separate inputs, the model is unlikely to overfit to these sections. We argue that ESWR presents advantages:

**Representative samples** Exploration sampling algorithms, for example diffusion sampling, are specifically designed to sample a network such that a given sample exhibits the same characteristics as the original network. This is not true of more simple methods like random node sampling.
**Size of dataset** Sampling *without* replacement would, after the first sample, be sampling a network with sections "missing", which would mean that samples have decreasing similarity to the original network. ESWR means that, while avoiding this issue, more nodes can be present in the sampled dataset than in the original network.

Algorithm 1 describes the application of ESWR to a network for generating an overall edgelist and a list of network identifiers. As previously, $G$ is the input network (Line 1), $E$ is a combined edgelist for all sampled sub-networks, and $V$ is a set of nodes. $G_{ind}$ (Line 2) is a list of identifiers to recover individual sub-networks from the combined edgelist $E$. SampleSize() (Line 5) randomly selects a size of sub-network from the normal distribution $N(\mu, \sigma)$. For each individual sample, a given set of nodes, $V_{sample}$, is sampled from the larger network, with the number of nodes, $|V_{sample}|$, given by SampleSize(), by sampling from $\mathcal{N}(\mu, \sigma^2)$, with $\mu = 400$ and $\sigma = 50$. ExplorationSampler($G, |V_{sample}|$) (Line 6) is call to an exploration sampling algorithm. An exploration sampler uses a graph traversal process to produce local samples of a given size from a network. For our exploration sampling algorithm, we use Metropolis Hastings Random Walk sampling, implemented in the Little Ball of Fur Python library [24].

The use of a normal distribution aims at giving a model the ability to learn a simple distribution for network size. We omit some elements for brevity, for example, re-indexing nodes for each sampled network.

### 3.2 GRAN Briefly

We now briefly describe the Graph Recurrent Attention Network (GRAN) [16], a recent and successful network generation model, that we employ in our work (see [16] for detailed description of GRAN). GRAN takes as input network dataset constructed via ESWR.

**Algorithm 1:** ESWR to construct a dataset of sub-networks from a single large network

```
1 $G \leftarrow$ LargeNetwork ;                // Original, $|G| > \mu$
2 $E \leftarrow []$;
3 for $i \leftarrow 0$ to $N_{networks}$ do
4     $|V_{sample}| \leftarrow$SampleSize();
5     $G_{sample} \leftarrow$ ExplorationSampler($G, |V_{sample}|$);
6     $E \leftarrow E + E_{sample}$
```

GRAN is a "graph-motif" auto-regressive model, meaning that it adds nodes and edges in blocks, with each motif at step $t$ denoted $C_t$. The primary issue that GRAN aims to solve is where to add new motifs $C_t$ to the previously constructed sub-network $G_{t-1}$.

With motif size $B = |C_i|$, at step $t$, there are $B(t-1)$ nodes in the current network. At step $t$, GRAN adds $B$ new nodes, and fully connects them to the existing network. From here GRAN employs a Message-Passing Neural Network (MPNN, [10]), and after several rounds of message passing, node states are passed to a multi-layer perceptron (MLP) to calculate a Bernoulli distribution for edge existence (Equation 1). $\theta_{i,j}$ parameterises the Bernoulli distribution for the edge existence between $i$ and $j$ via the aforementioned MLP$_\theta$. The resulting network is then used as input for the next block $C_{t+1}$, and so the model generates auto-regressively.

$$P(C_t|C_{<t}) = \prod_{B_{(t-1)}<i<B_t} \cdot \prod_{1\le j\le i} p(A^{\pi}_{i,j}|C_{<t}) \quad (1)$$

$$= \beta \prod_{B_{t-1}<i<B_t} \cdot \prod_{1\le j\le i} \theta_{i,j} \quad (2)$$

$$\theta_{i,j} = \text{Sigmoid}(\text{MLP}_\theta(h_{v_i} - h_{v_j})) \quad (3)$$

### 3.3 Model Use

For fair evaluation, as standard with most machine learning and deep-learning applications, we split the network dataset 80:20 into training and testing. We model realistic use as having access to moderate computing resources, as might be found in a desktop workstation, but not to a full HPC facility. We assume time constraints of a few days in constructing a synthetic dataset with around 24 hours in training time available. We do not include a full hyper-parameter optimisation process, and also avoid excessive training duration. The aim of this work is to evaluate the performance of a GNN model in this application under reasonable constraints, instead of the maximum performance possible with unlimited resources. Our aim is to demonstrate that a GNN model can perform well off-the-shelf, in the same manner as rule-based models.

