Privacy-Preserving Representation Learning on Graphs: A Mutual Information Perspective

Binghui Wang
Illinois Institute of Technology
& Duke University
bwang70@iit.edu

Jiayi Guo
Tsinghua University
guo-jy20@mails.tsinghua.edu.cn

Ang Li, Yiran Chen, and Hai Li
Duke University
ang.li630,yiran.chen,hai.li@duke.edu

ABSTRACT

Learning with graphs has attracted significant attention recently. Existing representation learning methods on graphs have achieved state-of-the-art performance on various graph-related tasks such as node classification, link prediction, etc. However, we observe that these methods could leak serious private information. For instance, one can accurately infer the links (or node identity) in a graph from a node classifier (or link predictor) trained on the learnt node representations by existing methods. To address the issue, we propose a privacy-preserving representation learning framework on graphs from the *mutual information* perspective. Specifically, our framework includes a primary learning task and a privacy protection task, and we consider node classification and link prediction as the two tasks of interest. Our goal is to learn node representations such that they can be used to achieve high performance for the primary learning task, while obtaining performance for the privacy protection task close to random guessing. We formally formulate our goal via mutual information objectives. However, it is intractable to compute mutual information in practice. Then, we derive tractable variational bounds for the mutual information terms, where each bound can be parameterized via a neural network. Next, we train these parameterized neural networks to approximate the true mutual information and learn privacy-preserving node representations. We finally evaluate our framework on various graph datasets.

CCS CONCEPTS

• **Security and privacy**; • **Computing methodologies** → **Machine learning**;

KEYWORDS

Graph representation learning, privacy, mutual information

ACM Reference Format:
Binghui Wang, Jiayi Guo, and Ang Li, Yiran Chen, and Hai Li. 2021. Privacy-Preserving Representation Learning on Graphs: A Mutual Information Perspective. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '21), August 14–18, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3447548.3467273

KDD '21, August 14–18, 2021, Virtual Event, Singapore

ACM ISBN 978-1-4503-8332-5/21/08…$15.00
https://doi.org/10.1145/3447548.3467273

1 INTRODUCTION

Graph is a powerful tool to represent diverse types of data including social networks, chemical networks, etc. Learning with graphs has been an active research topic recently and various representation learning methods on graphs [3, 4, 7, 8, 10, 11, 14, 17, 18, 22, 25–27, 30–32, 34, 36–38] have been proposed. Given a graph, representation learning on the graph aims to learn a node embedding function that maps each node in the observational space to a latent space by capturing the graph structural information. The learned node embedding function can be used for many graph-related tasks. For instance, node classification and link prediction are two basic tasks. Node classification aims to predict the label of unlabeled nodes in a graph based on a set of labeled nodes, while link prediction aims to predict the link status between a pair of nodes, based on a set of observed positive/negative links[1] in a graph.

Existing representation learning methods on graphs have demonstrated to achieve state-of-the-art performance on these tasks, e.g., node classification [14, 32] and link prediction [39]. However, due to that the learnt node representations are not task-specific, we note that existing methods could unintendedly leak important information. For instance, we observe that one can accurately infer the links in a graph from a node classifier trained on the learnt node representations; and one can also predict the node labels from a link predictor based on the node representations (See Table 2(b) in Section 5.2). Such information leakage could involve serious privacy issues. Take users in a social network (e.g., Twitter) as an example. Some users (e.g., celebrities) in the social network may just want to make their identities known to the public, but they do not want to expose their private social relationship (e.g., family relationship). Some other users (e.g., malicious users) do not want to reveal their identities, but do want to expose their social relationship with normal users to make themselves also look normal. Suppose the social network has deployed a user identity classification system (i.e., node classification) or friendship recommendation system (i.e., link prediction) using certain graph representation learning method. Then, if an adversary (e.g., insider) knows the method, he could thus infer the user's private friendship links/identities.

Our work: In this paper, we aim to address the above privacy violation issue and propose a privacy-preserving representation learning framework on graphs from the *mutual information* perspective. Specifically, our framework includes a primary learning task and a privacy protection task. We consider node classification and link prediction as the two tasks of interest[2]. Under this context, our framework includes three modules: node embedding function

[1] Positive means there exists a link between a pair of nodes, and negative means no link between them.
[2] Note that our framework can be generalized to any graph-related tasks.

(for node representation learning), link predictor (uses the node representations to perform link prediction), and node classifier (uses node representations to perform node classification).

Then, we target the following two problems:

- **Problem 1: Link prediction + node privacy protection.** The primary learning task is learning node representations such that the link predictor can achieve high link prediction performance, and the privacy protection task is to enforce that the learnt node representations cannot be used by the node classifier to accurately infer the node label.
- **Problem 2: Node classification + link privacy protection.** The primary learning task is learning node representations such that the node classifier can achieve high node classification performance, and the privacy protection task is to enforce that the learnt node representations cannot be used by the link predictor to accurately infer the link status.

We formally formulate our problems using two *mutual information* objectives, which are defined on the primary learning task and the privacy protection task, respectively. Then, for Problem 1, the goal of the two objectives is to learn an embedding function such that: 1) The representations of a pair of nodes retain as much information as possible of the respective link status. Intuitively, when the pair of node representations keep the most information about the link status, the link predictor trained on the node representations could have the highest link prediction performance. 2) The node representation contains as less information as possible about the node label. Intuitively, when the learnt node representation preserves the least information on the node label, the node classifier trained on the node representations could have the lowest node classification performance. Similarly, for Problem 2, the goal is to learn an embedding function such that: 1) The node representation contains as much information as possible to facilitate predicting the node label. 2) The representations of node pairs retain as less information as possible to prevent inferring the link status.

