Leveraging Homophily under Local Differential Privacy for Effective Graph Neural Networks

Yule Xie[1], Jiaxin Ding[1] ✉, Pengyu Xue[2], Xin Ding[1], Haochen Han[1], Luoyi Fu[1], and Xinbin Wang[1]

[1] Shanghai Jiao Tong University
{jiaxinding}@sjtu.edu.cn
[2] East China Normal University

Abstract. Graph Neural Networks (GNNs) have become indispensable tools for analyzing graph-structured data, with applications across numerous domains. However, collecting graph data that is locally stored by users in privacy-sensitive scenarios remains challenging. Existing methods applying Local Differential Privacy (LDP) fail to account for homophily—the tendency of connected nodes to share similar attributes, which consequently leads to suboptimal performance under limited privacy budgets. To address this challenge, we propose HPGR (Homophily Preserving Graph Reconstruction), a novel approach for collecting and modeling graph topology under LDP, while preserving homophily. Our method employs a homophily-aware querying and modeling mechanism that integrates homophily priors into the data collection process, and enables robust reconstruction of the underlying graph structure despite the injected noise. We provide theoretical analyses demonstrating that our method satisfies LDP requirements while effectively preserving homophily. Extensive experiments on benchmark datasets show that our approach significantly outperforms existing methods, achieving a superior balance between privacy protection and model utility.

Keywords: Graph neural network · Differential Privacy.

1 Introduction

Graph neural networks (GNNs) have emerged as indispensable tools across numerous domains, ranging from social networks to healthcare, fraud detection, and recommendation systems. Central to their success is the exploitation of *homophily* [23, 26]—the natural tendency for nodes with similar attributes to form connections—which underpins the effective propagation of information and the superior performance of GNNs. However, the collection of relational data required for these models increasingly conflicts with growing privacy regulations and decentralized data ownership paradigms.

Consider, for example, a decentralized social networking app like Briar [2], where user connection records are stored locally on individual devices to enable

peer-to-peer(P2P) interactions, such as sending messages, sharing files, or tracking contacts, without relying on a central server. While this decentralized approach inherently enhances user privacy, it obstructs centralized relational data aggregation for training GNN models on graphs to further enhance downstream tasks, such as user preference classification. Moreover, stringent privacy regulations, such as the GDPR [21], restrict the collection of user connection data without explicit consent, further complicating data usage for GNN training.

Local differential privacy (LDP) [5] emerges as a compelling solution, by ensuring that individual data is obfuscated [6] before it is shared or queried. In the previous scenario, LDP can be applied to perturb connection data with noise before it leaves the user's device, ensuring privacy while enabling data collection. However, most existing LDP mechanisms were designed for tabular or scalar data and fail to address the intricacies of graph-structured data [8]. Injecting noise via LDP mechanisms, such as the randomized response mechanism, into graphs can disrupt the intricate connectivity patterns by either falsely reporting a connection or masking an actual one, and more critically, undermine the homophily property. Despite several studies [27, 8, 13] on privacy-preserving GNNs, explicit preservation of homophily under LDP constraints remains underexplored. Existing approaches have primarily focused on maintaining graph sparsity, neglecting the key factors necessary to preserve the performance of GNNs.

In this paper, we propose HPGR (**H**omophily **P**reserving **G**raph **R**econstruction), a novel framework that reconciles the stringent privacy requirements of LDP with the structural imperatives of GNNs. Our key insight is to leverage server-side cluster labels (e.g., coarse user categories inferred from public profiles) to guide the privatization process, preserving inter-cluster connectivity patterns critical for homophily. HPGR operates in two phases. First, users perturb their cluster-specified degree vectors—counts of connections to predefined server-labeled clusters—using an LDP mechanism that minimizes noise while retaining community structure. Second, we adopt the Degree-Corrected Stochastic Block Model [9], along with a mixed membership extension, to reconstruct the graph from these privatized queries. We provide theoretical privacy-utility trade-off guarantees and empirically validate HPGR on benchmark datasets. Extensive experiments demonstrate the superiority of our method. GNNs trained on HPGR-reconstructed graphs retain 88% of their non-private accuracy on average under strict privacy budgets ($1 \leq \epsilon \leq 4$), achieving approximately a 5.8% improvement over SOTA. Our contribution can be summarized as follows[3]:

- We introduce a querying mechanism that gathers each user's cluster-specific degree vectors instead of raw edge data. This approach embeds homophily priors into the data collection process while strictly adhering to local differential privacy constraints.
- We propose a novel homophily-preserving graph reconstruction method and incorporate homophily priors to refine the reconstructed topology. This not only boosts GNN performance but also bridges the gap between degree-vector queries and graph homophily metrics.

[3] Experiment code can be found at: https://github.com/xyl-alter/HPGR.git

– We provide theoretical error analyses from four perspectives: privacy, sparsity, homophily, and accuracy. The experimental results further demonstrate significant improvements over existing approaches.

2 PRELIMINARY

2.1 Problem statement

This work focuses on protecting the local neighborhood information of nodes in a semi-supervised node classification task on undirected, unweighted graphs. GNN training needs three inputs: node features X, labels Y, and neighborhood relations encoded in the adjacency matrix A. In our decentralized setting, the server can access X and Y (e.g. from the public user profiles), but the graph topology A remains distributed-each node i locally stores its local neighborhood $A_i \in \{0,1\}^n$. The server is considered honest-but-curious (i.e., adheres to protocols but may infer sensitive information), aims to collect A to conduct GNN training. This raises a privacy challenge: transmitting raw neighborhood data risks exposing sensitive connections. We adopt ϵ-edge local differential privacy (LDP) [27] to protect edge privacy, which ensures that even when one node's local graph structure is altered, the privacy-preserving algorithm remains robust against inference of the actual graph topology.

Definition 1. *(ϵ-Edge Local Differential Privacy [27]). For $\epsilon > 0$, a randomized algorithm $\mathcal{A} : \{0,1\}^n \to \mathcal{O}$ satisfies ϵ-edge local differential privacy if for any two adjacent neighborhood vectors $A_i', A_i \in \{0,1\}^n$ of node i which differ only in one entry, i.e., $|A_i - A_i'|_1 = 1$, and for any possible output $O \in \mathcal{O}$, we have*

$$Pr(\mathcal{A}(A_i) = O) \leq e^{\epsilon} \cdot Pr(\mathcal{A}(A_i') = O). \tag{1}$$

The ϵ-edge LDP guarantees that modifying a single edge in a node's local neighborhood does not significantly alter the output of the privacy-preserving algorithm, thereby preventing the server from reliably inferring the existence or absence of any specific edge. The privacy budget ϵ indicates the level of privacy protection: the smaller the budget is, the stricter the privacy protection, and the greater the disturbance. Key notations in this paper are shown in Table 1.

