Motif-guided Heterogeneous Graph Deep Generation

Chen Ling[1], Carl Yang[1] and Liang Zhao[1*]

[1*]Department of Computer Science, Emory University, Atlanta, 30332, GA, USA.

*Corresponding author(s). E-mail(s): liang.zhao@emory.edu;
Contributing authors: chen.ling@emory.edu;
j.carlyang@emory.edu;

Abstract

The complex systems in the real-world are commonly associated with multiple types of objects and relations, and heterogeneous graphs are ubiquitous data structures that can inherently represent multi-modal interactions between objects. Generating high-quality heterogeneous graphs allows us to understand the implicit distribution of heterogeneous graphs and provides benchmarks for downstream heterogeneous representation learning tasks. Existing works are limited to either merely generating the graph topology with neglecting local semantic information or only generating the graph without preserving the higher-order structural information and the global heterogeneous distribution in generated graphs. To this end, we formulate a general, end-to-end framework - HGEN for generating novel heterogeneous graphs with a newly proposed heterogeneous walk generator. On top of HGEN, we further develop a network motif generator to better characterize the higher-order structural distribution. A novel heterogeneous graph assembler is further developed to adaptively assemble novel heterogeneous graphs from the generated heterogeneous walks and motifs in a stratified manner. The extended model is proven to preserve the local semantic and heterogeneous global distribution of observed graphs with the theoretical guarantee. Lastly, comprehensive experiments on both synthetic and real-world practical datasets demonstrate the power and efficiency of the proposed method.

Keywords: Heterogeneous Graph, Graph Generation, Deep Generative Models, Graph Neural Network

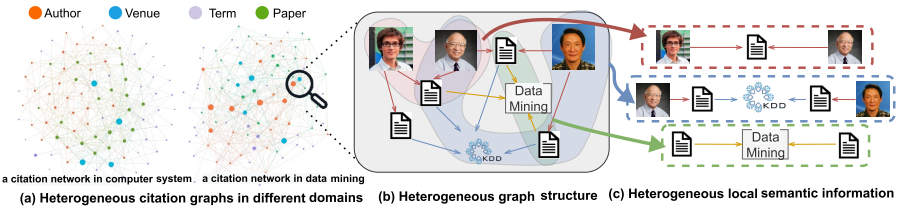


Fig. 1: Examples of heterogeneous graphs in the academic field.

1 Introduction

Graphs have emerged as an important data genre found in a wide class of applications. Researchers have devoted themselves to studying various types of graph problems, resulting in a rich literature of related papers and methods [1–7] in recent years, which can be primarily categorized into two directions: 1) graph representation learning aims at encoding graph topological and semantic information into vector space [8]; and 2) graph generation, which reversely aims at constructing graph-structured data from low-dimensional space containing the graph generation rules or distribution [9]. Many efforts have been devoted to studying both representation learning and graph generation on homogeneous graphs. However, as the superclass of the homogeneous graph, heterogeneous graphs come with different types of information attached to nodes and edges, which can contain considerably richer semantic information than homogeneous graphs [10]. Figure 1(b) shows a citation network with author, paper, venue, and term as nodes and "authorship", "containment" and "publishment" as edges. The local semantic information based on certain combinations of node types and edge types reflect the key patterns of heterogeneous graphs [11, 12], and such combinations of nodes and edges are typically referred to as *meta-path*. Meta-paths characterize the rich and diverse relations among nodes [12, 13]. For example, as shown in Figure 1(b), two authors can be connected via a meta-path since they both contribute to a paper, while two authors can alternatively be connected because their papers are accepted at the same venue.

As a more powerful, realistic, and generic superclass of traditional homogeneous graphs, heterogeneous graphs have recently been intensively studied. Existing literature focuses generally on learning network representations and latent embeddings for various network mining and analytical tasks, such as meta-relation detection [14, 15], heterogeneous node embedding learning [16, 17], and heterogeneous link prediction [18]. However, the other perspective of heterogeneous graph study - heterogeneous graph generation- remains paucity. Other than providing benchmarks for many heterogeneous graph studies, realistic heterogeneous graph generation has at least two advantages: 1) generating high-quality heterogeneous graphs requires us to comprehensively capture the latent graph distribution, which can significantly enrich our understanding of the implicit properties of heterogeneous graphs; 2) generating

heterogeneous graphs is helpful in specific downstream applications (e.g., recommendation system [19], knowledge graph reasoning [18], and node proximity search [11]). Given the importance of the research problem, there is only one work [20] that has tried to generate random heterogeneous graphs with handcrafted rules, which fails to decode the real data distribution underlying the observed graphs.

In the past few years, we have witnessed plenty of deep homogeneous graph generative models [2, 9, 21–23] that can learn the observed graph distribution without prescribed rules, which have shown advantages in preserving various static graph properties in the generated graphs. However, existing deep generative models designed for homogeneous graphs cannot be trivially adapted to heterogeneous graphs due to the following technical difficulties: 1) *Difficulties in preserving heterogeneous semantic information.* Current works for homogeneous graphs have been either using random walks as a tool to learn the graph topological distribution as learning the distribution of random walks ([22, 24]) or directly modeling an overall distribution of the edges ([23, 25]) over the homogeneous graphs. However, objects in heterogeneous graphs are interconnected via various meta-paths, as shown in Figure 1(c). As meta-paths carry the complex local semantic information, adapting current works to the heterogeneous graph scenario without any elaborations on meta-path would bring difficulties in learning and preserving the distribution of such complex semantic patterns spanning different graph entities (i.e., edges and nodes) in the newly generated heterogeneous graphs. 2) *Difficulties in preserving heterogeneous higher-order structural information.* In the study of heterogeneous graphs, meta-paths may also fall short of expressing more intricate relationships among nodes in heterogeneous graphs. As marked in Figure 1(b), some common and symmetric higher-order structures spanning meta-paths will likely be observed repeatedly, which forms a triangle or orbit structure (e.g., one author writes two papers that are accepted by the same venue, and two papers of an author focus on the same research topic). These higher-order connectivity patterns are known to be important in understanding the structure and organization of heterogeneous networks, and many works [26, 27] have proposed to utilize this information to boost the performance of downstream heterogeneous graph mining tasks. In terms of generating high-quality and realistic heterogeneous graphs, it is also inevitable to consider modeling the higher-order structural information. However, previous works either are designed for homogeneous graph generation [2, 22] that neglected the importance of the higher-order structural information or fail to consider integrating the higher-order structures into the overall generation block [28]. The distributions of these higher-order graph structures are also hard to capture in heterogeneous graphs, bringing more challenges to effective heterogeneous graph generation. 3) *Difficulties in preserving heterogeneous global information.* Meta-paths are also well-recognized to play a fundamental role in preserving the global patterns of heterogeneous graphs [11]. For example, the ratio of different node types, and edge types, and their meta-paths are apparently different between

the citation networks of the computer system domain and the data mining domain, as shown in Figure 1(a). It is essential to preserve the global distribution of meta-path patterns during heterogeneous graph generation, which is again extremely difficult as it is entangled with the preservation of node type ratios, edge type ratios, and graph topological patterns.

In coping with these challenges, we introduce an end-to-end graph generative framework, namely Heterogeneous Graph Generation (HGEN), whose goal is to generate novel heterogeneous graphs by preserving all the complex local semantic and heterogeneous global property through directly modeling the distribution of meta-paths in observed heterogeneous graphs. Particularly, HGEN learns a joint distribution of the random walks and the associated meta-paths from the observed heterogeneous graphs in order to capture the local semantic distribution. In order to tackle the second difficulty, we extend the meta-path-based generator in HGEN and make it capable of characterizing the higher-order structural distribution via directly modeling and generating network motifs. On top of that, we encode heterogeneous higher-order structural information into nodes via embedding learning and use it to guide the generation of meta-paths and network motifs that form different high-order heterogeneous structures. Finally, to tackle the third challenge, we develop a novel heterogeneous graph assembly method, which is theoretically proved to preserve the global heterogeneous graph patterns in node types, edge types, and meta-paths.

We conclude our major contributions as follows:

- **Problem Formulation.** We propose to formulate a new paradigm of heterogeneous graph generation, which can effectively identify and resolve its unique challenges in preserving various heterogeneous graph properties.
- **Framework Design.** We propose an end-to-end generative framework for heterogeneous graph generation. The proposed framework can effectively learn the underlying distribution of heterogeneous graphs. It generates heterogeneous graphs with ensuring the preservation of various heterogeneous graph properties.
- **Model Extension.** We further extend our proposed model to leverage network motifs to capture more intrinsic higher-order structural information as well as multiple meta relations on edges. We also adapt the proposed graph assembler to adaptively assemble novel graphs by various generated instances.
- **Evaluation.** We conduct extensive experiments on both synthetic and real-world heterogeneous graphs. Compared with state-of-the-art baselines, HGEN achieves competitive results in preserving most of the static graph properties. In addition, HGEN is shown to be capable of generating realistic heterogeneous graphs by preserving important meta-path information.

2 Related Work

2.1 Graph Generation.

Generative models for graphs have a rich history due to the wide range of applications in different domains, such as link prediction [22, 23], protein structure analysis [29], and information diffusion analysis in social networks [30]. Traditional graph generation methods (e.g., random graphs, stochastic block models, and Bayesian network models) fail to model complex dependencies in our real-world scenarios. In addition, they cannot effectively preserve the statistical properties of the observed graphs. In the last few years, there has been a surge in research focusing on deep graph generation. According to [9], the current deep graph generation can be divided into two categories: sequential-based and one-shot-based. Sequential-based graph generation methods [1, 2, 22] autoregressively generate the nodes and edges with the LSTM model. However, the sequential-based generation (e.g., GraphRNN [2]) is limited in following a fixed node/edge permutation order, which greatly loses the generation flexibility and model scalability. On the other hand, one-shot-based generation methods [22, 23, 28, 29, 31–33] try to build a probabilistic graph model based on the matrix representation that can generate graph topology as well as node/edge attributes in a one-shot, but most of them cannot easily be applied in large graphs due to the large time complexity. For example, GraphVAE [23] is a new and first-of-its-kind variational autoencoder for whole graph generation, though it typically only handles very small graphs and cannot scale well to large graphs in both memory and runtime. NetGAN [22] follows the GAN model [34] and uses a generator to generate synthetic random walks while discriminating synthetic walks from real random walks sampled from a real graph. Finally, multi-attributed graph generation [2, 21, 35, 36] aims at generating homogeneous graphs by preserving node/edge attributes. Instead, the key patterns of heterogeneous graphs are the higher-order local semantics reflected by the combinatorial of the types of nodes and edges, which cannot be captured by methods for homogeneous graphs.

