DPGS: Differentially Private Graph Synthesis via Graph Summarization

Xun Ran†, Qingqing Ye†*, Jian Lou‡, Haibo Hu†,

†The Hong Kong Polytechnic University, ‡Sun Yat-sen University

qi-xun.ran@connect.polyu.hk   qqing.ye@polyu.edu.hk   louj5@mail.sysu.edu.cn   haibo.hu@polyu.edu.hk

Abstract—**Many real-world systems, such as social and email networks, can be represented as graphs. Analyzing these structures is crucial for various applications but raises significant privacy concerns. Differential privacy (DP) provides a formal framework for protecting sensitive graph data, ensuring minimal impact from changing individual nodes or edges on the analysis results. A canonical approach in this context is to generate a synthetic graph under DP constraints, which preserves the characteristics of original graph, to support downstream graph analysis. However, existing graph synthesis solutions either perturb the adjacency matrix directly, leading to high noise sensitivity, or use alternative encoding methods that suffer from information loss. To address this, we propose DPGS, a novel framework for Differentially Private Graph Synthesis via graph summarization. By extracting a summarized representation of the original graph, DPGS encodes structural patterns before applying DP, improving noise resilience while preserving key properties. Experiments on real-world datasets show that DPGS outperforms existing methods in maintaining structural integrity under privacy constraints. Our approach enables effective, privacy-preserving graph analysis with enhanced utility.**

Index Terms—**Data privacy, Graph data, Differential privacy**

I. INTRODUCTION

Many real-world systems can be represented as graphs, such as social networks [1], email networks [2], and voting networks [3]. Analyzing these graph data is crucial in a wide range of applications [4]. For instance, Amazon recommends products by analyzing co-purchase networks, where users (i.e., nodes) are linked by shared purchases (i.e., edges) [5]. However, due to the sensitive nature of graph data, direct analysis without privacy protection raises significant concerns.

A classical approach to protecting privacy in graph analysis is anonymization, which removes node identification information [6], [7]. However, prior research has demonstrated that anonymized graphs can still be deanonymized when attackers possess auxiliary information [8], [9].

As a golden standard in the privacy community, differential privacy (DP) [10]–[13] has been applied to protect graph data [14], [15]. The core principle of DP is to ensure that a single node or edge has a limited impact on the analysis results. Most previous studies on differentially private graph analysis focus on developing tailored algorithms for specific graph tasks, such as degree distribution [16], subgraph counts [17], and community discovery [18]. However, such solutions are usually task-specific — for a different task, dedicated DP solutions must be designed from scratch. Unlike this paradigm, releasing a synthetic graph that preserves the semantic structure of the original graph while satisfying DP is a general approach. This method is more advantageous than task-specific algorithms because it supports a wide range of downstream graph analysis tasks.

Despite its benefits, designing an effective solution for accurate synthetic graph generation is non-trivial. This challenge arises from a fundamental trade-off between information retention and noise resilience. Fine-grained methods (such as those perturbing the adjacency matrix) preserve detailed structural information but are highly sensitive to noise, leading to degraded accuracy under tight privacy budgets. In contrast, coarse-grained or aggregated methods (such as clustering or hierarchical models) are more robust to noise but sacrifice fine details, limiting reconstruction fidelity.

Figure 1 summarizes existing works along these two aspects. Approaches like Top-m Filter (TmF) [19], which directly perturbs the adjacency matrix, sit at the high-information, low-resilience end of the spectrum. While they retain rich details, the direct perturbation of individual edges introduces large noise effects. Density-based exploration and reconstruction (DER) [20] improves noise resilience by grouping dense regions and applying quadtree-based aggregation, but it struggles with sparse regions and is computationally heavy. Hierarchical Random Graph (HRG) [21] further increases noise resilience by encoding the graph into a hierarchical form, though this distorts original structures and adds overhead. Node clustering methods like [22] aggregate graph information before adding noise, improving robustness but introducing approximation errors that reduce reconstruction accuracy. Some recent methods, such as PrivGraph [23], additionally rely on external statistics (e.g., degree distributions), which can introduce further privacy risks and distortions.

Our Proposal. In this work, we propose DPGS, a novel approach that strikes a balance between noise resilience and structural preservation by leveraging graph summarization. As illustrated in Figure 2 (a) and (c), graph summarization [24] represents the graph through supernodes and superedges, simplifying the structure while retaining key connectivity patterns. By perturbing this compressed representation, our method reduces the dimensionality and thus the noise impact, while maintaining sufficient structural information for high-quality synthetic graph generation. This strategy has been widely applied in fields like social network analysis, biological networks, and recommendation systems, making it a promising
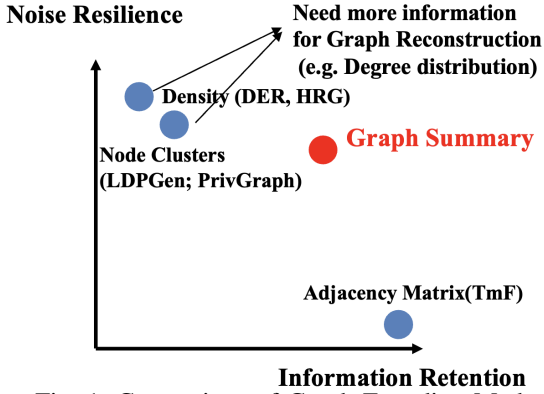

Fig. 1: Comparison of Graph Encoding Methods

direction for privacy-preserving graph synthesis.

Representing the graph at a higher level of abstraction by summarizing its structural patterns, graph summarization allows us to introduce less perturbation to satisfy DP. Specifically, by grouping nodes into supernodes and aggregating connections into superedges, summarization reduces the total number of elements (compared to the original adjacency matrix), which directly lowers the global sensitivity of the function — that is, the maximum change caused by modifying a single edge or node. This reduction enables the addition of less noise to achieve the same level of privacy.

Moreover, summarization preserves key structural information by capturing both intra-group and inter-group relationships, ensuring that essential connectivity patterns (such as community structures, dense regions, and cross-group links) are retained. As a result, even though fine-grained details are abstracted, the summary still holds sufficient information to support accurate reconstruction of graph properties or generate synthetic graphs that reflect meaningful global patterns.

Designing a method that balances utility and privacy remains challenging. Many existing graph summarization approaches, such as random search [23], [24], operate solely on structural signals and struggle to incorporate node features effectively. In contrast, GNNs naturally integrate both structure and features through message aggregation, enabling more meaningful node representations and better initial partitions. However, this advantage comes with privacy risks, as aggregation induces data dependencies that can amplify noise under DP constraints [25]. To address this trade-off, we propose a hybrid approach: we first leverage GNNs to obtain feature-aware initial clusterings, then refine them via random search. The GNN-induced noise is mitigated through a spectral clustering module, while the refinement phase employs the Exponential Mechanism to reduce noise accumulation and accelerate convergence.

Following the idea, DPGS first extracts a summarized representation of the graph, capturing both local and global structures. We then perturb this summary instead of the raw adjacency matrix, allowing for a more stable and noise-resilient encoding. To ensure fidelity, DPGS separately processes dense substructures and sparse interconnections, applying different perturbation strategies tailored to each. Finally,

a post-processing step refines the synthetic graph, ensuring consistency and enhancing utility.

Contributions. In summary, our main contributions are three-fold:

- We propose DPGS, a novel differentially private synthetic graph generation framework that leverages **graph summarization** to improve noise resilience while preserving structural integrity.
- We propose a private selection strategy that uses the exponential mechanism to achieve differentially private graph summarization while balancing utility and privacy.
- We conduct extensive experiments and a real-world case study across multiple datasets and evaluation metrics to demonstrate the effectiveness of DPGS.

II. PRELIMINARIES

In this section, we introduce the preliminaries of our study. We first present graph summarization techniques, which serve as the building block of this work, followed by the background of differential privacy and graph neural networks.

A. Graph Summarization

Input graph: Consider an undirected graph $G = (V, E, \boldsymbol{W}, \boldsymbol{X})$ where $V = \{v_1, v_2, \ldots, v_p\}$ is the node set, $E \subseteq V \times V$ is the edge set and $\boldsymbol{W}$ is the adjacency (weight) matrix. We assume that G is undirected without self-loops: $\boldsymbol{W}_{i,j} > 0$, if $(i,j) \in E$ and $\boldsymbol{W}_{i,j} = 0$ if $(i,j) \notin E$. Finally, $\boldsymbol{X} \in \mathbb{R}^{p \times n} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_p]^T$ is the feature matrix, where each row vector $\boldsymbol{x}_i \in \mathbb{R}^n$ is the feature vector associated with one of p nodes of the graph G. A graph can be conveniently represented by Laplacian matrix $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$, where $\boldsymbol{D} = \mathrm{diag}(\boldsymbol{W}\mathbf{1})$ is the degree matrix. The importance of the graph Laplacian matrix has been well recognized as a tool for embedding, manifold learning, spectral sparsification, clustering, and semi-supervised learning. Owing to these properties, Laplacian matrix representation is more desirable for building graph-based algorithms. We call nodes and edges in G subnodes and subedges, respectively, to distinguish them from those in summary graphs, described below.

Summary graph: A summary graph $\tilde{G} = (\tilde{V}, \tilde{E}, \tilde{\boldsymbol{W}}, \tilde{\boldsymbol{X}})$ of a graph $G = (V, E, \boldsymbol{W}, \boldsymbol{X})$ consists of a set of supernodes $\tilde{V}$, a set of superedges $\tilde{E}$, a weight matrix $\tilde{\boldsymbol{W}}$, and transformed feature matrix $\tilde{\boldsymbol{X}}$. Each superedge $(\tilde{V}^u, \tilde{V}^v) \in \tilde{E}$ connects two supernodes $\tilde{V}^u, \tilde{V}^v \in \tilde{V}$, and if $u = v$, then $\tilde{V}^u, \tilde{V}^v = \tilde{V}^v, \tilde{V}^v$ indicates the self-loop at the supernode $\tilde{V}^v$. We use $\Pi_{\tilde{V}} := \{(\tilde{V}^u, \tilde{V}^v) : \tilde{V}^u, \tilde{V}^v \in \tilde{V}\}$ to denote the all unordered pairs of supernodes, and then $\tilde{E} \subseteq \Pi_{\tilde{V}}$. For each superedge $(\tilde{V}^u, \tilde{V}^v) \in \tilde{E}$, the entries of the weight matrix is defined over the number of subedges between supernodes $\tilde{V}^u$ and $\tilde{V}^v$, i.e., $\tilde{\boldsymbol{W}}_{i,j} := |\{(u,v) \in E : u \in \tilde{V}^i, v \in \tilde{V}^j\}|$. See Figure 2 for an example summary graph.

Reconstructed graph: Given a summary graph $\tilde{G}$, we reconstruct a graph $\hat{G} = (\hat{V}, \hat{E}, \hat{\boldsymbol{W}}, \hat{\boldsymbol{X}})$ following the conventional approach in [26], [27]. The set of subnodes $\hat{V}$ is obtained by taking the union of all supernodes in $\tilde{V}$, while

the reconstructed edge set $\hat{E}$ includes all inter- and intra-supernode connections, defined as $\hat{E} := \{(u,v) \in \hat{V} \times \hat{V} : u \neq v, \ (\tilde{V}^u, \tilde{V}^v) \in \tilde{E}\}$. The reconstructed adjacency weights are computed as:

$$\hat{\boldsymbol{W}}_{u,v} := \frac{\tilde{\boldsymbol{W}}_{u,v}}{|\Pi_{\tilde{V}^u, \tilde{V}^v}|}, \tag{1}$$

where $\Pi_{\tilde{V}^u, \tilde{V}^v} := \{(u,v) : u \neq v, \ u \in \tilde{V}^u, \ v \in \tilde{V}^v\}$ denotes the set of all possible subedges between the two corresponding supernodes. Here, the denominator represents

B. Differential Privacy

Differential Privacy (DP) [10] was originally proposed for privacy-preserving scenarios, where a **trusted data curator** collects data from individual users and perturbs the aggregated results before releasing them. Intuitively, DP guarantees that the inclusion or exclusion of a single record has a limited influence on the output, thereby protecting individual privacy. Formally, we adopt the widely used relaxed definition known as (ε, δ)-differential privacy:

Definition 1 ((ε, δ)-Differential Privacy)**.** *An algorithm $\mathcal{A}$ satisfies (ε, δ)-differential privacy, where $\varepsilon > 0$ and $\delta \in (0, 1)$, if for any two neighboring datasets D and D', we have* $\forall T \subseteq Range(\mathcal{A}) : \ \Pr[\mathcal{A}(D) \in T] \leq e^\varepsilon \cdot \Pr[\mathcal{A}(D') \in T] + \delta$, *where $Range(\mathcal{A})$ denotes the set of all possible outputs of algorithm $\mathcal{A}$.*

We consider two datasets D and D' to be *neighbors*, denoted $D \simeq D'$, if they differ in only one record. In graph settings, two graphs G and G' are considered neighbors under edge-level privacy if one can be obtained from the other by adding or removing a single edge.

