PriGraph: Defending Against Inference Attacks on Graph Neural Networks via Policy-Based Adversarial Perturbations

Meng Shen, *Member, IEEE*, Hao Lu, Aijing Gu, Qi Li, *Senior Member, IEEE*, Ke Xu, *Fellow, IEEE*, and Liehuang Zhu, *Senior Member, IEEE*

***Abstract*—Graph-structured data have been widely used in modeling various systems, such as social networks and transportation networks. Leveraging the structure and properties of graphs, Graph Neural Networks (GNNs) have been proposed to perform in the graph domain. Previous research has shown that GNNs are vulnerable to inference attacks, which aim to infer sensitive information of training graph data, e.g., node membership and link status. Node Membership Inference Attacks (NMIAs) on GNNs infer whether a set of graph data records belongs to the training graph data of a target model. Link Status Inference Attacks (LSIAs) against GNNs aim to infer whether there exists a link between a pair of nodes in the graph used to train the target GNN model. Specifically, given black-box access to a GNN model, NMIAs and LSIAs are conducted by analyzing the outputs (e.g., confidence score vectors) from GNN models. The defense methods against these two score-based inference attacks face the challenges of achieving effective defense performance and maintaining the utility of GNN models. In this paper, we propose PriGraph, a defense mechanism to protect the node privacy and link privacy of training graph data, while maintaining the high accuracy of the target GNN models. PriGraph adds crafted adversarial perturbations to outputs of the target GNN model by deploying two key components, i.e., defense auxiliary classifier and adversarial perturbation generator, which are used to find the minimal adversarial perturbations that can reduce the attack accuracy while maintaining task performance of node classification. We evaluate PriGraph with different GNN models and multiple benchmark datasets. The results show that PriGraph can dramatically reduce the attack accuracy of NMIA and LSIA on GNNs, providing a superior trade-off between the model utility and privacy.**

***Index Terms*—Graph neural networks (GNNs), node membership inference attacks (NMIA), link status inference attacks (LSIAs), adversarial perturbations.**

Received 10 April 2023; revised 7 April 2025; accepted 10 September 2025. Date of publication 19 September 2025; date of current version 14 January 2026. This work was supported in part by the National Key R&D Program of China under Grant 2023YFB2703800, in part by NSFC Projects under Grant U23A20304 and Grant 62222201, and in part by Beijing Natural Science Foundation under Grant M23020. *(Corresponding author: Meng Shen.)*

Meng Shen is with the School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing 100081, China, and also with the State Key Laboratory of Cryptology, Beijing 100878, China (e-mail: shenmeng@bit.edu.cn).

Hao Lu, Aijing Gu, and Liehuang Zhu are with the School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing 100081, China (e-mail: haolu@bit.edu.cn; aijinggu@bit.edu.cn; liehuangz@bit.edu.cn).

Qi Li is with the Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing 100084, China (e-mail: qli01@tsinghua.edu.cn).

Ke Xu is with the Department of Computer Science, Tsinghua University, Beijing 100084, China (e-mail: xuke@tsinghua.edu.cn).

Digital Object Identifier 10.1109/TDSC.2025.3612195

I. Introduction

GRAPH structure data have widely existed in the real world, e.g., social networks [1], molecular networks [2] and traffic analysis [3]. Leveraging links (or edges), a graph describes the relationships between a set of entities represented by nodes (or vertexes). Graph Neural Networks (GNNs) are proposed as emerging Machine Learning (ML) models to operate on graph-structured data for a wide range of tasks, which can be categorized as node-level [4], link-level [5] and graph-level [6] tasks. Owing to their outstanding performances, GNNs have been employed in various applications including recommendation systems [7], fraud detection [8], and drug analysis [6].

Despite the prosperity of GNN applications, recent research indicates that GNNs are vulnerable to inference attacks which can lead to the leakage of private information about graph data. The inference attacks on GNNs can be conducted with black-box settings. The attacker is assumed to have access to the querying examples of the target models and the corresponding answers, which are score vectors (i.e., posterior distributions). According to the prediction results, inference attacks can infer the sensitive information of nodes or links over the label with the score vectors of the querying node.

Node Membership Inference Attacks (NMIA) and Link Status Inference Attacks (LSIA) are two typical inference attacks on GNNs with the invasion of node privacy and link privacy, respectively. As illustrated in Fig. 1, the attacker can conduct NMIA and LSIA by querying nodes of graph data from GNN models. NMIA is conducted to determine whether a target node is used in the training process of the target GNN, which can expose the node privacy of specific communities [9], [10], [11]. Different from NMIA, the attacker conducts LSIA [12], [13] to predict missing links in a graph that creates potential privacy risks, such as revealing an individual's political views, sexual orientation, or social circles. These privacy risks can lead to unintentional harm to users and society, such as discrimination or persecution of individuals, and undermining public trust. Therefore, We focus on defending against these two inference attacks for the privacy preservation of nodes and links.

The existing methods to protect the privacy of GNNs can be broadly categorized into three groups: Differential Privacy (DP) approaches [14], Federated Graph Learning [15], and Adversarial Privacy-Preserving methods [16]. DP approaches

1545-5971 © 2025 IEEE. All rights reserved, including rights for text and data mining, and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on November 15,2026 at 03:30:39 UTC from IEEE Xplore. Restrictions apply.

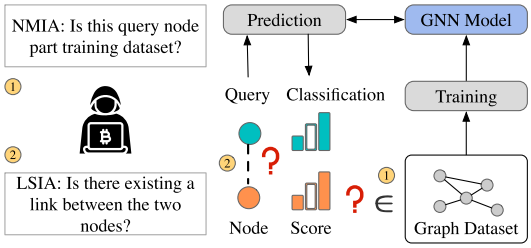


Fig. 1. The node membership inference attacks and link status inference attacks against the GNN model.

can provide strict theoretical proof for privacy protection for node privacy by adding controlled noise to the input the output of GNN models. However, DP approaches may compromise the utility of the target GNN model due to DP noises. Federated Graph Learning involves training the GNN models on distributed data while keeping the graph data locally on individual devices, thereby reducing the risk of privacy breaches. Federated Graph Learning requires sufficient graph data of participants to be effective, but the current commercial GNN models are released by a third party with the whole graph data training in a centralized manner and the inference attacks can be also conducted in this scenario [17]. Adversarial Privacy-Preserving methods [16] aim to mitigate privacy threats by training the model with adversarial examples. The current Adversarial Privacy-Preserving methods only focus on protecting the sensitive attributes of graph nodes and neglect adequate defense against NMIA and LISA.

Defending against inference attacks on GNNs faces the challenges that achieving defense performance against both NMIA and LISA and providing the utility of GNN models. Federated Graph Learning can be challenging to implement due to the heterogeneity of the participants and graph data. DP approaches can reduce the accuracy of the model by adding noise to the output of GNN models. Adversarial Privacy-Preserving methods may not be effective against all types of inference attacks, which require specialized expertise for defending NMIA and LISA. In graph modeling tasks, the overfitting of GNN models is the major reason for the success of NMIA models and LSIA models are conducted based on evaluating the distance of scores by querying two nodes from GNN models. We observe that both the score-based NMIA and LSIA models are vulnerable to adversarial perturbations added to the outputs of the GNN model, which can inspire an effective defense against these two inference attacks.

In this paper, we propose PriGraph, a defense mechanism via policy-based adversarial perturbations, which aims to defend against NMIA and LSIA to preserve the privacy of node membership and link status on GNNs while keeping the utility of GNN models. PriGraph provides three types of *privacy-preference-policy* for the user which depends on whether the user intends to put emphasis on protecting node membership privacy, link status privacy, or both. Adapting different privacy policies, PriGraph can generate adversarial perturbations to defend against corresponding inference attacks.

PriGraph contains two components, namely *defense auxiliary classifier* and *adversarial perturbation generator*. The defense auxiliary classifier serves as a local attacker to conduct the NMIA or LSIA on the target model, using full knowledge of the target model and its dataset to infer the privacy of the target model. Based on the feedback of the defense auxiliary classifier and the target model, the adversarial perturbation generator constructs the optimal perturbations that are adaptively applied to the outputs of the target model to preserve privacy while maintaining the classification accuracy. Thus, PriGraph employs a GAN-like model in which the defense auxiliary classifier attempts to infer private information from the perturbed output of the target model, while the adversarial perturbation generator perturbs the outputs of the target model to reduce the inference accuracy.

We conduct a comprehensive evaluation of PriGraph on four real-world datasets and three GNN models and compare it with the existing defenses. The experimental results show that PriGraph can effectively defend against black-box NMIA and LSIA on GNNs while guaranteeing the model's utility. For model utility, the loss of the test accuracy of the model trained with PriGraph incurs no more than a 5% drop compared to that trained without defense. For node membership and link status privacy preservation, PriGraph has achieved the lowest accuracy of NMIA and LISA compared to the existing defenses, significantly improving privacy on GNNs.

In summary, our contributions are as follows:

- We propose PrivGarph, a new defense scheme to defend against both NMIA and LSIA on GNN models under the black-box setting. It is built based on a key observation that adversarial perturbations can be added to the outputs of the GNN model to mitigate both NMIA and LSIA.
- We design a defense auxiliary classifier and adversarial perturbation generator to generate the adversarial perturbations. The defense auxiliary classifier assists the adversarial perturbation generator to find the optimal noises defending against attacks while keeping high classification accuracy. And the crafted adversarial perturbations are added to the outputs of the target model without tampering with the training process.
- We evaluate the performance of PriGraph on four real-world datasets and three GNN models. The experimental results demonstrate that it can defend against the NMIA and LSIA effectively and outperforms existing defenses.

II. Preliminaries

In this section, we present the background of GNNs and the notations used in this paper.

A. Notations

We use $\mathcal{D} = (\mathcal{G}, \mathcal{X}, \mathcal{Y})$ to denote a graph dataset, where $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ represents a graph with $\mathcal{V}$ and $\mathcal{E}$ respectively denoting the set of $\mathcal{N}$ nodes of this graph and links connecting them. For graph $\mathcal{G}$, the node attributes are defined as $\mathcal{X} = \{x_1, x_2, \ldots, x_{\mathcal{N}}\}$, where $x_i \in \mathbb{R}^d$ corresponds to the d-dimensional attributes of node v_i. And $\mathcal{Y} = \{y_1, y_2, \ldots, y_{\mathcal{N}}\}$ represent the predicted score vectors and labels of the set of nodes in graph $\mathcal{G}$, respectively. The adjacency matrix of garph $\mathcal{G}$ is denoted as

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on November 15,2026 at 03:30:39 UTC from IEEE Xplore. Restrictions apply.

