Unveiling Privacy Vulnerabilities: Investigating the Role of Structure in Graph Data

Hanyang Yuan∗
Zhejiang University
Hangzhou, China
yuanhy0408@zju.edu.cn

Jiarong Xu†
Fudan University
Shanghai, China
jiarongxu@fudan.edu.cn

Cong Wang
Peking University
Beijing, China
wangcong@gsm.pku.edu.cn

Ziqi Yang
Zhejiang University
Hangzhou, China
yangziqi@zju.edu.cn

Chunping Wang
Finvolution Group
Shanghai, China
wangchunping02@xinye.com

Keting Yin
Zhejiang University
Hangzhou, China
yinkt@zju.edu.cn

Yang Yang
Zhejiang University
Hangzhou, China
yangya@zju.edu.cn

ABSTRACT
The public sharing of user information opens the door for adversaries to infer private data, leading to privacy breaches and facilitating malicious activities. While numerous studies have concentrated on privacy leakage via public user attributes, the threats associated with the exposure of user relationships, particularly through network structure, are often neglected. This study aims to fill this critical gap by advancing the understanding and protection against privacy risks emanating from network structure, moving beyond direct connections with neighbors to include the broader implications of indirect network structural patterns. To achieve this, we first investigate the problem of Graph Privacy Leakage via Structure (GPS), and introduce a novel measure, the Generalized Homophily Ratio, to quantify the various mechanisms contributing to privacy breach risks in GPS. Based on this insight, we develop a novel graph private attribute inference attack, which acts as a pivotal tool for evaluating the potential for privacy leakage through network structures under worst-case scenarios. To protect users’ private data from such vulnerabilities, we propose a graph data publishing method incorporating a learnable graph sampling technique, effectively transforming the original graph into a privacy-preserving version.

Extensive experiments demonstrate that our attack model poses a significant threat to user privacy, and our graph data publishing method successfully achieves the optimal privacy-utility trade-off compared to baselines.

CCS CONCEPTS
• Security and privacy → Social network security and privacy.

KEYWORDS
Graph privacy protection, data release, adversarial learning

ACM Reference Format:

1 INTRODUCTION
In the era of big data, with the increasing involvement of personal data in information technology and shared on the web, privacy protection has emerged as a crucial concern [21, 24, 41]. In real-world scenarios, individuals often share some information publicly while safeguarding their private attributes. However, publicly available user information gives adversaries opportunities to infer private attributes, resulting in privacy breaches [40, 42, 44]. Furthermore, the inferred private data can facilitate malicious activities. For instance, in the 2010s, Cambridge Analytica collected personal data from 87 million Facebook users to infer their political stands, which were further used for political advertising, resulting in a scandal with over $100 billion in economic losses [20]. This incident emphasizes the urgent need for privacy protection mechanisms.

To prevent the exposure of user privacy, traditional works typically focus on the privacy leakage through users’ public attributes [21,
we aim to develop a graph data publishing method aimed at comprehensively protecting against potential privacy breaches arising from the network structure. Nevertheless, achieving this goal presents several challenges.

The first challenge lies in how to measure the extent of privacy exposure through network structures. Previous research mainly focuses on privacy leakage through direct connections between nodes, using homophily to quantify proximity-related exposure [23, 35, 49]. However, this approach falls short in assessing privacy risks from structure-role information. In this study, we introduce the Generalized Homophily Ratio (GHRatio), a novel measure to quantify privacy risks associated with network structures. The GHRatio is a general form that is adaptable to various structural features.

We explore two prevalent cases—proximity homophily, structure-role homophily and their combination—that contribute to privacy leakage.

The second challenge stems from the necessity to develop a private attribute inference attack model that utilizes proximity homophily, structure-role homophily, and their combination to launch attacks. Given that existing attack strategies merely exploit proximity homophily [3, 22, 40], suboptimal results are yielded (as depicted by the red bar in Figure 1 (b)). To overcome this challenge, our model is designed to account for all identified privacy breaches through a data-centric strategy. This strategy involves providing a graph neural network (GNN) with various data forms, thus enhancing its capacity to learn from different types of homophily. Consequently, our attack model effectively behaves like a worst-case adversary, as evidenced by the green bar in Figure 1 (b).

The last challenge lies in how to design a graph data publishing approach that can effectively defend the worst-case private attribute inference attack. Previous efforts in graph data publishing have primarily focused on differential privacy (DP) [57, 60] and graph sampling [5, 17], but DP often compromises the utility of the data [38]. In addition, many sampling methods are rule-based and reliant on domain-specific knowledge [37, 38], which restricts their applicability. We therefore propose a learnable graph sampling method for privacy protection, employing a generative network that selectively samples edges to block privacy leakage (as illustrated by × in Figure 1 (a)). This method ultimately produces a sampled graph suitable for publication.

Our contributions are summarized as follows:

- **Problem and measure**: Our work pioneers a comprehensive investigation into the problem of Graph Privacy leakage via Structure (GPS), introducing the innovative Generalized Homophily Ratio (GHRatio) as a measure of privacy leakage. This helps us unveil all identified mechanisms by which the network structure can lead to privacy breaches in a quantitative manner.

- **Attack model**: We introduce a novel private attribute inference attack leveraging a data-centric strategy to exploit all identified privacy breaches. By feeding a GNN various data forms, it gains the ability to learn from multiple homophily types that result in privacy risks.

- **Defensive model**: To counter the attacks, we propose a graph data publishing method that employs learnable graph sampling, rendering the sampled graph suitable for publication.

- **Extensive experiments**: Experiments in five real-world scenarios demonstrate that (1) our private attribute inference attack beats the best baselines by an average of +2.93%, and (2) our graph data publishing approach achieves the optimal privacy-utility trade-off, outperforming existing defensive methods when evaluated against worst-case attack scenarios.

## 2 PROBLEM DEFINITION

Network structure is a significant factor contributing to privacy breaches, which stands as a fundamental data source for various network analysis tasks, such as node classification and community detection. Previous research has proven that modifying network
structure is more effective than modifying node attributes in enhancing privacy protection [17]. Therefore, this work particularly focuses on a privacy-preserving graph data publishing problem with a specific emphasis on the network structure.

In this study, we refer to privacy as a particular attribute that nodes choose to keep hidden, which aligns with previous works [17, 28]. Let graph \( G = (V, E, X) \) denote an undirected network, where \( V = \{v_1, \ldots, v_n\} \) is the node set, \( E \subseteq V \times V \) is the edge set, and \( X \in \mathbb{R}^{n \times m} \) is the node attribute matrix. \( A \in \mathbb{R}^{n \times n} \) is the adjacency matrix of \( G \), where \( A_{ij} = 1 \) if there exists an edge \((i, j) \in E\), otherwise \( A_{ij} = 0 \). Each node \( v_i \in V \) is associated with a known/unknown private attribute \( Z_i \). Here, \( Z_i \in Z = Z_L \cup Z_U \), where \( Z_L \) denotes publicly available private attributes, and \( Z_U \) represents hidden private attributes. In this context, privacy in the graph \( G \) is defined by the set of hidden private attributes \( Z_U \).