2.2 Homophily in graphs

Homophily—the tendency of nodes with similar characteristics to form connections—is critical for GNN performance. We adopt a variant of the class-label homophily proposed in [12] as our homophily metric:

$$h = \frac{1}{c}\sum_{k=1}^{c} h_k, \quad h_k = \frac{\sum_{u \in C_k} d_u^{k_u}}{\sum_{u \in C_k} d_u} = \frac{M_{k,k}}{\sum_{j=1}^{c} M_{k,j}}, \tag{2}$$

where k_u is the cluster node u belongs to, $d_u^{k_u}$ is the number of edges between node u and nodes in k_u, and M is the cluster connection matrix indicating the number of edges within and between clusters.

Symbol	Description	Symbol	Description
X	feature matrix of nodes	d_i	cluster-specified degree vector of node i
Y	labels of nodes	$\mathbf{d}$	list of node degrees
A	graph adjacency matrix	Π	prior link probability matrix
n	the number of nodes	$\mathcal{P}$	posterior link probability matrix
c	the number of classes/clusters	C	node classification matrix
ϵ	the privacy budget	M	cluster connection matrix

Table 1: Basic Notations

Graph homophily is crucial for GNN performance. Figure 1 shows the impact of homophily on the node classification performance of GNN models, demonstrating a sharp decline in GNN performance as homophily decreases. This emphasizes the need to preserve homophily when querying private graphs to achieve optimal GNN results. However, conventional differential privacy mechanisms, such as the Laplace mechanism and randomized responses, fail to maintain homophily, highlighting the importance of allocating additional privacy budget for homophily-preserving queries.

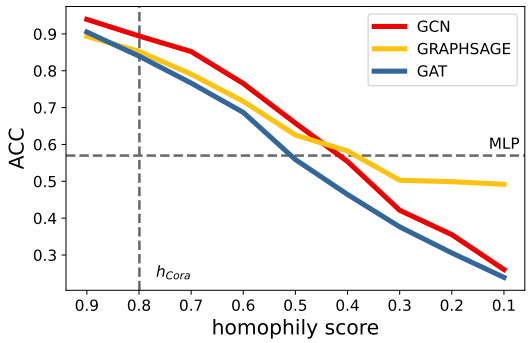

Fig. 1: Performance of three GNN models on SBM-reconstructed Cora dataset. Nodes with the same labels are connected with probability p while nodes with different labels are connected with probability q. The expected homophily score of the SBM (Stochastic Block Model)-generated graph is $h = p/(p + (c-1)q)$.

2.3 Degree-Corrected Stochastic Block Model (DC-SBM)

The Degree-Corrected Stochastic Block Model not only preserves the homophily of the graph but also maintains the degree distribution of nodes. In the DC-SBM model, the probability of an edge between node i and j is given by:

$$P(A_{ij} = 1) = \theta_i \theta_j \Phi_{g_i g_j}, \tag{3}$$

where θ_i, θ_j are degree correction parameters correlated to the degree of node i and node j, and $\Phi_{g_i g_j}$ is the probability that nodes in category g_i and g_j are connected, where g_i and g_j are the categories node i and node j belong to. $\Phi_{g_i g_j}$ is computed with

$$\Phi_{g_i g_j} = \frac{M_{g_i g_j}}{r_{g_i} r_{g_j}}, \tag{4}$$

where $M_{g_i g_j}$ is the number of edges between nodes in g_i and nodes in g_j, and r_{g_i} and r_{g_j} are the number of nodes in g_i and g_j.

3 METHODS

We propose **H**omophily-**P**reserving **G**raph **R**econstruction(HPGR) algorithm for effective graph neural networks under local differential privacy. The server makes homophily-aware queries to the nodes (clients) to reconstruct the graph for GNN training. During the query stage, the server first group local nodes into clusters. It queries both the neighborhood and the cluster-specified degree vector—number of edges connecting to each cluster, of each node. Local nodes respond by applying randomized response (RR) and Laplace (Lap) mechanism to perturb the adjacency lists the degree vectors. Thereafter, the server constructs a DC-SBM random graph based on the collected noisy degree vector and employs Bayesian estimation to refine the collected topology. The edge probabilities in the random graph serve as priors, while the noisy adjacency matrix acts as evidence, allowing the server to compute posterior probabilities for each potential link and reconstruct the graph for GNN training.

3.1 Homophily-aware topology collection under LDP

The server first clusters nodes into c clusters based on their labels, and then queries each node for its adjacency vector A_i and cluster-specified degree vector d_i. By collecting degree vectors, the server gathers information on node connectivity across clusters, reflecting graph homophily. In the cases where part of node labels are unavailable, the server uses pseudo labels for the unlabeled nodes. Pseudo labels are generated by an MLP, $C = \mathrm{MLP}(X, Y) \in R^{N \times c}$. To ensure privacy protection during the querying process, the server divides the total privacy budget ϵ into two components with proportion δ: $\epsilon_a = \epsilon \cdot \delta$ and $\epsilon_d = \epsilon \cdot (1 - \delta)$, which are allocated for querying adjacency and degree vectors respectively. By controlling the distribution of the privacy budget, the server can balance the need for better graph statistics (i.e., homophily) or more precise adjacency details.

The neighborhood query is straightforward, while for degree vector queries, the server should first send labels to each node. However, if the client is untrusted or potentially curious, revealing clustering information can lead to privacy leakage of node labels. In such cases, techniques such as homomorphic encryption and Secure Multi-Party Computation(SMPC) can be applied to avoid revealing the clustering information or the clients' local data.

Upon receiving a query, the local node calculates its degree vector by definition and uses random response (flipping one bit with probability $p = \frac{1}{1+exp(\epsilon_a)}$) [22] and Laplace mechanisms (adding Laplace noise with scale $\frac{1}{\epsilon_d}$) [6] to perturb the adjacency and degree vectors respectively to ensure LDP compliance. To better utilize the classification information provided by MLP, a soft query-response mechanism is also employed, where the server queries with a classification matrix $C \in \mathbf{R}^{n\times c}$ and the local node computes its degree vector with $A_i \cdot C$. Experiments show that the soft query mechanism often achieves better performance by acknowledging the uncertainty in pseudo labels, while at the cost of a higher communication overhead. The practical algorithm for the soft response on the client side is shown in Algorithm 1, with the hard response as a special case where C is one-hot.