However, the mutual information terms are challenging to calculate in practice, as they require to compute an intractable posterior distribution. Motivated by mutual information neural estimators [2, 5, 6], we convert the intractable mutual information terms to be the tractable ones via introducing variational (upper and lower) bounds. Specifically, each variational bound involves a variational posterior distribution, and it be parameterized via a neural network. Estimating the true mutual information thus reduces to training the parameterized neural networks. Furthermore, we propose an alternative training algorithm to train these neural networks.

We finally evaluate our framework on multiple benchmark graph datasets. Experimental results demonstrate that without privacy protection, the learnt node representations by existing methods for the primary learning task can be also used to obtain high performance on the privacy protection task. However, with our proposed privacy-protection mechanism, the learnt node representations can only be used to achieve high performance for the primary learning task, while obtaining the performance for the privacy protection task close to random guessing. Our key contributions can be summarized as follows:

- We propose the first work to study privacy-preserving representation learning on graphs.
- We formally formulate our problems via mutual information objectives and design tractable algorithms to estimate intractable mutual information.
- We evaluate our framework on various graph datasets and results demonstrate the effectiveness of our framework for privacy-preserving representation learning on graphs.

2 RELATED WORK

2.1 Representation Learning on Graphs

Various representation learning methods on graphs have been proposed [3, 4, 7, 8, 10, 11, 14, 17, 18, 22, 25–27, 30–32, 34–38] in the past several years. Graph representation learning based on graph neural networks have exhibit stronger performance than random walk- and factorization-based methods [3, 10, 22, 24, 30]. For instance, Graph Convolutional Network (GCN) [14] is motivated by spectral graph convolutions [9] and learns node representations, based on the graph convolutional operator, for node classification. HGCN [4] leverages both the expressiveness of GCN and hyperbolic geometry to learn node representations. Specifically, HGCN designs GCN operations in the hyperbolic space and maps Euclidean node features to embeddings in hyperbolic spaces with trainable curvatures at each layer. The learnt node representations make HGCN achieve both higher node classification performance and link prediction performance than Euclidean space-based GCNs.

A few recent works [21, 29, 33] propose to leverage mutual information to perform *unsupervised* graph representation learning. For instance, Peng et al. [21] propose a concept called Graphical Mutual Information (GMI), which measures the correlation between the entire graph and high-level hidden representations, and is invariant to the isomorphic transformation of input graphs. By virtue of GMI, the authors design an unsupervised model trained by maximizing GMI between the input and output of a graph neural encoder. The learnt node representations of GMI are used for node classification and link prediction and GMI achieves better performance than other unsupervised graph representation learning methods.

Note that although our framework also adopts mutual information, its goal is completely different from mutual information-based graph representation learning methods. Our goal is to learn privacy-preserving node representations that consider both a primary learning task and a privacy protection task, while these existing methods mainly focus on learning node representations that achieve high performance for a primary learning task.

2.2 Mutual Information Estimation

Estimating mutual information accurately between high dimensional continuous random variable is challenging [2]. To obtain differentiable and scalable mutual information estimation, recent methods [1, 2, 6, 12, 20, 23] propose to first derive mutual information (upper or lower) bounds by introducing auxiliary variational distributions and then train parameterized neural networks to estimate variational distributions and approximate true mutual information. For instance, MINE [2] treats mutual information as the KL divergence between the joint and marginal distributions, converts it into the dual representation, and obtains a lower mutual information bound. Cheng et al. [6] propose a Contrastive Log-ratio Upper Bound (CLUB) of mutual information. CLUB bridges

mutual information estimation with contrastive learning [20], and mutual information is estimated by the difference of conditional probabilities between positive and negative sample pairs.

2.3 Other Privacy-Preserving Techniques

Differential privacy (DP) and homomorphic encryption (HE) are two other types of methods that ensure privacy protection. However, DP incurs utility loss and HE incurs intolerable computation overheads. Mutual information is a recent methodology that protects privacy based on information theory [16]. Compared to DP and HE, the mutual information-based method is demonstrated to be more efficient or/and effective. Motivated by these advantages, we adopt mutual information to study privacy-preserving graph representation learning.

3 BACKGROUND & PROBLEM DEFINITION

3.1 Representation Learning on Graphs

Let $G = (\mathcal{V}, \mathcal{E}, \mathbf{A}, \mathbf{X})$ be an attributed graph, where $v \in \mathcal{V}$ is a node and $N = |V|$ is the total number of nodes; $(u, v) \in \mathcal{E}$ is a link between u and v; $\mathbf{A} \in \mathbb{R}^{N \times N}$ is the adjacency matrix, where $A_{u,v} = 1$, if $(u, v) \in \mathcal{E}$ and $A_{u,v} = 0$, otherwise; and $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N] \in \mathbb{R}^{D \times N}$ is the node feature matrix with $\mathbf{x}_u \in \mathbb{R}^D$ the node u's feature vector. The purpose of representation learning on graphs is to learn a node embedding function $f_\theta : \mathbb{R}^D \times \mathbb{R}^{N \times N} \to \mathbb{R}^d$, parameterized by θ, that maps each node u's feature vector $\mathbf{x}_u$ in the observational space to a feature vector $\mathbf{z}_u$ in a latent space by capturing the graph structural information, i.e., $\mathbf{z}_u = f_\theta(\mathbf{x}_u, \mathbf{A}) \in \mathbb{R}^d$, where we call $\mathbf{z}_u$ the *node representation*. The learnt node representations can be used for various graph-related tasks. In this paper, we mainly focus on two tasks of interest: node classification and link prediction.