Definition 2 (Edge-level (ε, δ)-Differential Privacy)**.** *A randomized mechanism $\mathcal{M}$ satisfies (ε, δ)-Edge-DP if, for any pair of neighboring graphs G and G' and any subset of outputs $\mathcal{Z} \subseteq Range(\mathcal{M})$, the following holds:*
$$\Pr[(|\ \mathcal{M}(G) \in \mathcal{Z}] \leq e^\varepsilon \cdot \Pr[(|\ \mathcal{M}(G') \in \mathcal{Z}] + \delta.$$

Gaussian Mechanism. Gaussian mechanism (GM) achieves (ε, δ)-differential privacy by adding random Gaussian noise to the function output. The noise scale depends on the *global sensitivity* Δ, defined as
$$\Delta = \max_{D \simeq D'} \|f(D) - f(D')\|_2,$$
where f is the query function, and $D \simeq D'$ are neighboring datasets that differ in a single individual. The mechanism $\mathcal{A}$ adds noise sampled from a zero-mean Gaussian distribution with variance calibrated to Δ:
$$\mathcal{A}_f(D) = f(D) + \mathcal{N}\left(0, \sigma^2 \Delta^2\right),$$
where $\sigma \geq \frac{\sqrt{2\ln(1.25/\delta)}}{\varepsilon}$ ensures (ε, δ)-DP. For vector-valued outputs, independent Gaussian noise is added to each component.

Exponential Mechanism. The Gaussian mechanism is applicable when the output of a function f is a real-valued vector. In contrast, the Exponential Mechanism (EM) [28] is designed for scenarios where the output is selected from a finite set. EM selects outputs with probabilities proportional to the exponential of their quality scores, favoring more accurate outcomes. Given an input dataset v, EM samples an output $o \in \mathcal{O}$ according to a predefined quality function $q(v, o)$, which evaluates how suitable o is given the input v. This mechanism requires careful design of the quality function q, which maps each pair (v, o) to a real-valued utility score. The privacy analysis of EM depends on the global sensitivity of q, defined as:
$$\Delta_q = \max_o \max_{v \simeq v'} |q(v, o) - q(v', o)|.$$
$\mathcal{A}$ satisfies ε-differential privacy under the following equation.
$$\Pr[\mathcal{A}_q(v) = o] = \frac{\exp\left(\frac{\varepsilon}{2\Delta_q} q(v, o)\right)}{\sum_{o' \in \mathcal{O}} \exp\left(\frac{\varepsilon}{2\Delta_q} q(v, o')\right)}$$

Composition Properties of (ε, δ)-DP. The following properties characterize how (ε, δ)-differential privacy behaves when multiple mechanisms are combined, enabling the design of complex private algorithms based on modular components.

1) Sequential Composition. If a sequence of k randomized algorithms $\{\mathcal{A}_1, \ldots, \mathcal{A}_k\}$ is applied to the same dataset, and each $\mathcal{A}_i$ satisfies $(\varepsilon_i, \delta_i)$-DP, then their combination satisfies $(\sum_{i=1}^k \varepsilon_i, \sum_{i=1}^k \delta_i)$-DP.
2) Parallel Composition. Suppose the dataset is partitioned into k disjoint subsets, and each subset is processed independently by a mechanism satisfying $(\varepsilon_i, \delta_i)$-DP. Then, the overall privacy guarantee is given by $(\max_i \varepsilon_i, \max_i \delta_i)$.
3) Post-processing Invariance. Any data-independent transformation of the output of a differentially private mechanism does not weaken the privacy guarantee. That is, if $\mathcal{A}$ satisfies (ε, δ)-DP, then for any (possibly randomized) function g, the output $g(\mathcal{A}(D))$ also satisfies (ε, δ)-DP.

C. Graph Neural Networks

Given a graph $G = (V, E, \boldsymbol{W}, \boldsymbol{X})$, an r-layer GNN is a parametric function that can be represented by the following operations:
$$\boldsymbol{H} = \text{GNN}(\boldsymbol{X}, \boldsymbol{W}; \boldsymbol{\Theta}_{\text{GNN}}),$$
where $\boldsymbol{H} \in \mathbb{R}^{p \times d'}$ is the node representations. GNNs can effectively capture local topological and feature-based patterns. The resulting representations $\boldsymbol{H}$ are then used to compute soft cluster assignments, which serve as the foundation for downstream summary construction and noise injection. This not only improves the quality of the summary graph but also ensures that important graph semantics are preserved in the differentially private synthetic graph generation pipeline.

III. PROBLEM STATEMENT

A common approach to privacy-preserving graph analysis is to design dedicated DP mechanisms to specific analysis tasks, such as community discovery and node classification [21]. However, different tasks may require iterative graph queries, each of which requires allocating a portion of the privacy budget. This becomes unsustainable as the number of queries grows, since the accumulation of privacy budget spending
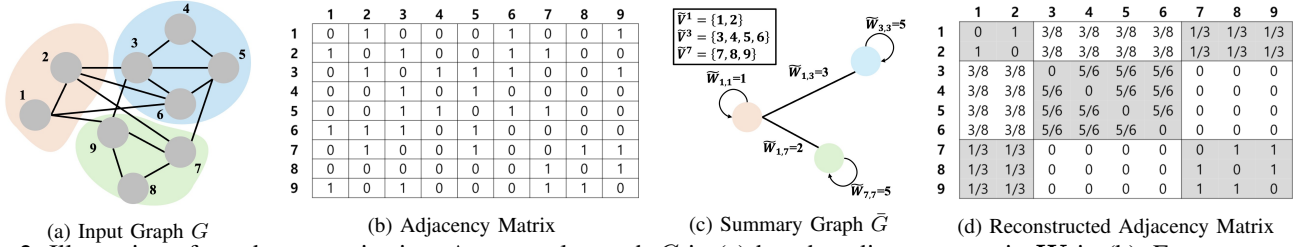
	1	2	3	4	5	6	7	8	9
1	0	1	0	0	0	1	0	0	1
2	1	0	1	0	0	1	1	0	0
3	0	1	0	1	1	1	0	0	1
4	0	0	1	0	1	0	0	0	0
5	0	0	1	1	0	1	1	0	0
6	1	1	1	0	1	0	0	0	0
7	0	1	0	0	1	0	0	1	1
8	0	0	0	0	0	0	1	0	1
9	1	0	1	0	0	0	1	1	0

	1	2	3	4	5	6	7	8	9
1	0	1	3/8	3/8	3/8	3/8	1/3	1/3	1/3
2	1	0	3/8	3/8	3/8	3/8	1/3	1/3	1/3
3	3/8	3/8	0	5/6	5/6	5/6	0	0	0
4	3/8	3/8	5/6	0	5/6	5/6	0	0	0
5	3/8	3/8	5/6	5/6	0	5/6	0	0	0
6	3/8	3/8	5/6	5/6	5/6	0	0	0	0
7	1/3	1/3	0	0	0	0	0	1	1
8	1/3	1/3	0	0	0	0	1	0	1
9	1/3	1/3	0	0	0	0	1	1	0

(a) Input Graph G (b) Adjacency Matrix   (c) Summary Graph $\bar{G}$ (d) Reconstructed Adjacency Matrix

Fig. 2: Illustration of graph summarization. An example graph G in (a) has the adjacency matrix W in (b). From a summary graph $\tilde{G}$ in (c), we restore a graph $\hat{G}$, whose weighted adjacency matrix is $\hat{W}$ in (d). Each subnode in G belongs to one supernode in $\hat{G}$, and the weight of each superedge corresponds to the number of subedges between the two supernodes.

eventually depletes the total budget, leading to poor utility in final results.

To overcome this limitation, we adopt an alternative paradigm. Instead of designing DP mechanisms separately to each analysis task or allocating privacy budgets to excessive queries, we focus on generating a differentially private *synthetic graph* that retains the structural properties of the original graph while satisfying Edge-DP. By doing so, any downstream graph analysis task can be considered as post-processing, without consuming additional privacy budget. This enables a more scalable and practical solution for privacy-preserving graph analytics.

Threat model. In the context of differentially private graph synthesis, an adversary with full access to the published synthetic graph aims to infer whether a specific edge exists in the original graph. For example, given a synthetic social network, the adversary seeks to determine whether a relationship occurred between two users in the original dataset.

Given a graph G, our objective is to generate a synthetic graph $\hat{G}$ that preserves key structural properties of G while satisfying Edge-DP. Instead of directly perturbing the adjacency matrix, we leverage *graph summarization* to encode structural information at a higher level, enhancing noise resilience and privacy protection. The synthetic graph $\hat{G}$ can be used for downstream graph analysis tasks without additional privacy loss.

By summarizing the graph before applying DP, our approach can enhance noise resilience while preserving these essential structural properties. To validate the superiority of our solution, we will follow prior work [15], [19], [20] and evaluate the similarity between $\hat{G}$ and G across five metrics widely used in graph analytics, including community structure, node attributes, degree distribution, path-based connectivity, and topology.

IV. OUR PROPOSAL

A. An Overview of DPGS

We propose a synthetic graph generation framework that achieves *differential privacy* through *graph summarization*, avoiding the need to directly perturb the raw graph structure. The method consists of two main components—**structure summarization** and **feature summarization**—followed by a **graph reconstruction** process. The key idea lies in summarizing information at a higher level of abstraction, which inher-

ently reduces sensitivity and enables accurate reconstruction under stronger privacy guarantees. Figure 3 shows an overview of DPGS.

Structure Summarization. To privately capture structural patterns, we design a GNN-based spectral clustering algorithm with DP guarantees. Unlike traditional methods that cluster solely based on structure, we incorporate node features to enhance clustering quality, leveraging the homophily property in real-world networks. To further improve assignment accuracy without exhausting the privacy budget, we introduce a *refinement step using the exponential mechanism* on a small set of uncertain nodes, amplifying privacy through selective intervention. The output is a noisy but informative *summary edge matrix* $\bar{W}$, representing the connectivity between supernodes with carefully added Gaussian noise. This summarization not only improves the privacy-utility trade-off but also significantly reduces the dimensionality of the graph.

Feature Summarization. To complement the summarized structure, we construct a *compressed feature matrix* $\bar{X}$ that preserves semantic information from the original features. Instead of perturbing raw features or training gradients—which may degrade utility or accumulate noise—we apply *objective perturbation* to a carefully designed convex optimization problem. This guarantees a one-shot DP feature transformation that maintains consistency with the summary graph structure.

Graph Reconstruction. Finally, we reconstruct a synthetic graph based on the summarized outputs. Using the learned cluster assignments, we propagate both structural and feature information from the summary graph back to the original node space. The summary structure and features are projected to obtain node-level similarities, which are then combined to infer the presence of edges. As this reconstruction relies solely on differentially private summary representations, it incurs no additional privacy cost.

Our approach achieves differentially private graph synthesis by first compressing and perturbing high-level summaries, and then performing utility-preserving reconstruction. This design minimizes sensitivity and avoids direct noise injection into fine-grained structures, offering improved accuracy under tight privacy constraints.