Algorithm 1 client-side response under LDP

Input: A_i: Adjacency list, C: Cluster matrix, ϵ: Privacy budget, δ: Privacy parameter
Output: $\tilde{A}_i \in \{0,1\}^n$: private adjacency vector, $\tilde{d}_i \in \mathbf{R}^c$: private degree vector

function INJECTION(A_i, C, ϵ, δ)
  $\epsilon_d = \delta\epsilon$ ▷ privacy budget for degree vector query
  $\epsilon_a = (1-\delta)\epsilon$ ▷ privacy budget for adjacency vector query
  for $j \in \{1, 2, \cdots, n\}$ **do**
    $\tilde{A}_{ij} = \begin{cases} A_{ij}, & \text{with probability } \frac{\exp(\epsilon_a)}{1+\exp(\epsilon_a)} \\ 1 - A_{ij}, & \text{with probability } \frac{1}{1+\exp(\epsilon_a)} \end{cases}$ ▷ apply RR mechanism
  end for
  $\hat{d}_i = A_i \cdot C$ ▷ calculating degree vector
  for $j \in \{1, 2, \cdots, c\}$ **do**
    sample $l_j \sim \text{Laplace}(0, \frac{1}{\epsilon_d})$
    $\tilde{d}_i[j] = \hat{d}_i[j] + l_j$ ▷ apply Laplace mechanism
  end for
  return $(\tilde{A}_i, \tilde{d}_i)$
end function

Theorem 1. *The client-side response algorithm above achieves ϵ-edge LDP.*

The proof of this theorem is straightforward, since the RR and Laplace mechanisms satisfy ϵ_a and ϵ_d-LDP respectively. According to the composition theorem, the total privacy budget is $\epsilon_a + \epsilon_d = \epsilon$.

3.2 Graph reconstruction

After receiving replies from local nodes, the server obtains noisy adjacency vectors $\tilde{A}_1, \tilde{A}_2, \cdots, \tilde{A}_n$ and degree vectors $\tilde{d}_1, \tilde{d}_2, \cdots, \tilde{d}_n$. The server then computes a prior distribution of edges with degree vectors and regards the adjacency vectors as evidence to estimate posterior probabilities through Bayesian estimation.

Prior distribution estimation given degree vectors Although DC-SBM is well suited for hard queries, applying it to soft queries (i.e., the mixed membership scenario [14]) is challenging, particularly in degree corrections for mixed membership. Here we introduce our **M**ixed **M**embership **DC-SBM** (MM-DCSBM) model, where the traditional DC-SBM is a special case when C is a one-hot matrix. Our MM-DCSBM algorithm requires three inputs: cluster connection matrix M (representing the number of edges between clusters), degree list $\mathbf{d}$ (reflecting the degree of each node), and the cluster matrix C obtained in Section 3.1.

Cluster connection matrix. The server computes $\tilde{M} = C^T \cdot \tilde{D}$, where $\tilde{D} = [\tilde{d}_1, \tilde{d}_2, \ldots, \tilde{d}_n]^T$ is the matrix of noisy degree vectors, and $\tilde{M}$ is the noisy cluster connection matrix. Due to Laplace noise injection, $\tilde{M}$ may not be symmetric, but the server can correct it with the following maximum likelihood estimation.

Theorem 2. *The maximum likelihood estimation of $\tilde{M}_{i,j}$ is*

$$\tilde{M}_{i,j,MLE} = \frac{n_j}{n_i + n_j}\tilde{M}_{i,j} + \frac{n_i}{n_i + n_j}\tilde{M}_{j,i} \sim N(M_{i,j}, \frac{4n_i n_j}{(n_i + n_j)\epsilon_d^2}), \tag{5}$$

where $n_i = \sum_{k\in[1,n]} C_{k,i}, n_j = \sum_{k\in[1,n]} C_{k,j}$ are the number of nodes in cluster i and cluster j, $M_{i,j}$ is the cluster connection matrix without noise injection.

Degree list. Instead of allocating extra privacy budget for querying node degrees, we sum the queried degree vectors to obtain unbiased estimates.

We represent our MM-DCSBM in Algorithm 2. The key difference between our algorithm and the traditional DC-SBM lies in degree regularization, where mixed membership precludes calculating the regularization factor R for each node independently. This results in Φ being computed via element-wise division in the DC-SBM algorithm, while through multiplication by the inverse matrix in our algorithm. To validate our algorithm, we present Proposition 1. A more detailed analysis of its properties is provided in Section 4.

Proposition 1. *When C is a one-hot matrix, meaning each node explicitly belongs to a cluster, the MM-DCSBM algorithm in Algorithm 2 reduces to the traditional DC-SBM algorithm introduced in Subsection 2.3.*

Posterior distribution estimation given noisy adjacency vector Inferring the posterior probability of each edge's existence with Bayesian estimation has been demonstrated to be both effective and efficient [27]. In line with previous works, we treat the edge existence probability in the reconstructed MM-DCSBM random graph as the prior probability and the adjacency vectors obtained through the random response mechanism as the evidence. Formally, the posterior edge existence probability between node i and j can be calculated with

$$\mathcal{P}_{ij} = \mathrm{P}[(A_{ij}, A_{ji}) = (1,1)|(\tilde{A}_{ij}, \tilde{A}_{ji})] = \frac{q_{ij}\Pi_{ij}}{q_{ij}\Pi_{ij} + q'_{ij}(1 - \Pi_{ij})}, \tag{6}$$

Algorithm 2 MM-DCSBM modeling algorithm

Input: M: cluster connection matrix, $\mathbf{d}$: node degree list, C: Node classification matrix
Output: Π: the prior link probability matrix

function MM-DCSBM($M, \mathbf{d}, C$)
  $R = C^T \cdot \text{diag}(\mathbf{d}) \cdot C$ ▷ Computer the regularization factor
  $\hat{M} = R^{-1} \cdot M \cdot R^{-1}$ ▷ normalize the cluster connection matrix
  $\Phi = C \cdot \hat{M} \cdot C^T$ ▷ compute the probability of links with SBM matrix
  $\Theta = \mathbf{d}^T \cdot \mathbf{d}$
  $\Pi = \Phi \odot \Theta$ ▷ correct SBM matrix with node degrees
  return Π
end function

where Π_{ij} is the prior distribution, q_{ij} and q'_{ij} are the joint distribution of $(\tilde{A}_{ij}, \tilde{A}_{ji})$. More specifically, we have