### 3.4 Evaluating Generated Networks

In network generation, as in other areas of data generation, measurement of quality is challenging. Unlike in other domains of generation, such as in tabular data, direct metrics such as Mean Absolute Error (MAE) or Mean Squared Error (MSE) could not be

**Table 1: Details of our large network datasets, sourced from [22, 23].**

| | Facebook | GitHub | Twitch | Deezer |
|---|---|---|---|---|
| Number of Networks | 200 | 200 | 200 | 9,629 |
| Minimum Nodes | 279 | 288 | 280 | 11 |
| Maximum Nodes | 531 | 525 | 533 | 363 |
| Minimum Communities | 9 | 13 | 12 | 1 |
| Maximum Communities | 24 | 21 | 19 | 12 |
| Minimum Density | 0.009,45 | 0.006,21 | 0.008,4 | 0.015 |
| Maximum Density | 0.036,1 | 0.013,8 | 0.014,5 | 0.909 |

applied here. Additionally a human qualitative understanding of performance, unlike in images or text, is not feasible. Graph metrics generally describe either the network as a whole, or a node within a network, respectively termed network-level and node-level metrics.

One way to compare effectiveness of a network generation method is to compute a characteristic of the network generated by the method and compare it with that of the original dataset. This can be as simple as comparing mean values [1], or more expressively make use of Maximum Mean Discrepancy (MMD) to allow variation across the dataset. Notably the metrics most often applied are not domain specific. We propose that demonstrating the quality of synthetic social networks requires consideration of their use in social network analysis. To this end we base metrics on common algorithms, specifically SIR (Susceptible, Infected, Recovered) simulation to model information movement across the network, Louvain community detection, and shortest path lengths [8].

## 4 EXPERIMENTS

Here we detail the experiments we perform and their results.

### 4.1 Data

We sample several large network datasets published by Rozemberczki et al. Rozemberczki et al. [22] using ESWR. We use the version available from the Stanford SNAP repositories [15]. For each dataset, we sample 200 sub-networks. Table 1 provides details for the resulting datasets after sampling. We also include a larger dataset of ego networks, which is sampled from the music social network site Deezer [1], published by Rozemberczki et al. [23]. Application of community detection here may be less useful, as we do not expect meaningful community structures in networks of this type.

#### 4.1.1 *Assessment of Network Characteristics.*

**Deezer.** Ego networks have the defining property of having a central "hub" node to which all other nodes connect. This is also true of the Deezer networks. The nature of these networks might prove difficult to encode and reproduce, as it is semantically complex, and could require modelling beyond a degree distribution in larger networks. The Deezer networks are generally small, so are easy to visualise, making this dataset something of a toy example to examine the potential benefits of using a GNN model.

[1]A music/social network platform, https://www.deezer.com/

**Table 2: Parameter selection for GRAN models.**

| Param | Value | Param | Value |
|---|---|---|---|
| Node Order | DFS | Mixture Components | 20 |
| Block Size | 1 | Sample Stride | 1 |
| Hidden Dimensions | 512 | Maximum Nodes | 550 |
| Embedding Dimensions | 512 | GNN Layers | 7 |
| GNN Prop. | 1 | | |

**Facebook.** For Facebook network samples, we expect tight clusters, joined by low degree "arcs" of nodes. This represents pages of the same type, or in the same community, with other pages bridging the gaps between these communities. As the number of nodes in a network increases, we expect a large range in numbers of communities found by application of Louvain detection. Small Facebook sampled networks will have higher density than those sampled from other networks, as seen in Table 1, and a wide range of node degrees.

**Twitch, GitHub.** Graphs sampled from Twitch are similar to those sampled from Facebook but with fewer and larger clusters. As seen from Table 1, this manifests in a smaller range in the number of communities found by Louvain detection, and lower minimum and maximum densities. GitHub sampled networks have lower densities than Twitch sampled networks. Due to the characteristics of these networks, graph layout does not show distinctive characteristics, so we do not use their visualisations.

### 4.2 GNN Model

We use GRAN [16] as a representative GNN model for network generation. Following the precedent set in the original work's training, we chose GRAN parameters and architecture depth as detailed in Table 2. We use these parameters for each dataset.

#### 4.2.1 *Training Parameters.*

We aim to constrain resources and computation time to less than a day, which we argue constitutes a rapid deployment, and so we limit training to 30,000 epochs for each dataset. An exception is the model fit on the Deezer networks, which is trained for 10,000 epochs.

The size in memory of a network varies considerably with the size of the node set, and further, networks with the same number of nodes may have greatly varying numbers of edges (to a limit of $0 \leq |E| \leq \frac{1}{2}|V|^2 - |V|$). This leads to batches with the same numbers of networks having hugely variable size in memory, so we limit batch sizes for the three sampled datasets (Facebook, Twitch, GitHub) to 50. The Deezer networks are smaller, reducing the uncertainty in memory usage, so we are able to set a batch size of 200. As in Liao et al. [16], we take a train:test split of 80:20. For the three datasets constructed through ESWR, this means a train set of 160 networks and a test set of 40. For the Deezer dataset, the train set is 7,703 networks and test set is 1,925 networks.

#### 4.2.2 *Training Resources.*

We conduct the training and model evaluation on a desktop workstation, in accordance with the constraints detailed in Section 3.3. Appendix B in the supplement includes the workstation specification. While this workstation is powerful compared to consumer-level computers, in the context of deep-learning research where access to HPC facilities can be assumed, it is not excessively so.