B. Structure Summarization

Spectral Clustering. We propose a differentially private spectral clustering framework, DP-SCGNN, which jointly lever-
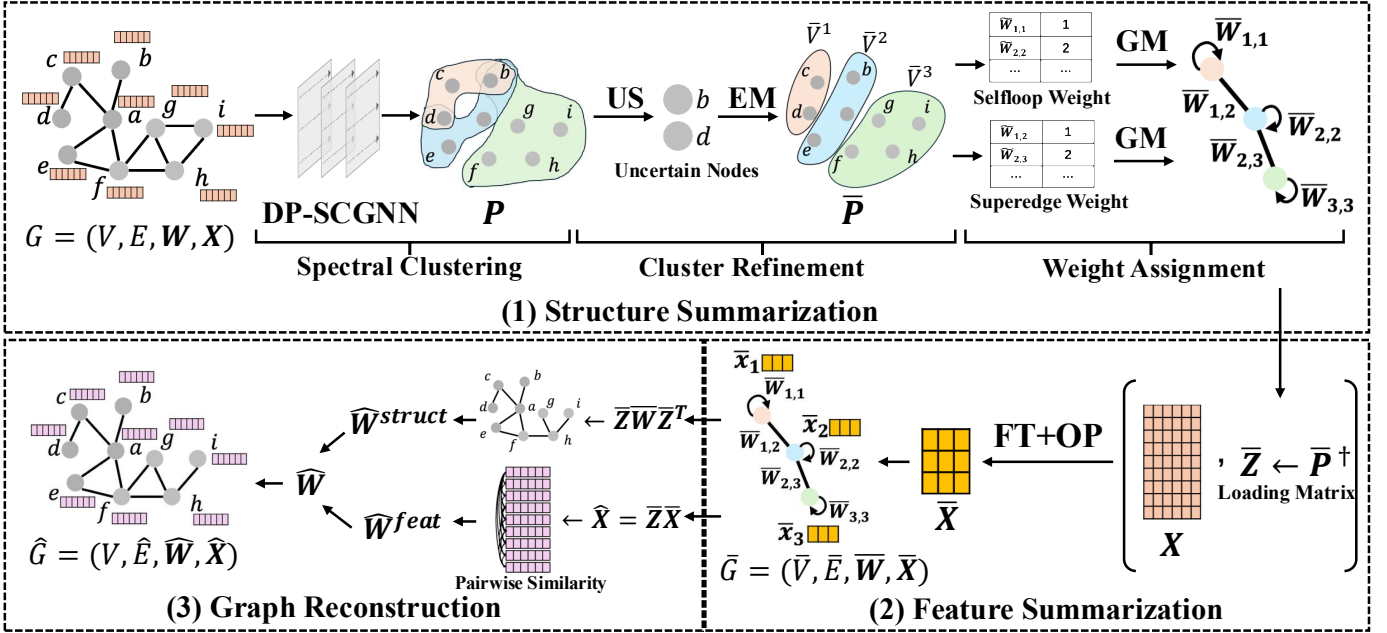
Fig. 3: Workflow of DPGS. The framework consists of three stages: (1) **Structure Summarization**, where the input graph $G = (V, E, W, X)$ is clustered using a differentially private spectral clustering GNN (DP-SCGNN), resulting in a soft assignment matrix $\boldsymbol{P}$. A subset of high-uncertainty nodes (US) is selected and refined using the exponential mechanism (EM), yielding a hard cluster assignment $\bar{\boldsymbol{P}}$. Gaussian noise (GM) is then added to the aggregated intra- and inter-cluster edge weights to obtain the summary graph $\bar{W}$; (2) **Feature Summarization**, where node features are transformed via feature transformation (FT) and objective perturbation (OP) into a compressed matrix $\bar{X}$ that aligns with the structure summary. The pseudo-inverse of $\bar{P}$ is used to compute the loading matrix $\bar{Z}$; (3) **Graph Reconstruction**, where a synthetic graph $\hat{G} = (\hat{V}, \hat{E}, \hat{W}, \hat{X})$ is reconstructed by propagating structural and semantic information back to the node level. Structural similarity $\hat{W}^{\text{struct}}$ and feature similarity $\hat{W}^{\text{feat}}$ are combined to infer edge existence.

ages graph topology and node features to generate clustering-aware representations. Motivated by the homophily property of real-world networks [29], our method encourages nodes with similar attributes and strong structural connectivity to be grouped together. Given a graph $G = (V, E, \boldsymbol{W}, \boldsymbol{X})$, to enable privacy-preserving representation for clustering, following the approach proposed in [42], we adopt a multi-scale GNN-based encoder with injected Gaussian noise:

$$\boldsymbol{H} = \texttt{COMBINE}\left(\left\{\texttt{MLP}^{(k)}(\texttt{Norm}(\boldsymbol{X}^{(k)} + \boldsymbol{\eta}^{(k)}))\right\}_{k=0}^{K}\right), \quad (2)$$

where $\boldsymbol{X}^{(k)} = \left(\boldsymbol{W}^{\top}\right)^{k}\boldsymbol{X}^{(0)}$ denotes the k-hop aggregated node features, and $\boldsymbol{\eta}^{(k)} \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$ is the Gaussian noise added to enforce differential privacy. The operator $\texttt{Norm}$ performs ℓ_2 normalization, and $\texttt{COMBINE}$ integrates representations across different multi-layer perceptron (MLP) layers.

The resulting embedding matrix $\boldsymbol{H}$ is then used to compute a cluster assignment of the nodes using a MLP with softmax on the output layer, which maps each node feature $\mathbf{x}_i \in \mathbb{R}^n$ into the ith column of a cluster assignment matrix $\boldsymbol{P} \in \mathbb{R}_{+}^{k \times p}$:

$$\boldsymbol{P} = \text{MLP}_{\text{cluster}}(\boldsymbol{H}; \boldsymbol{\Theta}_{\text{MLP}}), \quad (3)$$

where the softmax activation of the MLP guarantees that $\boldsymbol{P}_{i,j} \in [0, 1]$ and enforces the constraints $\boldsymbol{P}\mathbf{1}_k = \mathbf{1}_p$. Additionally, a temperature parameter can be introduced to the softmax function to adjust the sharpness of the assignment

distribution, thereby controlling the degree to which each column $\boldsymbol{P}_{:,i}$ approximates a one-hot vector—effectively tuning the level of fuzziness in the cluster assignments. The parameters $\boldsymbol{\Theta}_{\text{GNN}}$ and $\boldsymbol{\Theta}_{\text{MLP}}$ are jointly optimized by minimizing an unsupervised loss function $\mathcal{L}_u$ composed of two terms, which approximates the relaxed formulation of the MinCut problem [30]:

$$\mathcal{L}_u = \mathcal{L}_c + \mathcal{L}_o = \underbrace{-\frac{\text{Tr}\left(\boldsymbol{P}^T\tilde{\boldsymbol{W}}\boldsymbol{P}\right)}{\text{Tr}\left(\boldsymbol{P}^T\tilde{\boldsymbol{D}}\boldsymbol{P}\right)}}_{\mathcal{L}_c} + \underbrace{\left\|\frac{\boldsymbol{P}^T\boldsymbol{P}}{\|\boldsymbol{P}^T\boldsymbol{P}\|_F} - \frac{\boldsymbol{I}_k}{\sqrt{k}}\right\|_F}_{\mathcal{L}_o}$$

where $\tilde{\boldsymbol{W}} = \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{W}\boldsymbol{D}^{-\frac{1}{2}} \in \mathbb{R}^{p \times p}$, $\tilde{\boldsymbol{D}}$ denotes the normalized degree matrix $\boldsymbol{D}$, $\|\cdot\|_F$ indicates the Frobenius norm and k is the number of clusters. Minimizing $\mathcal{L}_c$ encourages strongly connected nodes to be clustered together. When $\tilde{\boldsymbol{W}}_{i,j}$ is large, the inner product $\langle\boldsymbol{P}_{:,i}, \boldsymbol{P}_{:,j}\rangle$ is encouraged to be large as well, thereby promoting similar cluster assignments for connected nodes. To penalize the degenerate minima of $\mathcal{L}_c$, the orthogonality loss term $\mathcal{L}_o$ encourages the cluster assignments to be orthogonal with clusters of similar size.

Remark. To enable differentially private training of DP-SCGNN—whose loss function depends on raw graph data—we adopt the DP-SGD mechanism [31]. At each training step, the Gaussian mechanism is applied to privatize the per-example gradients before updating model parameters. How-

ever, applying DP-SGD in the GNN context typically requires injecting substantial noise due to the interconnected nature of graph data. In the following sections, we present a formal privacy analysis and show that training the spectral clustering component satisfies differential privacy.

Cluster Refinement. To improve the reliability of cluster assignments under differential privacy, we design a refinement procedure that selectively corrects ambiguous assignments while preserving the graph's structural coherence. Instead of applying refinement to all nodes—which would lead to excessive privacy budget consumption—we focus on a small, highly uncertain subset of nodes and reassign them using the exponential mechanism. This design balances utility and privacy through targeted intervention.

As shown in Algorithm 1, we begin by measuring the uncertainty of each node's soft assignment using entropy (Lines 2-4). For node i, with assignment vector $\boldsymbol{P}_{:,i} \in \mathbb{R}^k$, we compute:

$$u_i = -\sum_{c=1}^{k} \boldsymbol{P}_{c,i} \log \boldsymbol{P}_{c,i}. \tag{4}$$

These uncertainty scores are normalized into a probability distribution $\pi_i = \frac{\exp(u_i)}{\sum_j \exp(u_j)}$, from which we sample a subset $\mathcal{R}$ of r nodes without replacement. This sampling not only concentrates effort on the most ambiguous nodes, but also benefits from privacy amplification by subsampling. For each sampled node $i \in \mathcal{R}$, we identify a small set of plausible candidate clusters by selecting the top-m entries in $\boldsymbol{P}_{:,i}$, denoted as $\mathcal{C}_i \subseteq \{1, \ldots, k\}$. To determine the most structurally appropriate cluster for node i, we define a utility score function over each candidate $c \in \mathcal{C}_i$, which counts how many of i's neighbors are confidently assigned to c:

$$s_i(c) \leftarrow \sum_{j:\boldsymbol{W}_{i,j}=1} \mathbb{I}\left[\arg\max_{c'} \boldsymbol{P}_{c',j} = c\right] \tag{5}$$

where $\{j : \boldsymbol{W}_{i,j} = 1\}$ denotes the 1-hop neighbors of node i in the adjacency matrix $\boldsymbol{W}$, and $\mathbb{I}[\cdot]$ is the indicator function (Lines 8-11). We then sample the final cluster using the exponential mechanism:

$$\Pr\left[c_i = c\right] \propto \exp\left(\frac{\epsilon \cdot s_i(c)}{2}\right), \quad \text{for } c \in \mathcal{C}_i, \tag{6}$$

where ε is the privacy budget and sensitivity $\Delta s = 1$ (Line 12). To finalize the assignment, we update $\boldsymbol{P}_{:,i}$ by retaining the original score of the selected cluster c_i and setting the remaining entries to zero (Line 13). For nodes not selected for refinement ($i \notin \mathcal{R}$), we similarly retain only the maximum entry. This enforces that the refined matrix $\bar{\boldsymbol{P}}$ is a orthogonal assignment matrix in which each column retains only one non-zero entry (Lines 15-17). This enables more interpretable downstream processing, while ensuring that the privacy budget is spent effectively.