TABLE I
THE MAIN NOTATIONS USED IN THIS PAPER

Notations	Descriptions
$\mathcal{D}_t$	Target dataset
$\mathcal{G}$	Graph data
$\mathcal{A}$	Adjacency matrix of graph $\mathcal{G}$
F	Target model
F^*	The defense model
C	Defense auxiliary classifier
G	Adversarial perturbation generator
$\mathcal{D}'$	Shadow dataset
F'	Shadow model
P	Node membership inference attack model
Q	Link status inference attack model
s_v	v's score from the target model
$s_v^{'}$	v's score from the defense model
n^*	Adversarial perturbations

$\mathcal{A} \in \mathbb{R}^{N\times N}$, if there is a link between nodes v_i and v_j, $\mathcal{A}_{ij} = 1$; otherwise, $\mathcal{A}_{ij} = 0$. Table I shows the main notations used in this paper.

B. Graph Neural Networks

To effectively exploit the graph structure data, Graph Neural Networks have been proposed to address graph-related tasks. Generally, GNNs adopt the message-passing mechanism to learn node representations, i.e., the representation of a node learned via aggregating its features and the information from its k-hop neighbors. Formally, the l-th layer of a GNN model can be defined as:

$$z_v^{(l)} = \text{AGGREGATE}\left(\left\{h_u^{(l-1)} : u \in \mathcal{N}(v)\right\}\right)$$
$$h_v^{(l)} = \text{UPDATE}\left(z_v^{(l)}\right) \quad (1)$$

where $\mathcal{N}(v)$ reprensents the neighborhood of node v. $z_v^{(l)}$ is the hidden state of node v and $h_v^{(l)}$ stands for its representation after the l-th GNN layer. $\text{AGGREGATE}(\cdot)$ and $\text{UPDATE}(\cdot)$ are the aggregation and update funcation, respecrively. At the beginning, the representation $h_v^{(0)}$ is initialized as x_v which is the attribute of this node. For different graph tasks, the representation h can be applied for predictions (e.g., node classification, graph classification, and link prediction). For node classification, the softmax layer is applied at the final layer L for the prediction of node v as follows:

$$y_i = \text{softmax}\left(z_i^{(L)}\mathcal{W}\right) \quad (2)$$

where $\mathcal{W}$ represents the learnable weight martix.

There are various GNN architectures and variations that have been proposed and are being actively researched, e.g., GraphSAGE [18], Graph Attention Network (GAT) [19], and Graph Isomorphism Networks (GIN) [20].

III. PROBLEM FORMULATION

In this section, we formally describe the problem and introduce the capabilities of three parties involved including *User*, *Attacker*, and *Defender*.

A. User

We assume that users can access two prediction services of the target GNN model. First, users can query specific node ID to obtain node predictions. Specifically, users provide personal information (attributes) and relationships (links) to construct the target training dataset $\mathcal{D}_t$, enabling the target model to predict all user nodes in $\mathcal{D}_t$ by learning partially labeled nodes. During the service phase, users query predictions according to node IDs in $\mathcal{D}_t$, and the prediction results are confidence vectors over the possible labels. For instance, a social network platform collects user information and provides services for searching for users with common preferences. Second, the user can query new graphs to obtain predictions for nodes in it. Such users may be researchers or social network analysts, who leverage the well-trained target model to gain insights pertinent to their datasets for further analysis.

B. Attacker

We assume that the attacker has black-box access to the target GNN model as a user, and obtains query results by querying node ID or graph. The attacker has a shadow dataset for conducting inference attacks that shares the same distribution as the target dataset and has its own graph structure, node features, and labels. It is noted that the attacker predicts potential links based on the node influence analysis in the vertically partitioned graph learning setting [13]. There is an interaction model where users can change training node features during inference, which does not align with realistic settings and is excluded from consideration in this paper. Following the black-box assumptions in the works of literature, [10], [12], we consider a threat model called score-based inference attack. Specifically, we summarize the attacks into two types, i.e., *NMIA* and *LSIA*.

Node membership inference attack: NMIA leverages the overfitting of the GNN models on the training dataset so that the query result of the training dataset and the test dataset follow different distributions. The attackers query the attack model with the 0-hop and 2-hop subgraphs to get the node's membership prediction [10]. Under the black-box setting, we assume that the attacker can access a target node v, a target GNN model F, and the shadow dataset $\mathcal{D}'$, which shares the same distribution as the target training dataset. The NMIA on GNNs is defined as the following:

$$\mathcal{NMIA} : v, F, \mathcal{D}' \mapsto \{\text{member, non-member}\} \quad (3)$$

Link status inference attack: LISA aims at inferring the existence of a link between nodes. Computing the variant distance between the prediction scores of the corresponding two nodes is the basic idea of LSIA [12]. Therefore, we assume that given target nodes v and u, a target GNN model F, and the shadow dataset $\mathcal{D}'$, which shares the same distribution of target training dataset, the LSIA on GNNs is demonstrated as the following:

$$\mathcal{LSIA} : v, u, F, \mathcal{D}' \mapsto \{\text{link-existence, link-inexistence}\} \quad (4)$$

The attack model architectures and parameters of both NMIA and LSIA are unknown to the defender.

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on November 15,2026 at 03:30:39 UTC from IEEE Xplore. Restrictions apply.

C. Defender

The defender is also the model publisher who conducts privacy-preserving mechanisms to defend against inference attacks before deploying the target GNN model. Considering that sensitive information may be involved in different aspects of graph data, i.e., node membership or link status, we define three privacy-preference-policies according to the protection priorities of the defender:

- *Policy ρ_1: Node_Mem_Privacy:* This policy is designed for the scenario where the defender aims to protect node membership privacy. For example, when detecting political ideology in social networks [21], the like or comment relationships are public, while users' political inclinations revealed by node membership should be private.
- *Policy ρ_2: Link_Sta_Privacy:* This policy is designed for the scenario where the defender prefers to protect the link status privacy. For example, when predicting user profiles (e.g., preferences, social influence) in social networks [22], links representing intimate relationships between users should be private.
- *Policy ρ_3: Joint_Privacy:* This policy is designed for the scenario where the defender expects effective protection of both node membership and link status. For example, when predicting the propagation of the COVID-19 virus in social networks [23], both infection membership and close contact relationships should be protected.

The defender has access to the target model in the white-box setting, as well as the target dataset. We define the node v with its features x_v and label y_v. Additionally, the confidence score vector is denoted as s_v. Likewise, we have node u with features x_u and label y_u. A_{vu} represents the link status between node u and v. We characterize the background knowledge of the defender as two scenarios.

Attack-type-aware: In this case, the attacker has a specific inference target according to different scenarios, i.e., NMIA or LSIA. For different attack types, the defender can adopt different defending mechanisms with the corresponding privacy-preference-policy, i.e., Policy ρ_1 and Policy ρ_2. In these two policies, the defense mechanism against NMIA and LSIA respectively are demonstrated as the following:

$$\mathcal{D}_{\mathcal{NMIA}} : \rho_1, v, p \mapsto \mathcal{RC}\{\text{member, non}-\text{member}\} \tag{5}$$

$$\mathcal{D}_{\mathcal{LSIA}} : \rho_2, v, u, q \mapsto \mathcal{RC}\{\text{link}-\text{existence, link}-\text{inexistence}\} \tag{6}$$

where P and Q represented the NMIA model and LSIA model respectively. The $\mathcal{RC}$ is the abbreviation for *Random Choose*, which indicates that the attack models fail to conduct inference attacks with defense, i.e., the attack model can not determine whether the target node is a member or not / exist a link between two nodes or not.

Attack-type-agnostic: In this case, both NMIA and LSIA coexist, while the defender has no knowledge of attack types. The defender supports both node membership privacy and link status privacy, which accords with Policy ρ_3: Joint_Privacy. In this policy, the defense mechanism against the synchronized attack can be formulated as follows:

$$\mathcal{D}_{\mathcal{MLA}} : \rho_3, v, u, p, q \mapsto \mathcal{RC}\{\text{member, non} - \text{member}\}, \mathcal{RC}\{\text{link} - \text{existence, link} - \text{inexistence}\} \tag{7}$$

The defender aims to defend against these two attacks simultaneously by confusing both NMIA and LSIA models.

IV. Design of PriGraph

In this section, we propose the PriGraph to defend against the NMIA and LSIA. We first introduce the workflow of PriGraph and then present each phase of PriGraph individually.

A. Overview

An ideal defense should effectively defend against the inference attacks as depicted in the attacker part while preserving the high performance of the primary graph learning task.

The design goals of defense can be described as follows.

- *Effectiveness:* The attack performance represents the node membership and link privacy leakage risks. For the defender, the goal is to minimize the attack performance, which indicates the effectiveness of the defense.
- *Utility Maintaince:* The performance of the model is important to achieve the prediction function in the node classification task. Thus, with defense, the utility of the GNN model cannot be severely impaired.
- *Adaptability:* The defense mechanism can be unified for defending both NMIA and LSIA with node classification tasks. In some cases, attackers intend to infer both node membership and link status. The defense against synchronized attacks should be also considered.

However, the defense faces two challenges in designing the defense scheme: 1) there is no access to the models of the attackers, and 2) the defense should not essentially damage the utility of the GNN models by the introduced perturbations.

To tackle these challenges, we employ a GAN-like model to construct PriGraph, which contains two components, i.e., a defense auxiliary classifier and an adversarial perturbation generator. The defense auxiliary classifier attempts to infer private information from the perturbed output of the target model, while the adversarial perturbation generator perturbs the outputs of the target model to reduce the inference accuracy and maintain the utility of the target model. The overview of PriGraph is shown in Fig. 2. To address the first challenge, the defender itself learns a classifier (i.e., the defense auxiliary classifier C) to carry out an auxiliary inference attack. The defender has access to the target GNN model in the white-box setting and target dataset. Note that the attacker only has access to the target GNN model in the black-box setting and shadow dataset. Therefore, leveraging the target GNN model and target dataset, the defender can imitate the behaviors of the attacker to attain the most effective attack models for the following perturbation-generating process. To meet the second challenge, the defender aims to add optimal adversarial perturbations to the output to protect the node privacy and link privacy of the GNN models. Our intuition is that the

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on November 15,2026 at 03:30:39 UTC from IEEE Xplore. Restrictions apply.