Node classification. Each node $v \in V$ in the graph G is associated with a label y_v from a label set $\mathcal{Y} = \{1, 2, \cdots, C\}$. Then, given a set of $\mathcal{V}_L \subset \mathcal{V}$ labeled nodes with the node representations $\{(\mathbf{z}_u, y_u)\}_{u \in \mathcal{V}_L}$ as the training nodes, node classification is to take the training nodes and their learnt representations as input and learn a node classifier $g_\psi : \mathbb{R}^d \to \mathbb{R}^{|\mathcal{Y}|}$, parameterized by ψ, that has a minimal loss on the training nodes. Suppose we use the cross-entropy loss. Then, the objective function of node classification is defined as follows:

$$\min_{\psi} \sum_{v \in \mathcal{V}_L} CE(g_\psi(\mathbf{z}_v), y_v) = - \sum_{v \in \mathcal{V}_L} \mathbf{1}_{y_v} \circ \log g_\psi(\mathbf{z}_v),$$

where $\mathbf{1}_{y_v}$ is an indicator vector whose y_v-th entry is 1, and 0, otherwise. With the learnt ψ^*, we can predict the label for each unlabeled node $u \in \mathcal{V} \setminus \mathcal{V}_L$ as $\hat{y}_u = \arg\max_i \ g_{\psi^*}(\mathbf{z}_u)_i$.

Link prediction. Given a set of positive links $\mathcal{E}_p \subset \mathcal{E}$ (i.e., $A_{uv} = 1$) and a set of negative links $\mathcal{E}_n \not\subset \mathcal{E}$ (i.e., $A_{uv} = 0$) as the training links. Link prediction is to take the training links and the associated nodes' representations as input and learn a link predictor $h_\phi : \mathbb{R}^d \times \mathbb{R}^d \to [0, 1]$, parameterized by ϕ, that has a minimal reconstruction error on the training links. Specifically, the objective function of link prediction we consider is as follows:

$$\min_{\phi} \sum_{(u,v) \in \mathcal{E}_p \cup \mathcal{E}_n} CE(h_\phi(\mathbf{z}_u, \mathbf{z}_v), A_{uv}) = \sum_{(u,v) \in \mathcal{E}_p \cup \mathcal{E}_n} -A_{uv} \circ \log h_\phi(\mathbf{z}_u, \mathbf{z}_v).$$

With the learnt ϕ^*, we predict a link between unlabeled pair of nodes u and v if $h^*_\phi(\mathbf{z}_u, \mathbf{z}_v) > 0.5$, predict no link, otherwise.

3.2 Problem Definition

Node classification and link prediction are two graph-related tasks. However, in existing graph representation learning methods, given a primary learning task, one can also obtain promising performance for the other task with the learnt node representations. That is, one can accurately infer the link status between nodes (or infer the node label) even if the primary learning task is node classification (or link prediction) (See Table 2(a) and Table 3(a) in Section 5.2). Such a phenomenon could induce privacy concerns in practical applications. For instance, a celebrity in Twitter just wants to share his identity, but does not want to reveal his private family relationship. A malicious user in Twitter does not want to expose his identity, but does want to make his social relationship with normal users known to the public, in order to let himself also look normal.

We highlight that the root cause of the above consequences is that when learning node representations for a task, existing methods do not consider protecting privacy for other tasks. To address the issue, we are motivated to propose privacy-preserving representation learning methods on graphs. We mainly consider the node classification and link prediction tasks, where one is the primary learning task and the other is the privacy protection task. Therefore, our problem involves three modules: node embedding function (for node representation learning), link predictor (uses the node representations to perform link prediction), and node classifier (uses node representations to perform node classification). In particular, we study the following two problems, each involving a primary learning task and a privacy protection task.

- **Problem 1: Link prediction + node privacy protection.** In this problem, our primary learning task is learning node representations such that the link predictor can achieve high link prediction performance, and our privacy protection task is to enforce that the learnt node representations cannot be used by the node classifier to accurately infer the node label.
- **Problem 2: Node classification + link privacy protection.** In this problem, our primary learning task is learning node representations such that the node classifier can achieve high node classification performance, and our privacy protection task is to enforce the learnt node representations cannot be used by the link predictor to accurately infer the link status.

In the next section, we will formally formulate our two problems and design algorithms to solve the problems.

4 PRIVACY-PRESERVING REPRESENTATION LEARNING ON GRAPHS

We formulate our problems via *mutual information*. Specifically, we define two mutual information objectives that are associated with the primary learning task and the privacy protection task, respectively. However, the mutual information terms are challenging to calculate in practice. Then, we convert them to be the tractable ones via designing variational bounds, and each bound can be estimated by a parameterized neural network. Finally, we propose algorithms to train these neural networks to achieve high performance for the primary learning task, and performance close to random guessing for the privacy protection task. Figure 1 overviews our privacy-preserving graph representation learning framework.

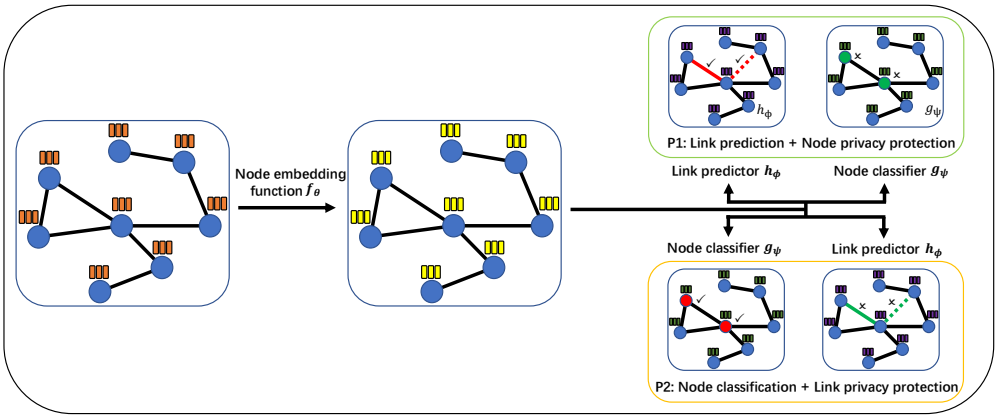


Figure 1: Overview of our privacy-preserving graph representation learning framework. It consists of a primary learning task and a privacy protection task. Our aim is to learn node representations such that they can be used to achieve high performance for the primary learning task, while obtaining performance for the privacy protection task close to random guessing.