Summary Weight Assignment. Based on the node clustering results $\bar{\boldsymbol{P}}$, our aim is to operate on a summary graph $\bar{G}$ constructed from clustered nodes. In this graph, edges are categorized into intra-supernode (within a cluster) or inter-supernode (across clusters) edges. Since intra-supernode edges

---

Algorithm 1: CLUSTERREFINEMENT

Input: Soft assignment matrix $\boldsymbol{P} \in \mathbb{R}^{k \times p}$, adjacency matrix $\boldsymbol{W} \in \{0,1\}^{p \times p}$, refinement size r, candidate size m, privacy budget ϵ
Output: Refined assignment matrix $\bar{\boldsymbol{P}} \in \mathbb{R}^{k \times p}$

1 **// Uncertainty-based sampling**
2 **for** $i = 1$ **to** p **do**
3     $u_i \leftarrow -\sum_{c=1}^{k} \boldsymbol{P}_{c,i} \log \boldsymbol{P}_{c,i}$
4 Normalize: $\pi_i \leftarrow \frac{\exp(u_i)}{\sum_{j=1}^{p} \exp(u_j)}$
5 Sample $\mathcal{R} \subset \{1, \ldots, p\}$ of size r from multinomial($\{\pi_i\}$) without replacement
6 **// Refine uncertain nodes**
7 Set $\bar{\boldsymbol{P}} \leftarrow \boldsymbol{0} \in \mathbb{R}^{k \times p}$
8 **for** *each* $i \in \mathcal{R}$ **do**
9     $\mathcal{C}_i \leftarrow$ indices of top-m values in $\boldsymbol{P}_{:,i}$
10   **for** *each* $c \in \mathcal{C}_i$ **do**
11        $s_i(c) \leftarrow \sum_{j:\boldsymbol{W}_{i,j}=1} \mathbb{I}\left[\arg\max_{c'} \boldsymbol{P}_{c',j} = c\right]$
12   Sample $c_i \in \mathcal{C}_i$ with probability $\propto \exp\left(\frac{\epsilon \cdot s_i(c)}{2}\right)$
13   Set $\bar{\boldsymbol{P}}_{c_i,i} \leftarrow \boldsymbol{P}_{c_i,i}$
14 **// Orthogonalize assignments**
15 **for** *each* $i \notin \mathcal{R}$ **do**
16     $c_i \leftarrow \arg\max_c \boldsymbol{P}_{c,i}$
17   Set $\bar{\boldsymbol{P}}_{c_i,i} \leftarrow \boldsymbol{P}_{c_i,i}$
18 **return** $\bar{\boldsymbol{P}}$

---

often dominate, treating these two types separately mitigates noise amplification. Instead of perturbing each edge, we add Gaussian noise to aggregated edge counts: self-loop weights for intra-supernode connections and superedge weights for inter-supernode links. This aggregation reduces sensitivity and yields a compact, noise-resilient representation of the original graph structure.

As shown in Algorithm 2, we begin by grouping nodes into supernodes based on the cluster assignment matrix $\bar{\boldsymbol{P}} \in \mathbb{R}_+^{k \times p}$. Each node is assigned to the cluster u with the highest membership score:

$$\bar{V}^u = \left\{ v_i \mid \arg\max_{c'} \bar{\boldsymbol{P}}_{c',i} = u \right\}, \quad \forall u \in \{1, \ldots, k\}.$$

This results in a partition of the node set into k disjoint clusters, which serve as the supernodes of the summary graph (Lines 1-2).

To define the connectivity between supernodes, we compute the normalized edge count between each pair (u, v) based on the original adjacency matrix $\boldsymbol{W} \in \{0,1\}^{p \times p}$. To ensure differential privacy, we perturb the raw edge counts using Gaussian noise. For $u \neq v$, the edge weight is defined as:

$$\bar{\boldsymbol{W}}_{u,v} = \frac{\sum_{v_i \in \bar{V}^u, v_j \in \bar{V}^v} \boldsymbol{W}_{i,j} + \mathcal{N}(0, \sigma^2)}{|\bar{V}^u| \cdot |\bar{V}^v|}.$$

For self-loops (intra-cluster connectivity), the weight is:

$$\bar{\boldsymbol{W}}_{u,u} = \frac{2 \sum_{i<j, v_i,v_j \in \bar{V}^u} \boldsymbol{W}_{i,j} + \mathcal{N}(0, 4\sigma^2)}{|\bar{V}^u|(|\bar{V}^u| - 1)}.$$

Note that the sensitivity of counting edges is 1, since adding or removing a single edge affects at most one pairwise count (Lines 3-8).

To preserve symmetry, we mirror each entry across the diagonal: $\bar{\boldsymbol{W}}_{v,u} \leftarrow \bar{\boldsymbol{W}}_{u,v}$ (Line 9). Finally, to ensure validity under differential privacy, we clip any negative weights to zero

(Line 10):
$$\bar{W}_{u,v} \leftarrow \max(0, \bar{W}_{u,v}).$$

The resulting matrix $\bar{W} \in \mathbb{R}^{k \times k}$ defines a connected, undirected, and non-negative weighted summary graph that can serve as the structural backbone for downstream graph reconstruction or representation tasks.

Algorithm 2: SUMMARYWEIGHTASSIGNMENT

Input: Adjacency matrix $W \in \{0,1\}^{p \times p}$, cluster assignment matrix $\bar{P} \in \mathbb{R}_+^{k \times p}$, Gaussian noise scale σ
Output: Noisy summary graph weight matrix $\bar{W} \in \mathbb{R}^{k \times k}$

1 **for** $u = 1$ to k **do**
2    $\bar{V}^u \leftarrow \{v_i \mid \arg\max_{c'} \bar{P}_{c',i} = u\}$
3 **for** $u = 1$ to k **do**
4   **for** $v = u$ to k **do**
5   **if** $u \neq v$ **then**
6        $\bar{W}_{u,v} \leftarrow \frac{\sum_{v_i \in \bar{V}^u, v_j \in \bar{V}^v} W_{i,j} + \mathcal{N}(0,\sigma^2)}{|\bar{V}^u| \cdot |\bar{V}^v|}$
7   **else**
8        $\bar{W}_{u,u} \leftarrow \frac{2\sum_{i<j, v_i,v_j \in \bar{V}^u} W_{i,j} + \mathcal{N}(0,4\sigma^2)}{|\bar{V}^u|(|\bar{V}^u|-1)}$
9      $\bar{W}_{v,u} \leftarrow \bar{W}_{u,v}$;   // Symmetrize
10      $\bar{W}_{u,v} \leftarrow \max(0, \bar{W}_{u,v})$;   // Normalize
11 **return** $\bar{W}$

C. Feature Summarization

Our objective then is to derive a suitable transformation F such that $\tilde{X} = F(X)$, i.e., it generates the summary graph's feature matrix from the original graph's features. It is essential that $\tilde{X}$ remains similar to X with only bounded information loss. We adopt the method of Kumar et al. (2023) [32] to learn a feature transformation while bounding the information loss. This is done by minimizing the objective function:

$$\min_{\tilde{X}} f(\tilde{X}) = \text{tr}\left(\tilde{X}^T Z^T L Z \tilde{X}\right) + \frac{\alpha}{2}\|Z\tilde{X} - X\|_F^2. \quad (7)$$

where $Z \in \mathbb{R}^{p \times k}$ is the pseudo inverse of the assignment matrix P, known as the loading matrix. Here, the first term of the objective function imposes the smoothness property on the summary graph [32], and the second term acts as a regularizer. This regularization summarizes the feature matrix of a larger graph $X \in \mathbb{R}^{p \times n}$ into a smaller one $\tilde{X} \in \mathbb{R}^{k \times n}$.

Objective Perturbation. According to the objective function in Eq. (7), directly solving it would involve dependencies on the raw data, such as the loading matrix Z, the Laplacian matrix L, and the feature matrix X, all of which are derived from the original graph. It is thus necessary to consider how to incorporate DP mechanism to learn differentially private feature matrix $\bar{X}$. To avoid potential structural damage caused by input perturbation and noise accumulation resulting from gradient perturbation, we perturb the objective funcion as

$$\min_{\bar{X}} f(\bar{X}) = \text{tr}(\bar{X}^T \bar{L}_s \bar{X}) + \frac{\alpha}{2}\|\bar{Z}\bar{X} - X\|_F^2 + \text{tr}(B^T \bar{X}). \quad (8)$$

Here, $\bar{L}_s$ and $\bar{Z}$ are the Laplacian matrix and loading matrix derived from the summarized graph structure $\bar{W}$ and the private assignment matrix $\bar{P}$, respectively. Since the structure summarization process already satisfies DP, these two matrices

are used to replace the structure-related data in Eq. (7). To ensure the privacy of the feature matrix, $\text{tr}(B^T \bar{X})$ is introduced. The third term of the private objective function adds noise into $\bar{X}$, where $B \sim \mathcal{N}(0, \Delta_B^2 \sigma^2 \mathbb{I})$ is the noise matrix. After feature normalization, we have $\Delta_B \leq \max_i \|X_i\|^2 \leq 2k$.

Optimization. As $\bar{L}_s$ and $\bar{Z}^T \bar{Z}$ are the positive semi-definite and definite matrices, the Hessian of $\bar{X}$, i.e., $\nabla^2 f(\bar{X}) = 2\bar{L}_s + \alpha\bar{Z}^T \bar{Z}$ is a positive definite matrix. This implies that Eq. (8) is a strongly convex optimization problem. For this, we can easily get the closed-form solution by setting the gradient to zero, i.e., $2\bar{L}_s\bar{X} + \alpha\bar{Z}^T(\bar{Z}\bar{X} - X) + \frac{1}{\alpha}B = 0$. Then we have

$$\bar{X} = \left(\frac{2}{\alpha}\bar{L}_s + \bar{Z}^T\bar{Z}\right)^{-1}\left(\bar{Z}^T X - \frac{1}{\alpha}B\right). \quad (9)$$

This optimization allows us to inject noise only once while directly solving for the optimal solution, effectively avoiding noise accumulation and preventing direct perturbation to the graph structure. Then, by integrating the summarized structure with the corresponding summarized features, the summary graph $\bar{G} = (\tilde{V}, \bar{E}, \bar{W}, \bar{X})$ is obtained. The whole graph summarization procedure is presented in Algorithm 3.

Algorithm 3: GRAPHSUMMARIZATION

Input: Adjacency matrix $W \in \{0,1\}^{p \times p}$, feature matrix $X \in \mathbb{R}^{p \times n}$, number of clusters k, noise scale σ, privacy budget ε
Output: Private summary graph $\bar{G} = (\tilde{V}, \bar{E}, \bar{W}, \bar{X})$

1 **// Structure Summarization.**
   $P \leftarrow$ DP-SCGNN(W, X, k, ε_1) ;   // GNN-based soft clustering
2 $\bar{P} \leftarrow$ CLUSTERREFINEMENT$(P, W, r, m, \varepsilon_2)$;
   // Entropy sampling + Exponential Mechanism
3 $\bar{W} \leftarrow$ SUMMARYWEIGHTASSIGNMENT$(W, \bar{P}, \sigma)$;
   // Noisy edge aggregation
4 **// Feature Summarization.**
5 Normalize feature matrix X column-wise: $X_{:,j} \leftarrow X_{:,j}/\|X_{:,j}\|_2, \quad \forall j \in \{1, \ldots, n\}$
6 Compute Laplacian matrix: $\bar{L}_s \leftarrow$ LAPLACIAN$(\bar{W})$;
7 Compute loading matrix: $\bar{Z} \leftarrow \bar{P}^\dagger$;
8 Generate noise: $B \sim \mathcal{N}(0, \Delta_B^2 \sigma^2 \mathbb{I})$;
9

$$\bar{X} = \left(\frac{2}{\alpha}\bar{L}_s + \bar{Z}^T\bar{Z}\right)^{-1}\left(\bar{Z}^T X - \frac{1}{\alpha}B\right)$$

// Return Summary Graph. Set $\tilde{V} = \{1, \ldots, k\}$, $\bar{E} = \{(u,v) \mid \bar{W}_{u,v} > 0\}$;
10 **return** $\bar{G} = (\tilde{V}, \bar{E}, \bar{W}, \bar{X})$;

D. Graph Reconstruction

Given the noisy summary weight matrix $\bar{W} \in \mathbb{R}^{k \times k}$, the compressed node features $\bar{X} \in \mathbb{R}^{k \times d}$, and the loading matrix $\bar{Z} \in \mathbb{R}^{p \times k}$, we reconstruct the original graph structure without requiring any additional information. The key idea is to propagate structural and semantic signals from the summary graph back to the original node space, thereby recovering the adjacency matrix in a privacy-preserving manner, as shown in Algorithm 4.

Structure-based similarity. We first reconstruct pairwise structural similarity among original nodes by projecting the noisy supernode weights back to the fine-grained space:

$$\hat{\boldsymbol{W}}^{\text{struct}} = \bar{\boldsymbol{Z}} \bar{\boldsymbol{W}} \boldsymbol{Z}^\top.$$

This step propagates the differentially private structure information to the original graph level.