$$q_{ij} = \begin{cases} p^2, & (\tilde{A}_{ij}, \tilde{A}_{ji}) = (0,0) \\ p(1-p), & (\tilde{A}_{ij}, \tilde{A}_{ji}) = (1,0) \\ p(1-p), & (\tilde{A}_{ij}, \tilde{A}_{ji}) = (0,1) \\ (1-p)^2, & (\tilde{A}_{ij}, \tilde{A}_{ji}) = (1,1) \end{cases}, \quad q'_{ij} = \begin{cases} (1-p)^2, & (\tilde{A}_{ij}, \tilde{A}_{ji}) = (0,0) \\ p(1-p), & (\tilde{A}_{ij}, \tilde{A}_{ji}) = (1,0) \\ p(1-p), & (\tilde{A}_{ij}, \tilde{A}_{ji}) = (0,1) \\ p^2, & (\tilde{A}_{ij}, \tilde{A}_{ji}) = (1,1) \end{cases},$$

where $p = \frac{1}{1+e^{\epsilon_a}}$ is the probability of bit flipping in the random response mechanism.

4 Utility analysis

In this section, we analyze the utility of our method from three aspects: sparsity, homophily, and precision. Before delving into the detailed analysis, we first introduce three key properties of our proposed MM-DCSBM algorithm, presented as Theorem 3.

Theorem 3. *Let Π be the prior distribution computed with the MM-DCSBM model proposed in Algorithm 2 given the cluster connection matrix $\tilde{M}$, degrees of nodes $\tilde{\mathbf{d}}$, and classification matrix C_{ps}, i.e. Π = MM-DCSBM($\tilde{M}, \tilde{\mathbf{d}}, C_{ps}$), then*

$$\begin{aligned} &C_{ps}^T \cdot \Pi \cdot C_{ps} = \tilde{M} \quad (7a), && |\Pi|_1 = |\tilde{M}|_1 \quad (7b), \\ &\frac{\sum_k \Pi_{ik}}{\sum_k \Pi_{jk}} = \frac{\tilde{\mathbf{d}}_i}{\tilde{\mathbf{d}}_j} \quad \forall i, j \in \{1, 2, \cdots, n\}, && C_{ps,i} = C_{ps,j} \quad (7c). \end{aligned} \tag{7}$$

Theorem 3 demonstrates that the MM-DCSBM model preserves homophily, sparsity, and degree distribution of the input graph by ensuring that the reconstructed graph maintains the connection probabilities between clusters (Equation 7a), the total number of edges (Equation 7b), and the relative degrees of nodes within the same cluster (Equation 7c). These properties enable transforming the algorithm's output back into the input during further analysis. Notably, without the MLE process, i.e., $\tilde{M} = C^T[\tilde{d}_1, \cdots, \tilde{d}_n]^T$ holds, Equation 7c can be enhanced with $\sum_k \Pi_{ik} = \tilde{\mathbf{d}}_i, \ \forall i \in \{1, 2, \cdots, n\}$, regardless of the value of C_{ps}.

4.1 Sparsity analysis

Sparsity is a common property of real-world graphs. However, topologies collected by differential privacy mechanisms, such as the random response mechanism, are often overly dense. For a graph with n nodes and $m \sim \mathcal{O}(n)$ edges, using the RR mechanism with flip probability p gives an expected edge count of $(n^2 - m)p + m(1-p) \sim \mathcal{O}(n^2)$. Overly dense graphs can lead to excessive computational overhead during GNN training, and incorrectly connected edges can severely degrade GNN performance.

We use $E(||\Pi|_1 - |A|_1|)$ as sparsity metric. Theorem 4 provides a bound on the sparsity gap between our prior link probability and the real topology.

Theorem 4. *Considering the sparsity distance between the prior distribution matrix Π and the original adjacency matrix A, we have*

$$E[||\Pi|_1 - |A|_1|] \leq \frac{2\sqrt{2}}{\epsilon_d}\sqrt{\frac{cn}{\pi}}. \tag{8}$$

Corollary 1. *The l_1-distance between the posterior probability and the original topology is bounded by*

$$E(|\mathcal{P} - A|_1) \leq 2|A|_1 + \frac{2\sqrt{2}}{\epsilon_d}\sqrt{\frac{cn}{\pi}}. \tag{9}$$

In real-world graphs, the number of edges is typically of order $\mathcal{O}(n)$, while our method's graph density estimation error is $\mathcal{O}(\sqrt{n})$. Comparing our method with Blink [27] which also emphasizes sparsity, their estimation error is constrained by $E[||\Pi|_1 - |A|_1|] \leq \frac{n}{2\epsilon_d}$, which is of a higher order than ours.

4.2 Homophily analysis

Since the homophily metric in Equation 2 only relates to the cluster connection matrix, we use $E(|M_{true} - \tilde{M}|_{1,1})$ as our utility metric, where $\tilde{M}$ is the cluster connection matrix collected by the server and M_{true} is the actual one. Under such metric, Theorem 5 provides a guarantee of homophily. As the core Theorem of this work, we provide a brief proof here.

Theorem 5. *Assume that M_{true} is the cluster connection matrix given true labels C_{true} and noise-free adjacency matrix A, and $\tilde{M}$ is that given pseudo labels C_{ps} and prior edge distribution matrix Π, denote $\mathbf{d}_i$ as the degree of node i, then we have*

$$\begin{aligned} E(|M_{true} - \tilde{M}|_{1,1}) &= E(|C_{true}^T A C_{true} - C_{ps}^T \Pi C_{ps}|_{1,1}) \\ &\leq \frac{2c}{\epsilon_d}\sqrt{\frac{cn}{\pi}} + \sum_{i=0}^{n} 2|C_{true,i} - C_{ps,i}|_1 \mathbf{d}_i. \end{aligned} \tag{10}$$

Proof. According to the triangle inequality,

$$\begin{aligned} &E(|M_{true} - \tilde{M}|_{1,1}) = E(|C_{true}^T A C_{true} - C_{ps}^T \Pi C_{ps}|_{1,1}) \\ &\leq E(|C_{true}^T A C_{true} - C_{ps}^T A C_{ps}|_{1,1}) + E(|C_{ps}^T A C_{ps} - C_{ps}^T \Pi C_{ps}|_{1,1}). \end{aligned} \tag{11}$$

For the first part, suppose $D = C_{true} - C_{ps}$, then

$$C_{true}^T A C_{true} - C_{ps}^T A C_{ps} = D^T A C_{ps} + C_{ps}^T A D + D^T A D. \tag{12}$$

If C_{true} and C_{ps} differ only in the i-th line, i.e., only the i-th row of D is non-zero:

1. $D^T A D = 0^{c \times c}$, since there's no self-loop and $D^T A D = A_{i,i} D_i^T D_i = 0^{c \times c}$.
2. $|D^T A C|_1 = |C_{ps,i} - C_{true,i}|_1 \mathbf{d}_i$, since $|D^T S|_1 = \sum_l D_{i,l} |S_i|_1 = |D_i|_1 |S_i|_1 = |C_{ps,i} - C_{true,i}|_1 \mathbf{d}_i$, where $S = AC$.