## 4.3 Benchmarking

As a benchmark for performance, we compare GRAN with R-MAT [5] and BTER [25], rule-based methods, which are currently employed for synthetic datasets. For evaluation R-MAT and BTER are, as with GRAN, applied to the test-set data. These benchmark models are fit and sampled from each test-set graph individually.

## 4.4 Metrics

We employ common node and network-level metrics, as well as metrics tailored to social network analysis, with quality calculated primarily via Maximum Mean Discrepancy (MMD). Lower MMD scores indicate more similar distributions with identical distributions scoring $MMD = 0$.

MMD requires the measurement of the distribution of a given quantity. GRAN implementation [16] packages the MMD scores for number of nodes, degree, clustering and spectral eigenvalues, to which we add metrics specific to social network analysis:

**MMD$_{\textbf{Nodes}}$** The number of nodes in each sub-network, $|V|$. We employ a Gaussian kernel to calculate MMD. Note that Liao et al. [16] use a Gaussian Total Variance (TV) kernel.

**MMD$_{\textbf{Degree}}$** The degree of each node, denoted $d(v)$ for a node $v$. Calculated using a Gaussian kernel.

**MMD$_{\textbf{Clustering}}$** The fraction of possible triangles through a node that exist, i.e., the extent to which a node and its neighbours are a complete network. Calculated using a Gaussian kernel. Clustering $c_v$ for a node $v$ is calculated as

$$c_v = \frac{2T(v)}{d(v)(d(v)-1)} \tag{4}$$

where $T(v)$ is the number of triangles through $v$, and $d(v)$ is the degree of $v$. Dense clusters in networks have high clustering coefficients, due to their high inter-connectedness, whereas more sparse regions have low clustering coefficients.

**MMD$_{\textbf{Spectral}}$** For an un-directed network $G$, we can construct a diagonal matrix of node degrees $D$. From this we can calculate the Laplacian of the network, $L = D - A$, where $A$ is the network’s adjacency matrix. The normalised Laplacian is then:.

$$N = D^{-1/2} L D^{-1/2} \tag{5}$$

Taking the eigenvalues of this normalised Laplacian allows us to treat the network as defined by a spectrum. Spectral analysis takes a view of global network properties, unlike degree or clustering, which are local statistics. We calculate MMD$_{\text{Spectral}}$ using a Gaussian TV kernel.

**MMD$_{\textbf{Paths}}$** All node-node shortest path lengths in a network. The shortest path between two nodes is the path which contains the fewest nodes. Calculated using a Gaussian kernel.

**MMD$_{\textbf{Steps}}$** The number of steps an SIR simulation [13] on a network takes before termination. We implement a simple SIR model. All nodes begin as “Susceptible”, then a randomly selected subset of size $N = 2$ are set as “Infected”. Infected nodes at each iteration have a $\kappa = 0.04$ probability of infecting each of their Susceptible neighbours. Once a node has been Infected for $\gamma = 5$ iterations it becomes Recovered, can no longer infect its neighbours, and in our implementation, also cannot become reinfected. We measure the number of iterations until no further infection is possible, or until 100 iterations have taken place. This simulation is performed multiple ($n = 20$) times to account for the random selection of seed nodes. Calculated using a Gaussian kernel.

**MMD$_{\textbf{Saturation}}$** The proportion of nodes in a network that are in a “Recovered” state at the termination point of an SIR simulation. Calculated using a Gaussian kernel.

**MMD$_{\textbf{Louvain}}$** The number of communities found by application of Louvain community detection[2]. We use the implementation from the Python library Python-Louvain [2], with default parameters, i.e., with a resolution of 1. We run the algorithm multiple times for each sub-network to account for randomness, but given the increased stability of Louvain compared to SIR simulations, we perform only $n = 5$ runs. Calculated using a Gaussian kernel.

There is an important caveatte to mention when using these metrics. Though ubiquitous in graph generation works, MMD scores and aggregate metrics both rely on simple node-level measurements, and as such may not be sufficiently expressive. Issues with MMD are discussed more specifically in O’Bray et al. [21]. These issues would generally bias results towards small scale structure (eg. degree, clustering) instead of more complex many-node patterns.

## 4.5 Results

We now detail the results of our experiments.

*4.5.1 Empirical Evaluation for Quality of Topologies.* We empirically evaluate the generated topologies using the testing set of each dataset. We generate 40 networks using GRAN for each sampled dataset (Facebook, Twitch, Github) to match the size of each test set. We generate 1926 networks for the Deezer Ego data, matching the size of the test-set. Notably, for each model, we take only the largest component of each generated graph for metric computation.

GRAN demonstrates a tendency to produce separate connected components, so for measurement, we take the largest of these. This means that the number of nodes in the samples actually used for measurement is often less than the number found in the training or test networks. The same measurements in Table 1 are computed for the test and generated sets, and shown in Table 3.

GRAN models fit on networks with areas of higher sparsity (Facebook, Twitch) display a higher tendency to produce separate connected components, hence the significantly lower maximum and minimum node numbers compared to their test sets. This reduction in number of nodes likely also explains their higher maximum densities compared to their test sets, as larger networks tend to have more sparse regions, leading to lower densities. In contrast, for the more consistent GitHub networks, GRAN is much more closely able to match numbers of nodes, and has a far lower tendency to produce separate components. We discuss this further in Section 5.2.