Let us consider the following attack scenario. In a public social network, some users choose to conceal their private attributes, while others make them public. The adversary aims to infer these hidden private attributes. The adversary is assumed to have access to the network structure, node attributes (typically non-private), and publicly available private attributes. Publicly available private attributes can come from users who do not consider this information private or who seek to maximize visibility by sharing extensive personal information.

Formally, we define this as graph private attribute inference attack problem.

**Problem 1 (Graph Private Attribute Inference Attack).** Given graph \( G = (V, E, X) \) and the publicly available private attributes \( Z_L \), the graph private attribute inference attack aims to learn a function

\[
    f : G, Z_L \rightarrow Z_U, \tag{1}
\]

that predicts the hidden private attribute \( Z_U \).

The primary objective of this work is to tackle the problem of privacy-preserving graph data publishing against the aforementioned graph private attribute inference attack. Specifically, instead of directly releasing the original graph, the data publisher is encouraged to generate a sampled graph for publishing, such that the sampled graph can defend against graph private attribute inference attack. Based on the above definition, we formulate our privacy-preserving graph data publishing problem as follows.

**Problem 2 (Privacy-Preserving Graph Data Publishing).** Given graph \( G = (V, E, X) \) and the publicly available private attributes \( Z_L \), the data publisher aims to sample a new graph \( G' = (V, E', X) \) by selectively removing edges in \( E \), resulting in a new edge set \( E' \). The sampled graph \( G' \) is expected to simultaneously achieve the following two objectives:

**Objective 1: privacy preservation.** The adversary with \( G' \) and \( Z_L \) cannot accurately infer the private attribute \( Z_U \), i.e.,

\[
    \min_{G'} \text{ perf } (f(G'), Z_U), \tag{2}
\]

where perf denotes a performance metric to evaluate how well the predicted value \( f(G', Z_L) \) aligns with the ground truth \( Z_U \), such as accuracy or ROC-AUC as used in our work.
Proximity-based direct connection

Info. propagated via

Learning proximity homophily

N(\( \delta \))

Given the general form of if \( 1 \) exceeds a certain threshold, and 0 otherwise; when considering r as the relation of adjacency, \( \delta(v_i,v_j,r) = 1 \) if node \( v_i \) and \( v_j \) are directly connected, and 0 otherwise.

Based on the homophily indicator, we can define the GHRatio as follows.

**Definition 1 (Homophily Indicator).** A homophily indicator \( \delta(v_i,v_j,r) \) assesses whether node \( v_i \) and \( v_j \) exhibit a shared structural characteristic or relation \( r \). For example, when \( r \) signifies similar local structure, we have \( \delta(v_i,v_j,r) = 1 \) if the similarity of \( v_i \) and \( v_j \) exceeds a certain threshold, and 0 otherwise; when considering \( r \) as the relation of adjacency, \( \delta(v_i,v_j,r) = 1 \) if node \( v_i \) and \( v_j \) are directly connected, and 0 otherwise.

**Two prevalent cases of GHRatio.** Given the general form of GHRatio, we further delve into two prevalent cases of it: proximity homophily, structure-role homophily, by defining specific graph structural feature associated with GHRatio.

1. **Proximity homophily ratio.** By defining the \( r \) in homophily indicator \( \delta(v_i,v_j,r) \) as the relation of adjacency, we have \( \delta(v_i,v_j,r) = 1 \) if \( v_i \) and \( v_j \) are connected, and 0 otherwise; We name this specific case of GHRatio as Proximity Homophily Ratio (GHRatio\( \text{prox} \)).

   \[
   \text{GHRatio}_{\text{prox}} = P(Z_i = Z_j | \delta(v_i,v_j,r) = 1), \quad j \neq i, \quad (3)
   \]

   where \( Z_i \) and \( Z_j \) are private attributes of node \( v_i \) and \( v_j \), respectively.

2. **Structure-role homophily ratio.** We define \( r \) in the homophily indicator \( \delta(v_i,v_j,r) \) as the similar local structure. Then, we have \( \delta(v_i,v_j,r) = 1 \) if the similarity between the local structures of \( v_i \) and \( v_j \) exceeds a certain threshold, and 0 otherwise. We name this case of GHRatio as Structure-Role Homophily Ratio (GHRatio\( \text{role} \)).

   \[
   \text{GHRatio}_{\text{role}} = \frac{|\{j \in N(i) \land Z_j = Z_i\}|}{|\{j \in N(i)\}|}, \quad (4)
   \]

   where \( N(i) \) denotes \( v_i \)’s neighborhood, and \( |\cdot| \) denotes the cardinality of a set. In fact, GHRatio\( \text{prox} \) aligns with the node-level homophily ratio defined in existing works [35, 49].

**Generalized Homophily Ratio.** As our goal is to investigate how network structure discloses privacy, we begin by defining homophily indicator, which can be used to characterize the structural characteristic or relation between two nodes.

**Definition 2 (Generalized Homophily Ratio (GHRatio)).** Given an observed graph \( G \) and private attribute matrix \( Z \), the GHRatio of node \( v_i \) is defined as the conditional probability that node \( v_i \) and any node \( v_j \) (\( j \neq i \)) have the same private attributes, given that \( \delta(v_i,v_j,r) = 1 \), i.e.,

   \[
   \text{GHRatio}_i = P(Z_i = Z_j | \delta(v_i,v_j,r) = 1), \quad j \neq i, \quad (3)
   \]

   where \( Z_i \) and \( Z_j \) are private attributes of node \( v_i \) and \( v_j \), respectively.

Two prevalent forms of GHRatio. Given the general form of GHRatio, we further delve into two prevalent forms of it: proximity homophily, structure-role homophily, by defining specific graph structural feature associated with GHRatio.

1. **Proximity homophily ratio.** By defining the \( r \) in homophily indicator \( \delta(v_i,v_j,r) \) as the relation of adjacency, we have \( \delta(v_i,v_j,r) = 1 \) if \( v_i \) and \( v_j \) are connected, and 0 otherwise; We name this specific case of GHRatio as Proximity Homophily Ratio (GHRatio\( \text{prox} \)).

   \[
   \text{GHRatio}_{\text{prox}} = \frac{|\{j \in N(i) \land Z_j = Z_i\}|}{|\{j \in N(i)\}|}, \quad (4)
   \]

   where \( N(i) \) denotes \( v_i \)’s neighborhood, and \( |\cdot| \) denotes the cardinality of a set. In fact, GHRatio\( \text{prox} \) aligns with the node-level homophily ratio defined in existing works [35, 49].