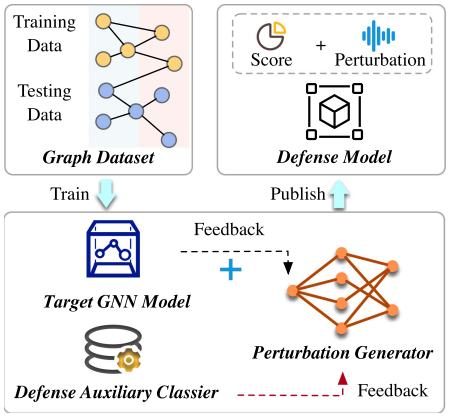


Fig. 2. The framework of PriGraph against inference attack.

attack models for NMIA or LSIA are essentially binary classifiers and for binary classifiers, adversarial examples [24] can involve flipping the label of the input from positive to negative or vice versa. We can train a neural network (i.e., the adversarial perturbation generator G) to generate adversarial perturbations that can mislead attack models to defend inference attacks and maintain the utility of GNN models.

PriGraph contains three phases in a defense model training process, we name them Target Model Training, Defense Auxiliary Classifier Training, and Adversarial Perturbation Generator Training.

1) Target Model Training: For the node classification task, given the graph dataset $\mathcal{D} = (\mathcal{G}, \mathcal{X}, \mathcal{Y})$, the target model F is trained to infer the labels and confidence score $F(v)$ of query node v. The target model can be utilized for the feedback of adversarial perturbation generator training. PriGraph does not tamper with the training process.

2) Defense Auxiliary Classifier Training: Defense auxiliary classifier is trained as an inference attack model with the strongest capacity, which leverages the precise knowledge of the target model. Specifically, training graph data and testing graph data can be seen as member data and non-member data respectively to get access to the query results (i.e., confidence score vectors) from the target model, which are constructed in the training dataset to train the defense auxiliary classifier g_n for node membership privacy. Similarly, the defense auxiliary classifier g_l can be trained for link privacy preservation.

3) Adversarial Perturbation Generator Training: With the assistance of the defense auxiliary classifier, the adversarial perturbation generator aims to find the optimal adversarial perturbations, which can be regarded as an optimization problem to achieve that: 1) fool the attack model to reduce its attack performance. 2) maintain the high accuracy of the target model. These perturbations are added to each confidence score vector predicted by the target model. The adversarial perturbation generator is trained and deployed to the defense model which is published for the user's query service.

B. Defense Auxiliary Classifier

In this subsection, we present the defense auxiliary classifier training process. The defender itself learns a defense auxiliary classifier to imitate the attacker performing inference attacks. The defender holds the information of the target dataset and target model. Therefore, the full knowledge of the target dataset and target model can be leveraged for the training of the defense auxiliary classifier. For the attacker, the whole information of the target dataset and model is inaccessible in the black-box setting. The defense auxiliary classifier is used to assist the adversarial perturbation generator to add effective and optimal perturbation vectors.

Algorithm 1: Defense Auxiliary Classifier Training for Node.

Input: Target model F, Target dataset $\mathcal{D}$
Output: Defense auxiliary classifier C_I

for i=1,2,...,epoch **do**
  for each batch in $\mathcal{D}$ **do**
    Get 0-hop and 2-hop subgraph $\mathcal{D}_{sub}$
    $logit_{0-hop}, logit_{2-hop} \leftarrow F(\mathcal{D}_{sub})$
    $feature \leftarrow$ GEN_FEA_NODE($logit_{0-hop}$, $logit_{2-hop}$)
    $logit_{C_I} \leftarrow C_I(feature)$
    /* Gradient descent */
    $\theta_{C_I} \leftarrow \theta_{C_I} - \eta \nabla_{\theta_{C_I}} \mathcal{L}(logit_{C_I}, label)$
  end for
end for
Return defense auxiliary classifier parameter θ_{C_I}

function GEN_FEA_NODE($logit_{0-hop}$, $logit_{2-hop}$)
  $logit_0 \leftarrow$ Top2($logit_{0-hop}$)
  $logit_2 \leftarrow$ Top2($logit_{2-hop}$)
  $feature \leftarrow$ Concatenate($logit_0$, $logit_2$)
  return feature
end Function

For node privacy preservation, we design the defense auxiliary classifier training process, and details are described in Algorithm 1. The defense auxiliary classifier C_I is a binary classifier, which leverages the target dataset and the outputs of the target model. All the nodes of the graph training dataset $\mathcal{D}_{train}$ are labeled as 1 and the nodes of the graph testing dataset $\mathcal{D}_{test}$ are labeled as 0. The corresponding prediction scores from the target model are treated as the features.

Likewise, we design the defense auxiliary classifier training process for link privacy preservation, and details are illustrated in Algorithm 2. The defense auxiliary classifier C_{II} is trained as a binary classifier, which leverages the graph topology structure of the target dataset and the target GNN models. The pairs of nodes with links are labeled 1 and the pairs without links are labeled 0. All pairs of nodes are selected to acquire the corresponding prediction scores by querying the target GNN models. The features are derived from computing the correlation between these pairs of scores.

C. Adversarial Perturbation Generator

The adversarial perturbation generator G is designed as a neural network, which crafts the optimal perturbation n^* for the confidence score output s_v of the target model. Since the attack model utilizes confidence scores to infer node membership

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on November 15,2026 at 03:30:39 UTC from IEEE Xplore. Restrictions apply.

Algorithm 2: Defense Auxiliary Classifier Training for Link.

Input: Target model F, Target dataset $\mathcal{D}$
Output: Defense auxiliary classifier C_{II}

for i=1,2,...,epoch **do**
  for each batch in $\mathcal{D}$ **do**
    Select 2*batch_size pairs of nodes v_i and v_j from the subgraph that contains the batch node
    $logit_i \leftarrow F(v_i)$
    $logit_j \leftarrow F(v_j)$
    $feature \leftarrow$ GEN_FEATURES_LINK($logit_i, logit_j$)
    $logit_{C_{II}} \leftarrow C_{II}(feature)$
    /* Gradient descent */
    $\theta_{C_{II}} \leftarrow \theta_{C_{II}} - \eta \nabla_{\theta_{C_{II}}} \mathcal{L}(logit_{C_{II}}, label)$
  end for
end for
return defense auxiliary classifier parameter $\theta_{C_{II}}$

function GEN_FEA_LINK($logit_i, logit_j$)
  $d \leftarrow Distance(logit_i, logit_j)$
  $s \leftarrow Similarity(logit_i, logit_j)$
  $feature \leftarrow$ Concatenate(d, s)
  Return feature
end Function

properties or link existence, the perturbed confidence score can be regarded as adversarial examples to evade the inference of attack models. Based on the knowledge of the defender, the adversarial perturbation generator will be constructed with the assistance of the defense auxiliary classifier C in Section IV-B.

We provide a formal description of the defense auxiliary classifier C. It is a binary classifier with parameters θ, and its input features are confidence score vectors of nodes in NMIA or node pairs in LSIA. In terms of NMIA defense, the auxiliary classifier C: $[0,1]^d \rightarrow \{0,1\}$ recognizes the input s_v as the label with the maximal probability, illustrated as follows:

$$C(s_v) = \arg\max_{c \in [0,1]} \mathcal{F}_c(s_v; \theta) \tag{8}$$

where d is the dimension of the confidence score vector, $\mathcal{F}_c$ computes the probability of mapping input confidence score s_v to label c.

Recall that the design goal of the adversarial perturbation generator is two-fold: 1) The generated adversarial perturbations added to the target GNN model output should effectively confuse the attack models for NMIA and LSIA. 2) The generated adversarial perturbations should minimize the impact on model prediction performance.

To achieve the first goal, the defender attempts to craft an adversarial example $s'_v = s_v + n^*$ to mislead the auxiliary classifier C. Our intuition is that, guided by auxiliary classifiers, the adversarial perturbation n^* can hide critical information in s_v and transfer such misleading to real attack models. Formally, we can define an indicator function I_{s_v}: $[0,1]^d \rightarrow \mathbb{R}$, where $I_{s_v}(s'_v)$ towards 0 indicates an adversarial perturbation that effectively confuses the attack models, avoiding correlation between attack

Algorithm 3: Adversarial Perturbation Generator for Node.

Input: Target model F, Target dataset $\mathcal{D}$, Defense auxiliary classifier C_I
Output: The defense model F^*

for i=1,2,...,epoch **do**
  for each batch in $\mathcal{D}$ **do**
    Get 0-hop and 2-hop subgraph $\mathcal{D}_{sub}$
    $logit_{0-hop}, logit_{2-hop} \leftarrow F(\mathcal{D}_{sub})$
    $logit \leftarrow logit_{0-hop}, logit_{2-hop}$
    $logit^* \leftarrow F^*(logit)$ + $logit$
    $feature \leftarrow$ GEN_FEA_NODE ($logit^*$)
    $logit_{C_I} = G_I(feature)$
    $\mathcal{L} \leftarrow \mathcal{L}_F(logit^*, logit) + \|logit_{C_I} - 0.5 \cdot \mathbf{1}\|_2$
    $\theta_{F^*} \leftarrow \theta_{F^*} - \eta \nabla_{\theta_{F^*}} \mathcal{L}$
  end for
end for
Return perturbation generator parameter θ_{F^*}

model predictions and true labels.

$$I_{s_v}(s'_v) = \max_{c \neq y} \mathcal{F}_c(s'_v; \theta) - \mathcal{F}_y(s'_v; \theta) \tag{9}$$

where y is the true attack label, e.g., node membership status.

To achieve the second goal, the node's predicted label should be unchanged after perturbation. We denote the confidence score for class i as s_{vi} with the following constraint:

$$\arg\max_i \{s_{vi} + n_i^*\} = \arg\max_i \{s_{vi}\} \tag{10}$$

where n_i^* is the perturbation. To satisfy the above objectives of adversarial perturbation for effective defense against inference attacks, we carefully design the optimization process of the adversarial perturbation generator G.

Formally, we aim to generate adversarial perturbations by solving the optimization problem with two core loss functions $\mathcal{L}_C$ and $\mathcal{L}_F$. The $\mathcal{L}_C$ aims to make the auxiliary classifier C recognize the perturbed confidence score vector as a random label, which is defined as:

$$\mathcal{L}_C = |I_{s_v}(s'_v)| = \|\mathcal{C}(s_v + G(s_v)) - 0.5 \cdot \mathbf{1}_{dim(2)}\|_2 \tag{11}$$

where $\mathcal{C}$ is the predicted function of the defense auxiliary classifier, and s_v is the confidence score output from the target GNN model. $G(\cdot)$ generates the adversarial perturbation for input s_v, and its parameter g is our optimization object. Equation (11) implies that the classifier C has a prediction probability of about 0.5 for each label, which represents an attack failure.