Fit on our largest dataset, the Deezer networks, GRAN produces far smaller networks. The two distributions of network sizes peak at the same point, but the GRAN distribution is far shorter-tailed than that for real networks. GRAN produces some especially small connected components, which are actually smaller than any present

[2]Louvain community detection is a modularity-optimising, un-supervised algorithm for finding communities in networks.

**Table 3: Measurements of the test set networks and networks generated by GRAN for each of our datasets.**

| | Facebook | | GitHub | | Twitch | | Deezer | |
|---|---|---|---|---|---|---|---|---|
| | Real | GRAN | Real | GRAN | Real | GRAN | Real | GRAN |
| Minimum Nodes | 283.000 | 183.000 | 294.000 | 288.000 | 309.000 | 128.000 | 11.000 | 3.000 |
| Maximum Nodes | 516.000 | 451.000 | 489.000 | 488.000 | 520.000 | 328.000 | 163.000 | 35.000 |
| Minimum Density | 0.010 | 0.010 | 0.006 | 0.008 | 0.009 | 0.011 | 0.041 | 0.095 |
| Maximum Density | 0.029 | 0.051 | 0.014 | 0.015 | 0.014 | 0.024 | 0.895 | 1.000 |
| Minimum Communities | 11.000 | 8.000 | 13.000 | 12.000 | 13.000 | 10.000 | 1.000 | 1.000 |
| Maximum Communities | 20.000 | 18.000 | 20.000 | 17.000 | 18.000 | 17.000 | 10.000 | 6.000 |

**Table 4: MMD scores for each model on each dataset summarised in Table 1. Lower is better; bold text indicates a superior result.**

| Dataset | Model | $\text{MMD}_{\text{Nodes}}$ | $\text{MMD}_{\text{Deg.}}$ | $\text{MMD}_{\text{Clus.}}$ | $\text{MMD}_{\text{Spec.}}$ | $\text{MMD}_{\text{Steps}}$ | $\text{MMD}_{\text{Sat.}}$ | $\text{MMD}_{\text{Paths}}$ | $\text{MMD}_{\text{Louvain}}$ |
|---|---|---|---|---|---|---|---|---|---|
| Facebook | R-MAT | **0.519** | 0.0110 | 0.555 | 0.0429 | 0.00674 | 0.00723 | 0.107 | 0.177 |
| | BTER | 0.696 | 0.00725 | 0.211 | 0.0196 | **0.00403** | **0.00399** | 0.0971 | **0.0153** |
| | GRAN | 0.646 | **0.00174** | **0.120** | **0.00611** | 0.00464 | 0.00507 | **0.0164** | 0.0964 |
| GitHub | R-MAT | 0.696 | 0.0258 | 1.16 | 0.0255 | 0.00832 | 0.0106 | 0.0226 | **0.0851** |
| | BTER | 0.757 | 0.0290 | **0.108** | 0.0407 | **0.00198** | **0.00458** | **0.00723** | 0.137 |
| | GRAN | **0.519** | **0.0116** | 0.233 | **0.0136** | 0.0110 | 0.0119 | 0.0462 | 0.200 |
| Twitch | R-MAT | **0.636** | 0.00542 | 1.06 | 0.0105 | **0.00573** | 0.00784 | 0.111 | 0.128 |
| | BTER | 0.726 | 0.0161 | 0.236 | 0.0341 | 0.00565 | **0.00652** | 0.0279 | **0.0405** |
| | GRAN | 0.787 | **0.00203** | **0.121** | **0.00662** | 0.00863 | 0.0107 | **0.0239** | 0.183 |
| Deezer Ego | R-MAT | **0.00** | 0.0351 | 0.0442 | 0.0715 | 0.00573 | 0.00612 | 0.259 | **0.141** |
| | BTER | 0.236 | 0.0232 | 0.165 | 0.0158 | 0.0177 | 0.0184 | 0.203 | 0.151 |
| | GRAN | 0.184 | **0.00863** | **0.00684** | **0.0107** | **0.00216** | **0.00258** | **0.00563** | 0.282 |

in its training data. Having generated R-MAT and BTER networks, we compute the previously detailed MMD scores, using the test set for each dataset. Table 4 details these results.

With the exception of the GitHub dataset, as expected, $\text{MMD}_{\text{Nodes}}$ is lowest for R-MAT. This is unsurprising, as each R-MAT network specifically fits the number of nodes in each test set network. We might expect similar performance from BTER, but we found that BTER had a tendency to produce multiple connected components, resulting in higher MMD scores when metrics are computed on only the largest.

For sampled datasets (Facebook, GitHub, Twitch), GRAN outperforms R-MAT. This is particularly true of structural metrics where MMD scores for GRAN are generally much lower than that for R-MAT networks. This might indicate that GRAN more accurately reproduces the more complex aspects of social network topologies. In some cases, in particular SIR simulation-based metrics, BTER outperforms R-MAT. However on almost all other metrics, in particular $\text{MMD}_{\text{Spec.}}$, where GRAN consistently out-performs the rule-based models.