2.2 Heterogeneous Network Motif (Meta-graph)

Compared to the commonly-adopted homogeneous graph, the heterogeneous graph carries much richer semantic information and has therefore gained much attention in recent literature [37]. The concept of meta-paths in a heterogeneous graph [11, 13] is one of the most important concepts proposed to capture numerous semantic relationships across multiple types of objects systematically. Compared to the commonly adopted heterogeneous meta-paths, heterogeneous Network Motifs (also known as meta-graph) [38] are proposed to capture more complex structural information in heterogeneous graphs. Specifically, a meta-graph is a special directed acyclic graph containing at least two embedded metapaths, such as a DAG containing as shown in Figure 1(b), where the higher-order structure *one author may publish two papers in a venue*

Table 1: Description of Important Notations

Notations	Descriptions
$\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$	A heterogeneous graph $\mathcal{G}$ with node set $\mathcal{V}$ and edge set $\mathcal{E} = \mathcal{V} \times \mathcal{V}$
$o = \phi(v_i)$	Each node $v_i \in \mathcal{V}$ is associated with a node type $o = \phi(v_i)$
$l = \phi(e_{ij})$	Each edge $e_{ij} \in \mathcal{E}$ is associated with a relation type $l = \psi(e_{ij})$
$(\mathbf{v}, \mathbf{o})$	A heterogeneous walk that consists of a random walk $(v_1, v_2, ..., v_i, ...)$ on $\mathcal{G}$ and an associated meta-path $\big((o_1, o_2, ..., o_n), (l_1, l_2, ..., l_{n-1})\big)$ of $\mathbf{v}$
$\hat{\mathbf{v}}, \hat{\mathbf{o}}$	Generated heterogeneous walk
S	Symmetric adjacency matrix with size $\mathcal{V} \times \mathcal{V}$ to record the sampled edge frequency

contains two meta-paths of *author-paper-venue*. Network motifs have served as a building block for learning latent embeddings that contain higher-order relationships in a graph [39]. However, in terms of graph generation, graph generative models are successful at retaining pairwise associations in the underlying networks but often fail to capture higher-order connectivity patterns known as network motifs. To date, one attempt [40] leverages network motifs as the basic unit to generate homogeneous graphs. However, this method only learns the structural distribution but fails to capture the meta-relation within the network motifs.

3 Problem Formulation

A heterogeneous graph [10, 37] is a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ with multiple types of objects and relations. $\mathcal{V}$ is the set of objects (i.e., nodes), where each node $v_i \in \mathcal{V}$ is associated with a node type $o = \phi(v_i)$. $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges, where each edge $e_{ij} \in \mathcal{E}$ is associated with a relation type $l = \psi(e_{ij})$. All notations are summarized in Table 1.

In the study of heterogeneous graphs, the concepts of meta-paths are widely considered as cornerstones and adopted to systematically capture numerous semantic relationships across multiple types of objects, which are defined as a path over the graph [10, 13]. Hence meta-paths are indispensable to be considered as basic units for heterogeneous graph generation. Concretely, a meta-path $\mathbf{o}$ is defined as a sequence of object types and edge types $\mathbf{o} = \big((o_1, o_2, ..., o_n), (l_1, l_2, ..., l_{n-1})\big) = o_1 \xrightarrow{l_1} o_2 \xrightarrow{l_2} ... \xrightarrow{l_{n-1}} o_n$, where each o_i and l_j are node type and edge type in the sequence, respectively. Each meta-path captures the rich semantic information between its two ends o_1 and o_n. In heterogeneous graphs, the local semantic information is carried on each of walks $\mathbf{v} = (v_0, v_1, ..., v_n)$ and its associated meta-path $\mathbf{o}$. We again take Figure 1(c) as an example, there exist two meta-paths between papers: (*Paper, Author, Paper*) and (*Paper, Venue, Paper*). The utilization of different meta-paths allow the heterogeneous graph to contain rich topological and semantics among diverse objects, which has been shown beneficial to many real-world graph mining applications [10, 16, 17].

With the preliminary notion of the heterogeneous graph, we formalize the heterogeneous graph generation problem as follows:

Problem 1 (**Heterogeneous Graph Generation**) The goal of the heterogeneous graph generation is to learn a distribution $p_{\text{data}}(\mathcal{G})$ from the observed heterogeneous graphs such that a new graph $\hat{\mathcal{G}}$ can be obtained by sampling $\hat{\mathcal{G}} \sim p_{\text{data}}(\mathcal{G})$.

Challenge 1 (**Difficulties in modeling the complex local semantic information.**) Although the existence of meta-paths allows heterogeneous graph to characterize the combinatorial of node types and edge types, it is unclear how to model their distributions and generatively assemble them into heterogeneous graphs.

Challenge 2 (**Difficulties in characterizing the heterogeneous structural patterns.**) The local structural patterns in heterogeneous graphs are often expressed in higher-order proximity among the nodes and edges (e.g., triangles, orbits, and other higher-order structures). Such a higher-order local structure may fuse multiple walks under one or more meta-paths with richer semantic information, yet brings more difficulties in learning its distribution.

Challenge 3 (**Difficulties in capturing heterogeneous global meta-path information.**) Meta-paths indeed play a significant role in preserving the global patterns of heterogeneous graphs. In heterogeneous graph generation, it is important yet challenging to preserve the global distribution of meta-path patterns since the distribution of meta-path patterns often involves node type ratios, edge type ratios, and graph topological patterns.

4 Heterogeneous Graph Generation

To address the above challenges, we propose a new heterogeneous graph generation framework, named HGEN. To address the first and second challenge, we propose a *heterogeneous walk generator* in Section 4.1 to jointly learn the distribution of local walks and the associated meta-paths so that both heterogeneous topological and local semantic information can be well captured. To overcome the second challenge, we leverage the heterogeneous node embedding to make the generator be aware of any potential higher-order structures that each node may be involved with. Finally, for the third challenge, we propose a novel *heterogeneous graph assembler* in Section 4.3, which can construct new heterogeneous graphs by capturing the global heterogeneous property, namely different meta-path ratios. We further prove that the global heterogeneous property can be well-preserved through our Theorem 1 introduced in Section 4.4.

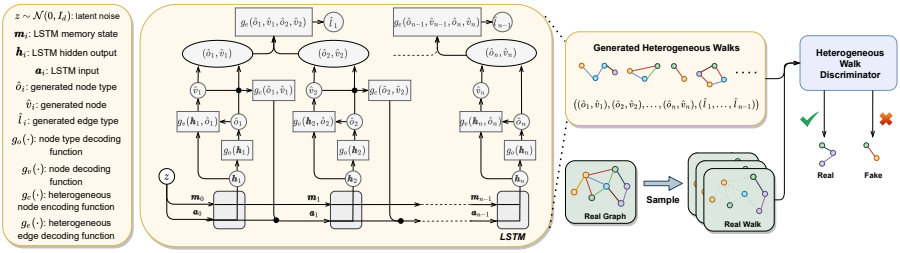


Fig. 2: The illustration of the heterogeneous walks generation in HGEN.

4.1 Heterogeneous Walk Generator

In the observed graph $\mathcal{G}$, a heterogeneous walk is defined as a tuple that consists of two components: a walk $\mathbf{v}$ and an associated meta-path $\mathbf{o}$. The proposed heterogeneous walk generator G is defined as a probabilistic sequential learning model to generate synthetic heterogeneous walks: $(\hat{\mathbf{v}}, \hat{\mathbf{o}}) = \big((\hat{v}_1, \hat{v}_2, ..., \hat{v}_n), ((\hat{o}_1, \hat{o}_2, ..., \hat{o}_n), (\hat{l}_1, \hat{l}_2, ..., \hat{l}_{n-1}))\big)$, where the $\hat{\mathbf{v}}$ and $\hat{\mathbf{o}}$ are denoted as the generated walk and associated meta-path, respectively. We use $\hat{v}_i$, $\hat{o}_i$, and $\hat{l}_i$ to denote each of the generated node, node type, and edge type in $(\hat{\mathbf{v}}, \hat{\mathbf{o}})$, respectively. Figure 2(a) illustratively summarizes the whole generative process of each synthetic heterogeneous walk.

Heterogeneous Walk Generation. We model G as a sequential learning process based on a recurrent architecture, and each unit f_θ in the sequential model is parameterized by θ so that it can generate a node type $\hat{o}$ and a corresponding node $\hat{v}$ that belongs to this node type in a hierarchical manner. Precisely, the node type $\hat{o}$ is determined based on the previously generated sequence, and the node $\hat{v}$ is then coherently determined by the generated node type as well as the generated sequence. Both generated node type $\hat{o}$ and node $\hat{v}$ together provide information for the generation of the next node type and node instance.

Specifically, at each recurrent block (i.e., time step) t, f_θ produces two outputs $(\boldsymbol{m}_t, \boldsymbol{h}_t)$, where the $\boldsymbol{m}_t$ is the current memory state and the $\boldsymbol{h}_t$ is a latent probabilistic distribution (i.e., hidden output of f_θ) denoting the information carried from previous time steps. We first sample the node type $\hat{o}_t \sim g_o(\boldsymbol{h}_t)$ based on the probability distribution $\boldsymbol{h}_t$, where the $g_o(\cdot)$ is a node type decoding function. We then sample the node $\hat{v}_t$ by a node decoding function $\hat{v}_t \sim g_v(\boldsymbol{h}_t, \hat{o}_t)$ that takes $\boldsymbol{h}_t$ and $\hat{o}_t$ as inputs. Lastly, the generated node type $\hat{o}_t$ and node $\boldsymbol{h}_t$ are fused by a heterogeneous node encoding function $g_c(\hat{o}_t, \hat{v}_t)$, which then serves as the input of next recurrent block.

Heterogeneous Node Sampling. To overcome the second challenge, we cannot uniformly sample $\hat{v}_t$ based on the node type $\hat{o}_t$ because such a way may cause the neglection of (1) *node structural* distribution and (2) *node semantic* distribution. For example, we may observe an author always tends to cite a paper with high citation (namely, high node degree of this paper

node). Then such distribution needs to be modeled with structural information. On the other hand, we may observe a data mining paper is unlikely to cite a computer system paper, and we may also need to characterize this tendency in the distribution. Both of the above distributions cannot be tackled by uniformly sampling. Therefore, to tackle this challenge, since latent node embedding could encode both topological and semantic information into the node, we propose to calculate a latent embedding $\tilde{v}_t$ of the next node v_t, then we select with a higher probability the closer embedding among all the embeddings that belong to node type $\hat{o}_t$ so that the next node v_t can be determined by the sampled embedding.