The $\mathcal{L}_F$ relaxes the constraints in (10), expecting the adversarial perturbations are minimized to maintain the accuracy of the classification task:

$$\mathcal{L}_F = CrossEntropy(s_v, s_v + G(s_v)) \tag{12}$$

In Policy ρ_1, the adversarial perturbation is crafted to protect the node membership privacy by defending against the NMIA. The adversarial perturbation generator learns the neural network to minimize the classification error and at the same time maximize the prediction error of the defense auxiliary classifier C_I.

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on November 15,2026 at 03:30:39 UTC from IEEE Xplore. Restrictions apply.

Algorithm 4: Adversarial Perturbation Generator for Link.

Input: Target model F, Target dataset $\mathcal{D}$, Defense auxiliary classifier C_{II}
Output: The defense model F^*

for i=1,2,...,epoch **do**
  for each batch in $\mathcal{D}$ **do**
    Select 2*batch_size pairs of nodes v_i and v_j from the subgraph that contains the batch node
    $logit_i \leftarrow F(v_i)$
    $logit_j \leftarrow F(v_j)$
    $logit_i^* \leftarrow F^*(logit_i) + logit_i$
    $logit_j^* \leftarrow F^*(logit_j) + logit_j$
    $feature \leftarrow$ GEN_FEA_LINK$(logit_i, logit_j)$
    $logit_{C_{II}} = C_{II}(feature)$
    $Loss_* \leftarrow \mathcal{L}_F(logit_i^*, logit_i) + \mathcal{L}_F(logit_j^*, logit_j)$
    $\mathcal{L} \leftarrow Loss_* + \|logit_{C_{II}} - 0.5 \cdot \mathbf{1}\|_2$
    /* Gradient descent */
    $\theta_{F^*} \leftarrow \theta_{F^*} - \eta \nabla_{\theta_{F^*}} \mathcal{L}$
  end for
end for
Return perturbation generator parameter θ_{F^*}

The loss of the adversarial perturbation generator G_I in policy ρ_1 is as follows:

$$\mathcal{L}_{G_I} = \mathcal{L}_F + \lambda_1 \mathcal{L}_{C_I} \tag{13}$$

where λ_1 is a hyperparameter that balances between model utility and node membership privacy. The defender can access the defense model F^* added the perturbations derived from the adversarial perturbation generator G_I and the training process for the node privacy preservation is described as Algorithm 3.

In Policy ρ_2, the adversarial perturbation is crafted to protect the link status privacy by defending against the LSIA. Similarly, the adversarial perturbation generator learns the neural network to minimize the classification error and at the same time maximize the prediction error of the defense auxiliary classifier C_{II}. The loss of the adversarial perturbation generator G_{II} in policy ρ_2 is as follows:

$$\mathcal{L}_{G_{II}} = \mathcal{L}_F + \lambda_2 \mathcal{L}_{C_{II}} \tag{14}$$

where λ_2 is a hyperparameter that balances between model utility and link status privacy. The adversarial perturbation generator G_I training process for the link privacy preservation is described as Algorithm 4.

In Policy ρ_3, the protection for node membership privacy and link status privacy are both demanded. The generated adversarial perturbation is required by defending against the NMIA and LSIA. The adversarial perturbation generator learns the neural network to minimize the classification error and at the same time maximize the prediction error of the defense auxiliary classifier C_I and C_{II}. The loss of the adversarial perturbation generator G_{III} in policy ρ_3 is as follows:

$$\mathcal{L}_{G_{III}} = \mathcal{L}_F + \lambda_1 \mathcal{L}_{C_I} + \lambda_2 \mathcal{L}_{C_{II}} \tag{15}$$

TABLE II
DATASET STATISTICS

Type	Homophilic		Heterophilic	
Dataset	**Citeseer**	**Pubmed**	**Chameleon**	**Texas**
Nodes	3,327	19,717	2,277	183
Edges	9,104	88,648	62,792	325
Features	3,703	500	128	1,703
Classes	6	3	5	5
H_{node}	0.71	0.79	0.17	0.10
H_{edge}	0.74	0.80	0.17	0.11
H_{class}	0.63	0.66	0.02	0.00

where λ_1 and λ_2 is a hyperparameter that balances between model utility, node membership privacy, and link relation privacy. The adversarial perturbation generator G_{III} training process is the combination of Algorithms 3 and 4.

V. EXPERIMENTAL SETUP

In this section, we describe the datasets used to evaluate PriGraph and introduce the architectures of the GNN models to perform the node-level classification task of the target model. The architectures of NMIA, LSIA models, defense auxiliary classifier, and adversarial perturbation generator models are demonstrated respectively.

A. Dataset Setting

Dataset: We conduct experiments on two homophilic datasets (i.e., Citeseer and Pubmed) and two heterophilic datasets (i.e., Chameleon and Texas) which are widely adopted in existing works [25]. Citeseer and Pubmed are citation datasets where nodes of graph data represent publications [26]. Chameleon is a web page dataset collected from Wikipedia [27]. Texas is a subdataset from the WebKB web-page dataset [28]. The linked nodes in heterophilic datasets have dissimilar features and different labels. The statistics of these datasets are provided in Table II. Here, the metrics H_{node}, H_{edge}, and H_{class} quantify the homophily of nodes, edges, and classes in these datasets, where lower values indicate stronger heterophily [25].

Dataset configuration: The dataset $\mathcal{D}$ is randomly split in half, creating the target dataset $\mathcal{D}_t$ and the attack auxiliary dataset $\mathcal{D}_{attack}$ with no edge intersection between them. Further, $\mathcal{D}_t$ is split into training dataset D_{train} and testing dataset D_{test} for the target model F. By querying the confidence scores from the target model F on D_{train} and D_{test}, the defense auxiliary datasets can be constructed. Similarly, for NMIA and LSIA, $\mathcal{D}_{attack}$ is randomly divided into two halves. The first half D'_{train} trains the shadow model F', and the second half D'_{test} tests the shadow model. Using the shadow model's query results according to the training data and testing data, the attack dataset can be constructed to train the attack model. Note that the divisions of $\mathcal{D}_t$ and $\mathcal{D}_{attack}$ intend to assign training nodes and do not modify the graph structure.

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on November 15,2026 at 03:30:39 UTC from IEEE Xplore. Restrictions apply.

B. Model Architectures

Target models: We utilize three GNN architectures (i.e., GraphSAGE [18], GAT [19], and GIN [20]) to construct our target models and shadow models with node classification tasks as primary tasks. We set the number of convolution layers to 2 and hidden channels to 32 for each target model. To reduce overfitting, the dropout rate is set to 0.5. Especially, for the multi-head attention mechanism of the GAT model, the number of heads is set to 2 at the first layer and 1 at the second layer. The optimizer is Adam and Cross-Entropy is used as the loss function. The learning rate is set to $3e^{-3}$. And the training epochs of the target and shadow model are set to 200.

Node membership inference attack models: Following He et al. [10], we adopt the combined strategy to construct the NMIA model. Specifically, the attack model takes the largest two values from the prediction vectors of 0-hop and 2-hop queries as inputs, which are fed into separate 64-neuron linear layers. The generated embeddings are concatenated for further membership inference. The dropout rate is set to 0.5. The learning rate is set to $3e^{-3}$ and the training epochs are set to 500. The attack model intercepts the Top-2 value instead of the full prediction vector as input, facilitating its membership inference across datasets with varying numbers of classes.

Link status inference attack models: Following attack training features designed by He et al. [12], we adopt MLP with 3 hidden layers as the LSIA model Q. Each hidden layer has 32 neurons and ReLU is adopted as the activation function. The optimizer is Adam and Cross-Entropy is used as the loss function. The dropout rate is set to 0.5. It is trained 50 epochs with a learning rate of 0.001.

Defense auxiliary classifiers: For defense against NMIA and LSIA, the defense auxiliary classifier models are trained by the defender. The defense auxiliary classifier for NMIA defense C_I and defense auxiliary classifier C_{II} for LSIA defense are both MLP with 3 hidden layers. Each hidden layer has 64 neurons. ReLU is used as the activation function and softmax is the output activation function. The dropout rate is set to 0.5. The training epoch is set to 100 with a learning rate of 0.001.

Adversarial perturbation generators: The adversarial perturbation generators G_I, G_{II} for defending against NMIA and LSIA respectively are constructed by MLP with 5 hidden layers. In order of the hidden layers, the numbers of neurons are 64, 128, 256, 128, and 64. ReLU is used as the activation function. The learning rate is set to $3e^{-3}$ and the training epochs are set to 50.

C. Metrics

To evaluate the primary task performance of GNN models with defense mechanisms, we use the test accuracy $\mathrm{Acc.}_{te}$ as the metric for utility maintenance (i.e., the percentage of the corrected labels predicted by the GNN model in the testing phase). To evaluate the membership and link leakage risk before and after the defense mechanism, we use the attack accuracy $\mathrm{Acc.}_{at}$ as the metric for effectiveness, i.e., the percentage of the corrected node membership (member or non-member) and link status (link-existence, link-inexistence) inferred respectively by the NMIA and LSIA model. Considering that in NMIA and LSIA models, the task of inferring node membership or link status often presents as an imbalanced binary classification problem, we further use precision $\mathrm{Pr.}_{at}$ and recall $\mathrm{Re.}_{at}$ as metrics to evaluate the effectiveness. All results are obtained on a machine equipped with an Intel(R) Core(TM) i7-13700K CPU and an NVIDIA RTX 3080 GPU.

D. Methods to Compare

For the three types of privacy-preference policies, we take no defense mechanism as the baseline and choose the following methods to compare.

Comparison on node privacy: We employ four defenses to defend against NMIA in Table III for comparison.

- *Node-Random Noise:* The gaussian noise is randomly added to the attribute vectors of the nodes in the graph dataset and the standard deviation is set to 0.04*norm.
- *Node-DP:* The differential privacy noise is added to the attribute vectors of the nodes in the graph dataset and the privacy budget is set to 30.
- *Centrality-based Noise:* Degree centrality represents the importance of a node in the graph data and refers to the number of other nodes associated with the node. According to the degree centrality of each node, noise is added with the corresponding probability to the attribute vectors of the nodes in the graph dataset.
- *Graph Reconstruction [29]. It reconstructs the graph structure by graph AutoEncoder (GAE) [31]:* We add Gaussian noise to the latent representations of nodes, and obtain the reconstructed perturbation graph with a threshold of 0.5. Following previous literature [29], the noise standard deviation is set to 0.3.