Replace one row in C_{ps} with the corresponding row in C_{true} in sequence, where C_i represents the i-th state. By accumulating $|C_i^T A C_i - C_{i-1}^T A C_{i-1}|_1$, we have

$$|C_{true}^T A C_{true} - C_{ps}^T A C_{ps}|_1 \leq \sum_{i=1}^{n} 2 |C_{true,i} - C_{ps,i}|_1 \mathbf{d}_i. \tag{13}$$

For the second part, according to Theorem 2 and Theorem 3, we have $C_{ps}^T A C_{ps} - C_{ps}^T \Pi C_{ps} = M - \tilde{M}_{MLE}$ and $\tilde{M}_{i,j,MLE} = \frac{n_j}{n_i + n_j} \tilde{M}_{i,j} + \frac{n_i}{n_i + n_j} \tilde{M}_{j,i}$, where $\tilde{M}_{i,j}$, $\tilde{M}_{j,i}$ are two random variables drawn from $N(M_{i,j}, 2n_i \frac{1}{\epsilon_d^2})$ and $N(M_{j,i}, 2n_j \frac{1}{\epsilon_d^2})$ respectively. Therefore, we have $\tilde{M}_{i,j,MLE} \sim N(M_{i,j}, \frac{4n_i n_j}{(n_i + n_j)\epsilon_d^2})$ and $|\tilde{M}_{i,j,MLE} - M_{i,j}|_1 \sim |N(0, \frac{4n_i n_j}{(n_i + n_j)\epsilon_d^2})|_1$. Denote $\sigma = \frac{4n_i n_j}{(n_i + n_j)\epsilon_d^2}$. Through integration, we have

$$E(|\tilde{M}_{i,j,MLE} - M_{i,j}|_1) = \int_0^\infty x \frac{1}{\sqrt{2\pi}\sigma} exp\{-\frac{x^2}{2\sigma^2}\} dx = \sqrt{\frac{2}{\pi}} \frac{2}{\epsilon_d} \sqrt{\frac{n_i n_j}{n_i + n_j}}.$$

According to the Cauchy's Inequality and Harmonic-Geometric Mean Inequality,

$$\sum_{i,j \in \{1,2,\cdots,c\}} \sqrt{\frac{n_i n_j}{n_i + n_j}} \leq \sqrt{c^2} \sqrt{\sum_{i,j \in \{1,2,\cdots,c\}} \frac{n_i n_j}{n_i + n_j}} \leq \frac{\sqrt{2}}{2} c \sqrt{cn}, \tag{14}$$

since $\sum_{i=1}^{c} n_i = n$. So we have

$$E(|C_{ps}^T A C_{ps} - C_{ps}^T \Pi C_{ps}|_1) = \sum_{i,j} E(|M_{i,j} - \tilde{M}_{i,j,MLE}|_1) \leq \sqrt{\frac{1}{\pi}} \frac{2}{\epsilon_d} c \sqrt{cn}. \tag{15}$$

Combining Equation 13 and Equation 15 yields the proof.

The error in Theorem 5 can be split into two parts: random error (the first term in Equation 10) caused by noise injection, and systematic error (the second term) caused by misclassification. The random error is inversely proportional to

ϵ_d and scales as $\mathcal{O}(\sqrt{n})$. The systematic error arises from the discrepancy between true and pseudo labels. In our approach, training set nodes are assigned true labels to reduce systematic error. For unlabeled nodes, provided that the MLP achieves a reasonable accuracy—an assumption that generally holds for real-world datasets—this term won't greatly compromise the homophily constraint.

Corollary 2. *Under the same classification matrix C_{true}, difference between the two cluster connection matrix is bounded by*

$$E(|C_{true}^T A C_{true} - C_{true}^T \Pi C_{true}|) \leq \frac{2c}{\epsilon_d}\sqrt{\frac{cn}{\pi}} + \sum_{i=0}^{n} 2|C_{true,i} - C_{ps,i}|_1(\mathbf{d}_i + \tilde{\mathbf{d}}_i),$$

where $\tilde{\mathbf{d}}_i$ is the degree of node i in Π.

4.3 Precision analysis

In this section, we analyze the effects of Bayesian estimation on the precision of the estimated edges, as stated in Theorem 6.

Theorem 6. *Suppose that the probability that the prior distribution is correct is q, where $q = \Pi_{ij}$ if $A_{ij} = 1$ and $1 - \Pi_{ij}$ otherwise. ϵ_a is the privacy budget allocated for the random response mechanism, then the probability that the posterior probability $\mathcal{P}_{ij}$ is correct satisfies*

$$\lim_{\epsilon_a \to 0} E(P(\mathcal{P}_{ij} = A_{ij}|A_{ij})) = q, \quad \lim_{\epsilon_a \to +\infty} E(P(\mathcal{P}_{ij} = A_{ij}|A_{ij})) = 1, \tag{16}$$

and we have that $P(\mathcal{P}_{ij} = A_{ij}|A_{ij}))$ is monotonic increasing with $\epsilon_a > 0$.

Theorem 6 shows that with a small privacy budget, the precision of the posterior probability depends on the precision of the prior estimation. While with an abundant budget, the RR mechanism makes the estimated graph approach the actual one. This indicates that as the privacy budget increases, the benefit of a better prior diminishes. Experiments shows that our method mainly improves performance under small privacy budgets and converges to the baseline GNN performance as the budget increases, which is consistent with this conclusion..

5 EXPERIMENTS

5.1 Experimental settings

Datasets. We conduct experiments on four real-world datasets: *Cora* [24], *Amazon* [19] (including *Computers* and *Photo*), *LastFM* [16] and *Facebook* [15]. These datasets cover several representative real-world network types, and also demonstrate the scalability of our method on graphs of different scales.