More specifically, we first calculate the latent node embedding $\tilde{v}_t$ based on the sampled node type $\hat{o}_t$ by a simple linear transformation. We then calculated the distance between $\tilde{v}_t$ and other node embedding $\tilde{v}_i^{(\hat{o}_t)}$, meaning any node $\tilde{v}_i$ belonging to the sampled node type $\hat{o}_t$. In this case, given a total number of k embeddings that belong to the type $\hat{o}_t$, the next node $\hat{v}_t$ can be sampled from a multinomial distribution:

$$\hat{v}_t \sim \text{Multi}(\tilde{v}_1^{(\hat{o}_t)}, \tilde{v}_2^{(\hat{o}_t)}, ..., \tilde{v}_k^{(\hat{o}_t)};\ p_1, p_2, ..., p_k),$$

where each $p_i = -\left\| d(\tilde{v}_t, \tilde{v}_i^{(\hat{o}_t)}) \right\|^2$ and $d(\cdot, \cdot)$ is a distance metric such as Euclidean distance. Note that the node embedding $\tilde{v}_i^{(\hat{o}_t)}$ can be obtained from a conventional heterogeneous node embedding technique such as [14].

In order to generate a variable-length heterogeneous walk, we incorporate a end-of-sequence token as an additional node type so that the heterogeneous walk generator stops when the sampled node type is the token at any steps. Therefore, the proposed generator is able to produce variable-length heterogeneous walks. Finally, the edge type l_t can be predicted by a simple edge decoding function $g_e(\hat{o}_t, \hat{v}_t, \hat{o}_{t-1}, \hat{v}_{t-1})$ that takes its two end nodes $\hat{v}_{t-1}$ and $\hat{v}_t$ as well as their node types $\hat{o}_{t-1}$ and $\hat{o}_t$ as inputs. In all, we summarize the overall generative process as follows:

$$\begin{aligned}
&\boldsymbol{a}_0 = 0,\ \boldsymbol{m}_0 = f_0(\boldsymbol{z}),\ \boldsymbol{z} \sim \mathcal{N}(0, 1)\\
&\boldsymbol{a}_1 = g_c(\hat{o}_1, \hat{v}_1),\ \hat{v}_1 \sim g_v(\boldsymbol{h}_1, \hat{o}_1),\ \hat{o}_1 \sim g_o(\boldsymbol{h}_1), (\boldsymbol{m}_1, \boldsymbol{h}_1) = f_\theta(\boldsymbol{m}_0, \boldsymbol{a}_0)\\
&\boldsymbol{a}_2 = g_c(\hat{o}_2, \hat{v}_2),\ \hat{v}_2 \sim g_v(\boldsymbol{h}_2, \hat{o}_2),\ \hat{o}_2 \sim g_o(\boldsymbol{h}_2), (\boldsymbol{m}_2, \boldsymbol{h}_2) = f_\theta(\boldsymbol{m}_1, \boldsymbol{a}_1)\\
&\hat{l}_1 = g_e(\hat{o}_2, \hat{v}_2, \hat{o}_1, \hat{v}_1)\\
&\cdots\\
&\hat{v}_n \sim g_v(\boldsymbol{h}_n, \hat{o}_n),\ \hat{o}_n \sim g_o(\boldsymbol{h}_n),\ (\boldsymbol{m}_n, \boldsymbol{h}_n) = f_\theta(\boldsymbol{m}_{n-1}, \boldsymbol{a}_{n-1})\\
&\hat{l}_{n-1} = g_e(\hat{o}_n, \hat{v}_n, \hat{o}_{n-1}, \hat{v}_{n-1})
\end{aligned}$$

In this work, we utilize LSTM as the recurrent architecture, and f_θ becomes a single LSTM unit. To initialize the whole generative process, G takes a random noise $\boldsymbol{z}$ as input, which is drawn from a standard Gaussian distribution.

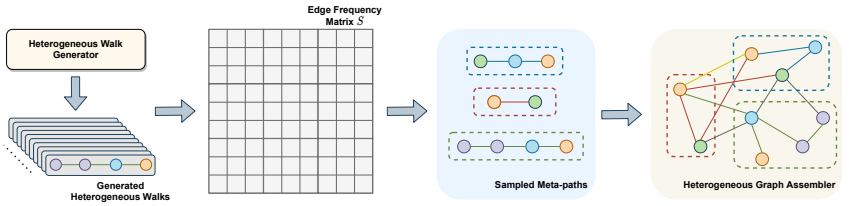


Fig. 3: The process of heterogeneous graph assembler.

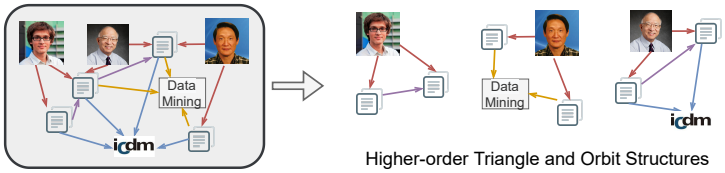


Fig. 4: Various graph motifs (meta-graphs) in academic heterogeneous network: one author publishes two paper that one paper cites the other one; one author publishes two papers that related to the same topic; and one author publishes two co-cited papers at one venue.

Additionally, for the node type decoding function $g_o(\cdot)$, we apply the Gumbel-softmax trick [41] in $g_o(\cdot)$ to make the whole sampling differentiable. Finally, in most of the real-world scenarios, the edge type l_t can be determined by the types of its two end nodes $\hat{o}_t$ and $\hat{o}_{t-1}$ if there does not exist multi-typed relations between two node types. In this case, the heterogeneous walk generator can be simplified only to generate node sequences and associated node types.

4.2 Extension of Heterogeneous Motif Generator.

In the previous section, the proposed heterogeneous walk generator can well characterize pairwise relationships within the heterogeneous graph and associated heterogeneous graph statistics via meta-paths; however, higher-order relationships (aka. heterogeneous network motifs) in a heterogeneous graph are fundamental for our understanding of the network behavior and function.

Definition 1 (Heterogeneous Network Motifs) A Heterogeneous Network Motif (Meta Graph) $\mathcal{M}$ is a directed acyclic graph (DAG) with a single source node v_s (i.e., with in-degree 0) and a single target node v_t (i.e., with out-degree 0) defined on a heterogeneous graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$. Then we define a Heterogeneous Network Motif as $\mathcal{M} = (\mathcal{V}_\mathcal{M}, \mathcal{E}_\mathcal{M}, v_s, v_t)$, where $\mathcal{V}_\mathcal{M} \subseteq \mathcal{V}$ and $\mathcal{E}_\mathcal{M} \subseteq \mathcal{V}$.

As we emphasized in Section 3, meta-path is the natural way to represent local semantics in heterogeneous networks; however, meta-path may

not be the best way to characterize the rich semantics, especially semantics encoded in higher-order structures. As can be clearly seen in Figure 4, separate meth-paths (i.e., author-paper-topic and author-paper-venue) can form an orbit structure, which cannot be simply described by meta-paths. Current heterogeneous graph generation methods either rely on meta-paths as the basic generation unit [28] or completely ignore the rich semantics encoded in heterogeneous meta-structures [20], which is a major shortcoming for applications that aim to generate heterogeneous graphs that realistically mimic real-world heterogeneous networks or predict unobserved heterogeneous higher-order structures.

In order to make our algorithm better preserve the higher-order structural distribution in the generated graph, other than utilizing heterogeneous node embedding, we generalize the proposed heterogeneous walk generator to be able to generate heterogeneous network motifs. While a complete enumeration of the network motifs present in a large-scale heterogeneous network is computationally prohibitive, we instead focus on three motif structures (e.g., triangle, orbit, overlapped triangle) as visualized in Figure 4.

Since graph motifs contain DAG structure, we may not trivially generate them as varying-length sequences and leverage the end-of-sequence token to indicate the stop. Instead, we propose to sample from a learnable logit τ, where each $\tau_i \in \tau$ represents the probability of the motif chosen to be generated and $\|\tau\| = 1$. Note that we initialize $\tau_0 \in \tau$ to be the probability of choosing meta-path to generate so that the motif-based generation model can be combined training with the meta-path-based generator. We provided the updated model overview as follows.

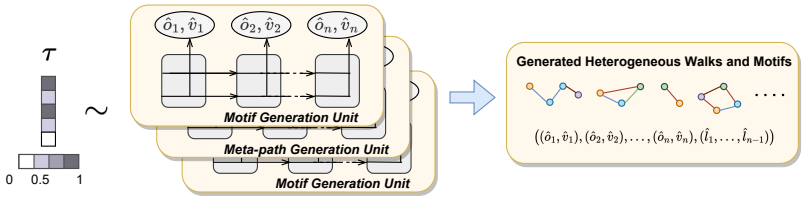


Fig. 5: The illustration of the extended heterogeneous walk/motif generation. We separate the generation unit for meta-paths and heterogeneous network motifs, and we leverage the sampling-based method to choose each unit.

4.3 Heterogeneous Generator Training and Utilization

In the following, we will introduce how to train the above-mentioned generator and how to use the heterogeneous walks and motifs generated by it to construct heterogeneous graphs. Since we extend our framework to be able to generate both meta-paths and motifs, we refer the generated meta-paths and

motifs to heterogeneous instances for the sake of simplicity in the following context. Concretely, we utilize a heterogeneous discriminator D to distinguish between real and fake heterogeneous instances, where the real instances are uniformly sampled from the observed graph. We then propose a heterogeneous graph assembler to construct new graphs based on the sampled heterogeneous instances. More details are presented as follows.

We first introduce the overall objective function of the Wasserstein heterogeneous GAN [34], which is written as:

$$\begin{aligned}\mathcal{L}_{\text{HGEN}} = \max \mathbb{E}_{(\mathbf{o},\mathbf{v})\sim p(\mathcal{G})}[D_o(\mathbf{o}) + D_v(\mathbf{v})] \\ - \mathbb{E}_{z\sim p(z)}[D_o(\hat{\mathbf{o}}) + D_v(\hat{\mathbf{v}})], s.t.\ G(z) = (\hat{\mathbf{o}}, \hat{\mathbf{v}}),\end{aligned} \tag{1}$$

where $\mathbf{v}$ and $\mathbf{o}$ are the random walk/motif and associated meta-path/meta-graph, respectively, directly sampled from the observed heterogeneous graph $\mathcal{G}$. They are the real data for training our heterogeneous generator G. Specifically, given an observed heterogeneous graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, we utilize random-walk-based method to uniformly sample a set of random walks $\{\mathbf{v}_1, \mathbf{v}_2, ...\}$, where each $\mathbf{v}_i$ is a node sequence s.t. $\mathbf{v}_i = (v_1, v_2, ..., v_n)$. In addition, we extract the meta information $\mathbf{o}_i = \big((o_1, o_2, ..., o_n), (l_1, l_2, l_{n-1})\big)$ from each $\mathbf{v}_i$.