Comparison on link privacy: We conduct three defenses to defend against LSIA in Table IV for comparison.

- *Link-Random Addition:* Constructing an adjacency matrix $\mathcal{A}$ of graph data of which $\mathcal{A}_{ij} = 0$ is randomly changed into $\mathcal{A}_{ij} = 1$.
- *Link-DP Conversion:* The change of the adjacency matrix $\mathcal{A}$ conforms to the different privacy mechanisms. The link statuses of the graph dataset are disturbed.
- *Influence-Link Reduce:* Subtract the links of the dataset with a certain probability which is on the basis of the centrality of two nodes that the link connects.
- *FairDefense [30]:* This method adds perturbation to the original graph structure during each training epoch, which modifies the edge existence with fixed probability. We set the privacy parameter γ to 0.01, which shows optimal defense results with acceptable accuracy loss.

Comparison on joint privacy: We employ the two noise-related defenses to defend against both NMIA and LISA in Table V for comparison.

- *Score-Random Noise:* The gaussian noise is randomly added to the probability distribution vectors of the target GNN model and the standard deviation is set to 0.04*norm.
- *Score-DP Noise:* The differential privacy noise is added to the probability distribution vectors of the outputs from the GNN model and the privacy budget is set to 30.

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on November 15,2026 at 03:30:39 UTC from IEEE Xplore. Restrictions apply.

TABLE III
DEFENSES AGAINST NMIA ON FOUR DATASETS WITH GNN MODEL GRAPHSAGE IN POLICY ρ_1

Defenses	Citeseer (%)				Pubmed (%)				Chameleon (%)				Texas (%)			
	Acc.$_{te}$	**Acc.**$_{at}$	**Pr.**$_{at}$	**Re.**$_{at}$	**Acc.**$_{te}$	**Acc.**$_{at}$	**Pr.**$_{at}$	**Re.**$_{at}$	**Acc.**$_{te}$	**Acc.**$_{at}$	**Pr.**$_{at}$	**Re.**$_{at}$	**Acc.**$_{te}$	**Acc.**$_{at}$	**Pr.**$_{at}$	**Re.**$_{at}$
No Defense	77.69	77.69	79.79	97.59	86.10	52.41	51.70	73.56	62.21	57.82	56.55	67.49	71.74	78.02	85.71	66.67
Node-Random Noise	67.43	54.29	62.82	20.94	85.35	50.78	51.16	13.84	59.23	52.46	51.41	89.98	67.39	60.44	90.91	22.22
DP-Node Noise	71.03	54.23	61.15	23.10	85.09	50.21	52.05	22.97	60.46	52.11	51.21	89.28	58.70	58.24	88.89	17.78
Centrality-based Noise	66.95	65.00	67.97	56.68	64.17	50.39	51.13	17.41	51.49	52.02	51.62	64.50	54.35	68.13	72.22	57.78
Graph Reconstruction [29]	72.48	73.16	75.54	68.03	81.13	50.66	50.87	81.65	60.87	50.26	47.00	16.52	67.56	48.35	40.00	8.88
PriGraph (ours)	**73.20**	**50.03**	**0**	**0**	**86.00**	**50.00**	**0**	**0**	**61.51**	**50.00**	**0**	**0**	**69.57**	**50.00**	**0**	**0**

TABLE IV
DEFENSES AGAINST LSIA ON FOUR DATASETS WITH GNN MODEL GRAPHSAGE IN POLICY ρ_2

Defenses	Citeseer (%)				Pubmed (%)				Chameleon (%)				Texas (%)			
	Acc.$_{te}$	**Acc.**$_{at}$	**Pr.**$_{at}$	**Re.**$_{at}$	**Acc.**$_{te}$	**Acc.**$_{at}$	**Pr.**$_{at}$	**Re.**$_{at}$	**Acc.**$_{te}$	**Acc.**$_{at}$	**Pr.**$_{at}$	**Re.**$_{at}$	**Acc.**$_{te}$	**Acc.**$_{at}$	**Pr.**$_{at}$	**Re.**$_{at}$
No Defense	77.69	83.14	81.50	85.76	86.10	82.17	79.14	87.31	62.21	65.49	60.98	86.02	71.74	64.06	58.71	94.79
Link-Random Addition	69.47	82.51	80.36	86.06	83.81	75.54	76.85	73.00	59.93	62.50	59.22	80.30	67.39	60.42	57.69	78.13
Link-DP Conversion	68.03	81.85	78.79	87.17	84.54	75.16	74.21	77.02	59.40	64.80	60.87	82.90	69.57	63.63	60.56	89.58
Influence-Link Reduce	73.92	83.22	80.00	88.58	84.70	73.19	81.07	60.41	57.12	61.37	60.11	67.62	60.87	58.85	56.91	72.92
FairDefense [30]	72.36	77.91	79.85	73.44	84.22	73.51	71.64	76.98	58.17	63.93	61.41	75.00	67.39	64.67	81.82	12.68
PriGraph (ours)	**72.72**	**50.00**	**50.00**	**100.00**	**85.64**	**49.93**	**49.93**	**100.00**	**61.86**	**50.00**	**50.00**	**100.00**	**71.74**	**50.00**	**0**	**0**

TABLE V
DEFENSES AGAINST NMIA AND LSIA ON TWO DATASETS WITH GNN MODEL GAT IN POLICY ρ_3

Defenses	Citeseer (%)							Texas (%)						
	Acc.$_{te}$	NMIA			LSIA			**Acc.**$_{te}$	NMIA			LSIA		
		Acc.$_{at}$	**Pr.**$_{at}$	**Re.**$_{at}$	**Acc.**$_{at}$	**Pr.**$_{at}$	**Re.**$_{at}$		**Acc.**$_{at}$	**Pr.**$_{at}$	**Re.**$_{at}$	**Acc.**$_{at}$	**Pr.**$_{at}$	**Re.**$_{at}$
No Defense	84.38	60.61	57.25	81.33	83.62	81.38	87.20	52.17	59.34	64.73	56.90	64.06	72.88	44.79
Score-Random Noise	32.45	51.11	51.17	47.53	60.21	80.00	25.03	30.43	49.45	49.45	100.00	53.13	80.00	8.33
Score-DP Noise	59.50	61.87	60.97	65.22	71.15	84.90	50.00	43.48	56.04	53.52	84.44	57.29	64.58	32.29
Graph Reconstruction [29]	72.11	55.58	80.38	40.67	65.38	88.44	33.12	43.24	50.54	50.00	2.22	62.18	56.25	5.07
FairDefense [30]	66.47	61.76	91.49	25.87	89.27	88.51	89.75	41.30	57.14	53.95	91.11	63.54	62.50	67.71
PriGraph (ours)	**84.31**	**50.39**	**0**	**0**	**50.00**	**50.00**	**100.00**	**47.83**	**50.00**	**0**	**0**	**50.00**	**50.00**	**100.00**

VI. PERFORMANCE EVALUATION

In this section, we evaluate the performance of PriGraph across different datasets and model architectures. First, in Section VI-A, we evaluate the effectiveness, utility, and adaptability of PriGraph. Then, in Section VI-B, we conduct experiments on the efficiency of PriGraph. Additionally, in Section VI-C, we perform experiments on the impact of attacker capability assumptions on PriGraph. Finally, in Section VI-D, we evaluate the impact of parameters on PriGraph.

A. Effectiveness, Utility and Adaptability of PriGraph

In this subsection, we evaluate PriGraph in terms of effectiveness, utility, and adaptability in different privacy-preference-policies. Since defenses are demanded to maintain model performance and defend against attacks simultaneously, we put the evaluation of the effectiveness and utility together.

Effectiveness and utility: In the attack-type-aware scenario, we evaluate the defense performance of PriGraph against NMIA and LSIA, respectively, as well as the test accuracy of the target model under the protection of PriGraph.

As shown in Table III, PriGraph effectively provides node membership privacy in both homophilic and heterophilic datasets and maintains the high utility of the target model. PriGraph can reduce the attack accuracy to about 50% and achieves a precision of 0% with a recall of 0%, indicating that the prediction of the attacker is close to random guessing and the attack model meaninglessly predicts all nodes as non-members. Compared with No Defense, the test accuracy of the target model in PriGraph drops by up to 4.49%, which is lower than that under other defenses.

Graph Reconstruction [29] achieves the closest defense performance to PriGraph on heterophilic datasets, reducing NMIA accuracy to 50.26% and 48.35%. This effectiveness is attributed to the fact that the reconstructed adjacency matrix alters the correlations between adjacent nodes. Such alteration is more pronounced in heterophilic datasets where the similarity of nodes is lower, thereby better concealing the information

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on November 15,2026 at 03:30:39 UTC from IEEE Xplore. Restrictions apply.

TABLE VI
TIME OVERHEAD OF DEFENDING AGAINST NMIA ON FOUR DATASETS WITH GRAPHSAGE IN POLICY ρ_1

Defenses	Citeseer		Pubmed		Chameleon		Texas	
	Offline Defense Time (s)	Online Defense Time (μs)	Offline Defense Time (s)	Online Defense Time (μs)	Offline Defense Time (s)	Online Defense Time (μs)	Offline Defense Time (s)	Online Defense Time (μs)
Node-Random Noise	0.18±0.01	/	1.32±0.08	/	0.13±0.01	/	0.03±0.01	/
DP-Node Noise	0.23±0.02	/	1.40±0.05	/	0.14±0.01	/	0.03±0.01	/
Centrality-based Noise	0.15±0.01	/	2.49±0.04	/	5.76±0.11	/	0.01±0.01	/
Graph Reconstruction [29]	0.62±0.02	/	76.51±0.32	/	0.96±0.02	/	0.37±0.02	/
PriGraph (ours)	163.45±1.99	499.96±0.19	489.57±5.83	500.44±0.39	107.68±0.51	499.81±0.30	5.39±0.04	500.12±0.30

1. / indicates that the corresponding defense has no time overhead in the offline or online phase.

regarding node membership. However, it still causes more significant test accuracy reduction of the target model than PriGraph. While other defenses demonstrate NMIA accuracy reduction capabilities, their NMIA accuracies remain above 50%, which are all higher than PriGraph. More critically, these defenses caused a more significant reduction in the test accuracy of the target model than PriGraph.