Experimental setup. For all datasets and models, we randomly split the nodes into train/validation/test nodes with a ratio of 2:1:1. We selected GCN [10],

GraphSAGE [7], and GAT [20] as the primary models for testing due to their representative approaches to graph neural networks. We test the performance of all prior-based method with $\epsilon \in \{1, 2, \cdots, 8\}$. We conduct grid search on δ and report the best performance. For other frameworks, we search for their best hyper-parameters respectively. We conduct injection 5 times to generate 5 noisy graphs, and we train GNNs on each graph 5 times to reduce randomness.

5.2 Experimental results

Compare to other prior-based methods Based on the Bayesian estimation framework, we compare the performance of other graph prior construction methods with ours. We use the following methods as baselines:

1. **Blink** [27]: Blink constructs the prior graph with β-model. Given a vector $\beta = [\beta_1, \beta_2, \cdots, \beta_n] \in R^n$, an edge between node i and node j is given with probability of $\Pi_{ij} = \frac{exp(\beta_i+\beta_j)}{1+exp(\beta_i+\beta_j)}$. Blink collects the degree of each node, and estimates β with maximum likelihood estimation. This work only utilizes the degree distribution, without taking advantage of graph's homophily.
2. **HAGEI** [25]: A work contemporaneous with ours likewise proposes executing degree vector queries, and it constructs the prior graph with the Chung-Lu model: $\Pi_{ij} = \frac{d_j^{c_i} \cdot \sum_{i \in \mathbf{V}} \frac{d_i^{c_i}}{|U_{c_i}|}}{\sum_{j \in U_{c_j}} d_j^{c_i} + \sum_{i \in \mathbf{V}} d_i^{c_i}} + \frac{d_i^{c_j} \cdot \sum_{j \in \mathbf{V}} \frac{d_j^{c_j}}{|U_{c_j}|}}{\sum_{i \in U_{c_i}} d_i^{c_j} + \sum_{j \in \mathbf{V}} d_j^{c_j}}$, where $d_j^{c_i}$ represents the noisy number of edges node j connects to cluster c_i, and $|U_{c_i}|$ represents the number of nodes in cluster U_{c_i}. However, this work provides no theoretical error analysis and thus fails to bridge the gap between degree vector queries and a specific homophily metric.

After fetching the posterior link probability matrix $\mathcal{P}$, we adopt the following two methods to sample a graph for GNN trainning:

- **Hard sampling**. An edge is sampled if its posterior probability is above 0.5. All the sampled edges are regarded as undirected and unweighted.
- **Soft sampling**. The probability of each edge's existence is used as the weight of the edge. Correspondingly, we adopt dense convolution in GNN models. We didn't train the GAT model under soft sampling strategy since it is not reasonable to let all nodes attend over all others.

Results in Table 2 show that our prior graph sampling method consistently outperforms other methods, particularly with a moderate privacy budget. Our approach not only outperforms other methods under soft sampling strategies but, more importantly, within our framework, soft sampling often yields better results than hard sampling. In comparison, hard sampling leads to a sparser graph due to the truncation threshold being set to 0.5. For example, when the server collects $e_{ij} = 1$ and $e_{ji} = 0$ due to the random response mechanism (common with small privacy budgets), Equation 6 shows that $\mathcal{P}ij \geq 0.5$ if and only if $\Pi ij \geq 0.5$. However, due to the inherent sparsity of the original graph,

mlp		eps	1	2	3	4	5	6	∞
cora 73.7	gcn	hard	72.6/72.6/**73.2**	72.6/72.6/**73.2**	72.6/72.6/**74.9**	77.6/**80.5**/79.9	85.2/**86.2**/85.1	86.8/**86.9**/86.8	87.2
		soft	58.0/59.5/**74.1**	58.8/51.4/**74.3**	67.6/62.6/**76.2**	79.0/77.3/**81.1**	83.9/83.8/**84.6**	84.8/84.8/**84.9**	85.8
	sage	hard	**72.5**/72.4/72.1	**72.5**/72.4/72.3	72.5/**75.4**/72.9	77.3/**82.4**/79.8	85.6/**86.4**/85.2	**87.7**/87.5/87.5	88.0
		soft	67.6/68.5/**74.2**	67.7/68.4/**74.4**	70.2/71.0/**75.6**	77.4/78.2/**80.7**	83.4/83.4/**84.2**	**85.4**/85.3/**85.4**	86.3
	gat	hard	72.4/72.4/**72.5**	**72.4**/**72.4**/**72.4**	72.4/72.4/**73.7**	74.5/76.4/**78.9**	83.9/**84.6**/84.2	**86.0**/**86.0**/**86.0**	87.0
computers 82.7	gcn	hard	81.6/82.6/**83.4**	81.6/82.6/**83.5**	81.6/83.2/**85.1**	82.9/82.6/**86.8**	87.2/87.3/**88.3**	88.4/**88.7**/**88.7**	89.1
		soft	28.8/59.6/**81.7**	34.1/59.3/**83.1**	58.8/58.9/**85.7**	81.6/80.7/**87.2**	85.7/85.6/**87.3**	86.6/86.9/**87.4**	87.4
	sage	hard	**82.9**/82.4/82.4	82.9/82.4/**83.5**	82.9/83.6/**85.8**	85.9/85.9/**87.8**	88.5/87.9/**89.4**	89.7/89.7/**90.0**	90.2
		soft	77.8/78.9/**83.6**	78.8/78.9/**85.4**	81.3/81.6/**86.8**	85.2/85.7/**87.8**	86.9/87.3/**87.9**	87.7/87.8/**88.0**	88.1
	gat	hard	**81.1**/79.8/81.0	80.8/79.8/**82.7**	80.0/81.7/**84.5**	79.9/79.8/**86.5**	86.3/87.0/**87.9**	88.4/88.5/**88.6**	88.7
lastfm 70.7	gcn	hard	67.5/67.5/**68.0**	67.5/67.5/**69.6**	67.9/70.6/**71.6**	73.6/71.1/**76.2**	79.4/79.4/**80.5**	82.7/82.7/**82.9**	83.4
		soft	42.0/47.8/**68.2**	45.7/47.3/**68.3**	61.9/52.0/**70.4**	69.0/64.4/**72.5**	75.0/74.8/**76.1**	**78.6**/**78.6**/**78.6**	81.0
	sage	hard	**68.6**/**68.6**/68.3	69.0/68.6/**69.7**	71.4/70.4/**72.5**	75.7/77.4/**77.5**	81.2/81.8/**82.5**	84.3/84.5/**85.0**	85.3
		soft	63.6/64.5/**70.8**	63.6/64.4/**71.1**	65.7/64.1/**71.9**	70.9/69.1/**73.6**	76.1/76.0/**78.2**	79.5/79.4/**80.3**	83.3
	gat	hard	**67.9**/**67.9**/67.8	67.9/67.9/**69.0**	67.9/69.4/**70.9**	70.4/67.9/**75.6**	77.8/78.0/**80.0**	82.0/82.1/**82.6**	83.0
facebook 74.7	gcn	hard	**73.9**/**73.9**/**73.9**	73.9/73.9/**74.4**	73.9/73.9/**76.3**	76.2/74.1/**79.5**	83.4/84.5/**85.0**	88.3/**88.7**/**88.7**	90.1
		soft	57.4/**76.3**/75.0	57.9/**76.5**/75.5	71.6/76.5/**77.9**	78.7/76.4/**82.0**	84.3/83.7/**85.5**	87.3/87.4/**87.7**	88.9
	sage	hard	74.2/74.2/**74.3**	74.1/74.2/**75.1**	**76.3**/74.2/**76.3**	78.7/74.7/**79.8**	84.0/**85.8**/85.3	88.6/**89.2**/89.0	91.0
		soft	74.5/73.7/**75.0**	74.6/74.9/**75.4**	76.6/77.4/**77.8**	80.0/80.5/**81.7**	84.9/84.7/**86.0**	88.4/88.3/**88.7**	90.1
	gat	hard	**73.8**/73.5/**73.8**	**73.8**/73.5/**73.8**	73.8/73.5/**75.4**	73.8/73.5/**78.7**	80.9/82.3/**84.0**	87.6/88.0/**88.1**	90.2