4.1 Problem 1: Link Prediction with Node Privacy Protection

4.1.1 Formulating Problem 1 with mutual information objectives. Suppose we have a set of samples $\{\mathbf{x}_u, y_u, A_{uv}\}$ consisting of node features $\mathbf{x}_u$, node label y_u, link status A_{uv}. We define the probability distribution associated with a node (i.e., node features $\mathbf{x}_u$ and node label y_u) as $p(\mathbf{x}_u, y_u)$. Moreover, we define the probability distribution associated with a link (i.e., a pair of node features $\mathbf{x}_u$ and $\mathbf{x}_v$, and the associated link status A_{uv}) as $p(\mathbf{x}_u, \mathbf{x}_v, A_{uv})$[3].

Our goal is to learn an embedding function f_θ to transform $\mathbf{x}_u$ to the representation $\mathbf{z}_u = f_\theta(\mathbf{x}_u)$ such that: 1) The node representation of a pair (e.g., u and v) retain as much information as possible on the link status (e.g., A_{uv}). Intuitively, when the representations of the node pair keep the most information about the link, the link predictor h_ϕ trained on the node representations could have the highest link prediction performance. 2) The node representation (e.g., u) contains as less information as possible about the node label (e.g., y_u). Intuitively, when the node representation preserves the least information on the node label, the node classifier g_ψ trained on the node representation could have the lowest node classification performance. Formally, to achieve 1) and 2), we have the following two respective mutual information objectives:

$$\textbf{Link prediction: } \max_\theta I(A_{uv}; \mathbf{z}_u, \mathbf{z}_v), \tag{1}$$

$$\textbf{Node privacy protection: } \min_\theta I(\mathbf{z}_u; y_u), \tag{2}$$

where $\mathbf{z}_u = f_\theta(\mathbf{x}_u)$ is the random vector after applying the embedding function on $\mathbf{x}_u$. $I(A_{uv}; \mathbf{z}_u, \mathbf{z}_v)$ is the mutual information between A_{uv} and the joint $(\mathbf{z}_u, \mathbf{z}_v)$, which indicates the information $(\mathbf{z}_u, \mathbf{z}_v)$ kept for the link variable A_{uv}. We maximize such mutual information to enhance the link prediction performance. $I(\mathbf{z}_u; y_u)$ is the mutual information between $\mathbf{z}_u$ and y_u and indicates the information $\mathbf{z}_u$ preserves for the label variable y_u. We minimize such mutual information to protect node privacy. Ideally, if $I(\mathbf{z}_u; y_u) = 0$, no node's label can be inferred from its node representation, i.e., no node classifier can perform better than random guessing.

4.1.2 Estimating mutual information via tractable variation lower bound and upper bound. In practice, the mutual information terms in Equations 1 and 2 are hard to compute as the random variables are potentially high-dimensional and mutual information terms require to know posterior distributions that are challenging to calculate. To address the challenge, we are inspired by existing mutual information neural estimation methods [1, 2, 6, 12, 20, 23], which convert the intractable mutual information calculation to the tractable one by designing variational bounds. Specifically, we first obtain a mutual information variational lower bound for Equation 1 and a mutual information variational upper bound for Equation 2 by introducing two auxiliary posterior distributions, respectively. Then, we parameterize each auxiliary distribution with a neural network, and approximate the true posteriors by maximizing the variational lower bound and minimizing the variational upper bound through training the involved neural network.

Maximizing the mutual information in Equation 1. To solve Equation 1, we derive the following variational lower bound:

$$\begin{aligned}
&I(A_{uv}; \mathbf{z}_u, \mathbf{z}_v) \\
=&H(A_{uv}) - H(A_{uv}|\mathbf{z}_u, \mathbf{z}_v) \\
=&H(A_{uv}) + \mathbb{E}_{p(\mathbf{z}_u, \mathbf{z}_v, A_{uv})}[\log p(A_{uv}|\mathbf{z}_u, \mathbf{z}_v)]] \\
=&H(A_{uv}) + \mathbb{E}_{p(\mathbf{z}_u, \mathbf{z}_v, A_{uv})}[KL\big(p(\cdot|\mathbf{z}_u, \mathbf{z}_v)||q_\phi(\cdot|\mathbf{z}_u, \mathbf{z}_v)\big)] \\
&\quad + \mathbb{E}_{p(\mathbf{z}_u, \mathbf{z}_v, A_{uv})}[\log q_\phi(A_{uv}|\mathbf{z}_u, \mathbf{z}_v)]] \\
\geq&H(A_{uv}) + \mathbb{E}_{p(\mathbf{z}_u, \mathbf{z}_v, A_{uv})}[\log q_\phi(A_{uv}|\mathbf{z}_u, \mathbf{z}_v)] \\
:=&I_{vLB}(A_{uv}; \mathbf{z}_u, \mathbf{z}_v),
\end{aligned} \tag{3}$$

where $KL[q(\cdot)||p(\cdot)]$ is the Kullback-Leibler divergence between two distributions $q(\cdot)$ and $p(\cdot)$ and is nonnegative. q_ϕ is an (arbitrary) auxiliary posterior distribution. $I_{vLB}(A_{uv}; \mathbf{z}_u, \mathbf{z}_v)$ is the variational lower bound of the true mutual information and $H(A_{uv})$ is a constant. Note that the lower bound is tight when the auxiliary distribution q_ϕ becomes the true posterior distribution p.