As shown in Table IV, PriGraph is effective in protecting link status privacy and preserves the high utility of the target model. PriGraph reduces the attack accuracy to around 50%. On Citeseer, Pubmed, and Chameleon datasets, the precision is around 50%, and the recall is 100%. On the Texas dataset, both the precision and recall are 0%. These indicate that the prediction of the attacker degrades to random guessing, meaninglessly predicting all nodes as either connected or unconnected. In terms of the test accuracy of the target model, compared with No Defense, it drops by up to 4.97%.

Although the test accuracy of the target model of Influence-Link Reduce is higher than that of PriGraph on the Citeseer dataset, it can hardly defend against the LSIA. FairDefense [30] fails to defend against LSIA on the Texas dataset. This is because, in the small-scale Texas dataset, the node paths are relatively short, resulting in a weaker link status obfuscation effect compared to large-scale datasets with longer node paths. In contrast to PriGraph, other defenses show more significant drops in test accuracy and smaller reductions in attack accuracy.

The results show that PriGraph can effectively protect node membership privacy and link status privacy in homophilic and heterophilic datasets, respectively, while achieving the minimum reduction of the test accuracy of the target model.

Adaptability: Given the substantial scale differences between the homophilic dataset Citeseer and the heterophilic dataset Texas, and the notable test accuracy changes of them shown in Tables III and IV, in the attack-type-agnostic scenario, we concurrently evaluate the defense performance of PriGraph against synchronized NMIA and LSIA on these datasets, as well as its performance in protecting the test accuracy. While Graph Reconstruction [29] and FairDefense [30] are designed to defend only against NMIA or LSIA, we test their performance against the other type of attack as comparisons.

As shown in Table V, PriGraph protects the privacy of node memberships and link status while maintaining a high level of model utility. The results demonstrate that PriGraph reduces the accuracies of both NMIA and LSIA to about 50%, while the accuracy of the target model decreases by at most 4.34%, and PriGraph consistently outperforms other defenses.

The Score-Random Noise reduces the accuracies of both NMIA and LSIA to around 50%, yet it causes the most substantial reduction in the test accuracy of the target model. Graph Reconstruction [29] can also reduce the accuracies of these two attacks. This is because the reconstructed adjacency matrix conceals the link status, which directly combats LSIA and the topological alterations influence the posterior predictions of nodes, thereby decreasing the membership information accessible for NMIA. However, in comparison with PriGraph, it still causes a more substantial decline in the test accuracy of the target model.

The results show that in the attack-type-agnostic scenario, PriGraph effectively defends against synchronized NMIA and LSIA on both homophilic and heterophilic datasets while maintaining high model utility.

B. *Efficiency of PriGraph*

In this section, we evaluate the time overheads of PriGraph when defending against NMIA and LSIA separately and simultaneously. The time overheads include the online defense time for perturbing a single output of the target model and the offline defense time for training the defense auxiliary classifier and the adversarial perturbation generator. Other defenses perturb the output during the online defense phase or the training dataset during the offline defense phase. We repeat these experiments three times and record the average value and standard deviation of the time overheads.

The time overhead of PriGraph defending against NMIA across four datasets and the GraphSAGE architecture is shown in Table VI. The offline defense time of PriGraph for training the defense auxiliary classifier and the adversarial perturbation generator is affected by the scale of the dataset. A larger data scale will lead to more offline defense time. It should be noted that the defense auxiliary classifier and the adversarial perturbation generator only need to be trained once to defend against attacks. Although other defenses based on the training data perturbation require less offline defense time than PriGraph, they cannot effectively preserve the utility of the target model while protecting node membership privacy, as shown in Table III. PriGraph only requires the online defense time of about 500 microseconds to perturb a single output and effectively defends against NMIA, indicating its high efficiency in practical deployment.

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on November 15,2026 at 03:30:39 UTC from IEEE Xplore. Restrictions apply.

TABLE VII
TIME OVERHEAD OF DEFENDING AGAINST LSIA ON FOUR DATASETS WITH GRAPHSAGE IN POLICY ρ_2

Defenses	Citeseer		Pubmed		Chameleon		Texas	
	Offline Defense Time (s)	Online Defense Time (μs)	Offline Defense Time (s)	Online Defense Time (μs)	Offline Defense Time (s)	Online Defense Time (μs)	Offline Defense Time (s)	Online Defense Time (μs)
Link-Random Addition	0.21±0.02	/	5.47±0.25	/	1.24±0.02	/	0.03±0.03	/
Link-DP Conversion	0.06±0.01	/	2.21±0.08	/	0.03±0.01	/	0.02±0.01	/
Influence-Link Reduce	0.01±0.01	/	0.07±0.01	/	0.20±0.01	/	0.01±0.01	/
FairDefense [30]	0.09±0.01	/	2.27±0.10	/	0.06±0.01	/	0.04±0.01	/
PriGraph (ours)	12.26±0.05	500.52±0.56	78.21±0.43	500.36±0.56	20.54±0.15	499.73±1.03	1.88±0.20	499.73±0.39

TABLE VIII
TIME OVERHEAD OF DEFENDING AGAINST NMIA AND LSIA ON TWO DATASETS WITH GAT IN POLICY ρ_3

Defenses	Citeseer		Texas	
	Offline Defense Time (s)	Online Defense Time (μs)	Offline Defense Time (s)	Online Defense Time (μs)
Score-Random Noise	/	499.73±0.01	/	501.43±0.01
Score-DP Noise	/	500.36±0.79	/	500.36±0.98
Graph Reconstruction [29]	0.56±0.02	/	0.29±0.01	/
FairDefense [30]	0.10±0.01	/	0.03±0.01	/
PriGraph (ours)	262.81±1.46	500.12±0.41	21.21±0.16	499.72±0.01

The time overhead of PriGraph in defending against LSIA across four datasets and the GraphSAGE architecture is presented in Table VII. Compared with its defense against NMIA, PriGraph incurs less offline defense time. This is because the input for training the defense auxiliary classifier of LSIA is composed of two nodes, thus reducing the scale of the training input and the number of iterations. Similar to defending against NMIA, although other defenses have lower time overheads, they cannot effectively defend against LSIA while maintaining the high utility of the target model, as shown in Table IV. With the offline defense time that only needs to be run once and the online defense time of around 500 microseconds, PriGraph achieves remarkable effectiveness in defending against LSIA and preserves the utility of the target model, demonstrating its practical applicability and feasibility.

The time overhead of PriGraph in simultaneously defending against NMIA and LSIA across two datasets and the GAT architecture is shown in Table VIII. Here, the defense auxiliary classifiers C_I and C_{II} are trained in parallel. Compared with the scenarios in Tables VI and VII, PriGraph requires more time to train the shadow models in the more complex GAT architecture, thus increasing the offline defense time. Considering its online defense time of around 500 microseconds, which is comparable to that of Score-Random Noise and Score-DP Noise, and its remarkable adaptability in simultaneously defending against NMIA and LSIA, PriGraph still demonstrates its practicality and feasibility.

The experimental results show that most of the time of PriGraph is spent in the offline phase to train the defense auxiliary classifier and the adversarial perturbation generator, which only need to be trained once. In the online phase, PriGraph only requires around 500 microseconds to perturb a single output to effectively defend against NMIA and LSIA, presenting high practical applicability and feasibility.

TABLE IX
DEFENSE AGAINST DIFFERENT MODEL ARCHITECTURES

(a) Defense against NMIA in Policy ρ_1

Defenses	F'	Acc.$_{te}$ (%)	Acc.$_{at}$ (%)	Pr.$_{at}$ (%)	Re.$_{at}$ (%)
No Defense	**GAT [19]**	76.66	69.94	63.78	92.32
	GIN [20]	76.66	62.04	58.72	81.09
PriGraph (ours)	**GAT [19]**	**76.22**	**50.00**	**0**	**0**
	GIN [20]	**76.22**	**50.00**	**0**	**0**

(b) Defense against LSIA in Policy ρ_2

Defenses	F'	Acc.$_{te}$ (%)	Acc.$_{at}$ (%)	Pr.$_{at}$ (%)	Re.$_{at}$ (%)
No Defense	**GAT [19]**	76.66	84.82	81.25	90.51
	GIN [20]	76.66	84.01	82.27	86.69
PriGraph (ours)	**GAT [19]**	**76.59**	**50.00**	**50.00**	**100.00**
	GIN [20]	**76.59**	**50.00**	**50.00**	**100.00**

C. Impact of Capability Assumption

In this subsection, we introduce the citation dataset Cora [26] to evaluate the impact of different capability assumptions of attackers on PriGraph. We investigate the performance of PriGraph under different capability assumptions of attackers: 1) The shadow datasets of attackers have different distributions from the target dataset. 2) The shadow model of the attacker is a different architecture from the target model. The results show that the performance of PriGraph still maintains good stability and decreases the attack performances, inclining NMIA and LISA toward random guessing.

The impact of different model architectures: We observe that the defense performance is relatively stable under different model architectures, illustrated in Table IX. We assume that a shadow model F' with the different architectures (i.e., GAT and GIN) from the target model GraphSAGE. For GraphSAGE trained on Cora, the primary node classification task accuracy is fixed as 76.66%.

Table IX(a) shows that when the shadow model architectures are GAT and GIN while the original NMIA accuracy is 69.94% and 62.04%, PriGraph achieves primary task accuracy only 0.44% loss and reduces NMIA accuracy by 19.94% and 12.04%. Table IX(b) demonstrates the performance of PriGraph under LISA. When the shadow models are GAT and GIN while the original LSIA accuracy is 84.82% and 84.01%, PriGraph achieves GNN accuracy with only 0.07% loss and reduces NMIA accuracy by 34.82% and 32.01%.

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on November 15,2026 at 03:30:39 UTC from IEEE Xplore. Restrictions apply.