Feature-based similarity. We then reconstruct the node features as:

$$\hat{\boldsymbol{X}} = \bar{\boldsymbol{Z}} \bar{\boldsymbol{X}},$$

and compute pairwise feature similarity via cosine similarity:

$$\hat{\boldsymbol{W}}^{\text{feat}}_{i,j} = \frac{\hat{\boldsymbol{X}}_i^\top \hat{\boldsymbol{X}}_j}{\|\hat{\boldsymbol{X}}_i\| \cdot \|\hat{\boldsymbol{X}}_j\| + \xi},$$

where $\xi > 0$ is a small constant to ensure numerical stability. This captures semantic relationships between nodes that may not be reflected in the structural signal.

Fusion of scores. We combine both sources of similarity into a unified score matrix:

$$\hat{\boldsymbol{W}}^{\text{score}} = \beta \cdot \hat{\boldsymbol{W}}^{\text{struct}} + (1-\beta) \cdot \hat{\boldsymbol{W}}^{\text{feat}},$$

where $\beta \in [0,1]$ controls the trade-off between structure and features.

Then, we threshold the fused scores to obtain the reconstructed binary adjacency matrix:

$$\hat{\boldsymbol{W}}_{i,j} = \begin{cases} 1 & \text{if } \hat{\boldsymbol{W}}^{\text{score}}_{i,j} \geq \tau, \\ 0 & \text{otherwise,} \end{cases}$$

where τ is a tunable threshold hyperparameter. We define the reconstructed node set as $V = \{v_1, \ldots, v_p\}$, where p is the number of rows in $\boldsymbol{Z}$. The predicted edge set is given by:

$$\hat{E} = \{(v_i, v_j) \mid \hat{\boldsymbol{W}}_{i,j} = 1\}.$$

This reconstruction procedure leverages only the available summary outputs, ensuring that no additional privacy budget is consumed. The integration of structure and feature signals enables accurate and utility-preserving recovery of the original graph topology.

V. PRIVACY ANALYSIS

Recall Figure 3, DPGS consists of two main stages: graph summarization and graph reconstruction. Among them, only the summarization stage involves direct access to the sensitive input data. Therefore, to analyze the privacy guarantee of DPGS, it suffices to analyze the privacy properties of the graph summarization algorithm. This module consists of three differentially private components: Structure Summarization with privacy budget ε_1, Weight Assignment with budget ε_2, and Feature Summarization with budget ε_3. Based on the composition of differential privacy, we can establish the following theorem:

Theorem 1 (Privacy Guarantee of DPGS). *Given a graph $G = (V, E, \boldsymbol{W}, \boldsymbol{X})$, where E is the edge set and $\boldsymbol{X}$ is the feature matrix, DPGS satisfies differential privacy if and only if its* GRAPHSUMMARIZATION *procedure satisfies differential privacy, since the* GRAPHRECONSTRUCTION *stage is a post-processing step without further data access. Specifically:*

- *For the graph structure E, the Graph Summarization algorithm, composed of structural clustering, weight assignment,*

Algorithm 4: GRAPHRECONSTRUCTION

Input: Summary edge matrix $\bar{\boldsymbol{W}} \in \mathbb{R}^{k \times k}$, summary feature matrix $\bar{\boldsymbol{X}} \in \mathbb{R}^{k \times d}$, loading matrix $\boldsymbol{Z} \in \mathbb{R}^{p \times k}$, fusion weight $\beta \in [0,1]$, threshold τ

Output: Reconstructed graph $\hat{G}$

1 // **Structure-based similarity**
2 $\hat{\boldsymbol{W}}^{\text{struct}} \leftarrow \boldsymbol{Z} \cdot \bar{\boldsymbol{W}} \cdot \boldsymbol{Z}^\top$
3 // **Feature-based similarity**
4 $\hat{\boldsymbol{X}} \leftarrow \boldsymbol{Z} \cdot \bar{\boldsymbol{X}}$
5 **for** $i = 1$ **to** p **do**
6   **for** $j = 1$ **to** p **do**
7       $\hat{\boldsymbol{W}}^{\text{feat}}_{i,j} \leftarrow \frac{\hat{\boldsymbol{X}}_i^\top \hat{\boldsymbol{X}}_j}{\|\hat{\boldsymbol{X}}_i\| \cdot \|\hat{\boldsymbol{X}}_j\| + \xi}$;   // Cosine similarity
8 // **Fusion**
9 $\hat{\boldsymbol{W}}^{\text{score}} \leftarrow \beta \cdot \hat{\boldsymbol{W}}^{\text{struct}} + (1-\beta) \cdot \hat{\boldsymbol{W}}^{\text{feat}}$
10 // **Thresholding (binary reconstruction)**
11 $\hat{E} \leftarrow \emptyset$
12 **for** $i = 1$ **to** p **do**
13   **for** $j = 1$ **to** p **do**
14   **if** $\hat{\boldsymbol{W}}^{score}_{i,j} \geq \tau$ **then**
15          $\hat{\boldsymbol{W}}_{i,j} \leftarrow 1$
16   Add (v_i, v_j) to $\hat{E}$;   // Add predicted edge
17   **else**
18          $\hat{\boldsymbol{W}}_{i,j} \leftarrow 0$
19 $V = \{v_i \mid \exists u \in \{1, \ldots, k\}, \bar{\boldsymbol{P}}_{u,i} \neq 0\}$.
20 **return** $\hat{G} = (V, \hat{E}, \hat{\boldsymbol{W}}, \hat{\boldsymbol{X}})$

---

and feature summarization, satisfies $(\varepsilon_1 + \varepsilon_2 + \varepsilon_3, \delta)$-Edge-DP.

- *For the node features $\boldsymbol{X}$ as the sensitive dataset, the feature summarization step alone satisfies (ε_3, δ)-DP under row-level changes.*

Therefore, the Graph Summarization module of DPGS simultaneously protects edge-level structure and node-level features, and the overall mechanism preserves differential privacy via composition.

Proof. Please refer to the appendix. $\square$

We next present the privacy accounting method for training DP-SCGNN. A common challenge when training GNNs under differential privacy lies in the dependencies among subgraphs in the training set. These dependencies amplify the sensitivity of the aggregation process, often leading to larger noise requirements and complicating privacy accounting. To address this issue, we adopt the accounting method in [34], which reduces inter-node dependency and improves the overall privacy-utility trade-off. This is formally proven using Rényi Differential Privacy (Rényi-DP) [35] as follows.

Theorem 2 (Privacy Guarantee for any r-Layer GNN). *Let N be the number of training instances V_{tr}, N_k be the maximum degree of the input graph, r be the number of GNN layers, and m be the batch size during training. For any choice of the noise standard deviation $\sigma > 0$ and clipping threshold η, every iteration t of DP-SGD is (α_ρ, γ) node-level Rényi DP,*

where:

$$\gamma = \frac{1}{\alpha_\rho - 1} \ln \mathbb{E}_\rho \left[\exp \left(\alpha_\rho(\alpha_\rho - 1) \cdot \frac{2\rho^2\eta^2}{\sigma^2} \right) \right],$$

$$\rho \sim \text{Hypergeometric} \left(N, \frac{N_k^{r+1} - 1}{N_k - 1}, m \right).$$

Hypergeometric *denotes the standard hypergeometric distribution [36]. By the standard composition theorem for Rényi Differential Privacy [35], over T iterations, spectral clustering is $(\alpha_\rho, \gamma T)$-Edge-Rényi DP.*

Proof. Please refer to the appendix. $\square$

VI. EXPERIMENTAL EVALUATION

In this section, we first introduce our experiment setup in Section VI-A, and then present the experimental results and our analysis in Section VI-B.

A. Experiment Setup

1) Datasets: Table I characterizes the 6 real-world datasets used to benchmark DPGS.

TABLE I: Summary statistics of used datasets.

Name	#Nodes	#Edges	#Features	#Classes
Cora	2708	5278	1433	7
CiteSeer	3327	9104	3703	6
PubMed	19717	44324	500	3
Coauthor	18333	163788	6805	15
DBLP	17716	105734	1639	4

2) Experimental Design: We evaluate the performance of our approach against a range of representative baselines across three categories. For **graph summarization baselines**: we include four state-of-the-art methods: Local Variation Neighbourhood (LVN), proposed in [37]; SSuMM, massive graph summarization method proposed in [38]; Kron, which applies Schur complement-based reduction of the Laplacian matrix [39]; and Algebraic Distance (AD), which leverages connection strength between nodes [40]. For **edge-DP GNN baselines**, we compare with DP-GNN [34], DPGCN [41], GAP [42], LPGNet [43], and Eclipse [44]. For **differentially private graph synthesis baselines**: For the private synthesis methods that do not incorporate node features, we consider five baselines: PrivHRG [15], DER [20], TmF [19], LDPGen [22], and PrivGraph [23]. To ensure a fair comparison, we adopt hyperparameter configurations recommended in the original studies. Since LDPGen was originally designed for the local differential privacy setting, we adapt its implementation to the standard DP model for consistency across all methods.

3) Evaluation Indicators: **Synthesis Quality.** To comprehensively evaluate the quality of the synthesized graphs, we adopt two representative metrics that capture different aspects of structural fidelity. Specifically, we use *relative error (RE)* of modularity and clustering coefficient to assess the preservation of **global topological structure**, and *eigenvector centrality (EVC)*-based metrics to evaluate the retention of **node-level importance**.

The RE of modularity and clustering coefficient are computed as:

$$\text{RE}_{\text{Mod}} = \frac{|\hat{Q} - Q|}{\max(\xi, Q)}, \quad \text{RE}_{\text{CC}} = \frac{|\hat{Y} - Y|}{\max(\xi, Y)},$$

where Q and $\hat{Q}$ denote the modularity of the original and synthetic graphs, Y and $\hat{Y}$ denote their average clustering coefficients, and ξ is a small constant to prevent division by zero.

To evaluate node importance preservation, we rank all nodes by EVC and compute two comparison metrics: (1) the proportion of overlapping nodes among the top 1% most influential nodes, and (2) the mean absolute error (MAE) of their centrality scores:

$$\text{Overlap}_{\text{Node}} = \frac{|N \cap \hat{N}|}{|N|}, \quad \text{MAE}_{\text{EVC}} = \frac{1}{T} \sum_{i=1}^{T} |\hat{s}_i - s_i|,$$

where N and $\hat{N}$ are the sets of top 1% nodes ranked by EVC in the original and synthetic graphs, and s_i, $\hat{s}_i$ are their corresponding EVC scores. This combination of metrics allows us to jointly assess structural and semantic fidelity of the generated graphs. Due to space constraints, additional evaluation metrics and results, please refer to the appendix.

Downstream Task Performance. To evaluate the utility of the synthesized graphs beyond structural similarity, we assess their effectiveness on two representative downstream graph learning tasks: node classification and link prediction. These tasks reflect how well the synthetic graph supports semantic inference and connectivity modeling, and are widely used to gauge the practical value of graph representations. Strong performance in these settings indicates that the synthetic graphs preserve not only topological patterns but also task-relevant signals essential for learning meaningful representations. We use accuracy to evaluate node classification and the Area Under the ROC Curve (AUC) to assess link prediction. Accuracy quantifies the proportion of correctly predicted node labels, while AUC measures the model's ability to distinguish between positive and negative links.

B. Experimental Results and Analysis

1) Synthesis Quality: **Topological Structure.** Figure. 6 reports the relative error of modularity and clustering coefficients on Cora,CiteSeer, PubMed and Coauthor datasets, capturing how well each method preserves structural properties under varying privacy budgets ε. Overall, DPGS achieves the lowest RE across both datasets and metrics, indicating superior preservation of topological features.

On CiteSeer, DPGS reduces RE(Modularity) from over 0.8 (typical for TmF, DER, and LDPGen) to below 0.5 at $\varepsilon = 1.0$, with further improvements as ε increases. Similar trends are observed in clustering coefficients, where DPGS consistently outperforms all baselines across privacy levels. In contrast, methods such as TmF and PrivHRG show minimal sensitivity to ε, likely due to their direct matrix perturbation or rigid modeling assumptions.