2. **Structure-role homophily ratio.** We define \( r \) in the homophily indicator \( \delta(v_i,v_j,r) \) as the similar local structure. Then, we have \( \delta(v_i,v_j,r) = 1 \) if the similarity between the local structures of \( v_i \) and \( v_j \) exceeds a certain threshold, and 0 otherwise. We name this case of GHRatio as Structure-Role Homophily Ratio (GHRatio\( \text{role} \)).

   \[
   \text{GHRatio}_{\text{role}} = \frac{|\{j \in N(i) \land g(i) \sim g(j) \land Z_j = Z_i\}|}{|\{j \in N(i) \land g(i) \sim g(j)\}|}, \quad (5)
   \]

   where \( g(i) \) is the ego network of node \( v_i \), \( g(i) \sim g(j) \) denotes that the ego networks \( g(i) \) and \( g(j) \) are sufficiently similar (e.g., this similarity can be understood in terms of structural similarity, specifically when the structural similarity between \( g(i) \) and \( g(j) \) exceeds a predetermined threshold). Empirically, we adopt the degree centrality to characterize the similarity between ego networks, which have been validated for its strong effectiveness and efficiency in previous studies [45, 58].

**4 PRIVATE ATTRIBUTE INFERENCE ATTACK**

This section introduces a novel private attribute inference attack model that leverages proximity homophily, structure-role homophily and their combination to disclose private information. We adopt a data-centric approach, feeding varied data forms into a GNN to extract the representations related to different homophily types (see § 4.1). Then, a routing operator is introduced for the adaptive integration of these homophily-related representations (see § 4.2). See Figure 4 (a) for an overview of our proposed attribute inference attack.

**4.1 Enhancing GNNs with Data-Centric Strategy**

Developing a GNN model capable of learning representations tailored to different homophily types is challenging. Existing works predominantly learn representations based on proximity information [15, 26] or high-order node dependencies [1, 61], which can not adequately learn structure-role information. Although some network representation methods [2, 16, 45] are designed to learn from node structure roles, their expressive power is limited.

In this work, we introduce a data-centric strategy designed to enhance GNNs’ capacity to learn representations tailored to proximity homophily and structure-role homophily. This is achieved by feeding different forms of data into GNNs. An illustrative example is provided in Figure 3. Our key insight is:

1. Feeding the entire graph to a GNN enables it to learn proximity homophily;
2. Feeding the set of nodes’ subgraphs (e.g., ego networks) to a GNN, where each node’s representation is computed as its subgraph’s representation, facilitates the learning of structure-role homophily.

**Learning proximity homophily from the entire graph.** This principle aligns with the message-passing mechanism in standard
GNNs, where nodes update their representations by aggregating information from their direct neighbors [26, 53]. This aggregation process naturally encourages similarity in the representations of connected nodes [53]. Specifically, we generate proximity-based representations \( H^{\text{prox}} \in \mathbb{R}^{n \times d} \) by feeding the entire graph to a GNN encoder \( \text{GNN}^{\text{prox}} \), i.e.,

\[
H_i^{\text{prox}} = \text{GNN}^{\text{prox}}(G).
\]  

**Learning structure-role homophily from subgraphs.** This approach first extracts \( k \)-hop ego networks for each node, resulting in a set of subgraphs. These subgraphs are subsequently fed to a GNN encoder. The resulting subgraph representation, which encapsulates the aggregated features of its constituent nodes, serves as the representation for the subgraph’s central node.

This benefits the GNN’s ability to learn structure-role homophily for two main reasons: First, by concentrating on subgraphs, the central node’s representation becomes exclusively reflective of its local structural context, isolating the central node from external influences of nodes outside the subgraph. Secondly, GNNs are particularly adept at learning structural information of smaller subgraphs. Although GNNs can identify structural patterns, comprehending complex structures in larger graphs remains challenging, as observed in [19]. Ultimately, this process allows the GNN to bring the representations of nodes with similar local structures closer in the latent space.

Specifically, we process the \( k \)-hop ego network of node \( i \), denoted as \( S_i = (V_i, A_i, X_i) \), and feed the subgraph to a GNN encoder \( \text{GNN}^{\text{role}} \). The node representation \( H_i^{\text{role}} \) is computed as the subgraph representation, which is the mean of the representations of all nodes in the subgraph:

\[
H_i^{\text{role}} = \text{Pooling}(\text{GNN}^{\text{role}}(S_i)),
\]

where Pooling is the mean pooling operator.

**Theoretical analysis of learning from subgraphs.** While capturing proximity homophily from the entire graph is widely acknowledged [25, 53, 56], few works explore learning structure-role homophily from subgraphs. We therefore theoretically investigate whether GNNs fed with subgraphs can learn structure-role homophily. The following theorem suggests that nodes with similar local structures can obtain similar node representations.

**Theorem 1.** Let \( S_i \) and \( S_j \) be two \( k \)-hop subgraphs induced from node \( v_i \) and \( v_j \). After employing a \( K \)-layer GNN encoder with a \( 1 \)-hop graph filter \( \Psi(L) \) on each subgraph, the representations of the center node \( v_i \) and \( v_j \) are obtained via a pooling function, i.e., \( H_i^{\text{role}} = \text{Pooling}(\text{GNN}^{\text{role}}(S_i)) \) and \( H_j^{\text{role}} = \text{Pooling}(\text{GNN}^{\text{role}}(S_j)) \). Without loss of generality, assume that the attribute of each node is a vector of ones, \( H_i^{\text{role}} \) and \( H_j^{\text{role}} \) satisfy:

\[
\|H_i^{\text{role}} - H_j^{\text{role}}\|_2 \leq \tau\|L_i - L_j\|_2,
\]

where \( \| \cdot \|_2 \) denotes \( L_2 \) norm of matrix or vector, \( \tau \) denotes a constant depending on \( \text{GNN}^{\text{role}} \), \( L_i \) denotes the normalised Laplacian matrix of \( S_i \).

In Theorem 1, the term \( \|L_i - L_j\|_2 \) measures the difference of local structure around \( v_i \) and \( v_j \). As similar local structures bring smaller differences in Laplacian matrices, the upper bound of node representation distance is reduced. Consequently, nodes with similar local structures become closer in the latent space. The proof of this theorem can be found in Appendix A.2.

**4.2 Routing Operator**

After obtaining the two types of representations, the challenge arises in determining the optimal method to integrate them, especially given the uncertainty about the extent of information that should be merged from each type. To tackle this issue, we introduce a *routing operator* that leverages our proposed GHRatio to effectively combine these representations.

Since GHRatio\(^{\text{prox}}\) and GHRatio\(^{\text{role}}\) serve to quantify the extent to which proximity and structural role disclose privacy, respectively. Utilizing these ratios, we can integrate the two types of representations, aiming for a balanced and informed combination that reflects the significance of both proximity and structural information in revealing privacy. However, calculating these ratios requires the knowledge of the private attributes of all nodes, which presents another difficulty. To address this, our approach involves estimating the ratios by employing the pseudo-labels of private attributes.