For networks generated based on Deezer networks results are more decisive. Structural scores are still superior for GRAN, but to a greater degree than on the other datasets. The most indicative result here is shortest paths, where the discrepancy in $\text{MMD}_{\text{Paths}}$ between GRAN, R-MAT and BTER is highest across any dataset we use. This may be due to R-MAT and BTER's inability to represent the "central node" nature of an ego network. The longest possible path across *any* ego network is two edges, but if R-MAT is unable to capture that ego networks are built around a central node, paths are longer. This is likely true of similar structures contained in larger graphs, but as previously stated, designing metrics that can express these topological dynamics is an open research problem.

*4.5.2 Visual Inspection for Quality of Topologies.* Figures 2 and 3 show networks generated from GRAN, R-MAT and BTER alongside their real counterparts for the Deezer Ego and Facebook datasets, respectively [3] Layouts are created via `networkx.spring_layout`[4], a force-based algorithm, with 100 iterations of simulation, and for GRAN-generated networks, only the largest connected component is displayed. We show only the largest connected components resulting from each generative method, and the reader should note that the R-MAT and BTER networks are not fit on the test-set networks displayed alongside them.

Visualisation of the Deezer networks in Figure 2 corroborates the findings of Section 4.5.1. The real ego networks have, crucially, one central node with which all other nodes share an edge. The real networks also have connections between non-central nodes, which in some cases are actually a complete network including the

[3] Appendix D in the supplement include additional figures comparing the networks generated from GRAN, R-MAT and BTER.
[4] https://networkx.org/documentation/stable/index.html

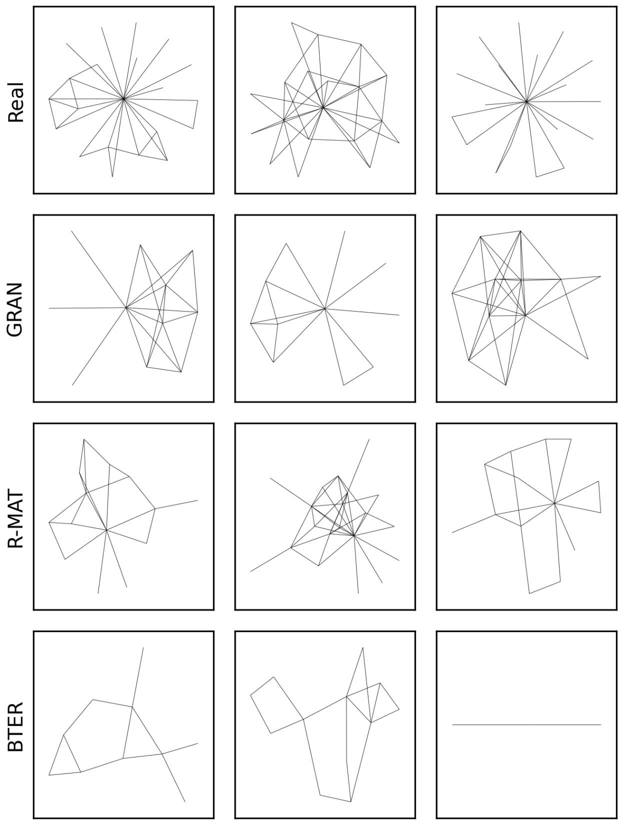

**Figure 2: Comparison of real networks and synthetic networks produced by GRAN and R-MAT trained on the Deezer dataset.**

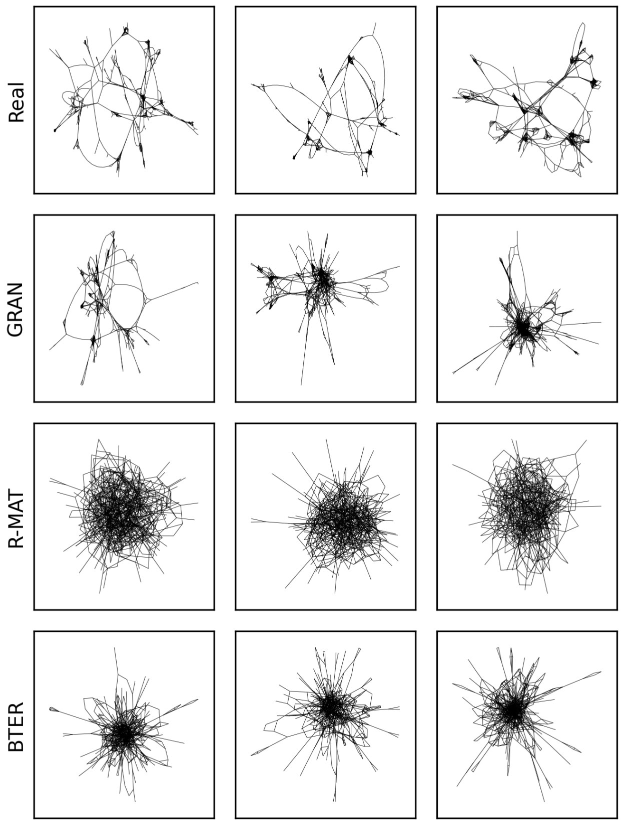

**Figure 3: Comparison of real networks and synthetic networks produced by GRAN and R-MAT trained on the Facebook dataset.**

central node. GRAN has properly re-created these features, albeit with fewer nodes than in the displayed test set examples.