[3]Note that for notation simplicity, we slightly abuse the notations A_{uv}, $\mathbf{x}_u$, and y_u. That is, these notations are originally used for the link status between nodes u and v, u's raw feature vector and u's label in the graph G. Here, we also use them to indicate random variable/vector.

Our target now is to maximize the lower bound by estimating the auxiliary posterior distribution q_ϕ via a parameterized neural network. Specifically, we have

$$\begin{aligned}
&\max_\theta I_{vLB}(A_{uv}; \mathbf{z}_u, \mathbf{z}_v) \\
\Leftrightarrow &\max_\theta \max_\phi \mathbb{E}_{p(A_{uv}, \mathbf{z}_u, \mathbf{z}_v)} [\log q_\phi(A_{uv}|\mathbf{z}_u, \mathbf{z}_v)] \\
= &\max_\theta \max_\phi \mathbb{E}_{p(A_{uv}, \mathbf{x}_u, \mathbf{x}_v)} [\log q_\phi(A_{uv}|f_\theta(\mathbf{x}_u), f_\theta(\mathbf{x}_v))] \qquad (4)
\end{aligned}$$

Minimizing the mutual information in Equation 2. To solve Equation 2, we leverage the variational upper bound in [6]:

$$\begin{aligned}
&I(\mathbf{z}_u; y_u) \\
\leq &I_{vCLUB}(\mathbf{z}_u; y_u) \\
= &\mathbb{E}_{p(\mathbf{z}_u, y_u)} [\log q_\psi(y_u|\mathbf{z}_u)] - \mathbb{E}_{p(\mathbf{z}_u)p(y_u)} [\log q_\psi(y_u|\mathbf{z}_u)],
\end{aligned}$$

where $q_\psi(y_u|\mathbf{z}_u)$ is an auxiliary distribution of $p(y_u|\mathbf{z}_u)$ that needs to satisfy the following condition [6]:

$$KL(p(\mathbf{z}_u, y_u)||q_\psi(\mathbf{z}_u, y_u)) \leq KL(p(\mathbf{z}_u)p(y_u)||q_\psi(\mathbf{z}_u, y_u)). \qquad (5)$$

To achieve Inequality 5, we need to minimize:

$$\begin{aligned}
&\min_\psi KL(p(\mathbf{z}_u, y_u)||q_\psi(\mathbf{z}_u, y_u)) \\
= &\min_\psi KL(p(y_u|\mathbf{z}_u)||q_\psi(y_u|\mathbf{z}_u)) \\
= &\min_\psi \mathbb{E}_{p(\mathbf{z}_u, y_u)} [\log p(y_u|\mathbf{z}_u)] - \mathbb{E}_{p(\mathbf{z}_u, y_u)} [\log q_\psi(y_u|\mathbf{z}_u))] \\
\Leftrightarrow &\max_\psi \mathbb{E}_{p(\mathbf{z}_u, y_u)} [\log q_\psi(y_u|\mathbf{z}_u)], \qquad (6)
\end{aligned}$$

where we have the last Equation because the first term in the second-to-last Equation is irrelevant to ψ.

Finally, achieving Equation 2 becomes solving the following adversarial training objective:

$$\begin{aligned}
&\min_\theta \min_\psi I_{vCLUB}(\mathbf{z}_u; y_u) \\
\Leftrightarrow &\min_\theta \max_\psi \mathbb{E}_{p(\mathbf{z}_u, y_u)} [\log q_\psi(y_u|\mathbf{z}_u)] \\
= &\min_\theta \max_\psi \mathbb{E}_{p(\mathbf{x}_u, y_u)} [\log q_\psi(y_u|f_\theta(\mathbf{x}_u))] \qquad (7)
\end{aligned}$$

Remark. The above objective function can be interpreted as an adversarial game between an adversary q_ψ who aims to infer the label y_u from $\mathbf{z}_u$ and a defender (i.e., the embedding function f_θ) who aims to protect the node privacy from being inferred.

Implementation via training parameterized neural networks. We solve Equation 4 and Equation 7 in practice via training two parameterized neural networks associated with the two auxiliary posterior distributions q_ϕ and q_ψ. With it, we expect to obtain high link prediction performance for our primary learning task and low node classification performance for our privacy protection task.

To solve Equation 4, we first sample a set of triplets $\{A_{uv}, \mathbf{x}_u, \mathbf{x}_v\}$ from the graph G. Then, we parameterize the variational posterior distribution q_ϕ via a link predictor h_ϕ defined on the node representations $f_\theta(\mathbf{x}_u)$ and $f_\theta(\mathbf{x}_v)$ of the sampled node pairs u and v. Suppose we are given a set of positive links $\mathcal{E}_p$ and a set of negative links $\mathcal{E}_n$, then we have

$$\begin{aligned}
&\max_\theta \max_\phi \mathbb{E}_{p(A_{uv}, \mathbf{x}_u, \mathbf{x}_v)} [\log q_\phi(A_{uv}|f_\theta(\mathbf{x}_u), f_\theta(\mathbf{x}_v))] \\
\approx &\max_\theta \max_\phi \sum_{(u,v) \in \mathcal{E}_p \cup \mathcal{E}_n} -CE(h_\phi(f_\theta(\mathbf{x}_u), f_\theta(\mathbf{x}_v)), A_{uv}) \\
= &\min_\theta \min_\phi \sum_{(u,v) \in \mathcal{E}_p \cup \mathcal{E}_n} CE(h_\phi(f_\theta(\mathbf{x}_u), f_\theta(\mathbf{x}_v)), A_{uv}). \qquad (8)
\end{aligned}$$