Table 2: Node classification accuracy comparison between different prior graph construction methods (Blink/HAGEI/HPGR(ours)) under various datasets, GNN models and privacy budgets. ∞ represents GNN performance without adopting privacy protection policies, and mlp represents model performance without utilizing graph topology. We bold the best method and underline the second best method for each setting. Values in the table are presented in the percentage form.

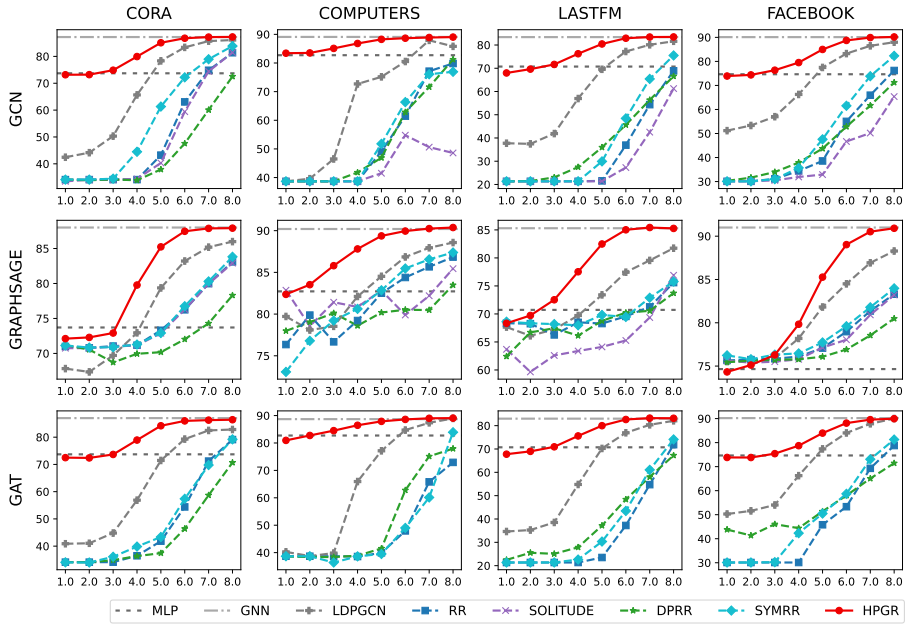

Fig. 2: Performance comparison between HPGR(ours) and other methods. X-axis represents privacy budget ϵ and y-axis represents test accuracy (%).

the prior probability Π_{ij} is often smaller than 0.5 even if an edge exists. This causes hard sampling to truncate values, which reduces noise but sacrifices fine-grained structural details. Excellent performance under soft sampling strategy suggests that our approach effectively utilizes detailed topological information while minimizing the impact of noise on downstream tasks.

Compare to other methods We also compare our method with other disparate locally differential private GNN frameworks, and they are:

1. **Random Response(RR)**: The server employs the RR mechanism to collect adjacency information without applying any additional denoising operations.
2. **Symmetric Random Response(SymRR)**: The server only collect the lower triangular adjacency matrix with RR, ensuring that the reconstructed graph remains undirected.
3. **Degree-Preserving Random Response(DPRR)** [8]: DPRR performs an unbiased estimation of node degrees in the original topology based on the observed degrees in the collected topology. Subsequently, it randomly samples an equal number of collected edges for each node, corresponding to its estimated degree, to reconstruct the corrected topology.
4. **Locally Differential Private GNN(LDPGNN)**: LDPGNN, a variant of DPGNN proposed in [18], utilizes the Laplace mechanism to collect the

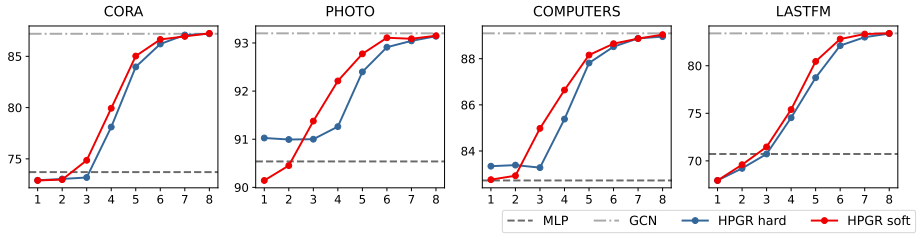

Fig. 3: Comparison between soft and hard queries (instead of sampling strategies). X-axis represents privacy budget and y-axis represents test accuracy(%).

private topology. The server then estimates the unbiased number of edges in the entire graph and retains the corresponding number of highest-weighted entries in the collected topology.

5. **Solitude** [13]: Solitude formulates the sparsity of the collected topology as an optimization objective for GNN training. Its optimization goal is given by $\min_{\hat{A},\theta} \mathcal{L}(\hat{A}|\theta) + \lambda_1|\hat{A} - \tilde{A}|_F + \lambda_2|\hat{A}|_1$.