The heterogeneous discriminator D in Equation (1) is designed as a parallel recurrent architecture in order to individually distinguish whether each unit in the heterogeneous component are valid or not. Specifically, at each recurrent block (i.e., each step) t, the discriminator D takes two inputs: the generated node type $\hat{o}_t$ and node index $\hat{v}_t$, each of which is fed into an individual recurrent unit. After processing both sequences, the discriminator returns a single score $D_v(\mathbf{v}) + D_o(\mathbf{o})$ that represents the probability of the heterogeneous component being real.

4.3.1 Heterogeneous Graph Assembler

To assemble a heterogeneous graph from the generated heterogeneous instances, we further propose a novel stratified heterogeneous edge sampling strategy to achieve the following steps: 1) it first samples a node $\hat{v}_i$ and its type $\hat{o}_i$ from all of the generated heterogeneous walks; 2) based on the node type $\hat{o}_i$, we then sample a meta-path that starts with $\hat{o}_i$; 3) we iteratively sample the next node $\hat{v}_{i+1}$ in the sampled meta-path if both of the node type $\hat{o}_{i+1}$ and edge type $\hat{l}_i$ fits the meta-path pattern.

More specifically, the generator G firstly produces a sufficient number of heterogeneous walks as shown in Figure 3(a). We then construct an symmetric adjacency matrix S with size $|\mathcal{V}|\times|\mathcal{V}|$ to record the count of edges observed from the sampled heterogeneous walks in each entry S_{ij}, where the $|\mathcal{V}|$ is the size of the node set. Next, we collect all of the meta-path patterns generated by the generated heterogeneous walks, as shown in Figure 3(b-c). For the first step of the stratified heterogeneous edge sampling, we sample the a node $\hat{v}_i$ and its type type $\hat{o}_i$ based on the node degree distribution $\frac{\sum_j S_{ij}}{|\mathcal{V}|}$. For the

second step, among all the meta-paths $\{\mathbf{o}_1^{(f)}, \mathbf{o}_2^{(f)}, ...\}$ that start with the node type $\hat{o}_i$, we sample a meta-path $\mathbf{o}_i^{(f)}$ based on the probability $\frac{c(\mathbf{o}_i^{(f)})}{T^{\hat{o}_i}}$, where $T^{\hat{o}_i}$ is the total count of generated meta-paths that starts with node type $\hat{o}_i$ and $c(\mathbf{o}_i^{(f)})$ is the count of meta-path pattern $\mathbf{o}_i^{(f)}$. For the third step, by following this meta-path pattern $\mathbf{o}_r = (o_1, o_2, ..., o_n)$, we iteratively sample all the nodes whose node types are regulated by the the meta-path. Precisely, we sample the next node v_j by sampling all the neighbors of the current node v_i with the probability $p_{v_i v_j} = (S_{ij})/(\sum_s S_{is})$ such that all the nodes v_s belong to the specific node type o_j following the meta-path $\mathbf{o}_i^{(f)}$. The sampled node sequence $\mathbf{v}_r = (v_0, v_1, ...)$ is then added to the score matrix S. We continue the stratified heterogeneous edge sampling strategy until the desired amount of edges is reached. The final assembled graph is visualized in Figure 3 (d).

Extension of Heterogeneous Graph Assembler with Motif Consideration. To assemble a heterogeneous graph from the generated heterogeneous instances, we further extend our stratified heterogeneous graph assembler and leverage the learned logit τ in order to make the assembler generate heterogeneous graph that has the closer higher-order structural distribution. Specifically, after the generator G produces a sufficient number of heterogeneous instances, we leverage the learned τ to firstly sample the exact heterogeneous instance pattern (i.e., walk, triangle, or orbit). After collecting such a pattern, we then follow the aforementioned stratified heterogeneous edge sampling strategy to sample exact nodes under the specific pattern. This strategy allows us to more closely model the local semantic, higher-order, and global distribution if the learned τ can correctly characterize the ratio of different heterogeneous component patterns in the observed heterogeneous graph.

4.3.2 Complexity Analysis.

The computational complexity of HGEN is $O(W \cdot L)$, where W is the weights of a single LSTM unit, and L is the length of the generated heterogeneous instances. However, the length of our proposed heterogeneous walk is considerably small ($1 \leq L \leq 3$) while the walk length in other random-walk-based graph generative method [22] is (≥ 16). For auto-regressive graph generation models [2, 3], the time complexities are at least $O(|\mathcal{V}|^2 \cdot W)$, where $|\mathcal{V}|$ is the cardinality of the node set. They convert graph as a long sequence by performing a large number of breadth-first-search (BFS) enumerations for each graph. Additionally, HGEN also has linear complexity in graph assembly, it only needs to run the trained model T_s times to sample heterogeneous walks for constructing the score matrix S. To sum up, the overall complexity of HGEN can be reduced to $O(W + T_s)$, which makes our proposed model highly efficient for handling large graphs, since the overall process is not sensitive to the number of nodes at all.

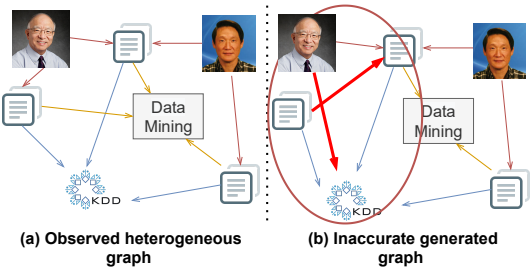


Fig. 6: Example of two heterogeneous graphs with different semantic information: the observed meta-path patterns are different, although the node and edge distribution are the same between two graphs. Specifically, since we do not observe a direct link between (author, venue) and (paper, paper) in the observed graph Figure 6(a). It is not accurate for the generated graph Figure 6(b) that generate such links.

4.4 Meta-path Information Preservation Analysis

As we discussed in Sec. 3, it is significant to preserve the meta-path information in our generated graph. Taking Fig. 6 as an example, although both graphs have exactly the same structure, they are still regarded as two different heterogeneous graphs since their meta-path distributions are different. Given the importance of the meta-path information in heterogeneous graph generation, we further show that our framework can successfully preserve this meta-path information as proved in Theorem 1.

Theorem 1 The distribution of meta-path patterns $\overline{\mathcal{O}}^{(r)}$ of the generated heterogeneous graph equals the distribution of meta-path patterns $\overline{\mathcal{O}}$ in the observed heterogeneous graph, namely $p(\overline{\mathcal{O}}^{(r)}) = p(\overline{\mathcal{O}})$.

Proof . We will prove that the ratio of the meta-path patterns can be preserved in three steps: 1) the ratio of different meta-path patterns can be preserved during the sampling procedure; 2) the ratio of generated meta-path patterns can be preserved during the generation procedure; 3) the meta-path patterns can be preserved during the graph assembling procedure.

Meta-path Ratio Preservation in Sampling. Let $\overline{\mathcal{O}} = (\overline{\mathbf{o}}_1, \overline{\mathbf{o}}_2, ...)$ be the collection of meta-paths obtained from the observed heterogeneous graph $\mathcal{G}$, each $\overline{\mathbf{o}}_i$ is a meta-path in one-hot format $\overline{\mathbf{o}}_i \in \{0,1\}^{1\times R}$, where the R is the total number of different meta-path patterns. $\overline{\mathcal{O}}^{(\tau)} = (\overline{\mathbf{o}}_1^{(\tau)}, \overline{\mathbf{o}}_2^{(\tau)}, ..., \overline{\mathbf{o}}_K^{(\tau)})$ is the sequence of sampled meta-paths with sampling size K, where each meta-path $\overline{\mathbf{o}}_j^{(\tau)} \in \{0,1\}^{1\times R}$ is drawn independent and identically distributed ($i.i.d$) from $\overline{\mathcal{O}}$.

Suppose that $\boldsymbol{\mu} = [\mu_1, \mu_2, ..., \mu_R]^T$ denotes the probability of each individual meta-path pattern in $\overline{\mathcal{O}}$, it is obvious that $\mathbb{E}[\overline{\mathbf{o}}_i|\boldsymbol{\mu}] = \sum_{\overline{\mathbf{o}}_i} p(\overline{\mathbf{o}}_i|\boldsymbol{\mu})\overline{\mathbf{o}}_i = [\mu_1, \mu_2, ..., \mu_R]^T = \boldsymbol{\mu}$. Now consider the total K observations $\overline{\mathcal{O}}^{(\tau)} =$

	# of Nodes	# of Edges	Average Degree	# of Node Types	# of Edge Types
Syn_{100}	100	490	9.8	3	6
Syn_{200}	200	1,090	10.9	3	6
Syn_{500}	500	2,987	11.95	3	6
$\text{Syn}_{\text{Multi}}$	300	1,352	12.13	3	8
PubMed	1,565	13,532	17.29	4	10
IMDB	1,653	4,267	5.432	4	4
DBLP	11,240	47,885	8.52	4	3

Table 2: Dataset Overview.

$(\overline{\mathbf{o}}_1^{(\tau)}, \overline{\mathbf{o}}_2^{(\tau)}, ..., \overline{\mathbf{o}}_K^{(\tau)})$, the corresponding likelihood function takes the form:

$$p(\overline{\mathcal{O}}^{(\tau)}|\boldsymbol{\mu}) = \prod_{i}^{R}\prod_{j}^{K}\mu_j^{\overline{\mathbf{o}}_{ij}^{(\tau)}} = \prod_{j}^{K}\mu_j^{\sum_n \overline{\mathbf{o}}_{nj}^{(\tau)}} = \prod_{j}^{K}\mu_j^{m_j} \tag{2}$$

We see that the likelihood function depends on the K data points only through the R quantities: $m_j = \sum_n \overline{\mathbf{o}}_{nj}^{(\tau)}$. Since the number of observations of $\overline{\mathbf{o}}_j^{(\tau)}$ equals 1, we achieved sufficient statistics for this distribution. Therefore, $p(\overline{\mathcal{O}}^{(\tau)}) = p(\overline{\mathcal{O}})$ can be proved.