To solve Equation 7, we first sample a set of labeled nodes $\{\mathbf{x}_u, y_u\}$, and then we parameterize q_ψ via a node classifier g_ψ defined on the node representations $\{f_\theta(\mathbf{x}_u)\}$ of these labeled nodes. Suppose we sample a set of labeled nodes $\mathcal{V}_L$, then we have

$$\begin{aligned}
&\min_\theta \max_\psi \mathbb{E}_{p(\mathbf{x}_u, y_u)} [\log q_\psi(y_u|f_\theta(\mathbf{x}_u))] \\
\approx &\min_\theta \max_\psi \sum_{v \in \mathcal{V}_L} -CE(g_\psi(f_\theta(\mathbf{x}_v)), y_v). \qquad (9)
\end{aligned}$$

Combining Equation 8 and Equation 9, we have the final objective function for our **Problem 1** as follows:

$$\begin{aligned}
\min_\theta \big(\lambda \min_\phi &\sum_{(u,v) \in \mathcal{E}_p \cup \mathcal{E}_n} CE(h_\phi(f_\theta(\mathbf{x}_u), f_\theta(\mathbf{x}_v)), A_{uv}) \\
&- (1-\lambda) \max_\psi \sum_{v \in \mathcal{V}_L} CE(g_\psi(f_\theta(\mathbf{x}_v)), y_v)\big), \qquad (10)
\end{aligned}$$

where λ is a trade-off factor to balance between achieving high link prediction performance and low node classification performance.

Note that Equation 10 involves optimizing three neural networks: the node embedding function f_θ, the link predictor h_ϕ, and the node classifier g_ψ. We alternatively train the three neural networks. Specifically, in each round, we perform several iterations of gradient descent to update ϕ, several iterations of gradient ascent to update ψ, and several iterations of gradient descent to update θ. We iteratively perform these steps until reaching a predefined maximal number of rounds or the convergence condition. Algorithm 1 in Appendix illustrates the training procedure of these networks.

4.2 Problem 2: Node Classification with Link Privacy Protection

4.2.1 Formulating Problem 2 with mutual Information. In this problem, our goal is to learn an embedding function f_θ to transform $\mathbf{x}_u$ to the representation $\mathbf{z}_u = f_\theta(\mathbf{x}_u)$ such that: 1) The representation of a node (e.g., u) contains as much information as possible to facilitate predicting the node label (e.g., y_u). 2) The representation of node pairs (e.g., u and v) retain as less information as possible to prevent inferring the link status (e.g., A_{uv}). Formally, to achieve 1) and 2), we have the following two mutual information objectives:

$$\textbf{Node classification: } \max_\theta I(\mathbf{z}_u; y_u), \qquad (11)$$

$$\textbf{Link privacy protection: } \min_\theta I(A_{uv}; \mathbf{z}_u, \mathbf{z}_v), \qquad (12)$$

4.2.2 Estimating mutual information via tractable variation lower bound and upper bound. Similarly, we first obtain a lower bound for Equation 11 and an upper bound for Equation 12 by introducing two auxiliary posterior distributions, respectively. Then, we parameterize each auxiliary distribution with a neural network, and train each neural network to maximize the lower bound or minimize the upper bound, respectively.

Maximizing the mutual information in Equation 11. To solve Equation 11, we have the following variational lower bound

$$\begin{aligned} & I(\mathbf{z}_u; y_u) \\ =& H(y_u) - H(y_u|\mathbf{z}_u) \\ =& H(y_u) + \mathbb{E}_{p(\mathbf{z}_u, y_u)}[\log P(y_u|\mathbf{z}_u)] \\ =& H(y_u) + \mathbb{E}_{p(\mathbf{z}_u, y_u)}[KL(p(\cdot|\mathbf{z}_u)||q_\psi(\cdot|\mathbf{z}_u)] \\ & + \mathbb{E}_{p(\mathbf{z}_u, y_u)} \log p(y_u|\mathbf{z}_u)] \\ \geq& H(y_u) + \mathbb{E}_{p(\mathbf{z}_u, y_u)}[\log q_\psi(y_u|\mathbf{z}_u)] \\ :=& I_{vLB}(\mathbf{z}_u; y_u). \end{aligned} \tag{13}$$

Note that the variational lower bound is tight when the auxiliary distribution q_ψ becomes the true posterior distribution p. Now, we maximize the variational lower bound to achieve Equation 11 by estimating q_ψ. Specifically, we have

$$\begin{aligned} & \max_\theta I_{vLB}(\mathbf{z}_u; y_u) \\ \Leftrightarrow & \max_\theta \max_\psi \mathbb{E}_{p(\mathbf{z}_u, y_u)}[\log q_\psi(y_u|\mathbf{z}_u)] \\ = & \max_\theta \max_\psi \mathbb{E}_{p(\mathbf{x}_u, y_u)}[\log q_\psi(y_u|f_\theta(\mathbf{x}_u))]. \end{aligned} \tag{14}$$

Minimizing the mutual information in Equation 12. To solve Equation 12, we derive the vCLUB motivated by [6] and have

$$\begin{aligned} & I(A_{uv}; \mathbf{z}_u, \mathbf{z}_v) \\ \leq & I_{vCLUB}(A_{uv}; \mathbf{z}_u, \mathbf{z}_v) \\ =& \mathbb{E}_{p(A_{uv}, \mathbf{z}_u, \mathbf{z}_v)}[\log q_\phi(A_{uv}|\mathbf{z}_u, \mathbf{z}_v)] \\ & - \mathbb{E}_{p(\mathbf{z}_u, \mathbf{z}_v)p(A_{uv})}[\log q_\phi(A_{uv}|\mathbf{z}_u, \mathbf{z}_v)], \end{aligned} \tag{15}$$

where $q_\phi(A_{uv}|\mathbf{z}_u, \mathbf{z}_v)$ is an auxiliary distribution of $p(A_{uv}|\mathbf{z}_u, \mathbf{z}_v)$ that needs to satisfy the following condition:

$$\begin{aligned} & KL(p(\mathbf{z}_u, \mathbf{z}_v, A_{uv})||q_\phi(\mathbf{z}_u, \mathbf{z}_v, A_{uv})) \\ \leq & KL(p(\mathbf{z}_u, \mathbf{z}_v)p(A_{uv})||q_\phi(\mathbf{z}_u, \mathbf{z}_v, A_{uv})). \end{aligned} \tag{16}$$

That is, I_{vCLUB} is a mutual information upper bound if the variational joint distribution $q_\phi(\mathbf{z}_u, \mathbf{z}_v, A_{uv})$ is closer to the joint distribution $p(\mathbf{z}_u, \mathbf{z}_v, A_{uv})$ than to $p(\mathbf{z}_u, \mathbf{z}_v)p(A_{uv})$.