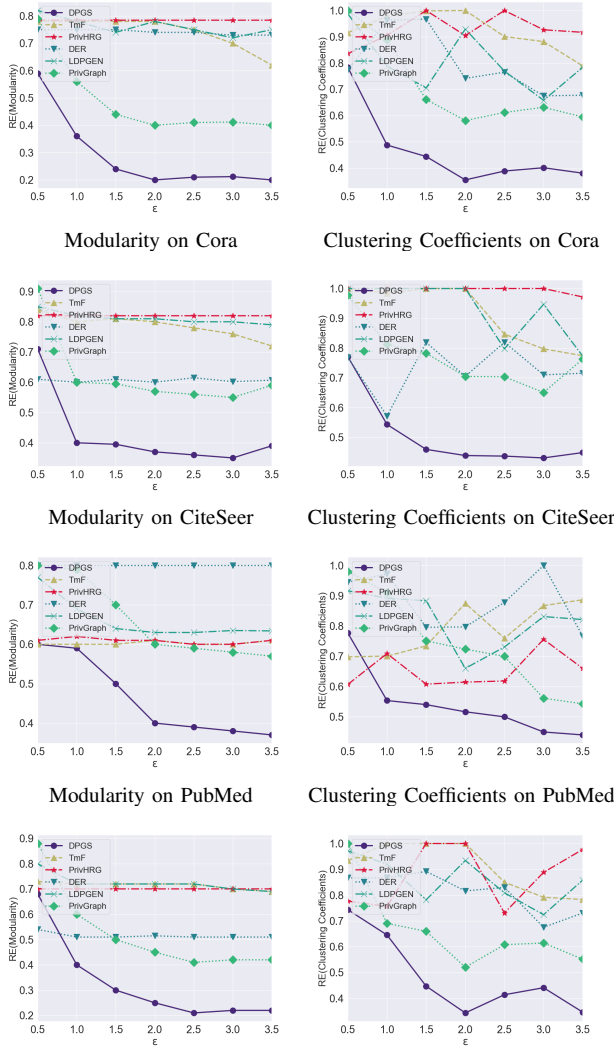PrivGraph also underperforms compared to DPGS, particularly on CiteSeer. Although it benefits from community-

Fig. 4: Overall results on Modularity and Clustering Coefficients over datasets Cora, CiteSeer, PubMed and Coauthor

Fig. 5: Overall results on Overlap and MAE(EVC) over datasets Cora, CiteSeer, PubMed and Coauthor

level modeling, its hard clustering strategy in the community adjustment phase forces a full re-clustering of all nodes at each iteration. This introduces cumulative noises and optimization instability, often leading to suboptimal convergence and inconsistent structural recovery. These limitations are reflected in its relatively high RE in both modularity and clustering coefficient metrics, especially under small ε.

On PubMed—a larger and sparser dataset—DPGS maintains RE(Modularity) below 0.4 for $\varepsilon \geq 2$, while all other methods, including PrivGraph, exceed 0.6. Its RE on clustering coefficients is also the lowest and most stable, demonstrating robustness to both graph size and sparsity.

Node-level Importance. Figure 5 presents the comparison results of node-level importance preservation using two complementary metrics: the overlap of eigenvector centrality nodes between the original and synthetic graphs, and the MAE of their EVC scores. Across four datasets, DPGS consistently outperforms all baselines once the privacy budget exceeds $\varepsilon =$
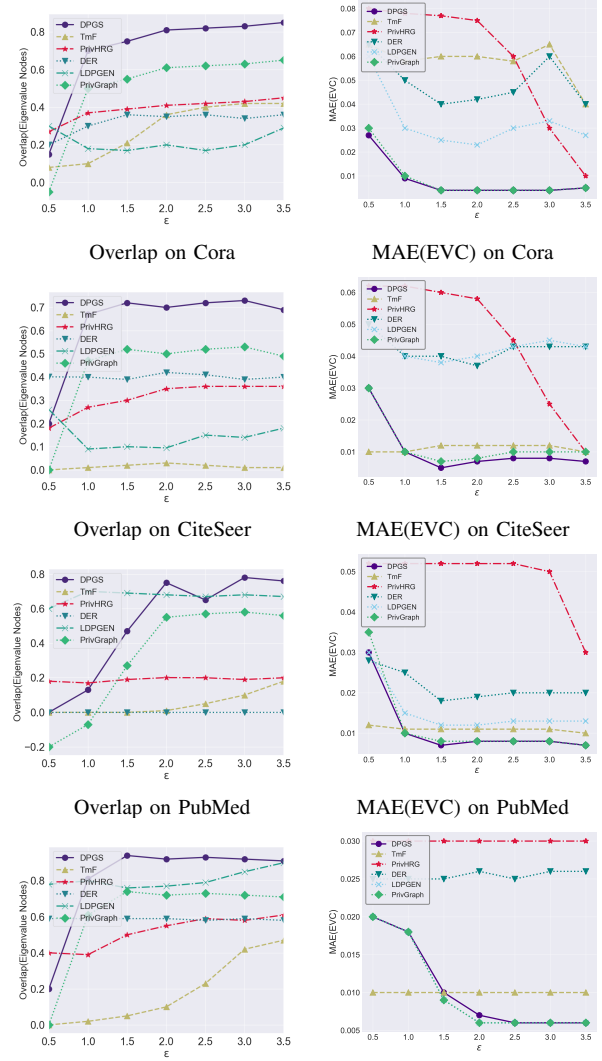
1.5. This improvement can be attributed to its summarization-based design: by clustering structurally similar nodes and performing edge reconstruction at the community level, DPGS mitigates the effect of perturbation and better maintains global spectral properties. As a result, it yields significantly higher node overlap (e.g., > 0.7 on both CiteSeer and PubMed for $\varepsilon \geq 2$) and lower MAE compared to TmF, PrivHRG, and DER.

At low privacy budgets ($\varepsilon = 0.5$), the overlap scores of DPGS remain modest, which is expected since noisy clustering leads to degraded reconstruction fidelity. However, the performance rapidly improves with increasing ε, while other baselines remain flat or improve marginally. This trend suggests that DPGS is better able to capitalize on larger budgets to enhance structural recovery.

In contrast, PrivGraph and LDPGen, while competitive on certain datasets, exhibit limited adaptability. For example, PrivGraph's overlap plateaus on PubMed, likely due to its

reliance on fixed statistics and hard clustering, which restrict flexibility under noise. LDPGen performs relatively well on PubMed but struggles on CiteSeer, possibly because the EVC score differences between top nodes are less pronounced, making ranking sensitive to noise.

TmF and PrivHRG, which directly perturb edge presence or rely on coarse probabilistic hierarchies, consistently underperform due to accumulated noise and limited expressiveness. DER, although data-adaptive, shows unstable results across datasets and generally falls short of DPGS's performance.

2) Downstream Task Performance: **Node Classification.** To evaluate the utility of DPGS under different privacy constraints, we conduct node classification experiments across four benchmark datasets with privacy budgets $\varepsilon \in 0.1, 10$ under the Edge-DP setting. As reported in Table II, DPGS consistently achieves the highest accuracy across all datasets and privacy levels. Although the performance margin over strong baselines (e.g., Eclipse and LPGNet) is narrower compared to earlier settings, DPGS remains the most robust method, particularly under low ε values, where most other methods suffer significant performance degradation.

For instance, on the Cora and CiteSeer datasets with $\varepsilon = 0.1$, DPGS attains 70.70% and 66.14% accuracy respectively, while the second-best methods (Eclipse) reach only 65.11% and 63.07%. This demonstrates DPGS's effectiveness in preserving task-relevant information even under strict privacy constraints. On larger graphs like PubMed and Coauthor, DPGS maintains its superiority, achieving over 82% and 91% accuracy under $\varepsilon = 0.1$, respectively—substantially outperforming all alternatives.

The effectiveness of DPGS can be attributed to its two-stage architecture: it first summarizes the input graph into low-sensitivity structural and semantic representations, and then applies privacy-preserving mechanisms at this intermediate level. By avoiding direct noise injection into the input graph or training gradients, DPGS significantly reduces utility loss. In contrast, baselines such as DP-GNN and DPGCN suffer from accumulated noise across multiple stages of training or inference, leading to noticeable degradation in predictive performance.

Link Prediction. Since most private learning methods are primarily designed for node classification, we evaluate the effectiveness of DPGS on the link prediction task, comparing it with several summarization-based baselines, including SSumM, LVN, Kron, and AD. Following established practice, we vary the summarization ratio from 5% to 30% for Cora and CiteSeer, and from 1% to 5% for the larger PubMed dataset to assess scalability under compression.

As shown in Table III, DPGS consistently achieves the highest AUC across all datasets and summarization levels. On Cora, for instance, DPGS reaches an AUC of 82.42% at a 30% summary size and maintains a strong lead even when compressed to 5%. Similarly, on CiteSeer and PubMed, DPGS outperforms all baselines by a notable margin, especially under extreme compression (e.g., 1% summary on PubMed), where it still achieves 72.17% AUC compared to AD's 68.15%.

These improvements are primarily due to the DPGS framework, which jointly summarizes both structural patterns and node features into a compact representation with reduced sensitivity. Privacy-preserving noise is applied only once during the summarization phase, and no additional noise is introduced during reconstruction. In contrast, competing methods either disregard node attributes (e.g., LVN, Kron) or apply heuristics that lead to cumulative distortions. DPGS effectively balances utility and privacy, especially under aggressive compression, making it more robust and scalable in practice.

3) Robustness against MIA Attacks: We evaluate the resilience of DPGS to membership inference attacks (MIA) under the setting of [45], where the adversary aims to determine whether a given node was part of the target model's training set. Such attacks typically exploit overfitting by using confidence scores from a shadow model trained on a similar distribution. As shown in Table IV, all methods yield AUC values close to 50%, indicating that the attack model performs no better than random guessing. Notably, DPGS maintains consistently low AUC values across all datasets and privacy levels, with minimal variation as ε increases. This demonstrates that DPGS effectively limits information leakage, offering strong membership privacy even under relaxed privacy budgets.

VII. Related Work

A. Differentially Private Graph Neural Networks

Integrating differential privacy (DP) into Graph Neural Networks (GNNs) has gained traction due to the sensitive nature of graph data. Various methods aim to enforce edge- or node-level DP during training. GAP [42] perturbs the neighborhood aggregation step, supporting both edge and node DP with calibrated noise to preserve utility. DPDGC [46] decouples graph convolution into structural and feature components, enabling fine-grained gradient sensitivity control under DP-SGD.

ProGAP [47] introduces progressive noise injection to stabilize training and improve the privacy-utility trade-off. LPGNet [43] modifies message-passing rules to protect edge information without excessive noise. Eclipse [44] applies low-rank SVD to the adjacency matrix before perturbation, reducing sensitivity while retaining structure. Works such as LinkTeller [41] demonstrates that many graph model defenses remain vulnerable to inference attacks, underscoring the need for stronger protections and refined threat models.

B. Privacy Attacks and Threat Models on Graphs

Graph data is highly vulnerable to privacy attacks due to its rich relational structure. Adversaries may infer hidden edges via structure-based attacks, which exploit topological patterns [25], [48], or feature-based attacks, which rely solely on node attributes [49], [50]. De-anonymization attacks [51], [52] further attempt to re-identify nodes using auxiliary data and matching algorithms. DPGS mitigates such risks by disrupting correlations between observed signals and sensitive edges, reducing susceptibility to both structure- and feature-based attacks. Nevertheless, attacks outside the DP framework

TABLE II: Node classification performance across datasets under different ε values.

Dataset	ε	DPGCN	Eclipse	LPGNet	DP-GNN	GAP	DPGS
Cora	0.1	36.95 ± 0.42	65.11 ± 0.38	48.14 ± 0.21	27.36 ± 1.08	34.66 ± 0.25	**70.70 ± 0.20**
	10.0	72.34 ± 1.00	66.38 ± 0.61	70.70 ± 0.91	39.59 ± 0.71	57.46 ± 0.20	**72.78 ± 0.38**
CiteSeer	0.1	35.23 ± 1.06	63.07 ± 0.07	49.38 ± 0.87	28.00 ± 0.29	32.87 ± 0.47	**66.14 ± 0.15**
	10.0	62.28 ± 0.68	62.88 ± 0.48	63.61 ± 0.52	31.23 ± 0.43	56.43 ± 0.41	**71.80 ± 0.35**
PubMed	0.1	54.65 ± 0.62	71.72 ± 0.24	66.82 ± 0.88	62.24 ± 0.64	59.99 ± 0.30	**82.17 ± 0.25**
	10.0	62.16 ± 0.97	71.56 ± 0.82	75.04 ± 0.46	65.73 ± 1.13	72.08 ± 0.16	**83.73 ± 0.28**
Coauthor	0.1	63.26 ± 0.30	88.01 ± 0.14	67.46 ± 0.64	52.90 ± 0.77	80.21 ± 0.63	**91.59 ± 0.12**
	10.0	88.01 ± 0.13	89.07 ± 0.12	90.71 ± 0.10	59.05 ± 0.78	90.52 ± 0.20	**93.11 ± 0.06**

TABLE III: Link prediction performance across datasets under different summary size.