R-MAT generated networks for the Deezer dataset do not have a central node. The degree distribution is close to correct, with one or two high-degree nodes and many low-degree nodes, but not the hub-and-spoke topologies in the test set or generated by GRAN. BTER graphs are of noticeably lower realism than those from GRAN or indeed R-MAT, with either many small connected components, or no central hub-and-spoke topology.

The real Facebook networks in Figure 3 show clusters and sparsity in-between. This is characterised visually by what we term "arcs", where nodes have two neighbours, leading to strings of nodes with no further structure. This is shown in the GRAN generated networks and to a lesser extent in the R-MAT and BTER networks. However, the rule-based networks seem essentially to be one large grouping of nodes, without any complex structure. Qualitative analysis here has confirmed the results of our quantitative analysis, in that GRAN out-performs rule-based models to a wide extent, in particular where structure is semantically complex or has a wide range of local densities within a network.

## 4.6 Takeaways

We find that GRAN can produce realistic synthetic social networks. The networks produced correspond closely to their real counterparts, including when measured using SIR simulations and Louvain community detection. However, GRAN shows a tendency to produce multiple connected components. The largest of these can be significantly smaller than those in the training data.

The size of network samples in future work should be topologically significant instead of simply sampling a given number of nodes. This may allow GRAN to more properly encode network size, and might alleviate the issues we find with multiple connected components. The sampling method could instead be Community Sampling With Replacement (CSWR), where repeated applications of community detection could be used to produce topologically meaningful partitions.

We compared the networks generated by the rule-based methods R-MAT and BTER with those generated by GRAN. We use this comparison to evaluate the compromise researchers might face when choosing between types of model.

On most quantitative metrics, GRAN performs better than our rule-based benchmarks, . This is especially true for structural and social network specific metrics, where GRAN strongly out-performs

R-MAT. The existing issues with rule-based models are clear in networks generated based on the Deezer networks. Here, R-MAT and BTER are unable to reproduce the essential nature of an ego network, whereas the more expressive GRAN produces realistic ego networks, as demonstrated by both visualisations and the distribution of shortest path lengths. We hypothesise that this would also be true of complex structures in larger networks, but that the current metrics employed in network generation are not expressive enough to show this.

Rule-based models can be used where reasonable but not realistic approximations of social networks are needed, for example, in applications where power-law degree distributions are paramount, or in rapid early testing of models. In situations where realism is essential, GRAN or another GNN model should be employed. This is especially true for those wishing to create synthetic datasets for distribution.

### 4.7 Code and Data Availability

Our code is available on GitHub[5], or as a compressed file on Google Drive[6], both published through anonymous accounts. Processed data for the Deezer, Facebook, Twitch and GitHub datasets is included through a bash script, which also pulls trained models. We also include individual edgelist files for each real, GRAN generated and rule-based generated social network. The original (unprocessed) data is available through the Stanford SNAP project [15].

## 5 ASSUMPTIONS, RISKS TO VALIDITY AND DIRECTIONS

We now detail our assumptions, along with other risks to the validity of our findings.

### 5.1 Sampling Procedure

In using ESWR, we assume that an exploration sampler is able to take representative samples. Leskovec and Faloutsos [14] find that exploration samplers perform well in this task, but also note that there is a lower limit to the size of a sample, which they set at around 15%. Our sub-network samples are (on average) below this size, which was in order to remain tractable as training samples for GRAN.

### 5.2 GRAN Generated Graph Size

GRAN adds nodes in blocks, then determines which nodes already in the network should share edges with the newly added nodes. GRAN can determine that a new node *shouldn't share any edges* with the existing network, in which case it is a new component of size one. The potential issue with our datasets is that the size of a network is determined not by network structure, but as an input parameter, which may not be encoded through topology.

This may be why GRAN was found to generate multiple unconnected components. If a single node mid-way through the network building process is un-connected, new nodes can be connected either to the existing network or to this new separate component, and successive nodes could be more likely to be attached to the new component as it grows. A potential remedy might be to sample sub-networks not according to number of nodes but instead according to some other network measure, for example communities or number of edges. These might be better encoded by GRAN or other deep-learning models.

[5]GitHub repo: https://github.com/anonymous-synonymous/GNN-Realistic-Social-Networks

[6]Google Drive: https://drive.google.com/file/d/1y0UwF56k0IiuBGmKKOqh9mYumjoWBM5r/view?usp=sharing

### 5.3 GRAN Training

A significant factor to consider is that we do not aim to produce the best possible performance with GRAN, and so we have not performed any hyper-parameter optimisation process, and trained only for a comparatively short time. Liao et al. Liao et al. [16] do not provide details of training duration, number of epochs, batch sizes and so on. Based on configuration files packaged with GRAN, specifically for the DB point cloud dataset from Neumann et al. [20], they train GRAN for 20k epochs on a dataset of 41 much larger networks than we use. These networks, while larger than ours, are $k$ nearest neighbour networks, so are arguably less topologically complex. We suspect that training GRAN for a longer duration—as appears to have been the case with the original Liao et al. [16] work—would bring greater performance. Performance could likely be improved further through hyper-parameter optimisation specific to each dataset.