The experimental results in Figure 2 show that our method outperforms baseline approaches. When the privacy budget is limited, both the random response (RR) and Laplace (Lap) mechanisms significantly distort the graph's topology, making it challenging for GNNs to extract meaningful signals from graphs with a low signal-to-noise ratio. As a result, methods relying on the RR or Lap mechanisms (such as RR, DPRR, SYMRR, and LDPGCN) exhibit poor performance. However, despite the differential privacy mechanisms' ability to obscure the presence or absence of individual edges, relatively accurate statistical information can still be derived. By leveraging homophily statistics, we are able to preserve node connectivity within and across classes. In homophilic graphs, the prior distribution encodes the knowledge that "nodes of the same class are more likely to be connected", enabling the GNN to extract low-frequency signals from the graph.

Comparison between hard and soft queries We analyze the difference between the proposed soft and hard query mechanisms from Section 3.1, with results shown in Figure 3. In most cases, the soft query outperforms the hard query, particularly when the privacy budget is relatively sufficient, as it reduces bias from MLP misclassification by accounting for classification uncertainty. However, the soft query requires transmitting an $n \times c$ classification probability matrix instead of an n-sized classification vector, increasing the communication burden. This represents a trade-off between resource consumption and performance.

6 RELATED WORK

Differentially Private GNNs. Differentially private GNNs can be divided into two categories based on the goal of privacy protection: global differentially private GNNs and local differentially private GNNs.

Global differentially private GNNs aim to train node embedding vectors or labels privately at the sever side. GAP [17] achieves edge-level and node-level differential privacy by adding noise during the convolution process. DPDGC [3] improves upon the node-level differential privacy definition proposed in [17], achieving better performance by injecting noise into the graph topology and node features individually. DPGCN [4] extends the DP-SGD method [1] to the GNN domain, achieving node-level differential privacy.

Local differentially private GNNs focus on protecting node features and graph topology at the client side. LDPGNN [18] focuses on protecting node features and labels, while LDP-GE [11] focuses on protecting only node features. Solitude [13], DPRR [8], and Blink [27] aim at protecting the graph topology, and these methods have been introduced in detail in Section 5.

7 Conclusion

In this paper, we introduce HPGR, a method that effectively leverages homophily to enhance the performance of GNNs under LDP. HPGR not only advances the state of the art in privacy-preserving graph learning but also highlights the critical importance of integrating domain-specific structural insights—such as homophily—into privacy mechanisms. This work opens new avenues for developing secure, high-utility graph-based machine learning applications in decentralized and privacy-sensitive environments.

8 Acknowledgment

This work was supported by NSF China under Grant No. T2421002, 62202299, 62020106005, 62061146002.

References

1. Abadi, M., Chu, A., Goodfellow, I., McMahan, H.B., Mironov, I., Talwar, K., Zhang, L.: Deep learning with differential privacy. In: CCS (2016)
2. Briar Project: Briar: Secure messaging, anywhere. https://briarproject.org/, 2024
3. Chien, E., Chen, W.N., Pan, C., Li, P., Ozgur, A.: Differentially private decoupled graph convolutions for multigranular topology protection. In: NeurIPS (2024)
4. Daigavane, A., Madan, G., Sinha, A., Thakurta, A.G., Aggarwal, G., Jain, P.: Node-level differentially private graph neural networks. arXiv:2111.15521 (2021)
5. Duchi, J.C., Jordan, M.I., Wainwright, M.J.: Local privacy and statistical minimax rates. In: FOCS (2013)
6. Dwork, C.: Differential privacy. In: International colloquium on automata, languages, and programming (2006)
7. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: NeurIPS (2017)
8. Hidano, S., Murakami, T.: Degree-preserving randomized response for graph neural networks under local differential privacy. arXiv:2202.10209 (2022)

9. Karrer, B., Newman, M.E.: Stochastic blockmodels and community structure in networks. Phys. Rev. E (2011)
10. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: ICLR (2017)
11. Li, Z., Li, R.H., Liao, M., Jin, F., Wang, G.: Locally differentially private graph embedding. arXiv:2310.11060 (2023)
12. Lim, D., Hohne, F., Li, X., Huang, S.L., Gupta, V., Bhalerao, O., Lim, S.N.: Large scale learning on non-homophilous graphs: New benchmarks and strong simple methods. In: NeurIPS (2021)
13. Lin, W., Li, B., Wang, C.: Towards private learning on decentralized graphs with local differential privacy. TIFS (2022)
14. Moyer12, D., Gutman, B., Prasad, G., Ver Steeg, G.: Mixed membership stochastic blockmodels for the human connectome. neuro-degenerative diseases (2015)
15. Rozemberczki, B., Allen, C., Sarkar, R.: Multi-scale attributed node embedding. Journal of Complex Networks (2021)
16. Rozemberczki, B., Sarkar, R.: Characteristic functions on graphs: Birds of a feather, from statistical descriptors to parametric models. In: CIKM (2020)
17. Sajadmanesh, S., Shamsabadi, A., Bellet, A., Gatica-Perez, D.: Gap: Differentially private graph neural networks with aggregation perturbation. In: Security (2023)
18. Sajadmanesh, S., Gatica-Perez, D.: Locally private graph neural networks. In: CCS (2021)
19. Shchur, O., Mumme, M., Bojchevski, A., Günnemann, S.: Pitfalls of graph neural network evaluation. arXiv:1811.05868 (2018)
20. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. In: ICLR (2018)
21. Voigt, P., Von dem Bussche, A.: The eu general data protection regulation (gdpr). A Practical Guide, 1st Ed., Cham: Springer International Publishing (2017)
22. Warner, S.L.: Randomized response: A survey technique for eliminating evasive answer bias. Journal of the American statistical association (1965)
23. Wu, K., Xie, Y., Ding, J., Ren, Y., Fu, L., Wang, X., Zhou, C.: Characterizing the influence of topology on graph learning tasks. In: DASFAA (2024)
24. Yang, Z., Cohen, W., Salakhudinov, R.: Revisiting semi-supervised learning with graph embeddings. In: ICML (2016)
25. Zhang, G., Cheng, X., Pan, J., Lin, Z., He, Z.: Locally differentially private graph learning on decentralized social graph. Knowledge-Based Systems (2024)
26. Zhu, J., Yan, Y., Zhao, L., Heimann, M., Akoglu, L., Koutra, D.: Beyond homophily in graph neural networks: Current limitations and effective designs. In: NeurIPS (2020)
27. Zhu, X., Tan, V.Y., Xiao, X.: Blink: Link local differential privacy in graph neural networks via bayesian estimation. In: CCS (2023)