Dataset	Size	Ground Truth	SSumM	LVN	Kron	AD	DPGS
Cora	30%	—	68.35 ± 0.15	70.20 ± 0.58	77.10 ± 0.24	78.02 ± 0.30	**82.42 ± 0.04**
	10%	84.31 ± 0.75	66.08 ± 0.56	67.82 ± 0.22	75.13 ± 0.13	77.02 ± 0.36	**80.21 ± 0.13**
	5%	—	63.32 ± 1.05	63.00 ± 0.95	73.28 ± 0.24	75.05 ± 0.13	**77.89 ± 0.03**
CiteSeer	30%	—	72.05 ± 0.47	71.45 ± 0.25	73.01 ± 0.14	75.13 ± 0.17	**76.86 ± 0.03**
	10%	78.62 ± 0.62	69.71 ± 0.38	69.53 ± 0.58	69.83 ± 0.36	73.88 ± 0.30	**75.66 ± 0.03**
	5%	—	62.89 ± 0.82	63.93 ± 0.64	68.13 ± 0.39	72.00 ± 0.45	**72.29 ± 0.04**
PubMed	5%	—	61.30 ± 0.83	62.10 ± 0.90	67.05 ± 0.40	77.12 ± 0.33	**81.19 ± 0.04**
	3%	83.28 ± 0.57	60.66 ± 0.58	62.13 ± 0.57	66.28 ± 0.22	72.30 ± 0.23	**75.85 ± 0.06**
	1%	—	57.35 ± 0.58	61.60 ± 0.23	66.22 ± 0.38	68.15 ± 0.45	**72.17 ± 0.02**

TABLE IV: MIA performance across datasets.

Dataset	Method	$\varepsilon = 1$	$\varepsilon = 2$	$\varepsilon = 4$	$\varepsilon = 8$
Cora	GAP	50.12 ± 0.03	50.23 ± 0.04	50.42 ± 0.02	51.10 ± 0.04
	DP-GNN	50.24 ± 0.04	50.25 ± 0.02	50.20 ± 0.01	50.22 ± 0.03
	DPGS	50.18 ± 0.02	50.22 ± 0.02	50.39 ± 0.03	50.61 ± 0.04
CiteSeer	GAP	50.02 ± 0.09	50.18 ± 0.03	50.30 ± 0.05	51.58 ± 0.01
	DP-GNN	50.06 ± 0.02	50.05 ± 0.02	50.62 ± 0.04	50.67 ± 0.05
	DPGS	50.08 ± 0.03	50.08 ± 0.02	50.28 ± 0.03	50.46 ± 0.05
PubMed	GAP	50.08 ± 0.05	50.35 ± 0.04	51.12 ± 0.02	51.29 ± 0.03
	DP-GNN	50.06 ± 0.03	50.06 ± 0.04	50.35 ± 0.02	50.66 ± 0.02
	DPGS	50.29 ± 0.02	50.30 ± 0.02	50.50 ± 0.06	50.96 ± 0.02
Coauthor	GAP	50.03 ± 0.05	50.18 ± 0.04	50.52 ± 0.05	51.48 ± 0.02
	DP-GNN	50.00 ± 0.02	50.08 ± 0.04	50.57 ± 0.02	50.93 ± 0.06
	DPGS	50.23 ± 0.02	50.30 ± 0.07	50.33 ± 0.04	50.44 ± 0.05

remain an open challenge, calling for alternative privacy notions and structural obfuscation techniques.

C. Graph Summarization and Coarsening

Graph summarization and coarsening reduce graph size while preserving key properties, enabling efficient storage [53], scalable analytics [26], and visualization [54]. Traditional methods rely on heuristics, whereas recent approaches adopt information-theoretic or optimization-based formulations. SSumM [38] uses the Minimum Description Length (MDL) principle to merge nodes and sparsify edges, achieving compression with fidelity guarantees. Kumar et al. [32] coarsen both structure and features by optimizing a feature-aware similarity objective, preserving semantic consistency for downstream tasks. These techniques not only improve efficiency but also complement DP by reducing data sensitivity before noise injection, making them well-suited for privacy-preserving graph analytics.

VIII. CONCLUSION

We propose DPGS, a novel framework for differentially private graph synthesis that integrates structure and feature summarization to enable accurate and privacy-preserving graph generation. By applying differential privacy during the summarization phase, DPGS significantly reduces sensitivity and avoids noise injection in downstream tasks. Our framework supports structure-feature fusion and uses a one-shot perturbation strategy to achieve strong utility under strict privacy budgets. Extensive experiments on node classification, link prediction, and membership inference demonstrate that DPGS consistently outperforms state-of-the-art baselines in both performance and privacy. These results validate the effectiveness of summarization-based synthesis as a promising paradigm for private graph publishing.

Proof of Theorem 1

Proof: We analyze the privacy of the DPGS mechanism by decomposing the GRAPHSUMMARIZATION procedure into its core components and applying the composition and post-processing properties of differential privacy.

(1) Structure Summarization via DP-SCGNN. The spectral clustering component injects Gaussian noise into each layer of the GNN's message passing (Eq. (2)) to achieve differential privacy:

$$\boldsymbol{\eta}^{(k)} \sim \mathcal{N}(0, \sigma^2 \mathbb{I}).$$

This corresponds to DP-SGD over a sequence of gradient updates, and its privacy guarantee is formally captured using Rényi differential privacy. As a result, the output cluster assignment matrix $\boldsymbol{P}$ satisfies $(\varepsilon_1, \delta_1)$-Edge-DP.

(2) Cluster Refinement via Exponential Mechanism. For each uncertain node $i \in \mathcal{R}$, DPGS selects a refined cluster assignment using the exponential mechanism:

$$\Pr\left[[] c_i = c\right] \propto \exp\left(\frac{\varepsilon_2 \cdot s_i(c)}{2}\right),$$

where $s_i(c)$ counts how many neighbors of node i are confidently assigned to cluster c, and the sensitivity $\Delta_q = 1$ due to the addition/removal of a single edge. Therefore, this step satisfies ε_2-Edge-DP.

(3) Weight Assignment via Gaussian Mechanism. The summary weight matrix $\bar{\boldsymbol{W}}$ is computed by aggregating intra- and inter-cluster edges, followed by the addition of calibrated Gaussian noise:

$$\bar{\boldsymbol{W}}_{u,v} = \frac{\sum_{i \in \bar{V}^u, j \in \bar{V}^v} \boldsymbol{W}_{i,j} + \mathcal{N}(0, \sigma^2)}{|\bar{V}^u| \cdot |\bar{V}^v|}.$$

Given that each edge affects at most one count, the sensitivity is 1, and the Gaussian mechanism ensures $(\varepsilon_2, \delta_2)$-Edge-DP.

(4) Feature Summarization via Objective Perturbation. To protect the node features $\boldsymbol{X}$, DPGS solves a convex optimization problem with a perturbed objective:

$$\min_{\bar{\boldsymbol{X}}} \operatorname{tr}(\bar{\boldsymbol{X}}^\top \bar{\boldsymbol{L}}_s \bar{\boldsymbol{X}}) + \frac{\alpha}{2} \|\bar{\boldsymbol{Z}}\bar{\boldsymbol{X}} - \boldsymbol{X}\|_F^2 + \operatorname{tr}(\boldsymbol{B}^\top \bar{\boldsymbol{X}}),$$

where $\boldsymbol{B} \sim \mathcal{N}(0, \Delta_B^2 \sigma^2 \mathbb{I})$. Since the objective is strongly convex and $\bar{\boldsymbol{L}}_s, \bar{\boldsymbol{Z}}$ are derived from DP-compliant summaries, the entire mechanism satisfies $(\varepsilon_3, \delta_3)$-DP with respect to $\boldsymbol{X}$ under row-level changes.

(5) Post-processing via GRAPHRECONSTRUCTION. The final synthetic graph $\hat{G}$ is generated solely using $\bar{\boldsymbol{W}}, \bar{\boldsymbol{X}}, \bar{\boldsymbol{Z}}$, which are outputs of DP mechanisms. According to the post-processing property of DP, any function applied to these outputs does not incur additional privacy loss.

(6) Composition. By sequential composition of the above components, the total privacy budget over the structure is:

$$(\varepsilon_1 + \varepsilon_2 + \varepsilon_3, \delta_1 + \delta_2 + \delta_3),$$

and for the feature-level protection, we have:

$$(\varepsilon_3, \delta_3)\text{-DP}.$$

Letting $\delta := \delta_1 + \delta_2 + \delta_3$ completes the proof. ∎

Proof of Theorem 2

We provide a detailed proof for the following theorem:

Theorem 2 [Privacy Guarantee for any r-Layer GNN] Let N be the number of training instances V_{tr}, N_k be the maximum number of cluster neighbors any node can have in the coarsened graph (i.e., cluster degree), r be the number of GNN layers, and m be the batch size during training. For any choice of the noise standard deviation $\sigma > 0$ and clipping threshold η, every iteration t of DP-SGD is (α_ρ, γ) node-level Rényi DP, where:

$$\gamma = \frac{1}{\alpha_\rho - 1} \ln \mathbb{E}_\rho \left[\exp\left(\alpha_\rho(\alpha_\rho - 1) \cdot \frac{2\rho^2 \eta^2}{\sigma^2}\right)\right],$$

$$\rho \sim \text{Hypergeometric}\left(N, \frac{N_k^{r+1} - 1}{N_k - 1}, m\right).$$

Proof. Let G and G' be two node-adjacent graphs differing by a single node v. For a GNN with r layers, the output of each node in the training set V_{tr} is influenced by its r-hop neighborhood. Under the assumption that each node in the coarsened (clustered) graph has degree at most N_k, the number of subgraphs in which the removed node v could appear is bounded by $d = \frac{N_k^{r+1} - 1}{N_k - 1}$.

Let $\mathcal{B}_t$ be the mini-batch sampled at iteration t, and ρ be the number of affected subgraphs from the total d that fall into $\mathcal{B}_t$. Then, ρ follows a hypergeometric distribution Hypergeometric(N, d, m).

Assume each clipped per-node gradient has ℓ_2-norm at most η. Then the total sensitivity of the summed gradient $\mathbf{u}_t$ is upper bounded by $2\eta \cdot \rho$. Adding Gaussian noise of variance σ^2 implies by Corollary 3 of [35] that the iteration satisfies:

$$(\alpha_\rho, \gamma_\rho)\text{-RDP}, \quad \gamma_\rho = \frac{\alpha_\rho \cdot (2\rho\eta)^2}{2\sigma^2}.$$

Taking expectation over ρ:

$$\gamma = \frac{1}{\alpha_\rho - 1} \ln \mathbb{E}_\rho \left[\exp\left(\alpha_\rho(\alpha_\rho - 1) \cdot \frac{2\rho^2 \eta^2}{\sigma^2}\right)\right].$$

Therefore, each iteration is (α_ρ, γ)-RDP, and over T iterations, by the composition property of Rényi DP, the full algorithm satisfies $(\alpha_\rho, \gamma T)$-RDP. $\square$

Additional Experimental results

We evaluate the quality of the synthetic graphs using two additional metrics: **Normalized Mutual Information (NMI)** and **Kullback–Leibler (KL) divergence**. NMI quantifies the alignment between community detection results on the original and synthetic graphs, while KL divergence measures the difference in degree distributions between the two graphs. Below are the formal definitions of both metrics.

a) Normalized Mutual Information (NMI).: Let $A = \{A_1, A_2, \ldots, A_R\}$ and $B = \{B_1, B_2, \ldots, B_S\}$ be two partitions of a graph $G = (V, E)$. Their overlap can be represented using a contingency table H, where H_{ij} denotes the number of nodes shared by clusters A_i and B_j. Let x_i (resp. y_j) be the sum of the i-th row (resp. j-th column) in H. The NMI between A and B is computed as:

$$\text{NMI}(A, B) = \frac{-2 \sum_{i=1}^{R} \sum_{j=1}^{S} H_{ij} \log\left(\frac{H_{ij} n}{x_i y_j}\right)}{\sum_{i=1}^{R} x_i \log\left(\frac{x_i}{n}\right) + \sum_{j=1}^{S} y_j \log\left(\frac{y_j}{n}\right)}. \quad (10)$$

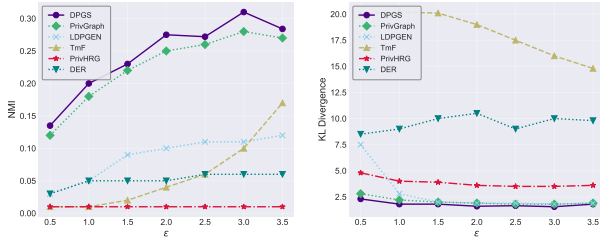| NMI on Cora	KL Divergence on Cora

Fig. 6: Results on NMI and KL Divergence over dataset Cora

b) Kullback–Leibler (KL) Divergence.: Given the degree distributions $P(x)$ and $\hat{P}(x)$ of the original and synthetic graphs respectively, the KL divergence is defined as:

$$D_{\mathrm{KL}}(P \parallel \hat{P}) = \sum_{x \in \mathcal{X}} P(x) \log \left(\frac{P(x)}{\hat{P}(x)} \right). \qquad (11)$$

The results are shown in Figure 6.