## 6 CONCLUSIONS

We evaluate the potential of recent deep Graph Neural Network (GNN) models in generating synthetic social networks. For a benchmark comparison, we use Recursive-MATrix (R-MAT) [5] and the Block Two-level Erdos-Renyi (BTER) [25] rule-based models. GRAN demonstrated a propensity to generate multiple connected components in networks with regions of sparsity. A remedy to this issue with GRAN will be to use a more topologically consistent sampling method. As we simply employ an exploration sampler up until a given number of nodes are sampled, the size of the resulting network is not necessarily encode-able by GNN models. A sampling method with termination based on a topologically meaningful metric, employed in the same manner as we use in this work, is an area for future research.

GRAN, on all other metrics, out-performs R-MAT. On the majority of metrics it also out-performs BTER. This is particularly true of structural or social-network specific measurements such as shortest path lengths, where complex topological structure plays a large part. Our results do not mean that every GNN model would outperform every rule-based method, but they highlight the potential of GNN models for synthetic social network dataset creation.

Other avenues for future work include generating directed graphs, allowing the generated synthetic graphs to model common relationships in social networks such as when a user "follows" another user, e.g., a content creator, and is not followed back. It would also be useful to investigate methods to increase the potential number of nodes in generated graphs. Finally, because the main motivation behind generating synthetic social networks is to allow data sharing with anonymity preservation, it is important to evaluate privacy guarantees in the generated graphs.

## REFERENCES

[1] Awrad Mohammed Ali, Hamidreza Alvari, Alireza Hajibagheri, Kiran Lakkaraju, and Gita Sukthankar. 2014. Synthetic Generators for Cloning Social Network Data. In *ASE International Conference on Social Informatics*. 1–9.

[2] Vincent D. Blondel, Jean Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment* 2008, 10 (10 2008), 1–12. https://doi.org/10.1088/1742-5468/2008/10/P10008

[3] Aleksandar Bojchevski, Oleksandr Shchur, Daniel Zugner, and Stephan Gunnemann. 2018. NetGAN: Generating Graphs via random walks. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, Vol. 2. International Machine Learning Society (IMLS), 973–988.

[4] Umit Can and B. Alatas. 2019. A new direction in social network analysis: Online social network analysis problems and applications. *Physica A: Statistical Mechanics and its Applications* 535 (12 2019), 122372. https://doi.org/10.1016/J.PHYSA.2019.122372

[5] Deepayan Chakrabarti, Yiping Zhan, and Christos Faloutsos. 2004. R-MAT: A Recursive Model for Graph Mining. In *Proceedings of the SIAM International Conference on Data Mining*. Society for Industrial and Applied Mathematics, Philadelphia, 442–446. https://doi.org/10.1137/1.9781611972740.43

[6] Sajad Darabi, Piotr Bigaj, Dawid Majchrowski, Pawel Morkisz, and Alex Fit-Florea. 2022. A Framework for Large Scale Synthetic Graph Dataset Generation. *ArXiv e-print* 1 (10 2022), 1–12. http://arxiv.org/abs/2210.01944

[7] Nicola De Cao and Thomas Kipf. 2018. MolGAN: An implicit generative model for small molecular graphs. In *Proceedings of the Workshop on Theoretical Foundations and Applications of Deep Generative Models*. 1–11.

[8] Pasquale De Meo, Emilio Ferrara, Giacomo Fiumara, and Alessandro Provetti. 2011. Generalized Louvain method for community detection in large networks. *International Conference on Intelligent Systems Design and Applications (ISDA)* (2011), 88–93.

[9] P. Erdos and A. Renyi. 1960. ON THE EVOLUTION OF RANDOM GRAPHS. (1960).

[10] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. 2017. Neural Message Passing for Quantum Chemistry. *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2053–2070.

[11] Furkan Gursoy and Bertan Badur. 2019. A Community-aware Network Growth Model for Synthetic Social Network Generation. In *Proceedings of the 5th International Management Information Systems Conference*. Springer, Ankara, 1–10. http://arxiv.org/abs/1901.03629http://dx.doi.org/10.6084/m9.figshare.7582082

[12] Xun Jian, Yue Wang, and Lei Chen. 2023. Publishing Graphs under Node Differential Privacy. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 35, 4 (April 2023), 4164–4177. https://doi.org/10.1109/TKDE.2021.3128946

[13] David G Kendall. 1956. Deterministic and stochastic epidemics in closed populations. In *Proceedings of the 3rd Berkeley Symposium on Mathematical Statistics and Probability*, Vol. 4. 149–165.

[14] Jure Leskovec and Christos Faloutsos. 2006. Sampling from large graphs. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, Vol. 2006. ACM, Philadelphia, 631–636. https://doi.org/10.1145/1150402.1150479

[15] Jure Leskovec and Andrej Krev. 2014. SNAP datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data.