Meta-path Ratio Preservation in Generation. Since we have proved the meta-path ratio can be preserved during the sampling, the next step is to show that the distribution of generated meta-paths $p(\overline{\mathcal{O}}^{(g)})$ is equal to $p(\overline{\mathcal{O}}^{(\tau)})$. Proving $p(\overline{\mathcal{O}}^{(g)}) = p(\overline{\mathcal{O}}^{(\tau)})$ is equivalent to prove whether $p_{data} = p_g$ in the GAN setting. As being proved in the works of GANs and their variants [34, 42], it showed that the objective function of the generator G is equivalent to optimize the distribution distance between p_{data} and p_g if the discriminator D is optimal. Therefore, global optimality of $p_g = p_{data}$ can be achieved if both generator G and discriminator D have enough capability. Therefore, $p(\overline{\mathcal{O}}^{(g)}) = p(\overline{\mathcal{O}}^{(\tau)})$ if both G and D are optimal in our framework.

Meta-path Ratio Preservation in Assembling. Finally, we show that our graph assembling method can also preserve the meta-path ratio from the generated data $\overline{\mathcal{O}}^{(g)}$ such that $p(\overline{\mathcal{O}}^{(g)}) = p(\overline{\mathcal{O}}^{(r)})$. As discussed in Sec. 4.3, the new graph $\hat{\mathcal{G}}$ is directly assembled by meta-paths $(\overline{\mathbf{o}}_1^{(g)}, \overline{\mathbf{o}}_2^{(g)}, ..., \overline{\mathbf{o}}_Q^{(g)})$ that are sampled *i.i.d* from $\overline{\mathcal{O}}^{(g)}$ with sampling size Q, which is exactly the reverse procedure of Eq. (2).

Therefore, if both generator G and discriminator D are optimal, the multinomial distribution $p(\overline{\mathcal{O}})$ of distinct meta-path patterns can be preserved in all three steps of our generation framework. □

5 Experiment

In this section, we compare HGEN to the adaption of closest state-of-the-art baselines, demonstrating its effectiveness in generating realistic heterogeneous graphs in diverse settings. The code and dataset can be found at: https://github.com/lingchen0331/HGEN.

5.1 Data

We perform experiments on three synthetic heterogeneous graph datasets and three real-world heterogeneous graph datasets. We summarize the statistics of datasets in Table 2.

Synthetic Datasets. We synthesis random heterogeneous graphs of different sizes through the combination of N overlapping homogeneous graphs, where the overlap is accomplished by node sharing. We generate three random heterogeneous graphs (named as Syn_{100}, Syn_{200}, and Syn_{500}) with node size 100, 200, and 500, respectively. The number of node types in each of the synthetic heterogeneous graph is 3. In addition, we sample a random heterogeneous graph Syn_{Multi} with node size 300 that contains multiple edge types between two nodes.

Real-world Datasets. We also employ three large-scale real-world heterogeneous graph datasets in our experiment.

- **PubMed**. This dataset consists of four classes of nodes: Gene (G), Disease (D), Chemical (C), and Species (S). We construct a sub-graph that relates to all Chemical nodes labeled in [10]. There are $1,565$ nodes and $13,532$ edges.
- **IMDB**. This movie-related heterogeneous graph is adopted from [17], which contains three node types: Director (D), Actor (A), Movie (M), and Genre (G). We construct a subgraph that contains all the movies with a score ≥ 7.5. This graph contains $1,653$ nodes and $4,267$ edges.
- **DBLP**. This heterogeneous graph adopted from [17] contains Paper (P), Author (A), Venue (V), and Term (T) as node types. We sample a subgraph that is related to five computer science venues: *KDD*, *WSDM*, *WWW*, *ICDM*, and *ICML*. There are $1,565$ nodes and $47,885$ edges.

5.2 Experiment Setting

In our experiment, we focus on meta-paths with length 1, 2, and 3 as they are the most common ones in heterogeneous graphs [11]. We sample 10 graphs from each of the trained models and report their average results and standard deviation in Table 3. We randomly select 60% of the edges for training, and the remaining graph is used for testing.

Baselines. Since no baseline is available for the novel task of heterogeneous graph generation, we carefully adapt four state-of-the-art graph generation methods: NetGAN [22], GraphVAE [23], VGAE [25], and GraphRNN [2]. We utilize node type information as node features of the input graph in GraphVAE and VGAE. In addition, we modify NetGAN and GraphRNN to make them available to generate node types. HGEN refers to the model that does not generate network motifs for the proposed method. We further compare HGEN-Motif, which generates network motifs along with meta-paths.

Evaluation Metrics. The evaluation of heterogeneous graph generation can be divided into three aspects. 1) *Graph Statistical Properties*: we focus on six typical statistics as widely used in [22, 31, 36] for measuring the structural similarity, including LCC (the size of the largest connected component),

TC (Triangle count), Clustering Coef. (clustering coefficient); Powerlaw Coef. (power-law distribution of the node degree distribution), Assortativity, and Degree Distribution Dist. (Node degree distribution Maximum Mean Discrepancy distance). 2) *Graph Novelty and Uniqueness.* Ideally, we would want the generated graphs to be diverse and similar, but not identical. To quantify this aspect, we check the uniqueness between the generated graphs by calculating their edit distances. Specifically, we align the node order between the test graph and the generated graph, and calculate the EO Rate (edge overlapping rate) between the generated graphs and the testing graphs for measuring the novelty of the generated graphs. A higher EO Rate indicates the generation method tends to generate more similar graphs than other approaches. The uniqueness is utilized to capture the diversity of generated graphs. To calculate the uniqueness of a generated graph, we let each model to generate 100 graphs, and the generated graphs that are subgraph isomorphic to some other generated graphs are first removed. The percentage of graphs remaining after this operation is defined as uniqueness. For example, if the model generates 100 graphs, all of which are identical, the uniqueness is $1/100 = 1\%$. 3) *Meta-path Ratio Properties*: We measure the preservation of meta-path distribution in two metrics. Firstly, we measure the meta-path length ratio preservation. Secondly, under different meta-path lengths, we also measure the distribution of the frequent meta-path patterns.

5.3 Quantitative Analysis

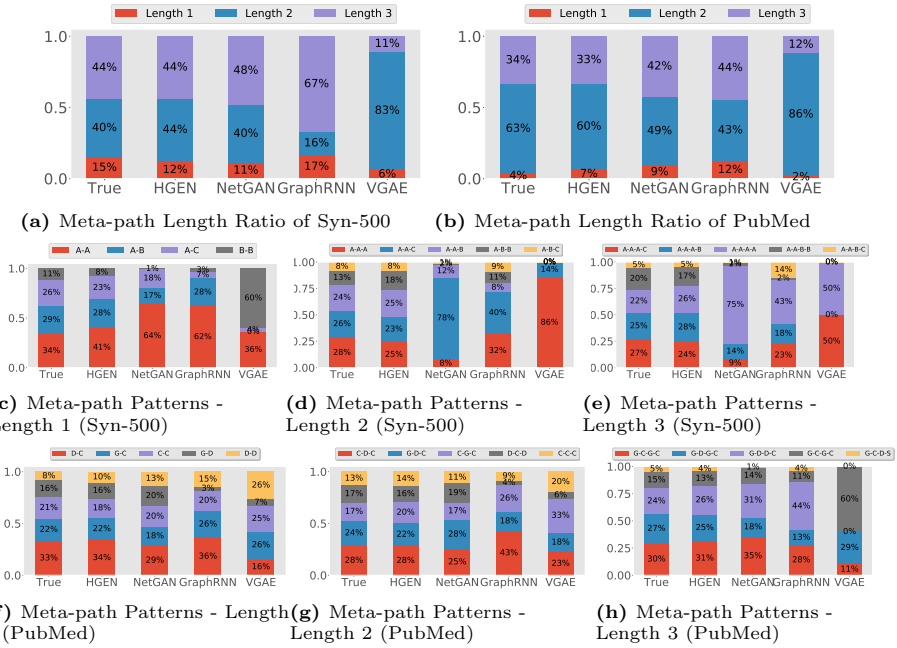
Preservation of Graph Statistical Properties. We evaluate the performance of HGEN and its extended model HGEN-Motif against all the baselines on the standard graph statistics, and the results are shown in Table 3. Overall, HGEN and HGEN-Motif achieve competitive performance with very few exceptions on all metrics over synthetic and real-world datasets. We report several observations from the table: 1) *Node-level similarity*: HGEN-based models are the dominant performer in most node-level metrics. Although there are no significant differences in both Assortativity and Power-law Coef. among all the algorithms, HGEN rank top with very few exceptions in the node degree distribution distance with at least 40% improvement, which indicates that HGEN can effectively capture the degree distribution of all types of nodes through jointly learning both meta-path and random walk distribution. 2) *Graph level similarity*: HGEN-based models still exceed other baselines by effectively preserving the community distribution. Specifically, for all the datasets with rich local community information (e.g., PubMed and synthetic datasets), HGEN-based models can utilize the heterogeneous node embedding for preserving the higher-order structural information in the generated heterogeneous walks, which leads to better performance in metrics like LCC, TC, and Clustering Coef.. However, in heterogeneous graphs with rare high-order structures, the performance of HGEN-based models are comparatively less impressive. 3) As shown in Table 3, the random-walk based method HGEN and NetGAN can generally achieve stable performance than one-shot based (e.g., VGAE and