REFERENCES

[1] J. Leskovec, D. Huttenlocher, and J. Kleinberg, "Signed Networks in Social Media," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2010, pp. 1361–1370.

[2] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graph Evolution: Densification and Shrinking Diameters," *ACM transactions on Knowledge Discovery from Data (TKDD)*, vol. 1, no. 1, pp. 2–es, 2007.

[3] P. J. Mucha and M. A. Porter, "Communities in Multislice Voting Networks," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 20, no. 4, p. 041108, 2010.

[4] D. Sharma, R. Shukla, A. K. Giri, and S. Kumar, "A Brief Review on Search Engine Optimization," in *2019 9th International Conference on Cloud Computing, Data Science & Engineering*. IEEE, 2019, pp. 687–692.

[5] F. Jiang, C. K. Leung, and A. G. Pazdor, "Big Data Mining of Social Networks for Friend Recommendation," in *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE, 2016, pp. 921–922.

[6] Y. Wang, L. Xie, B. Zheng, and K. C. Lee, "High Utility K-Anonymization for Social Network Publishing," *Knowledge and Information Systems*, vol. 41, no. 3, pp. 697–725, 2014.

[7] J. Cheng, A. W.-c. Fu, and J. Liu, "K-Isomorphism: Privacy Preserving Network Publication against Structural Attacks," in *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, 2010, pp. 459–470.

[8] S. Ji, P. Mittal, and R. Beyah, "Graph Data Anonymization, De-Anonymization Attacks, and De-Anonymizability Quantification: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 2, pp. 1305–1326, 2016.

[9] J. Qian, X.-Y. Li, C. Zhang, L. Chen, T. Jung, and J. Han, "Social Network De-Anonymization and Privacy Inference with Knowledge Graph Model," *IEEE Transactions on Dependable and Secure Computing*, vol. 16, no. 4, pp. 679–692, 2017.

[10] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating Noise to Sensitivity in Private Data Analysis," in *Theory of Cryptography Conference*. Springer, 2006, pp. 265–284.

[11] L. Du, Z. Zhang, S. Bai, C. Liu, S. Ji, P. Cheng, and J. Chen, "AHEAD: Adaptive Hierarchical Decomposition for Range Query under Local Differential Privacy," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 1266–1288.

[12] Z. Zhang, T. Wang, N. Li, S. He, and J. Chen, "CALM: Consistent Adaptive Local Marginal for Marginal Release under Local Differential Privacy," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 212–229.

[13] T. Wang, J. Q. Chen, Z. Zhang, D. Su, Y. Cheng, Z. Li, N. Li, and S. Jha, "Continuous Release of Data Streams under both Centralized and Local Differential Privacy," in *ACM CCS*, 2021.

[14] A. Sala, X. Zhao, C. Wilson, H. Zheng, and B. Y. Zhao, "Sharing Graphs Using Differentially Private Graph Models," in *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, 2011, pp. 81–98.

[15] Q. Xiao, R. Chen, and K.-L. Tan, "Differentially Private Network Data Release via Structural Inference," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2014, pp. 911–920.

[16] M. Hay, C. Li, G. Miklau, and D. Jensen, "Accurate Estimation of the Degree Distribution of Private Networks," in *2009 Ninth IEEE International Conference on Data Mining*. IEEE, 2009, pp. 169–178.

[17] H. Sun, X. Xiao, I. Khalil, Y. Yang, Z. Qin, H. Wang, and T. Yu, "Analyzing Subgraph Statistics from Extended Local Views with Decentralized Differential Privacy," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019.

[18] T. Ji, C. Luo, Y. Guo, Q. Wang, L. Yu, and P. Li, "Community Detection in Online Social Networks: A Differentially Private and Parsimonious Approach," *IEEE transactions on Computational Social Systems*, vol. 7, no. 1, pp. 151–163, 2020.

[19] H. H. Nguyen, A. Imine, and M. Rusinowitch, "Differentially Private Publication of Social Graphs at Linear Cost," in *2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE, 2015, pp. 596–599.

[20] R. Chen, B. Fung, P. S. Yu, and B. C. Desai, "Correlated Network Data Publication via Differential Privacy," *The VLDB Journal*, vol. 23, no. 4, pp. 653–676, 2014.

[21] A. Clauset, C. Moore, and M. E. Newman, "Hierarchical Structure and the Prediction of Missing Links in Networks," *Nature*, vol. 453, no. 7191, pp. 98–101, 2008.

[22] Z. Qin, T. Yu, Y. Yang, I. Khalil, X. Xiao, and K. Ren, "Generating Synthetic Decentralized Social Graphs with Local Differential Privacy," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 425–438.

[23] Q. Yuan, Z. Zhang, L. Du, M. Chen, P. Cheng, and M. Sun, "{PrivGraph}: differentially private graph data publication by exploiting community information," in *32nd USENIX Security Symposium (USENIX Security 23)*, 2023, pp. 3241–3258.

[24] Y. Liu, T. Safavi, A. Dighe, and D. Koutra, "Graph summarization methods and applications: A survey," *ACM computing surveys (CSUR)*, vol. 51, no. 3, pp. 1–34, 2018.

[25] X. Xian, T. Wu, Y. Liu, W. Wang, C. Wang, G. Xu, and Y. Xiao, "Towards Link Inference Attack against Network Structure Perturbation," *Knowledge-Based Systems*, vol. 218, p. 106674, 2021.

[26] M. Riondato, D. García-Soriano, and F. Bonchi, "Graph summarization with quality guarantees," *DMKD*, vol. 31, no. 2, pp. 314–349, 2017.

[27] K. LeFevre and E. Terzi, "Grass: Graph structure summarization," in *SDM*, 2010.

[28] F. McSherry and K. Talwar, "Mechanism Design via Differential Privacy," in *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*. IEEE, 2007, pp. 94–103.

[29] M. McPherson, L. Smith-Lovin, and J. M. Cook, "Birds of a feather: Homophily in social networks," *Annual review of sociology*, vol. 27, no. 1, pp. 415–444, 2001.

[30] I. S. Dhillon, Y. Guan, and B. Kulis, "Kernel k-means: spectral clustering and normalized cuts," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2004, pp. 551–556.

[31] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 308–318.

[32] M. Kumar, A. Sharma, S. Saxena, and S. Kumar, "Featured graph coarsening with similarity guarantees," in *International Conference on Machine Learning*. PMLR, 2023, pp. 17953–17975.

[33] X. Ran, Q. Ye, J. Lou, and H. Hu, "Differentially private implicit matrix factorization," https://github.com/marlowe518/CCS-2025/blob/main/dpgs_full.pdf, 2025.

[34] A. Daigavane, G. Madan, A. Sinha, A. G. Thakurta, G. Aggarwal, and P. Jain, "Node-level differentially private graph neural networks," *arXiv preprint arXiv:2111.15521*, 2021.

[35] I. Mironov, "Rényi differential privacy," in *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*. IEEE, 2017, pp. 263–275.

[36] C. Forbes, M. Evans, N. Hastings, and B. Peacock, *Statistical Distributions*. Wiley, 2011. [Online]. Available: https://books.google.co.in/books?id=YhF1osrQ4psC

[37] A. Loukas and P. Vandergheynst, "Spectrally approximating large graphs with smaller graphs," in *International conference on machine learning*. PMLR, 2018, pp. 3237–3246.

[38] K. Lee, H. Jo, J. Ko, S. Lim, and K. Shin, "Ssumm: Sparse summarization of massive graphs," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 144–154.

[39] F. Dorfler and F. Bullo, "Kron reduction of graphs with applications to electrical networks," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, no. 1, pp. 150–163, 2012.

[40] J. Chen and I. Safro, "Algebraic distance on graphs," *SIAM Journal on Scientific Computing*, vol. 33, no. 6, pp. 3468–3490, 2011.

[41] F. Wu, Y. Long, C. Zhang, and B. Li, "Linkteller: Recovering private edges from graph neural networks via influence analysis," in *2022 ieee symposium on security and privacy (sp)*. IEEE, 2022, pp. 2005–2024.

[42] S. Sajadmanesh, A. S. Shamsabadi, A. Bellet, and D. Gatica-Perez, "{GAP}: Differentially private graph neural networks with aggregation perturbation," in *32nd USENIX Security Symposium (USENIX Security 23)*, 2023, pp. 3223–3240.

[43] A. Kolluri, T. Baluta, B. Hooi, and P. Saxena, "Lpgnet: Link private graph networks for node classification," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, 2022, pp. 1813–1827.

[44] T. Tang, Y. Niu, S. Avestimehr, and M. Annavaram, "Edge private graph neural networks with singular value perturbation," *arXiv preprint arXiv:2403.10995*, 2024.

[45] I. E. Olatunji, W. Nejdl, and M. Khosla, "Membership inference attack on graph neural networks," in *2021 Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*. IEEE, 2021, pp. 11–20.

[46] E. Chien, W.-N. Chen, C. Pan, P. Li, A. Ozgur, and O. Milenkovic, "Differentially private decoupled graph convolutions for multigranular topology protection," *Advances in Neural Information Processing Systems*, vol. 36, pp. 45 381–45 401, 2023.

[47] S. Sajadmanesh and D. Gatica-Perez, "Progap: Progressive graph neural networks with differential privacy guarantees," in *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, 2024, pp. 596–605.

[48] Y. Zhang, M. Humbert, B. Surma, P. Manoharan, J. Vreeken, and M. Backes, "Towards Plausible Graph Anonymization," in *Proceedings of the 27th Annual Network and Distributed System Security Symposium*, 2020.

[49] C. Yang, L. Zhong, L.-J. Li, and L. Jie, "Bi-Directional Joint Inference for User Links and Attributes on Large Social Graphs," in *Proceedings of the 26th International Conference on World Wide Web Companion*, 2017, pp. 564–573.

[50] N. Eagle, A. Pentland, and D. Lazer, "Inferring Friendship Network Structure by Using Mobile Phone Data," *Proceedings of the National Academy of Sciences*, vol. 106, no. 36, pp. 15 274–15 278, 2009.

[51] W.-H. Lee, C. Liu, S. Ji, P. Mittal, and R. B. Lee, "How to Quantify Graph De-Anonymization Risks," in *Information Systems Security and Privacy*, 2017, pp. 84–104.

[52] Y. Zhao and I. Wagner, "Using Metrics Suites to Improve the Measurement of Privacy in Graphs," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 1, pp. 259–274, 2020.

[53] H. Toivonen, F. Zhou, A. Hartikainen, and A. Hinkka, "Compression of weighted graphs," in *KDD*, 2011.

[54] D. Koutra, U. Kang, J. Vreeken, and C. Faloutsos, "Vog: Summarizing and understanding large graphs," in *SDM*, 2014.