To achieve Inequality 16, we need to minimize the KL-divergence $KL(p(\mathbf{z}_u, \mathbf{z}_v, A_{uv})||q_\phi(\mathbf{z}_u, \mathbf{z}_v, A_{uv}))$ as follows:

$$\begin{aligned} & \min_\phi KL(p(A_{uv}, \mathbf{z}_u, \mathbf{z}_v)||q_\phi(A_{uv}, \mathbf{z}_u, \mathbf{z}_v)) \\ = & \min_\phi KL(p(A_{uv}|\mathbf{z}_u, \mathbf{z}_v)||q_\phi(A_{uv}|\mathbf{z}_u, \mathbf{z}_v)) \\ = & \min_\phi \mathbb{E}_{p(A_{uv}, \mathbf{z}_u, \mathbf{z}_v)}[\log p(A_{uv}|\mathbf{z}_u, \mathbf{z}_v)] - \mathbb{E}_{p(A_{uv}, \mathbf{z}_u, \mathbf{z}_v)}[\log q_\phi(A_{uv}|\mathbf{z}_u, \mathbf{z}_v)] \\ \Leftrightarrow & \max_\phi \mathbb{E}_{p(A_{uv}, \mathbf{z}_u, \mathbf{z}_v)}[\log q_\phi(A_{uv}|\mathbf{z}_u, \mathbf{z}_v)]. \end{aligned}$$

Finally, our target to achieve Equation 12 becomes the following adversarial training objective:

$$\begin{aligned} & \min_\theta \min_\phi I_{vCLUB}(A_{uv}; \mathbf{z}_u, \mathbf{z}_v) \\ \Leftrightarrow & \min_\theta \max_\phi \mathbb{E}_{p(A_{uv}, \mathbf{z}_u, \mathbf{z}_u)}[\log q_\phi(A_{uv}|\mathbf{z}_u, \mathbf{z}_v)] \\ = & \min_\theta \max_\phi \mathbb{E}_{p(A_{uv}, \mathbf{x}_u, \mathbf{x}_v)}[\log q_\phi(A_{uv}|f_\theta(\mathbf{x}_u), f_\theta(\mathbf{x}_v))] \end{aligned} \tag{17}$$

Remark. The above objective function can be interpreted as an adversarial game between an adversary q_ϕ who aims to infer the link A_{uv} from the pair of node representations $f_\theta(\mathbf{x}_u)$ and $f_\theta(\mathbf{x}_v)$; and a defender (i.e., the embedding function f_θ) who aims to protect the link privacy from being inferred.

Table 1: Dataset statistics.

Dataset	#Nodes	#Edges	#Features	#Labels
Cora	**2,708**	**5,429**	**1,433**	**7**
Citeseer	**3,327**	**4,732**	**3,703**	**6**
Pubmed	**19,717**	**44,338**	**500**	**3**

Implementation via training parameterized neural networks. We solve Equation 14 and Equation 17 in practice via training two parameterized neural networks. With it, we expect to obtain a high node classification performance for our primary learning task and a low link prediction performance for our privacy protection task.

Similar to solving **Problem 1**, to solve Equation 14, we first sample a set of labeled nodes $\{\mathbf{x}_u, y_u\}$, and then we parameterize the variational posterior distribution q_ψ via a node classifier g_ψ defined on the node representation $\{f_\theta(\mathbf{x}_u)\}$ of these labeled nodes. Suppose we sample a set of labeled nodes $\mathcal{V}_L$, then we have

$$\begin{aligned} & \max_\theta \max_\psi \mathbb{E}_{p(\mathbf{x}_u, y_u)}[\log q_\psi(y_u|f_\theta(\mathbf{x}_u))] \\ \approx & \max_\theta \max_\psi \sum_{v \in \mathcal{V}_L} -CE(g_\psi(f_\theta(\mathbf{x}_v)), y_v) \\ = & \min_\theta \min_\psi \sum_{v \in \mathcal{V}_L} CE(g_\psi(f_\theta(\mathbf{x}_v)), y_v) \end{aligned} \tag{18}$$

To solve Equation 17, we first sample a set of triplets $\{A_{uv}, \mathbf{x}_u, \mathbf{x}_v\}$ from the graph G. Then, we parameterize q_ϕ via a link predictor h_ϕ defined on the node representation $f_\theta(\mathbf{x}_u)$ and $f_\theta(\mathbf{x}_v)$ of the sampled node pairs u and v. Depending on the real scenarios, we can protect a set of positive links with $A_{uv} = 1$ or/and a set of negative links with $A_{uv} = 0$. In our experiments, we consider protecting both positive links and negative links. Suppose we are given a set of positive links $\mathcal{E}_p$ and a set of negative links $\mathcal{E}_n$, then we have

$$\begin{aligned} & \min_\theta \max_\phi \mathbb{E}_{p(A_{uv}, \mathbf{x}_u, \mathbf{x}_v)}[\log q_\phi(A_{uv}|f_\theta(\mathbf{x}_u), f_\theta(\mathbf{x}_v))] \\ \approx & \min_\theta \max_\phi \sum_{(u,v) \in \mathcal{E}_p \cup \mathcal{E}_n} -CE(h_\phi(f_\theta(\mathbf{x}_u), f_\theta(\mathbf{x}_v)), A_{uv}) \end{aligned} \tag{19}$$