Graphs	Models	LCC	TC	Clustering Coef.	Powerlaw Coef.	Assortativity	Degree Distribution Dist.	EO Rate	Uniqueness
Syn-100	GraphRNN	78.43 ± 2.23	16.62 ± 5.42	0.002 ± 0.01	1.611 ± 0.09	**−0.153 ± 0.07**	2.19e−2 ± 3.21e−3	37.21% ± 1.08%	33.09% ± 7.06%
	NetGAN	80.12 ± 3.45	6.79 ± 1.76	0.001 ± 0.00	1.524 ± 0.21	−0.213 ± 0.09	1.33e−2 ± 6.46e−3	8.74% ± 0.82%	**94.03%** ± **0.49%**
	GraphVAE	99.01 ± 0.00	224.81 ± 5.13	0.70 ± 0.04	4.579 ± 0.05	−0.73 ± 0.05	3.71e−1 ± 1.98e−2	11.5% ± 1.09%	65.54% ± 2.98%
	VGAE	48.9 ± 4.63	63.7 ± 46.25	0.184 ± 0.06	**1.87 ± 0.10**	0.1 ± 0.03	2.23e−1 ± 6.08e−2	**3.23%** ± **0.09%**	51.1% ± 3.04%
	HGEN	81.13 ± 2.42	53.12 ± 3.78	**0.079 ± 0.01**	1.782 ± 0.01	−0.114 ± 0.03	**8.79e−3 ± 3.12e−3**	10.2% ± 0.17%	92.97% ± 0.72%
	HGEN-Motif	**80.54 ± 3.12**	**47.34 ± 2.69**	0.089 ± 0.01	1.673 ± 0.02	−0.207 ± 0.06	6.28e−2 ± 1.77e−3	13.7% ± 0.17%	89.32% ± 1.32%
	Real	85	36	0.072	1.832	-0.169	N/A	N/A	N/A
Syn-200	GraphRNN	132.76 ± 1.08	2.54 ± 0.77	0.001 ± 0.00	1.603 ± 0.01	−0.05 ± 0.01	5.15e−2 ± 3.07e−3	25.81% ± 2.65%	27.72% ± 3.07%
	NetGAN	153 ± 1.56	2.24 ± 0.35	0.001 ± 0.00	1.579 ± 0.31	−0.008 ± 0.001	6.43e−2 ± 4.2e−3	11.32% ± 0.77%	95.88% ± 3.19%
	GraphVAE	195.43 ± 1.12	51.32 ± 1.01	0.002 ± 0.001	5.377 ± 0.21	−0.75 ± 0.05	5.38e−1 ± 1.7e−2	**1.78%** ± **0.41%**	64.37% ± 2.94%
	VGAE	86.2 ± 16.93	860.4 ± 185.9	0.23 ± 0.04	**1.787 ± 0.08**	0.2 ± 0.15	8.53e−2 ± 2.14e−2	3.74% ± 0.08%	59.65% ± 1.46%
	HGEN	158.5 ± 2.64	38.5 ± 5.26	**0.043 ± 0.01**	1.732 ± 0.02	**−0.065 ± 0.04**	2.25e−2 ± 5.5e−3	4.22% ± 0.67%	**96.31%** ± **5.11%**
	HGEN-Motif	**169 ± 4.89**	**29.28 ± 3.43**	0.049 ± 0.01	1.72 ± 0.02	−0.015 ± 0.08	**3.73e−3 ± 2.5e-4**	8.6% ± 1.3%	93.57% ± 3.61%
	Real	180	28	0.037	1.809	-0.089	N/A	N/A	N/A
Syn-500	GraphRNN	311.59 ± 2.14	11.53 ± 5.57	0.004 ± 0.001	1.862 ± 0.01	1.862 ± 0.002	4.05e−2 ± 1.1e−3	21.87% ± 0.86%	29.54% ± 4.32%
	NetGAN	305.81 ± 14.28	**3 ± 1.21**	**0.001 ± 0.001**	1.812 ± 0.07	0.03 ± 0.12	4.83e−2 ± 7.4e−4	6.72% ± 0.13%	93.98% ± 0.21%
	VGAE	97.0 ± 29.24	4346.2 ± 453.62	0.193 ± 0.02	1.77 ± 0.06	−0.022 ± 0.09	2.22e−1 ± 2.4e−2	5.46% ± 1.12%	63.65% ± 3.1%
	HGEN	347.88 ± 7.63	74.88 ± 4.78	0.031 ± 0.01	**1.865 ± 0.02**	**−0.097 ± 0.01**	2.81e−2 ± 3.4e−3	**1.49% ± 0.11%**	**95.89%** ± **1.18%**
	HGEN-Motif	**398 ± 7.23**	59.63 ± 5.68	0.024 ± 0.03	1.72 ± 0.02	−0.015 ± 0.08	**3.73e−3 ± 2.5e-4**	10.32% ± 2.5%	89.67% ± 1.32%
	Real	417	8	6.5e−3	1.978	-0.12	N/A	N/A	N/A
Syn-Multi	GraphRNN	219.32 ± 32.65	23.67 ± 13.88	0.003 ± 0.001	0.962 ± 0.02	1.373 ± 0.04	3.77e−2 ± 2.9e−3	36.71% ± 4.66%	42.12% ± 4.99%
	NetGAN	215.48 ± 6.99	23 ± 4.74	0.02 ± 0.001	1.076 ± 0.03	0.67 ± 0.22	3.78e−2 ± 3.5e−3	19.87% ± 0.54%	89.13% ± 0.34%
	VGAE	106.8 ± 37.12	1432.8 ± 283.12	0.087 ± 0.03	1.48 ± 0.09	−0.039 ± 0.12	2.22e−1 ± 2.4e−2	7.93% ± 0.78%	27.78% ± 9.3%
	HGEN	295.82 ± 3.91	35.71 ± 5.76	0.019 ± 0.008	**1.381 ± 0.03**	**−0.055 ± 0.03**	1.38e−2 ± 1.3e−3	**1.49% ± 0.11%**	**95.89%** ± **1.18%**
	HGEN-Motif	**282.78 ± 3.79**	**43.32 ± 2.99**	**0.023 ± 0.003**	1.087 ± 0.05	−0.079 ± 0.04	**1.21e−2 ± 2.4e−4**	2.88% ± 0.57%	91.32% ± 1.07%
	Real	275	47	0.027	1.243	-0.36	N/A	N/A	N/A
PubMed	GraphRNN	1563.23 ± 32.46	1549.79 ± 33.62	0.01 ± 0.007	**1.753 ± 0.04**	−0.03 ± 0.01	1.61e−1 ± 3.71e−2	13.41% ± 1.24%	54.62% ± 4.32%
	NetGAN	793.2 ± 41.5	18.3 ± 0.9	0.001 ± 0.00	1.47 ± 0.11	−0.12 ± 0.02	6.69e−2 ± 1.5e−3	4.32% ± 0.54%	78.03% ± 0.19%
	VGAE	347.9 ± 7.03	70,982.2 ± 4,086.53	0.234 ± 0.01	2.48 ± 0.01	−0.466 ± 0.01	1.38e−1 ± 4.8e−3	≈ 0%	22.87% ± 1.68%
	HGEN	825.6 ± 22.1	1569.3 ± 31.3	0.034 ± 0.003	1.634 ± 0.07	−0.143 ± 0.08	**3.92e−2 ± 7.5e−4**	0.07% ± 0.01%	93.91% ± 0.12%
	HGEN-Motif	**897.3 ± 12.5**	**1972.3 ± 46.7**	**0.051 ± 0.002**	1.505 ± 0.04	**−0.162 ± 0.07**	5.21e−2 ± 6.2e−4	0.12% ± 0.03%	**94.12%** ± **0.33%**
	Real	948	2,114	0.068	1.75	−0.208	N/A	N/A	N/A
IMDB	GraphRNN	1425.47 ± 121.5	142.13 ± 5.87	0.179 ± 0.02	2.97 ± 0.05	0.05 ± 0.04	1.98e−1 ± 2.61e−3	9.87% ± 0.51%	21.52% ± 3.31%
	NetGAN	932.5 ± 8.49	**0.0 ± 0.0**	**0.0 ± 0.0**	2.08 ± 0.01	**−0.25 ± 0.07**	1.36e−1 ± 1.89e−3	7.62% ± 0.07%	82.69% ± 1.27%
	VGAE	635.2 ± 4.16	7,752.4 ± 281.32	0.141 ± 0.01	2.02 ± 0.02	−0.49 ± 0.15	1.9e−1 ± 2.33e−3	≈ 0%	42.71% ± 1.47%
	HGEN	**945.2 ± 11.54**	26.0 ± 3.28	3.56e−3 ± 0.0	2.16 ± 0.01	−0.19 ± 0.04	4.36e−2 ± 4.25e−4	2.69% ± 0.04%	**88.71%** ± **0.39%**
	HGEN-Motif	932.12 ± 7.58	53.1 ± 4.66	4.66e−3 ± 0.0	**2.23 ± 0.04**	−0.21 ± 0.03	**1.29e−2 ± 3.13e−3**	5.32% ± 0.13%	87.43% ± 0.52%
	Real	1,074	1	4.43e−4	2.51	-0.235	N/A	N/A	N/A
DBLP	NetGAN	10,353 ± 72.71	**0.0 ± 0.0**	**0.0 ± 0.0**	3.308 ± 0.41	−0.059 ± 0.03	5.03e−1 ± 2.1e−2	5.48% ± 0.32%	**72.51%** ± **0.32%**
	VGAE	3,771 ± 236.29	1214.69 ± 452.61	0.271 ± 0.06	1.579 ± 0.07	−0.44 ± 0.11	8.71e−2 ± 1.77e−3	≈ 0%	17.26% ± 0.41%
	HGEN	5,163 ± 21.41	1068 ± 12.83	0.018 ± 0.001	**1.793 ± 0.21**	**−0.157 ± 0.03**	5.82e−3 ± 1.67e−4	1.55% ± 0.09%	66.59% ± 0.17%
	HGEN-Motif	**5,624 ± 78.32**	788 ± 65.32	0.012 ± 0.001	**1.705 ± 0.14**	**−0.157 ± 0.03**	**3.57e−3 ± 2.55e−4**	1.26% ± 0.1%	58.89% ± 1.79%
	Real	5,513	0.0	0.0	1.855	−0.201	N/A	N/A	N/A

Table 3: Performance evaluation over compared baselines. The *Real* rows include the values of real graphs, while the rest are the evaluation results of different algorithms. The best performance (the closest to real value) achieved under each metric for a particular dataset is highlighted in bold font. Note that we do not include GraphVAE in datasets with (≥ 300) nodes and GraphRNN in datasets with ($\geq 10,000$) nodes because the programs return errors.

GraphVAE) and sequential-based (GraphRNN) generative models across all datasets. The reason is that random walk based methods learn the overall graph distribution by learning the distribution of its discrete random walks, which is not sensitive to various graph characteristics. 4) Table 3 also shows that VGAE cannot produce realistic graphs even though it achieves the best performance in some metrics, which is expected since the primary purpose of VGAE is learning node embeddings but not generating entire graphs. In addition, as the size of the graph increases, GraphRNN also fails to generate realistic graphs because of the weak scalability of auto-regressive models.

Graph Novelty and Uniqueness. The results of graph novelty and uniqueness are reported in the right two columns in Table 3. Specifically, HGEN achieves a generally lower EO rate across all datasets, indicating that HGEN does not purely memorize the seen heterogeneous walks in the training data.