TABLE X
DEFENSES AGAINST DIFFERENT SHADOW DATASETS

(a) Defenses against NMIA in Policy ρ_1

Defenses	D'	Acc.$_{te}$ (%)	Acc.$_{at}$ (%)	Pr.$_{at}$ (%)	Re.$_{at}$ (%)
No Defense	Citeseer	74.00	53.32	52.42	71.94
	Pubmed	74.00	55.91	54.07	78.58
PriGraph (ours)	Citeseer	**73.12**	**50.81**	**50.00**	**100.00**
	Pubmed	**73.12**	**50.96**	**50.48**	**99.85**

(b) Defenses against LSIA in Policy ρ_2

Defenses	D'	Acc.$_{te}$ (%)	Acc.$_{at}$ (%)	Pr.$_{at}$ (%)	Re.$_{at}$ (%)
No Defense	Citeseer	74.00	85.22	82.74	89.01
	Pubmed	74.00	55.15	99.30	10.37
PriGraph (ours)	Citeseer	**72.74**	**50.00**	**50.00**	**100.00**
	Pubmed	**72.74**	**51.58**	**50.80**	**99.93**

TABLE XI
DEFENSES AGAINST DIFFERENT MODEL ARCHITECTURES AND DATA DISTRIBUTIONS

(a) Defenses against NMIA in Policy ρ_1

Defenses	Acc.$_{te}$ (%)	Acc.$_{at}$ (%)	Pr.$_{at}$ (%)	Re.$_{at}$ (%)
No Defense	67.21	52.81	53.42	43.87
PriGraph (ours)	**64.40**	**50.00**	**50.00**	**100.00**

(b) Defenses against LSIA in Policy ρ_2

Defenses	Acc.$_{te}$ (%)	Acc.$_{at}$ (%)	Pr.$_{at}$ (%)	Re.$_{at}$ (%)
No Defense	67.21	84.72	83.34	86.80
PriGraph (ours)	**66.74**	**50.00**	**50.00**	**100.00**

The impact of different data distributions: The defense performance of PriGraph has no effect under different data distributions. We assume that a shadow dataset D' with different distributions (i.e., Citseer and Pubmed) from the target dataset D Cora. The performance of PriGraph against NMIA and LSIA is shown in Table X. For GAT trained on Cora, the primary node classification task accuracy is 74.00%. As Table X(a) shows, when the shadow datasets are Citeseer and Pubmed while the original NMIA accuracy is 53.32% and 55.91%, PriGraph achieves primary task accuracy only 0.88% loss and reduces NMIA accuracy by 2.51% and 4.95%. Table X(b) demonstrates the performance of PriGraph under LISA. The original LSIA accuracy is 85.22% and 55.11%, PriGraph achieves primary task accuracy of 1.26% loss and reduces LSIA accuracy by 35.22% and 3.67%.

The impact of different model architectures and data distributions: The defense performance of PriGraph is also stable under the assumption of different model architectures and data distributions, as summarized in Table XI. We assume that a shadow model F' is GAT and shadow dataset D' is Citeseer, choosing GIN as the target GNN model architecture which is trained on Cora. For GIN trained on Cora, the primary node classification task accuracy is 67.21%. As Table XI(a) shows, when the shadow model architectures are GAT while the original NMIA accuracy is 52.81%, PriGraph achieves primary task accuracy by 2.81% loss and reduces NMIA accuracy by 2.81%. Table XI(b) demonstrates the performance of PriGraph under LISA. The original LSIA accuracy is 67.21%, PriGraph achieves primary task accuracy only 0.47% loss and reduces LISA accuracy by 34.72%.

TABLE XII
IMPACT OF PRIVACY-UTILITY HYPERPARAMETER

Parameters		Acc.$_{te}$ (%)	Acc.$_{at}^{node}$ (%)	Acc.$_{at}^{link}$ (%)
λ_1	λ_2			
0	0	70.07	67.10	87.59
1	0	69.95	50.03	83.59
0	1	67.30	50.03	77.52
1	1	65.62	50.03	81.67
1	10	45.07	50.03	70.79
10	1	68.38	50.03	80.12
1	2.5	63.01	50.03	72.56

D. Impact of Privacy-Utility Parameters

In PriGraph, we design a privacy-utility parameter to balance the node and link privacy protection and task performance. To further explore the impact of hyperparameters (i.e., λ_3) of G_{III} (cf. Section IV-C) on the defense performance of PriGraph, for the GIN model trained on Citeseer, we fix the hyperparameter of $\mathcal{L}_F$ in (12) to 1 and record the attack accuracy and model accuracy after deploying PriGraph under different λ_1 and λ_2. The hyperparameters λ_1 and λ_2 correspond to NMIA and LISA defense respectively, and we set them to 0 as the baseline for comparison with no defense. The experiment results are depicted in Table XII.

From Table XII, we can observe the correlation between PriGraph's defense performance and the hyperparameters. Specifically, when only λ_1 or λ_2 is set to 1, it decreases the success rate of corresponding NMIA and LISA attacks by 17.07% and 10.07%, respectively, within 3% model accuracy loss. We also find that the defense targeting one attack also works for another. This is because both NMIA and LSIA rely on node predictions to make inferences. Taking policy $\rho_1(\lambda_1{=}1, \lambda_2{=}0)$ as an example, when the membership properties of the nodes are hidden, the prediction consistency features between nodes also change accordingly.

Then, in policy ρ_3, we set all hyperparameters to 1. PriGraph can reduce NMIA and LSIA accuracy by 17.07% and 5.92% while the model accuracy is maintained at 65.62%. However, compared with single defense policies, in policy ρ_3, PirGraph has lower model accuracy and is 4.15% less effective against LISA than policy ρ_2. To enhance the defense effectiveness, we further amplify λ_1 and λ_2 to 10, respectively. As we can see, when $\lambda_1{=}1$ and $\lambda_2{=}10$, PriGraph achieves the best performance on both NMIA and LSIA. Nevertheless, its model accuracy drops to 45.07%, which is caused by the decrease in the proportion of $\mathcal{L}_f$ optimization terms. We also extract the classifications of classes of graph data and project them into

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on November 15,2026 at 03:30:39 UTC from IEEE Xplore. Restrictions apply.



Fig. 3. Node classification projected into a 2-dimension space using t-SNE. (a) No defense λ_1: 0 λ_2: 0, (b) With PriGraph λ_1: 1 and λ_2: 10.

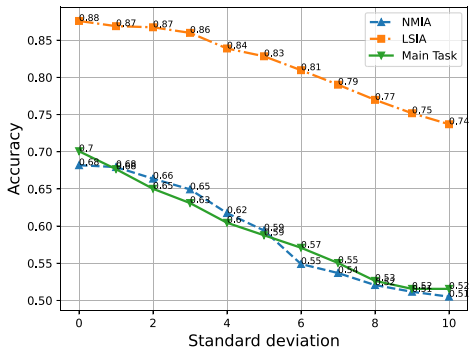


Fig. 4. NMIA/LSIA attack accuracy of Defense Auxiliary Classifiers and main task accuracy on different noise.

a 2-dimension space using t-Distributed Stochastic Neighbor Embedding (t-SNE) to show the effect of perturbations, as Fig. 3 shows. By comparison, we can see that with PriGraph the node classification can be clustered into a more compact position, which indicates the defense method can compact the boundary between node classes which can affect the primary task of GNN models. When λ_1=10 and λ_2=1, the model utility is recovered, while the attack performance of LSIA also improves. Given the above phenomenon, selecting proper hyperparameters in policy ρ_3 is crucial to achieving effective defense against both NMIA and LISA under acceptable model performance loss.

Following, we provide a tactic for selecting λ_1 and λ_2, which perturbs the model outputs with Gaussian noise to determine the vulnerability of the target model to privacy leakage under different attacks. Specifically, we add Gaussian noise of varying scales to the model logit outputs and utilize the Defense Auxiliary Classifier (cf. Section IV-C) to carry corresponding attacks on the processed node prediction vectors. Then, based on the attack performance loss of each classifier with fixed noise scales, we can evaluate the removal difficulty of each attack on the target model, and thereby select larger hyperparameters for attacks that are more difficult to defend against when generating perturbation.

The above selection strategy is lightweight since it does not require a tuning process. We take the GIN model trained on Citeseer as an example for a detailed explanation. First, we gradually scale up the standard deviation σ of the Gaussian noise from 0 to 10 and record the changes in NMIA/LSIA attack accuracy of Defense Auxiliary Classifiers. We plot them in Fig. 4, which shows that the two metrics decrease to varying degrees as the noise increases. Second, we adopt σ when the attack accuracy first drops below a threshold s (55% in this paper). With the σ of 6, we calculate the proportions p of accuracy reduction compared with no noise, which are 19.48% and 7.56% for NMIA and LSIA. The larger attack accuracy reduction corresponds smaller hyperparameter. Thus, we calculate the reciprocal of these two values as follows:

$$r = 1/(p + \epsilon) \tag{16}$$

where ϵ is the smoothing term, we set it to $1e^{-6}$ in this paper. The ratio of calculated results is adopted as the ratio of corresponding hyperparameters for NMIA and LSIA, which is 1 : 2.5 in our case. Besides, we also show the proportion p_t of main task accuracy reduction in Fig. 4, which is 18.52% with σ of 6. We donate the accuracy reduction proportion of NMIA as p_N, and we set the ratio of λ_1 and λ_3 to $p_t : p_N$. Specifically, when $p_N < p_t$, the attack is harder to defend, and λ_3 corresponds to drop for better defense. Conversely, λ_3 shifts for better performance. We fix λ_3 to 1, and calculate that the final settings of λ_1 and λ_2 are 1 and 2.5, respectively.

We also test the defense performance in Table XII under the above setting. In policy ρ_3, compared to other settings where model performance is maintained above 60%, our hyperparameters selection strategy has the lowest attack accuracy, which means an effective trade-off between model utility and defense performance. Notably, we find that ρ_3 introduces a larger accuracy loss compared with other policies, and thus ρ_1 and ρ_2 are more appropriate under a single privacy-preference.

VII. Related Work

Recent research has revealed that GNNs are vulnerable to privacy attacks. In this section, we mainly review existing privacy attacks and privacy preservations on GNNs.

A. Privacy Attacks on Graph Neural Networks

According to the goals of extracting sensitive information, privacy attacks on GNNs are various. Inference attacks [9], [10], [11], [12], [13], [32] on GNNs aims to infer some sensitive information from the training graph data such as node/subgraph membership, attributes of nodes, connections between nodes and properties of nodes. Launching the reconstruction attack [32] on GNNs is to reconstruct the target graph dataset, which is more challenging than a property inference attack. Additionally, the model extraction attack focuses on stealing the models [33].

Property inference attack: The property inference attack is to infer the properties that hold for a subset of the training data, and these properties are irrelevant to the training task of the target model. For instance, the training task of the model is to train a classifier that can distinguish genders, and the attacker can infer whether the data has the property of wearing glasses. Given the graph embedding, the properties (i.e., the numbers of nodes and links or the density) of the target graph can be determined [32]. The subgraph inference attack aims to infer whether a subgraph is contained in the target graph [9].

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on November 15,2026 at 03:30:39 UTC from IEEE Xplore. Restrictions apply.

Reconstruction attack: The dataset reconstruction attack aims to fully reconstruct the training data to obtain all the privacy, including the membership information and properties of the training data. Given the graph embedding, the graph reconstruction attack aims to reconstruct a graph that shares a similar graph topology and node attributes that are similar to the target graph [32].