[16] Renjie Liao, Yujia Li, Yang Song, Shenlong Wang, Will Hamilton, David K Duvenaud, Raquel Urtasun, and Richard Zemel. 2019. Efficient Graph Generation with Graph Recurrent Attention Networks. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Vancouver, 4255–4265.

[17] Xiaoxiao Ma, Jia Wu, Shan Xue, Jian Yang, Chuan Zhou, Quan Z. Sheng, Hui Xiong, and Leman Akoglu. 2022. A Comprehensive Survey on Graph Anomaly Detection with Deep Learning. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* (2022). https://doi.org/10.1109/TKDE.2021.3118815

[18] David F. Nettleton. 2016. A synthetic data generator for online social network graphs. *Social Network Analysis and Mining* 6, 44 (2016), 1–33. https://doi.org/10.1007/S13278-016-0352-Y/TABLES/18

[19] David F. Nettleton, Sergio Nettleton, and Marc Canal i. Farriol. 2021. MEDICI: A Simple to Use Synthetic Social Network Data Generator. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 12898 LNAI (2021), 273–285. https://doi.org/10.1007/978-3-030-85529-1{_}22/TABLES/3

[20] Marion Neumann, Plinio Moreno, Laura Antanas, Roman Garnett, and Kristian Kersting. 2013. Graph kernels for object category prediction in task-dependent robot grasping. In *Proceedings of the 11th Workshop on Mining and Learning with Graphs*. 1–6.

[21] Leslie O'Bray, Max Horn, Bastian Rieck, and Karsten Borgwardt. 2021. Evaluation Metrics for Graph Generative Models: Problems, Pitfalls, and Practical Solutions. In *ICLR 2022 - 10th International Conference on Learning Representations*.

[22] Benedek Rozemberczki, Ryan Davies, Rik Sarkar, and Charles Sutton. 2019. GemSec: Graph embedding with self clustering. In *Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. ACM, New York, 65–72. https://doi.org/10.1145/3341161.3342890

[23] Benedek Rozemberczki, Oliver Kiss, and Rik Sarkar. 2020. Karate Club: An API Oriented Open-source Python Framework for Unsupervised Learning on Graphs. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management (CIKM)*. ACM, Online, 3125–3132. http://arxiv.org/abs/2003.04819

[24] Benedek Rozemberczki, Oliver Kiss, and Rik Sarkar. 2020. Little Ball of Fur: A Python Library for Graph Sampling. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management (CIKM)*. ACM, Online, 3133–3140. https://doi.org/10.1145/3340531.3412758

[25] C. Seshadhri, Tamara G. Kolda, and Ali Pinar. 2012. Community structure and scale-free collections of Erdos-Rényi graphs. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics* 85 (5 2012), 056109. Issue 5. https://doi.org/10.1103/PhysRevE.85.056109

[26] Martin Simonovsky and Nikos Komodakis. 2018. GraphVAE: Towards Generation of Small Graphs Using Variational Autoencoders. In *Proceedings of the International Conference on Artificial Neural Networks (ICANN)*. Springer, Rhodes, 412–422.

[27] Hongwei Wang, Jia Wang, Jialin Wang, Miao Zhao, Weinan Zhang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2017. GraphGAN: Graph Representation Learning with Generative Adversarial Nets. In *Proceedings Of the 32nd AAAI Conference on Artificial Intelligence (AAAI)*. AAAI Press, New Orleans, 2508–2515.

[28] Duncan J. Watts and Steven H. Strogatz. 1998. Collective dynamics of 'small-world' networks. *Nature* 393, 6684 (June 1998), 440–442.

[29] Mingxuan Yuan, Lei Chen, Philip S. Yu, and Ting Yu. 2013. Protecting Sensitive Labels in Social Network Data Anonymization. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 25, 03 (March 2013), 633–647. https://doi.org/10.1109/TKDE.2011.259

[30] Liming Zhang, Liang Zhao, Shan Qin, Dieter Pfoser, and Chen Ling. 2021. TG-GAN: Continuous-time temporal graph deep generative models with time-validity constraints. In *Proceedings of the World Wide Web Conference (WWW)*. ACM, Online, 2104–2116. https://doi.org/10.1145/3442381.3449818

[31] Ziwei Zhang, Peng Cui, and Wenwu Zhu. 2022. Deep Learning on Graphs: A Survey. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 34, 1 (Jan. 2022), 249–270. https://doi.org/10.1109/TKDE.2020.2981333

[32] Liang Zhao. 2020. A Systematic Survey on Deep Generative Models for Graph Generation. *arXiv e-print* 1, 1 (2020), 1–36. https://doi.org/10.1145/nnnnnnn.nnnnnnn

[33] Yanqiao Zhu, Yuanqi Du, Yinkai Wang, Yichen Xu, Jieyu Zhang, Qiang Liu, and Shu Wu. 2022. A Survey on Deep Graph Generation: Methods and Applications. *ArXiv* 1, 1 (3 2022), 1–14. https://doi.org/10.48550/arxiv.2203.06714