(a) Meta-path Length Ratio of Syn-500

(b) Meta-path Length Ratio of PubMed

(c) Meta-path Patterns - Length 1 (Syn-500)

(d) Meta-path Patterns - Length 2 (Syn-500)

(e) Meta-path Patterns - Length 3 (Syn-500)

(f) Meta-path Patterns - Length 1 (PubMed)

(g) Meta-path Patterns - Length 2 (PubMed)

(h) Meta-path Patterns - Length 3 (PubMed)

Fig. 7: The meta-path distribution comparison. 7a and 7b are the generated meta-path length distribution for Syn_500 dataset and PubMed dataset, respectively. 7c - 7e and 7f - 7h are frequent meta-path patterns distribution with length 1, 2, 3 for Syn_500 dataset and PubMed dataset, respectively.

In contrast, GraphRNN has a higher EO rate, indicating GraphRNN regenerates graphs it saw during training. In addition, VGAE achieves the lowest EO rate since it fails to generate realistic heterogeneous graphs. For Uniqueness, HGEN also exceeds other one-shot and sequential-based algorithms by an evident margin, demonstrating the generated graphs' diversity.

Preservation of Graph Semantic Properties To further demonstrate the performance of HGEN, we evaluate the performance of meta-path distribution preservation with other baselines. Specifically, we measure the meta-path distribution from two aspects: 1) the overall meta-path length ratio preservation in generated graphs and 2) frequent meta-path patterns under each length. In general, all the methods can approximately maintain the meta-path length ratio except for VGAE. However, HGEN can constantly achieve a better performance as shown in Figure 7a and 7b. 2) As shown in Figure 7c - 7e and 7f - 7h, HGEN can outperform other methods by at least 10% in preserving the ratio of specific meta-path patterns under each length, which is expected since HGEN is able to learn and maintain the meta-path distribution from the observed graphs while others cannot.

Method	NetGAN		GraphRNN		GraphVAE		VGAE		HGEN		HGEN-Motif	
	F-1	AUC	F-1	AUC	F-1	AUC	F-1	AUC	F-1	AUC	F-1	AUC
Syn_100	73.24	88.93	79.67	89.67	54.32	78.63	66.78	87.89	78.93	92.63	**81.52**	**93.11**
Syn_200	76.29	81.63	**82.54**	**93.41**	58.67	77.63	59.63	77.89	77.92	89.68	79.25	92.66
Syn_500	81.32	89.86	80.54	92.67	77.85	84.63	61.76	83.81	73.69	89.63	**84.32**	**93.42**
PubMed	68.54	77.63	71.32	78.77	56.53	75.45	54.32	76.42	**72.89**	**79.96**	71.32	79.63
IMDB	64.96	77.82	52.42	71.63	62.53	77.64	67.21	81.53	**70.83**	80.82	68.32	**82.78**
DBLP	70.54	81.32	64.32	84.31	52.53	59.63	62.53	82.81	72.57	**89.92**	**73.58**	74.51
Syn_Multi	–	–	–	–	–	–	–	–	69.54	87.63	**72.45**	**80.32**

Table 4: Link Prediction Performance (in %). We randomly sampled 60% edges as a training graph and the rest of the edges as testing. For Syn_Multi dataset, since no existing methods are capable of generating different meta-relations between edges, we only compare HGEN with its variant HGEN-Motif.

	HGEN-S	HGEN-E	HGEN-A	HGEN	HGEN-Motif	Real
LCC	1563.76	824.14	819.32	825.6	**897.3**	948
TC	1453.23	784.34	863.53	1569.3	1972.3	2114
OC	512.38	379.63	432.67	453.28	**509.45**	576
Clustering Coef.	0.026	0.015	0.016	0.034	**0.051**	0.068
Power Law Coef.	1.649	**1.652**	1.621	1.634	1.505	1.75
Assortativity	-0.09	-0.132	-0.131	−0.143	**−0.162**	-0.208
Node Degree Dist.	**0.0354**	0.0388	0.0515	0.0392	0.0521	N/A

Table 5: Ablation Study in PubMed Dataset

5.4 Link Prediction

Link prediction is commonly used as an evaluation to predict the existence of unobserved links (i.e., edges) in a given observed graph, and we use it to evaluate the generalization power of HGEN and other approaches. We randomly mask out 40% of the edges as a testing set and report the performance with two commonly used metrics: area under the ROC curve (AUC) and F1 score (F1). We conducted the experiments with other approaches on all datasets; note that the Syn-Multi dataset contains multiple edge relations while other approaches cannot handle the multi-typed edge generation job. We, therefore, only compare HGEN variants.

The results are reported in Table 4. Although there is no overall dominant method, HGEN-based methods still achieve comparably more impressive performance. With the effort to preserve local semantic distribution and higher-order structural information, HGEN-based models can leverage observed heterogeneous information to complement the rest. In addition to the normal link prediction, HGEN-based models can still perform well in recovering the multi-edge type information, proving HGEN can characterize different meta-path distributions in the observed heterogeneous graph.

5.5 Ablation Study

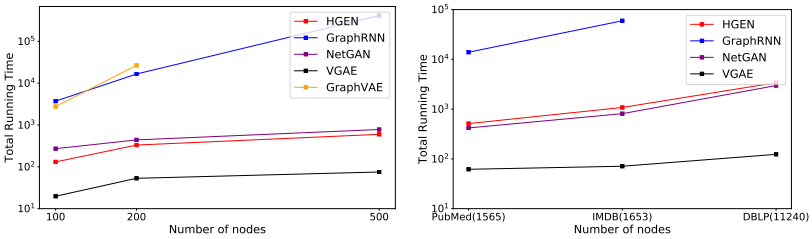
We further conduct ablation studies on the PubMed dataset to evaluate the effect of different components in HGEN, and the results are exhibited

in Table 5. The ablative experiments are conducted based on each of the essential components in our architecture. Specifically, we select a single large heterogeneous walk length - 8 to replace the heterogeneous walk length 1, 2, and 3 in our model, and the resulting model is called HGEN-S. We also independently remove the heterogeneous node embedding to let the generator uniformly sample the next node, and the resulting model is named HGEN-E. Moreover, we replace the heterogeneous graph assembler with a probability-based graph assembler, namely HGEN-A. Lastly, we add another evaluation metric - OC (Orbit Count) to quantify how HGEN and HGEN-Motif perform when preserving higher-order structures.

As shown in Table 5, all the ablative models achieve similar results in node-level metrics like Powerlaw Coef., Assortativity, which is because HGEN can well capture this node-level information through learning the heterogeneous walk distribution. Other than that, we observe: 1) HGEN-S can construct a larger sub-graph since the length of the heterogeneous walk is largely greater than HGEN, but the large subgraph doe not makes any improvements in terms of capturing the heterogeneous structural information. The reason is there are rarely long meta-path in the heterogeneous graph since longer meta-paths are highly redundant because of the shared sub-parts [11]. We instead choose 1, 2, and 3 as our meta-path lengths to make the whole generation more flexible. 2) removing the heterogeneous node embedding would make HGEN-E hard to capture the local graph structure since HGEN relies on the encoded neighborhood information to make the node sampling aware of the local structure. 3) as shown in the node degree distribution evaluation, replacing the heterogeneous graph assembler with a probabilistic graph assembler would cause HGEN-A hard to capture the latent heterogeneous node distribution because it uniformly samples edges from the generated walks and completely neglects the generated meta-path information. However, HGEN takes meta-paths as a basic unit to sample edges so that it can effectively preserve the overall distribution of meta-paths as proved in Theorem 1. Therefore, the node degree distribution under each type can be well preserved. Finally, HGEN-Motif performs better than HGEN in generating the most similar triangle and orbit counts with the observed graph, which also justifies the choice of adding motif as the based generation unit.

5.6 Running Time Comparison

Figure 8 shows the results of our running time experiments. The running times on both synthetic and real-world datasets, including both training and inference time, are shown with respect to the growth of the number of nodes in both synthetic and real-world datasets. All running times are in the $log-10$ scale. As shown in both figures, random-walk-based generative models (HGEN and NetGAN) have a constant running time growth in terms of the number of nodes, which is especially important when dealing with large graphs. Even though VGAE is much faster in running time, it is indeed a representation



(a) Synthetic Dataset Running Time **(b)** Real Dataset Running Time

Fig. 8: Running time comparison of different models in both synthetic and real-world datasets. It is clear that GraphVAE is not scalable in generating graphs with more than 200 nodes. GraphRNN also fails in generating large graphs (with more than $10,000$ nodes). The proposed HGEN exhibits a linear running time growth in terms of the growth of graph size.

learning framework based on GCN and lacks the ability to generate realistic heterogeneous graphs, and the results are also reflected in Table 3. Both GraphRNN and GraphVAE fail to compare with HGEN in model scalability because their designs require at least $O(|\mathcal{V}|^2)$ to process the transformed node sequence and adjacency matrix.

5.7 Graph Visualization

Since it is nearly impossible to judge whether a graph is realistic only by statistics, we visualize the generated graph to further demonstrate the performance of HGEN (Figure 9). Visually, HGEN looks the most similar, while both GraphVAE and VGAE is the most dissimilar. This result is consistent with the quantitative results obtained in Table 3. For one-shot based generative models, GraphVAE and VGAE, they fail to capture the structural similarity of the observed heterogeneous graph. For the sequential-based and random walk based graph generative methods, GraphRNN and NetGAN can successfully mimic the structure similarity but fail to preserve the global heterogeneous graph properties (e.g., overall meta-path ratio).

6 Conclusion

This paper focuses on a new problem: heterogeneous graph generation. To achieve this, we propose a novel framework - HGEN for the heterogeneous graph generation. Specifically, the proposed method consists of a novel heterogeneous walk/motif generator that can hierarchically generate meta-paths and a heterogeneous graph assembler that can construct new graphs by sampling from the generated heterogeneous walks in a stratified manner. As the extension of the meta-path-based HGEN, this paper proposes a novel module HGEN-Motif that considers the network motif as one of the basic generation units in order to better capture the higher-order structural distribution.