B. Defense of Privacy Attacks on Graph Neural Networks

Several defenses against membership inference attacks have been proposed such as, DP approaches [14], [34], [35], federated GNNs [15], [36] and adversarial privacy-preserving approaches [16], [37].

Differential privacy approach: DP [38] has been widely used for privacy-preserving ML, which can provide a theoretical privacy guarantee of training data. DPNE [34] adopts the objective perturbation on the loss function of learning network embedding to achieve the differentially private graph embedding method. Leveraging the paradigm of knowledge distillation, a student GNN model is trained on the public graph in a DP manner to provide privacy guarantee [14]. Based on Local Differential Privacy (LDP), the locally private GNN [35] adds noises to perturb the features of user nodes at the first GNN layer. To reduce the negative effect of the injected noise, a graph convolution layer is designed as a denoising mechanism with multi-hop aggregation of the nodes' features.

Federated graph neural networks: Federated learning is a distributed ML technology that keeps user data locally. Users can collaborate on model training without uploading local data [39]. Wu et al. [15] proposes a federated framework for GNN training. The server aggregates the local gradients of the GNN model with LDP, which are trained on the user-item graph to protect user-sensitive information. Under graph data vertically partitioned setting, Chen et al. [36] split the computation graph into private data and non-private data with the related computations carried out by data holders and the semi-honest server respectively for privacy concerns.

Adversarial privacy-preserving approaches: Adversarial learning is adopted to defend against attribute inference attack-based graph embedding [16]. With disentangling and purging mechanisms, the graph embedding algorithm protects node representation privacy. Liao et al. [37] propose an adversarial training framework to locally filter out attributes that are more vulnerable to be inferred to defense attribute inference attack with acceptable accuracy loss.

Edge-based perturbation approaches: Besides focusing on attribute/output perturbations, recently, some studies have investigated the influence of edge on GNN privacy leakage and proposed edge perturbation strategies. Assisted by graph reconstruction algorithms, Wang et al. [29] add noise to the vector of nodes, and reconstruct the perturbed adjacency matrix to defend against NMIA. For defending against LSIA, zhong et al. [30] unveil the disparity in subgroup vulnerability (DSV), and propose *FairDefense*, which perturbs the structure of the training graph with a given probability to mitigate DSV.

Unfortunately, the current defenses do not effectively achieve the trade-off between utility and privacy. Defense for providing privacy preservation on both NMIA and LISA while maintaining high accuracy remains further exploration.

VIII. Conclusion

In this paper, we proposed PriGraph that leverages crafted adversarial perturbations to defend against score-based NMIA and LSIA on GNNs. PriGraph contains two components, i.e., defense auxiliary classifier and adversarial perturbation generator, which are used to simulate the process of inference attacks and generate the adversarial noises respectively to strongly protect the privacy with negligible accuracy loss of the target GNN models. The experimental results demonstrated that PriGraph can effectively defend against score-based inference attacks while maintaining the high utility of target GNN models with better performance than existing defenses.

References

[1] H. Ohtsuki, C. Hauert, E. Lieberman, and M. A. Nowak, "A simple rule for the evolution of cooperation on graphs and social networks," *Nature*, vol. 441, no. 7092, pp. 502–505, 2006.
[2] D. K. Duvenaud et al., "Convolutional networks on graphs for learning molecular fingerprints," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 2224–2232.
[3] M. Shen, J. Zhang, L. Zhu, K. Xu, and X. Du, "Accurate decentralized application identification via encrypted traffic analysis using graph neural networks," *IEEE Trans. Inf. Forensics Secur.*, vol. 16, pp. 2367–2380, 2021.
[4] T. Zhao, X. Zhang, and S. Wang, "Graphsmote: Imbalanced node classification on graphs with graph neural networks," in *Proc. 14th ACM Int. Conf. Web Search Data Mining*, 2021, pp. 833–841.
[5] A. Sankar, Y. Liu, J. Yu, and N. Shah, "Graph neural networks for friend ranking in large-scale social platforms," in *Proc. Web Conf.*, 2021, pp. 2535–2546.
[6] D. Jiang et al., "Could graph neural networks learn better molecular representation for drug discovery? a comparison study of descriptor-based and graph-based models," *J. Cheminformatics*, vol. 13, no. 1, pp. 1–23, 2021.
[7] W. Fan et al., "Graph neural networks for social recommendation," in *Proc. World Wide Web Conf.*, 2019, pp. 417–426.
[8] H. Du, M. Shen, R. Sun, J. Jia, L. Zhu, and Y. Zhai, "Malicious transaction identification in digital currency via federated graph deep learning," in *Proc. IEEE Conf. Comput. Commun. Workshops*, 2022, pp. 1–6.
[9] I. E. Olatunji, W. Nejdl, and M. Khosla, "Membership inference attack on graph neural networks," 2021, *arXiv:2101.06570*.
[10] X. He, R. Wen, Y. Wu, M. Backes, Y. Shen, and Y. Zhang, "Node-level membership inference attacks against graph neural networks," 2021, *arXiv:2102.05429*.
[11] B. Wu, X. Yang, S. Pan, and X. Yuan, "Adapting membership inference attacks to GNN for graph classification: Approaches and implications," in *Proc. IEEE Int. Conf. Data Mining*, 2021, pp. 1421–1426.
[12] X. He, J. Jia, M. Backes, N. Z. Gong, and Y. Zhang, "Stealing links from graph neural networks," in *Proc. 30th USENIX Secur. Symp.*, 2021, pp. 2669–2686.
[13] F. Wu, Y. Long, C. Zhang, and B. Li, "Linkteller: Recovering private edges from graph neural networks via influence analysis," 2021, *arXiv:2108.06504*.
[14] I. E. Olatunji, T. Funke, and M. Khosla, "Releasing graph neural networks with differential privacy guarantees," 2021, *arXiv:2109.08907*.
[15] C. Wu, F. Wu, Y. Cao, Y. Huang, and X. Xie, "FeDGNN: Federated graph neural network for privacy-preserving recommendation," 2021, *arXiv:2102.04925*.
[16] K. Li, G. Luo, Y. Ye, W. Li, S. Ji, and Z. Cai, "Adversarial privacy-preserving graph embedding against inference attack," *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6904–6915, Apr. 2021.
[17] C. Chen et al., "Vertically federated graph neural network for privacy-preserving node classification," 2020, *arXiv:2005.11903*.

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on November 15,2026 at 03:30:39 UTC from IEEE Xplore. Restrictions apply.

[18] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1024–1034.
[19] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," 2017, *arXiv:1710.10903*.
[20] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?," 2018, *arXiv:1810.00826*.
[21] H. Lyu and J. Luo, "Understanding political polarization via jointly modeling users, connections and multimodal contents on heterogeneous graphs," in *Proc. 30th ACM Int. Conf. Multimedia*, 2022, pp. 4072–4082.
[22] F. Gafarov, A. Berdnikov, and P. Ustin, "Predicting success of the online social network user by graph neural networks," *Int. J. Adv. Intell. Informat.*, vol. 8, no. 3, pp. 285–298, 2022.
[23] E. Dai et al., "A comprehensive survey on trustworthy graph neural networks: Privacy, robustness, fairness, and explainability," 2022, *arXiv:2204.08570*.
[24] M. Shen, C. Li, Q. Li, H. Lu, L. Zhu, and K. Xu, "Transferability of white-box perturbations: Query-efficient adversarial attacks against commercial DNN services," in *Proc. 33rd USENIX Secur. Symp.*, 2024.
[25] X. Zheng et al., "Graph neural networks for graphs with heterophily: A survey," 2022, *arXiv:2202.07082*.
[26] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI Mag.*, vol. 29, no. 3, pp. 93–93, 2008.
[27] B. Rozemberczki, C. Allen, and R. Sarkar, "Multi-scale attributed node embedding," *J. Complex Netw.*, vol. 9, no. 2, 2021, Art. no. cnab014.
[28] A. P. Garcıá-Plaza, V. Fresno, R. M. Unanue, and A. Zubiaga, "Using fuzzy logic to leverage HTML markup for web page representation," *IEEE Trans. Fuzzy Syst.*, vol. 25, no. 4, pp. 919–933, Aug. 2017.
[29] K. Wang, J. Wu, T. Zhu, W. Ren, and Y. Hong, "Defense against membership inference attack in graph neural networks through graph perturbation," *Int. J. Inf. Secur.*, vol. 22, no. 2, pp. 497–509, 2023.
[30] D. Zhong, R. Yu, K. Wu, X. Wang, J. Xu, and W. H. Wang, "Disparate vulnerability in link inference attacks against graph neural networks," in *Proc. Privacy Enhancing Technol.*, vol. 2023, no. 4, pp. 149–169, 2023.
[31] T. N. Kipf and M. Welling, "Variational graph auto-encoders," 2016, *arXiv:1611.07308*.
[32] Z. Zhang, M. Chen, M. Backes, Y. Shen, and Y. Zhang, "Inference attacks against graph neural networks," in *Proc. 31th USENIX Secur. Symp.*, 2022, pp. 1–18.
[33] Y. Shen, X. He, Y. Han, and Y. Zhang, "Model stealing attacks against inductive graph neural networks," 2021, *arXiv:2112.08331*.
[34] D. Xu, S. Yuan, X. Wu, and H. Phan, "Dpne: Differentially private network embedding," in *Proc. Pacific-Asia Conf. Knowl. Discov. Data Mining*, Springer, 2018, pp. 235–246.
[35] S. Sajadmanesh and D. Gatica-Perez, "Locally private graph neural networks," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2021, pp. 2130–2145.
[36] J. Zhou et al., "Vertically federated graph neural network for privacy-preserving node classification," 2020, *arXiv:2005.11903*.
[37] P. Liao et al., "Information obfuscation of graph neural networks," in *Proc. Int. Conf. Mach. Learn.*, PMLR, 2021, pp. 6600–6610.
[38] C. Dwork, "Differential privacy," in *Encyclopedia of Cryptography and Security*. Berlin, Germany: Springer, 2011, pp. 338–340.
[39] S. Sun, W. Chen, J. Bian, X. Liu, and T.-Y. Liu, "Slim-DP: A multi-agent system for communication-efficient distributed deep learning," in *Proc. 17th Int. Conf. Auton. Agents MultiAgent Syst. Int. Found. Auton. Agents Multiagent Syst.*, 2018, pp. 721–729.

Authorized licensed use limited to: BEIJING INSTITUTE OF TECHNOLOGY. Downloaded on November 15,2026 at 03:30:39 UTC from IEEE Xplore. Restrictions apply.