Formally, we apply Multilayer Perceptrons (MLPs) to obtain inference results from proximity-based representation \( H^{\text{prox}} \) and structure role-based representation \( H^{\text{role}} \). Then we use the estimated GHRatios as the proportions to integrate them. The integrated result \( \hat{Z}_i \) in turn serves as the pseudo-labels. Due to the interdependence between pseudo-labels and GHRatios, we initialize GHRatios with constants and iteratively update them during training. The above process can be described as follows:

\[
\hat{Z}_i^{\text{prox}} = \text{MLP}(H_i^{\text{prox}}), \quad \hat{Z}_i^{\text{role}} = \text{MLP}(H_i^{\text{role}}), \\
\hat{Z}_i = \text{GHRatio}_i^{\text{prox}} \cdot \hat{Z}_i^{\text{prox}} + \text{GHRatio}_i^{\text{role}} \cdot \hat{Z}_i^{\text{role}},
\]

where \( \hat{Z}_i \) denotes the final inference result for node \( v_i \).

The model is optimized as follows:

\[
\min_\Theta L_{\text{adv}} = \frac{1}{|V_1|} \sum_{v_i \in V_1} \sum_{c=1}^{C} \sum_{i \in c} Z_{i,c} \log \hat{Z}_{i,c},
\]

where \( C \) denotes the number of categories for the private attribute, \( V_1 \) denotes the nodes with known private attributes, and \( \Theta \) denotes the parameters of the attack model.
5 PRIVACY-PRESERVING GRAPH DATA PUBLISHING

In this section, we introduce a learnable graph sampling approach for privacy-preserving graph data publishing. We first outline the learnable strategy for graph sampling in § 5.1. Following this, the optimization objectives and training algorithm are detailed in § 5.2 and § 5.3, respectively. Figure 4 (b) provides an overview of the proposed graph data publishing method.

5.1 Learnable Graph Sampling

To generate a sampled graph $G'$ with adjacency matrix $A'$, we first feed the graph $G$ into a GNN encoder to obtain node representation $H^\text{samp}$:

$$H^\text{samp} = \text{GNN}(G).$$  \hspace{1cm} (10)

Given any connected node pair $[v_i, v_j]$, an MLP with sigmoid activation takes the concatenation of their node representations $H_i^\text{samp}$ and $H_j^\text{samp}$ as input to compute the probability $\mathcal{T}_{ij}$ of preserving the edge between $[v_i, v_j]$:

$$\mathcal{T}_{ij} = \sigma(\text{MLP}(H_i^\text{samp} \oplus H_j^\text{samp})), \hspace{1cm} (11)$$

where $\sigma$ denotes sigmoid activation, and $\Phi$ represent the concatenation operation. With an edge’s $\mathcal{T}_{ij}$ calculated, we sample whether to retain the edge in the synthetic graph $G'$ according to this probability, where

$$A'_ij = \text{Bernoulli}(\mathcal{T}_{ij}).$$  \hspace{1cm} (12)

In particular, the Gumbel-Softmax reparameterization trick [36] is utilized to tackle the non-differentiable nature of the sampling process. In doing so, we obtain a continuous sampling result, i.e.,

$$A'_ij = \sigma((\log U - \log (1 - U) + \log \mathcal{T}_{ij})/\epsilon),$$

where $U \sim \text{Uniform}(0, 1)$. As the temperature hyper-parameter $\epsilon$ tends to zero, the reparameterized result smoothly converges to binary values, while maintaining the relative order of each Gumbel [36].

5.2 Optimization Problem

We propose three optimization objectives for training learnable parameters within the graph sampling procedure: one aimed at defending against worst-case attacks, one designed for a broader spectrum of attacks, and another dedicated to preserving essential graph properties. These objectives collectively ensure that the sampled synthetic graph maintains user privacy while simultaneously achieving desirable data utility.

Defending Against Worst-Case Attack. Given the proposed inference attack, the most straightforward and effective approach is to defend against this attack under the worst-case. Specifically, to obtain the worst-case attack, we maximize the performance of the attack model, and subsequently, we defend against such an attack. This can be formulated as

$$\min_{\Phi} \max_{\Theta} -L_{\text{adv}}(G'(\Phi), \Theta), \hspace{1cm} (13)$$

where $\Phi$ and $\Theta$ denote the parameters of the sampling component and the attack model respectively. $\Phi$ determines $G'$ by influencing $A'$.

Defending Against a Broad Spectrum of Attacks via GHRatio. Eq. (13) ensures that our publishing method can defend against the proposed worst-case attack. However, in real-world scenarios, the released graph may face various attacks, and not all of them necessarily reach the worst case [34, 43]. In such scenarios, we devise a universal protection strategy via GHRatio, which serves as a measure independent of a specific attack.

Essentially, GHRatio quantifies how much information the network structure can disclose for inferring the private attribute. By minimizing GHRatio, we can mitigate the risk of privacy leakage in an attack-agnostic manner:

$$\min_{\Phi} L_{\text{dis}} = \frac{1}{|V|} \sum_{v_i \in V} \| \text{GHRatio}_1(\Phi) - \text{GHRatio}_0 \|_1, \hspace{1cm} (14)$$

where $\text{GHRatio}_1(\Phi)$ represents the new GHRatio of the sampled graph $G'$. $\text{GHRatio}_0$ represents $P(Z_i = Z_j)$, indicating the probability that node $v_i$ and any node $v_j (j \neq i)$ have the same private attribute. When $L_{\text{dis}} = 0$, the structural characteristic in GHRatio provides no benefits for attribute inference.

Note that since $Z_i$ is given, we have $\text{GHRatio}_0^0 = P(Z_i)$. In practice, we propose to optimize the two prevalent cases of GHRatio, GHRatio$^{\text{prox}}$ and GHRatio$^{\text{role}}$, being described as:

$$\min_{\Phi} L_{\text{dis}} = \frac{1}{|V|} \sum_{v_i \in V} \| \text{GHRatio}_1^{\text{prox}}(\Phi) - \hat{P}(Z_i) \|_1 + \| \text{GHRatio}_1^{\text{role}}(\Phi) - \hat{P}(Z_i) \|_1, \hspace{1cm} (15)$$

where $\hat{P}(Z_i)$ represents the empirical estimation of $P(Z_i)$.

Regularization for Ensuring Utility. To ensure the utility of the sampled graph, we aim to align the properties of $G'$ with those of $G$. Thus, we incorporate a reconstruction-based regularization term to control the deviation of the sampled graph:

$$\min_{\Phi} L_{\text{reg}} = -\frac{1}{|E|} \sum_{(i,j) \in E} A_{ij} \log(T_{ij}), \hspace{1cm} (16)$$

where $T_{ij}$ denotes the sampling probability of edge $(i, j)$. By Eq. (16), $G'$ will retain as many edges as possible from $G$.