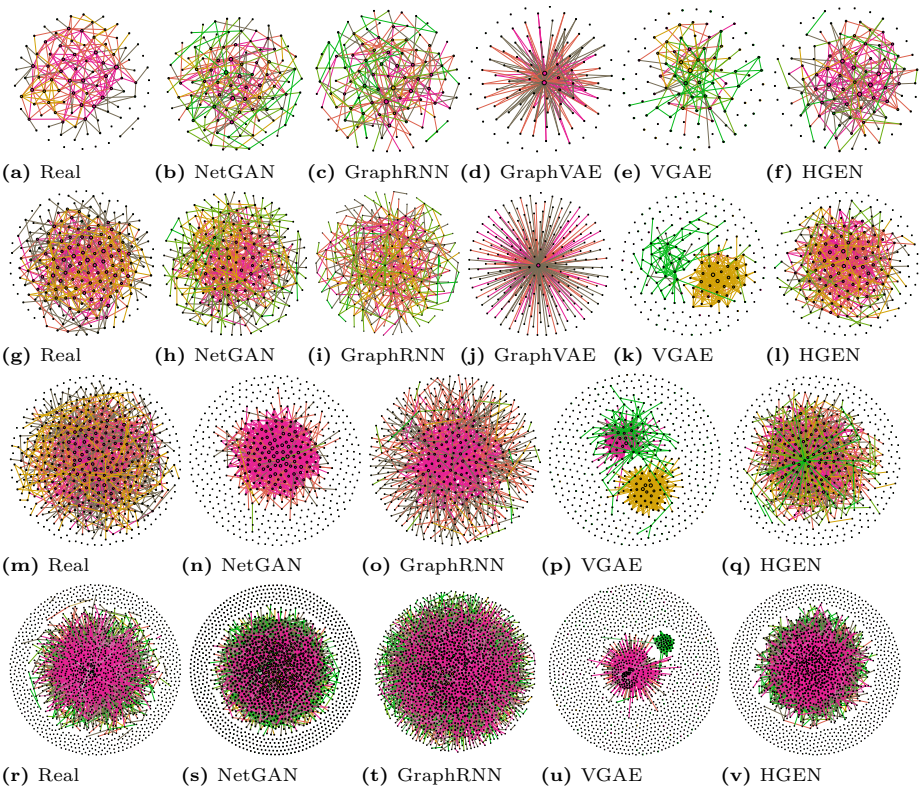


Fig. 9: 9a - 9f are the generated graph of the Syn_100 dataset, 9g - 9l are the generated graph of the Syn_200 dataset, 9m - 9q are the generated graph of the Syn_500 dataset, and 9r - 9v are the generated graphs of the PubMed dataset. (Better to see with color)

We further unified the training framework to enable the generator to generate various heterogeneous instances to meet different statistics of the observed heterogeneous graph. Compared to existing deep graph generation methods, HGEN is tailored for heterogeneous graph generation and can provide more insights into heterogeneous graph mining studies. It is evaluated from the experiments that existing deep graph generation methods cannot well preserve the local semantic, higher-order structural, and global distribution of an observed heterogeneous graph, and thus cannot handle the unique job of heterogeneous graph generation. As the first-of-its-kind heterogeneous graph generation method, HGEN can not only provide benchmarks for the many heterogeneous graph-related studies, but it can also enrich our understanding of the implicit properties of heterogeneous graphs.

References

[1] Sun, M., Li, P.: Graph to graph: a topology aware approach for graph structures learning and generation. In: Proc. of the AISTATS (2019)

[2] You, J., Ying, R., Ren, X., Hamilton, W., Leskovec, J.: Graphrnn: Generating realistic graphs with deep auto-regressive models. In: Proc. of the ICML (2018)

[3] Yun, S., Jeong, M., Kim, R., Kang, J., Kim, H.J.: Graph transformer networks. In: Proc. of the NIPS, pp. 11983–11993 (2019)

[4] Wu, L., Cui, P., Pei, J., Zhao, L.: Graph Neural Networks: Foundations, Frontiers, and Applications, (2021)

[5] Ling, C., Chowdhury, T., Jiang, J., Wang, J., Zhang, X., Chen, H., Zhao, L.: Deepgar: Deep graph learning for analogical reasoning. In: Proc. of the ICDM (2022)

[6] Ling, C., Jiang, J., Wang, J., Liang, Z.: Source localization of graph diffusion via variational autoencoders for graph inverse problems. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 1010–1020 (2022)

[7] Zhao, L.: Event prediction in the big data era: A systematic survey. CSUR **54**(5), 1–37 (2021)

[8] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Philip, S.Y.: A comprehensive survey on graph neural networks. TNNLS (2020)

[9] Guo, X., Zhao, L.: A systematic survey on deep generative models for graph generation. arXiv preprint arXiv:2007.06686 (2020)

[10] Yang, C., Xiao, Y., Zhang, Y., Sun, Y., Han, J.: Heterogeneous network representation learning: A unified framework with survey and benchmark. TKDE (2020)

[11] Sun, Y., Han, J., Yan, X., Yu, P.S., Wu, T.: Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. Proc. of the VLDB Endowment **4**(11), 992–1003 (2011)

[12] Sun, Y., Han, J.: Meta-path-based search and mining in heterogeneous information networks. Tsinghua Science and Technology (2013)

[13] Sun, Y., Norick, B., Han, J., Yan, X., Yu, P.S., Yu, X.: Pathselclus: Integrating meta-path selection with user-guided object clustering in heterogeneous information networks. TKDD (2013)

[14] Fu, T.-y., Lee, W.-C., Lei, Z.: Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning. In: Proc. of the CIKM (2017)

[15] Dong, Y., Chawla, N.V., Swami, A.: metapath2vec: Scalable representation learning for heterogeneous networks. In: Proc. of the KDD (2017)

[16] Hu, Z., Dong, Y., Wang, K., Sun, Y.: Heterogeneous graph transformer. In: Proc. of the WebConf, pp. 2704–2710 (2020)

[17] Wang, X., Ji, H., Shi, C., Wang, B., Ye, Y., Cui, P., Yu, P.S.: Heterogeneous graph attention network. In: Proc. of the WebConf (2019)

[18] Zhang, W., Paudel, B., Wang, L., Chen, J., Zhu, H., Zhang, W., Bernstein, A., Chen, H.: Iteratively learning embeddings and rules for knowledge graph reasoning. In: Proc. of the WebConf, pp. 2366–2377 (2019)

[19] Shi, C., Zhang, Z., Luo, P., Yu, P.S., Yue, Y., Wu, B.: Semantic path based personalized recommendation on weighted heterogeneous information networks. In: Proc. of the CIKM, pp. 453–462 (2015)

[20] Gupta, A.: Generating large-scale heterogeneous graphs for benchmarking. In: Specifying Big Data Benchmarks, pp. 113–128 (2012)

[21] Guo, X., Zhao, L., Nowzari, C., Rafatirad, S., Homayoun, H., Dinakarrao, S.M.P.: Deep multi-attributed graph translation with node-edge co-evolution. In: Proc. of the ICDM, pp. 250–259 (2019)

[22] Bojchevski, A., Shchur, O., Zügner, D., Günnemann, S.: Netgan: Generating graphs via random walks. In: Proc. of the ICML (2018)

[23] Simonovsky, M., Komodakis, N.: Graphvae: Towards generation of small graphs using variational autoencoders. In: Proc. of the ICANN, pp. 412–422 (2018)

[24] Caridá, V., Jalilifard, A., Mansano, A., Cristo, R.: Can netgan be improved on short random walks? In: Proc. of the BRACIS (2019)

[25] Kipf, T.N., Welling, M.: Variational graph auto-encoders. arXiv preprint arXiv:1611.07308 (2016)

[26] Carranza, A.G., Rossi, R.A., Rao, A., Koh, E.: Higher-order clustering in complex heterogeneous networks. In: Proc. of the KDD, pp. 25–35 (2020)

[27] Li, J., Peng, H., Cao, Y., Dou, Y., Zhang, H., Philip, S.Y., He, L.: Higher-order attribute-enhancing heterogeneous graph neural networks. IEEE Transactions on Knowledge and Data Engineering **35**(1), 560–574 (2021)

[28] Ling, C., Yang, C., Zhao, L.: Deep generation of heterogeneous networks. In: 2021 IEEE International Conference on Data Mining (ICDM), pp. 379–388 (2021)

[29] De Cao, N., Kipf, T.: Molgan: An implicit generative model for small molecular graphs. arXiv preprint arXiv:1805.11973 (2018)

[30] Wang, H., Wang, J., Wang, J., Zhao, M., Zhang, W., Zhang, F., Xie, X., Guo, M.: Graphgan: Graph representation learning with generative adversarial nets. In: Proc. of the AAAI (2018)

[31] Yang, C., Zhuang, P., Shi, W., Luu, A., Li, P.: Conditional structure generation through graph variational generative adversarial nets. In: Proc. of the NIPS, pp. 1340–1351 (2019)

[32] Wang, S., Guo, X., Zhao, L.: Deep generative model for periodic graphs. In: Proc. of the NeurIPS (2022)

[33] Ling, C., Cao, H., Zhao, L.: Stgen: Deep continuous-time spatiotemporal graph generation. In: 2022 European Conference on Machine Learning and Principles of Knowledge Discovery in Databases (2022)

[34] Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein generative adversarial networks. In: Proc. of the ICML (2017)

[35] Honda, S., Akita, H., Ishiguro, K., Nakanishi, T., Oono, K.: Graph residual flow for molecular graph generation. arXiv preprint arXiv:1909.13521 (2019)

[36] Goyal, N., Jain, H.V., Ranu, S.: Graphgen: A scalable approach to domain-agnostic labeled graph generation. In: Proc. of the WebConf, pp. 1253–1263 (2020)

[37] Shi, C., Li, Y., Zhang, J., Sun, Y., Philip, S.Y.: A survey of heterogeneous information network analysis. TKDE **29**(1), 17–37 (2016)

[38] Huang, Z., Zheng, Y., Cheng, R., Sun, Y., Mamoulis, N., Li, X.: Meta structure: Computing relevance in large heterogeneous information networks. In: Proc. of the KDD, pp. 1595–1604 (2016)

[39] Sun, L., He, L., Huang, Z., Cao, B., Xia, C., Wei, X., Philip, S.Y.: Joint embedding of meta-path and meta-graph for heterogeneous information networks. In: Proc. of the ICBK, pp. 131–138 (2018)

[40] Gamage, A., Chien, E., Peng, J., Milenkovic, O.: Multi-motifgan (mmgan): Motif-targeted graph generation and prediction. In: Proc. of the ICASSP, pp. 4182–4186 (2020)

[41] Jang, E., Gu, S., Poole, B.: Categorical reparameterization with gumbel-softmax. arXiv preprint arXiv:1611.01144 (2016)

[42] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Proc. of the NeurIPS (2014)