To sum up, we formalize the overall optimization problem:

$$\min_{\Phi} \max_{\Theta} L = -\gamma \cdot L_{\text{adv}} + \eta \cdot L_{\text{dis}} + \lambda \cdot L_{\text{reg}}, \hspace{1cm} (17)$$

where $\gamma, \eta, \lambda \geq 0$ are hyper-parameters.

For different scenarios, we can also modify Eq. (17) to obtain different variants. If the goal is to protect against the proposed worst-case attack, only retaining $L_{\text{adv}}$ and $L_{\text{reg}}$ would be sufficient. On the other hand, if the goal is not specifically for the worst case but to be effective against a broad range of attacks, retaining $L_{\text{dis}}$ and $L_{\text{reg}}$ is suitable.

5.3 Training Algorithm

In the training phase, the parameters $\Phi$ of the sampling component and the parameters $\Theta$ of the proposed attack model are jointly trained. Specifically, The training algorithm iterates through the following main steps: (1) Learn $\Phi$ to minimize $-L_{\text{adv}}, L_{\text{reg}}$ and $L_{\text{dis}}$ while keeping $\Theta$ fixed, and (2) Learn $\Theta$ to maximize $-L_{\text{adv}}$ while keeping $\Phi$ fixed. Repeat these steps until the maximum iteration is reached. The detailed training algorithm and complexity analysis are summarized in Appendix A.3.
6 EXPERIMENTS

In this section, we evaluate the effectiveness of both the proposed attack model and the defensive model of data publishing.

6.1 Experiment Setting

Datasets. We conduct attribute inference and data publishing experiments on three datasets: Pokec-n, Pokec-z, and NBA [8, 33]. Following prior works [8], we treat country as the private attribute in NBA, and region as the private attribute in Pokec-n and Pokec-z. Additionally, we also treat users’ age as another private attribute in Pokec-n and Pokec-z, categorizing it according to the split in [18]: Young (18-24), Young-Adult (25-34), Middle-aged (35-49), and Senior (> 49). All private attributes are randomly split, with 10% publicly available and the remaining 90% hidden. In the experiments of data publishing, we also consider salary as the label in NBA and working field as the label in Pokec-n and Pokec-z [8, 33]. We conduct node classification on these labels as downstream tasks (using a training-testing split of 0.1:0.9), and assess the utility of published graphs by evaluating the performance of these tasks. The statistics of the three datasets are summarized in Table 6.

Implementation details. All experiments are conducted on a machine of Ubuntu 20.04 system with AMD EPYC 7763 (756GB memory) and NVIDIA RTX3090 GPU (24GB memory). All models are implemented in PyTorch version 2.0.1 with CUDA version 11.8 and Python 3.8.0. Each experiment is repeated 5 times to report the average performance with standard deviation.

For the attack model, the encoder GNNprox and GNNrole are implemented by two 2-layer GIN [54] encoders, with the same model architecture. The hidden dimensions are set to 128 in Pokec-z, Pokec-n, and 64 in NBA. The two MLPs are both implemented by 1-layer linear transformations. The model is trained by AdamW optimizer with a learning rate of 0.001 for 300 epochs in Pokec-n, Pokec-z, and 500 epochs in NBA. For the defensive model, the sampling component consists of a two-layer Graphsage [15] encoder and a two-layer MLP. The hidden dimension of the Graphsage encoder is 64, and the hidden dimension of the MLP is 32 in all datasets. The model is trained by AdamW optimizer with a learning rate of 0.002 for 200 epochs in Pokec-n, Pokec-z, and 100 epochs in NBA. The weight decay is consistently set as 0.0005. Both models use ReLU as the non-linear activation function. For hyper-parameters settings. We perform a grid search of the degree similarity threshold (see Eq. 5) in [0.20] with a step size of 5 in all datasets. We set λ to 1, and vary γ and η (see Eq. 17) within [0.10, 0.90] in NBA and [0.20] in Pokec-z and Pokec-n, with a step size of 5. Our codes are available at https://github.com/zjunet/GPS_KDD.

6.2 Experiments on Private Attribute Inference

Baselines. We compare with the following attack models, which are divided into three types: (1) MLP: multilayer perceptions; (2) GCN [26], GAT [50], GraphSAGE, Mixhop [1] and H2GCN [61]: three foundational GNNs and two heterogeneous GNNs, used as comparisons to evaluate the effectiveness of the proposed attack model in capturing privacy leakage from both proximity homophily and structure-role homophily; (3) AttriInfer [22], ComInfer [40], AI-N2V, AI-DW [13]: four methods designed for private inference on graphs. AttriInfer and ComInfer are non-deep learning models, based on Markov random fields and community detection respectively. AI-N2V and AI-DW are representation-based models that utilize Node2Vec and DeepWalk to obtain node representations.

Comparison results. For evaluation, Table 1 presents the performance results of the proposed attack model in comparison with other baseline methods on the five aforementioned private attributes. The results reveal several key insights: Firstly, the notable performance drop observed in the MLP model empirically demonstrates the crucial role of GPS. Secondly, AI-N2V and AI-DW exhibit lower performance, possibly due to their limited ability to simultaneously capture both proximity and structure-role information in these representation methods. Additionally, the performance of AttriInfer and ComInfer is weaker than that of the GNN models. The reason may be attributed to the more powerful expressive capabilities of the latter models. Thirdly, the proposed attack model outperforms other baseline methods on all datasets, highlighting its remarkable efficacy in capturing GPS from both structure-role homophily and proximity homophily.

Ablation study. We conduct ablation studies to demonstrate the efficacy of each component within our attack model, including three variants: (1) Ours-prox: the inference results are solely obtained by aggregating on the entire graph; (2) Ours-role: the results are solely obtained by aggregating within each subgraph. (3) Ours-equal: dropping the GHratios and combining the two prediction results in a 0:0.5 ratio. Table 2 shows the results on the five private attributes. Notably, the complete model consistently surpasses the performance of all variants, showing the effectiveness and necessity of simultaneously leveraging proximity homophily and structure-role homophily to capture privacy leakage.

6.3 Experiments on Data Publishing

Table 1: Accuracy (age) and ROC-AUC (rest) of graph private attribute inference, where Ours denotes the proposed attack model. The best results are bolded.

<table>
<thead>
<tr>
<th></th>
<th>Pokec-n</th>
<th>Pokec-z</th>
<th>NBA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age</td>
<td>Region</td>
<td>Age</td>
<td>Region</td>
</tr>
<tr>
<td>MLP</td>
<td>54.01 (3.42)</td>
<td>58.61 (1.36)</td>
<td>56.34 (3.53)</td>
</tr>
<tr>
<td>GCN</td>
<td>65.37 (1.03)</td>
<td>81.34 (0.52)</td>
<td>66.53 (1.28)</td>
</tr>
<tr>
<td>GAT</td>
<td>64.20 (1.44)</td>
<td>77.37 (2.28)</td>
<td>65.09 (2.01)</td>
</tr>
<tr>
<td>SAGE</td>
<td>65.29 (1.04)</td>
<td>81.15 (0.85)</td>
<td>67.14 (0.79)</td>
</tr>
<tr>
<td>MixHop</td>
<td>64.46 (0.50)</td>
<td>85.93 (0.87)</td>
<td>66.94 (0.38)</td>
</tr>
<tr>
<td>H2GCN</td>
<td>63.13 (0.27)</td>
<td>80.26 (1.34)</td>
<td>65.78 (1.61)</td>
</tr>
<tr>
<td>AttriInfer</td>
<td>64.83 (0.11)</td>
<td>63.85 (0.53)</td>
<td>63.18 (0.39)</td>
</tr>
<tr>
<td>ComInfer</td>
<td>34.18 (0.59)</td>
<td>62.35 (0.29)</td>
<td>39.64 (0.81)</td>
</tr>
<tr>
<td>Ours</td>
<td>61.13 (0.52)</td>
<td>72.12 (0.42)</td>
<td>59.42 (0.76)</td>
</tr>
<tr>
<td>AI-DW</td>
<td>62.25 (0.69)</td>
<td>75.30 (2.18)</td>
<td>58.81 (0.19)</td>
</tr>
<tr>
<td>Ours-prox</td>
<td>66.69 (1.03)</td>
<td>89.39 (0.35)</td>
<td>68.80 (0.89)</td>
</tr>
</tbody>
</table>
Utility
We select the best three trade-off points for each method and visualize them in Figure 5. The upper-left corner represents the ideal performance. The lower-right corner presents higher attack performance. Note that we also report the privacy preservation performance by evaluating the attack metric, comparing the degree distribution and clustering coefficient distribution of the original graph with those of the published graph of each method. Specifically, the properties of the published graph should closely resemble those of the original graph. Therefore, we employ the Maximum Mean Discrepancy (MMD) distance as our evaluation metric, which measures the extent to which the published graph deviates from the original graph. To ensure fairness, we tune the hyperparameters of these models to achieve comparable results in terms of privacy-preserving performance. The MMD scores in Table 3 (the smaller, the better) demonstrate that our model outperforms the other methods in terms of privacy leakage from both structure-role homophily and proximity homophily, thereby achieving less optimal trade-offs. In addition, it presents higher computational complexity.

Evaluation of graph property change. To evaluate from a broader perspective, we characterize the utility by measuring the extent to which the sampled graph deviates from the original graph. Specifically, the properties of the published graph should closely resemble those of the original graph. Therefore, we employ the Maximum Mean Discrepancy (MMD) distance as our evaluation metric, comparing the degree distribution and clustering coefficient distribution of the original graph with those of the published network under our model and baselines. To ensure fairness, we tune the hyperparameters of these models to achieve comparable results in terms of privacy-preserving performance. The MMD scores in Table 3 (the smaller, the better) demonstrate that our model outperforms the other methods in terms of privacy leakage from both structure-role homophily and proximity homophily, thereby achieving less optimal trade-offs. In addition, it presents higher computational complexity.

Table 3: MMD distance of the degree distributions (d.d) and clustering coefficient distribution (c.c) between the original and the published graph of each method.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Ours-prox</th>
<th>Ours-role</th>
<th>Ours-equal</th>
<th>RABV</th>
<th>NetFense</th>
<th>Betw.</th>
<th>Deg.</th>
<th>Eigen</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pokec-n Age</td>
<td>1.50</td>
<td>1.40</td>
<td>1.35</td>
<td>1.35</td>
<td>1.56</td>
<td>1.40</td>
<td>1.55</td>
<td>1.40</td>
</tr>
<tr>
<td>Pokec-n Region</td>
<td>1.50</td>
<td>1.40</td>
<td>1.35</td>
<td>1.35</td>
<td>1.56</td>
<td>1.40</td>
<td>1.55</td>
<td>1.40</td>
</tr>
<tr>
<td>NBA Age</td>
<td>1.50</td>
<td>1.40</td>
<td>1.35</td>
<td>1.35</td>
<td>1.56</td>
<td>1.40</td>
<td>1.55</td>
<td>1.40</td>
</tr>
<tr>
<td>NBA Region</td>
<td>1.50</td>
<td>1.40</td>
<td>1.35</td>
<td>1.35</td>
<td>1.56</td>
<td>1.40</td>
<td>1.55</td>
<td>1.40</td>
</tr>
<tr>
<td>Country</td>
<td>1.50</td>
<td>1.40</td>
<td>1.35</td>
<td>1.35</td>
<td>1.56</td>
<td>1.40</td>
<td>1.55</td>
<td>1.40</td>
</tr>
</tbody>
</table>

Pokec-z Age. In contrast, methods such as Rand., Deg., and Eigen do not take the private attribute into account during perturbation, thus compromising data utility. RABV exhibits suboptimal privacy preservation effects when introducing additional noise to the network structure. NetFense fails to adequately capture privacy leakage from both structure-role homophily and proximity homophily, thereby achieving less optimal trade-offs. In addition, it presents higher computational complexity.
whether the defensive model can perform effectively against other attack models, we aim to assess its transferability. In other words, we evaluate methods in protecting against various attack models, demonstrating its outstanding transferability.

Evaluation of transferability. As our defensive model adopts the proposed inference as the worst-case adversary during training, we aim to assess its transferability. In other words, we evaluate whether the defensive model can perform effectively against other attribute inference attack models. Table 4 reports the performance of our defensive model compared with other baselines on Pokec-n. The results show that our defensive model outperforms other methods in protecting against various attack models, demonstrating its outstanding transferability.

7 RELATED WORK

Private attribute inference. Early approaches to attribute inference have primarily focused on using user-individual public attributes. These attributes include profile labels [14, 29], textual content [42, 44], and location information from users’ public posts [27]. These approaches heavily rely on the correlation between public and hidden private attributes to build inference models. Despite their demonstrated effectiveness, these methods often overlook valuable information from the connections between users, resulting in a noticeable performance decline.

Subsequent explorations of attribute inference leverage network structure [13, 22, 40, 59] and involve the utilization of graph propagation algorithms, such as GCN [52] and MRF [22], to facilitate the propagation of information across connected nodes. These models aggregate information from adjacent nodes [22, 59], leverage community structures [40], or random walk [13] to infer private attributes. While they underscore the exploitation of proximity homophily [3], they often overlook the other crucial aspect of structure-role homophily, thereby achieving less than optimal performance.

Privacy-preserving learning on graph. Privacy-preserving techniques are crucial to graph data publishing, among which anonymization [11, 60], sampling-based [17], model training-based [28, 51] and differential privacy [6, 10, 31, 46, 57, 62] defense methods have been proposed. The anonymization methods [11, 60] face constraints due to the need to mitigate operational complexities and often compromise privacy and utility for efficiency. The sampling-based method [17] proposes an edge perturbation technique to defend against GNN-based inference on binary private attributes. However, it fails to consider privacy leaks from structural information and has high computational complexity. Regarding model training-based defense methods [28, 51], they often fall short when dealing with complex scenarios that require direct processing of graph data. In addition, differential privacy (DP) is a common privacy protection technique. Early efforts [6, 55, 62] extend DP to correlated settings, where data records are assumed to be correlated with each other (e.g., network structure). They primarily rely on noise injection for privacy preservation. In contrast, our method systematically addresses the attack-defense problem by considering the complex relationships and structural patterns encompassed in graph data. Recently, DP-DGAE [31] perturbs the objective function of graph auto-encoders to prevent attackers from re-identifying nodes. Local DP [7] allows individuals to locally perturb their graph metrics, such as node degree and adjacency list before aggregation to mitigate the risk of privacy leakage [10, 46, 57]. Striking a balance between utility and privacy remains a challenge for them in graph data publication. Note that our method differs from DP in two key aspects: first, the determination of the edge sampling probability in DP is established according to predetermined mechanisms with respect to network structure. In contrast, our method learns the sampling probability based on the risks associated with GPS. Second, DP aims to preserve membership privacy, that is, altering one sample (e.g., node or edge) doesn’t significantly change the output distribution, while the privacy we investigate in this work pertains to attribute-wise privacy.

8 CONCLUSION

In this work, we delve into the problem of GPS and uncover the underlying mechanisms, including structure-role homophily, proximity homophily, and their intricate interplay. Based on this understanding, we introduce a novel data-centric approach for graph private attribute inference, capable of capturing privacy leaks from these mechanisms. Serving as the worst-case adversary, this method provides a comprehensive evaluation of potential privacy risks. To combat GPS, we propose a learnable graph sampling model for privacy-preserving data publishing. Our model enhances privacy security by learning the risks associated with each edge in GPS. Extensive experiments validate the effectiveness of our attack method and demonstrate the advantageous balance achieved by our defensive model between privacy preservation and utility retention.

ACKNOWLEDGMENTS

This work was supported in part by NSFC (62232606, 62271059, 62322606, 62441605, 72101007), SMP-IDATA Open Youth Fund, CCF-Tencent Rhino-Bird Open Research Fund, Joint Funds of Zhejiang Provincial NSFC (LHZSD24F020001), Zhejiang Province “LingYan” Research and Development Plan Project (2024C01114), and Zhejiang Province High-Level Talents Special Support Program “Leading Talent of Technological Innovation of Ten-Thousands Talents Program” (2022R52046).
Table 5: Major notations used in this work.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$G, V, E$</td>
<td>Graph, node set, and edge set</td>
</tr>
<tr>
<td>$X, A$</td>
<td>Attribute matrix and adjacency matrix</td>
</tr>
<tr>
<td>$Z_L, Z_U$</td>
<td>Known/hidden private attributes</td>
</tr>
<tr>
<td>$G', E'$</td>
<td>Sampled graph and sampled edge set</td>
</tr>
<tr>
<td>$\delta()$</td>
<td>Homophily indicator</td>
</tr>
<tr>
<td>$\mathcal{N}(), g()$</td>
<td>Neighborhood and ego network</td>
</tr>
<tr>
<td>$\text{GNN}^{\text{prox}}, \text{GNN}^{\text{role}}$</td>
<td>GNN encoders</td>
</tr>
<tr>
<td>$H^{\text{prox}}, H^{\text{role}}$</td>
<td>Node representations</td>
</tr>
<tr>
<td>$S_i, L_i$</td>
<td>Induced subgraph and Laplacian matrix</td>
</tr>
<tr>
<td>$\Psi()$</td>
<td>1-hop graph filter</td>
</tr>
<tr>
<td>$\Theta, \Phi, \eta, \lambda, \gamma$</td>
<td>Model parameters</td>
</tr>
<tr>
<td>$\tilde{Z}$</td>
<td>Inferred private attributes</td>
</tr>
<tr>
<td>$C$</td>
<td>Number of private attribute’s classes</td>
</tr>
<tr>
<td>$T_{i,j}, A'_{i,j}$</td>
<td>Sampling probability, sampled adjacency</td>
</tr>
<tr>
<td>$P_i(), P()$</td>
<td>Probability and its empirical estimation</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>Sigmoid function</td>
</tr>
<tr>
<td>$\oplus$</td>
<td>Concatenation operation</td>
</tr>
<tr>
<td>$\otimes$</td>
<td>Homophily indicator</td>
</tr>
</tbody>
</table>


A APPENDIX

A.1 Notations

To facilitate clarity in our presentation, Table 5 summarizes major notations used in this work.

A.2 Proof of Theorem 1

**Lemma 1.** For any $A \in \mathbb{R}^{n \times d}$, let $a_l \in \mathbb{R}^{1 \times d}$ denote the i-th row of $A$. We have

$$\|a_l\|_2 \leq \|A\|_2, \forall l = 1, ..., n$$

**(Proof.** Let $e_i = [0, ..., 1, ..., 0]^T$ be a vector of zero, except for the i-th element. By the definition of matrix norm, we have $\|A^T e_i\|_2 \leq \|A^T\|_2 \|e_i\|_2 = \|A\|_2$. Then we have $\|a_l\|_2 = \|A^T e_i\|_2 \leq \|A^T\|_2 \|e_i\|_2 = \|A\|_2$. □

We now give the proof of Theorem 1.

**Proof.** Without loss of generality, we prove Theorem 1 by instantiating the encoder as a $K$-layer GCN encoder and a 1-hop graph filter $\Psi(L) = I_d - L$. For simplicity, we denote the i-th layer’s node representation in $S_I$ as $H^I_S$, which is obtained as

$$H^I_S = \sigma(\Psi(L)H^{I-1}_S W^I)$$

where $\sigma$ denotes a $\tau_l$-Lipschitz activation function, $W^I \in \mathbb{R}^{d \times d}$ denotes the learnable parameters in the $I$-th layer.

Assume that $S_I$ and $S_J$ have the same number of nodes, and $\max_l \|H^I_{S_I}\|_2 \leq \tau_I$ and $\max_l \|W^I\|_2 \leq \tau_w$. Then $\forall l = 1, ..., K$, we have

$$\|H^I_{S_I} - H^I_{S_J}\|_2 \leq \|\sigma(\Psi(L)H^{I-1}_S W^I) - \sigma(\Psi(L)H^{I-1}_S W^I)\|_2

\leq \tau_l \|\Psi(L)\|_2 \|H^{I-1}_S W^I - H^{I-1}_S W^I\|_2

\leq \tau_l \|\Psi(L)\|_2 \|H^{I-1}_S - H^{I-1}_S\|_2 + \|\Psi(L)\|_2 \|W^I\|_2

\leq \tau_l \|\Psi(L)\|_2 \|H^{I-1}_S - H^{I-1}_S\|_2 + \tau_w \|\Psi(L)\|_2 \|W^I\|_2

\leq \|\Psi(L)\|_2 \|H^{I-1}_S - H^{I-1}_S\|_2 + \tau_w \|\Psi(L)\|_2 \|W^I\|_2

\leq \|\Psi(L)\|_2 \|H^{I-1}_S - H^{I-1}_S\|_2 + \tau_w \|\Psi(L)\|_2 \|W^I\|_2$$

The above equation can be rewritten as $R_I \leq a + bR_{I-1}$, then we have

$$R_I \leq a + bR_{I-1} \leq a(b + 1) + b^2 R_{I-2} \leq \cdots \leq b^I - 1 \leq a + b^IR_0$$

where $a = \tau_l \|\Psi(L)\|_2 \|W^I\|_2$ and $b = \|\Psi(L)\|_2 \|H^{I-1}_S - H^{I-1}_S\|_2 + \tau_w \|\Psi(L)\|_2 \|W^I\|_2$. □
Let \( l = K \) and let \( ||H_{S_l} - H_{S_j}||_2 = ||H^K_{S_l} - H^K_{S_j}||_2 \) denote the representation difference in the last layer, we have
\[
||H_{S_l} - H_{S_j}||_2 \leq \frac{(\tau_s \tau_w)^K - 1}{\tau_s \tau_w} ||\Psi(L_l) - \Psi(L_j)||_2 \\
+ (\tau_s \tau_w)^K ||\Psi(L_j)||_2 ||X_i - X_j||_2 
\]
(22)
Since we assume that the attribute of each node is a vector of ones, we have ||\(X_i - X_j||_2 = 0\). Since the graph Laplacians are normalized, we assume \( min_i ||\Psi(L_l)||_2 \leq \tau_l \). Thus
\[
||H_{S_l} - H_{S_j}||_2 \leq \frac{(\tau_s \tau_w)^K - 1}{\tau_s \tau_w} ||\Psi(L_l) - \Psi(L_j)||_2 \\
\leq \tau ||L_l - L_j||_2 
\]
(23)
where \( \tau = \frac{(\tau_s \tau_w)^K - 1}{\tau_s \tau_w} \).

By a pooling function of mean, we obtain the final center node representation \( H^i_{role} \), \( H^j_{role} \) of \( S_i, S_j \). From Lemma 1, we have:
\[
||H^i_{role} - H^j_{role}||_2 \leq \frac{1}{n} \sum_{t=1}^{r_n} ||H_{S_{i,t}} - H_{S_{j,t}}||_2 \leq \tau ||L_l - L_j||_2 
\]
(24)
which completes the proof. \( \square \)

A.3 Training Algorithm and Complexity Analysis

Algorithm 1 Learnable graph sampling method

**Input:** Graph \( G = (V, E, X) \), available private attributes \( Z_t \), number of steps \( t \), update interval \( l \), and \( \gamma, \eta, \lambda \).

**Output:** The sampled graph \( G' = (V, E', X) \).

1. Initialize GHRatio\textsuperscript{prox} and GHRatio\textsuperscript{role} for all nodes as 0.5.
2. for epoch \( e = 1, 2, \ldots \) do
3. Calculate the edge sampling probability by Eq. (11).
4. Calculate the training loss by Eq. (13), Eq. (15) and Eq. (16).
5. Update the parameters \( \Phi \) of the attack model by maximizing Eq. (17).
6. Update the parameters \( \Phi \) of the sampling component by minimizing Eq. (17).
7. if \( e \% l = 0 \) then
8. Update GHRatio\textsuperscript{prox} and GHRatio\textsuperscript{role} by Eq. (4) and (5).
9. end if
10. end for
11. Use the learned edge sampling probability to obtain \( G' \).

(1) Preprocessing: In this phase, we extract the subgraphs of all nodes. Let graph \( G = (V, E, X) \), the complexity of extraction is \( O(nd^K) \), where \( k \) is the number of hops, \( n \) is the number of nodes, \( d \) is the average degree of nodes. This step is computed only once.

(2) Sampling: In this phase, we compute the sampling probability for each edge and perform graph sampling. The complexity of obtaining node representations for each node through Graph-SAGE is \( O(nKt) \), where \( r \) is the number of sampled neighbors, \( K \) is the number of layers, and \( t \) is the number of iterations. Here, we omit the time complexity of matrix operations. Then, obtaining edge sampling probability and performing sampling has a complexity of \( O(m) \), where \( m \) represents the number of edges. Therefore, the total complexity of this step is \( O(rnKt + m) \).

(3) Inference: In this phase, we perform attribute inference on the sampled graph. Firstly, the time complexity of obtaining predictions using GIN on the entire graph is \( O(nKt) \). To mitigate the time overhead caused by repeated subgraph extraction, we store the global masks of corresponding edges in each node’s \( k \)-hop subgraph during the preprocessing step. After each sampling step, we only need to determine which edges in the \( k \)-hop subgraph of each node are retained based on these masks. And using GIN on the subgraph for prediction has a time complexity of \( O(nd^K) \). Thus, the overall complexity of this step is \( O(nKt) \).

(4) Loss calculation: the complexities of computing \( L_{adv} \), \( L_{dis} \), and \( L_{reg} \) are \( O(n) \), \( O((d + \bar{b})n) \), and \( O(m) \). Where \( d \) denotes the average degree (number of neighbors) for a node, and \( \bar{b} \) denotes the average number of nodes with a similar degree to a given node. The overall complexity of this step is \( O((d + \bar{b})n + m) \).

A.4 Datasets statistics
We provide the statistics of the three used datasets in this work.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Pokec-n</th>
<th>Pokec-z</th>
<th>NBA</th>
</tr>
</thead>
<tbody>
<tr>
<td># nodes</td>
<td>66,569</td>
<td>67,797</td>
<td>403</td>
</tr>
<tr>
<td># node attr.</td>
<td>59</td>
<td>59</td>
<td>39</td>
</tr>
<tr>
<td># edges</td>
<td>729,129</td>
<td>882,765</td>
<td>16,570</td>
</tr>
<tr>
<td>Private attr.</td>
<td>region/age</td>
<td>region/age</td>
<td>country</td>
</tr>
<tr>
<td>Label</td>
<td>working field</td>
<td>working field</td>
<td>salary</td>
</tr>
</tbody>
</table>