Combining Equation 18 and Equation 19, we have the final objective function for our **Problem 2** as follows:

$$\begin{aligned} \min_\theta \big(& \lambda \min_\psi \sum_{v \in \mathcal{V}_L} CE(g_\psi(f_\theta(\mathbf{x}_v)), y_v) \\ & - (1-\lambda) \max_\phi \sum_{(u,v) \in \mathcal{E}_p \cup \mathcal{E}_n} CE(h_\phi(f_\theta(\mathbf{x}_u), f_\theta(\mathbf{x}_v)), A_{uv}) \big), \end{aligned} \tag{20}$$

where λ is a trade-off factor to balance between achieving high node classification performance and low link prediction performance.

Similar to **Problem 1**, our objective function for Problem 2 in Equation 20 involves optimizing three neural networks: the node embedding function f_θ, the link predictor h_ϕ, and the node classifier g_ψ. We alternatively train the three neural networks. Algorithm 2 in Appendix illustrates the training procedure of these networks.

Table 2: Results on primary learning task: link prediction + privacy-protection task: node classification.

(a) **Without node privacy protection**

Dataset	Method	Link prediction	Node classification	
		AUC	Acc	Rand
Cora	GCN	89.33%	72.00%	14.29%
	GAT	92.95%	71.80%	14.29%
	HGCN	93.78%	75.30%	14.29%
Citeseer	GCN	91.52%	67.40%	16.67%
	GAT	95.03%	67.00%	16.67%
	HGCN	94.65%	67.20%	16.67%
Pubmed	GCN	91.43%	72.70%	33.33%
	GAT	94.44%	78.50%	33.33%
	HGCN	95.18%	76.60%	33.33%

(b) **With node privacy protection using our framework**

Dataset	Method	Link prediction	Node classification	
		AUC	Acc	Rand
Cora	GCN	79.41%	28.50%	14.29%
	GAT	84.12%	21.40%	14.29%
	HGCN	85.01%	14.40%	14.29%
Citeseer	GCN	85.55%	15.40%	16.67%
	GAT	85.44%	21.40%	16.67%
	HGCN	86.51%	18.20%	16.67%
Pubmed	GCN	81.24%	42.50%	33.33%
	GAT	84.65%	41.80%	33.33%
	HGCN	85.39%	40.70%	33.33%

Table 3: Results on primary learning task: node classification + privacy-protection task: link prediction.

(a) **Without link privacy protection**

Dataset	Method	Link prediction		Node classification
		AUC	Rand	Acc
Cora	GCN	82.73%	50.00%	81.60%
	GAT	77.32%	50.00%	81.80%
	HGCN	80.83%	50.00%	79.50%
Citeseer	GCN	83.30%	50.00%	67.50%
	GAT	82.12%	50.00%	71.00%
	HGCN	79.12%	50.00%	68.50%
Pubmed	GCN	78.90%	50.00%	78.80%
	GAT	78.45%	50.00%	79.20%
	HGCN	77.10%	50.00%	80.00%

(b) **With link privacy protection using our framework**

Dataset	Method	Link prediction		Node classification
		AUC	Rand	Acc
Cora	GCN	49.91%	50.00%	79.70%
	GAT	50.00%	50.00%	81.30%
	HGCN	54.49%	50.00%	75.50%
Citeseer	GCN	53.29%	50.00%	65.80%
	GAT	50.00%	50.00%	70.70%
	HGCN	49.36%	50.00%	64.50%
Pubmed	GCN	49.57%	50.00%	78.60%
	GAT	50.00%	50.00%	78.50%
	HGCN	53.22%	50.00%	78.50%

5 EVALUATION

5.1 Experimental Setup

Dataset description. We use three benchmark citation graphs (i.e., Cora, Citeseer, and Pubmed) [28] to evaluate our method. In these graphs, each node represents a documents and each edge indicates a citation between two documents. Each document treats the bag-of-words feature as the node feature vector and also has a label. Table 1 shows basic statistics of these citation graphs.

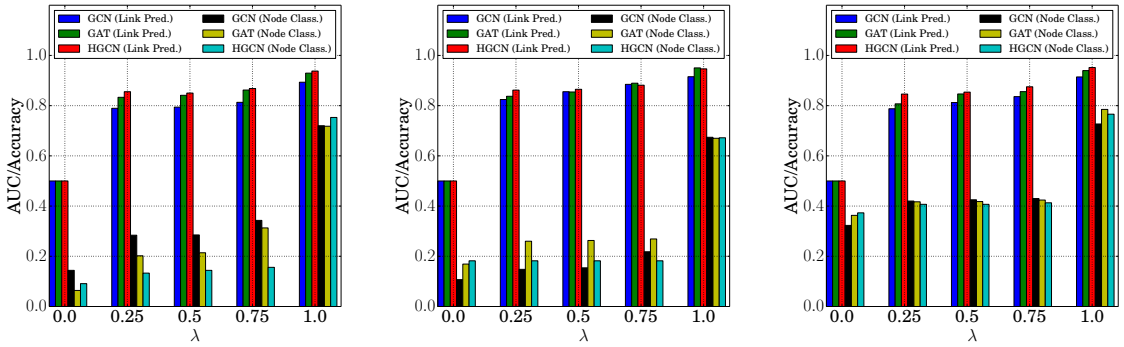
Representation learning methods. We select three graph neural networks, i.e., GCN [14], GAT [32], HGCN [4] as the representative graph representation learning methods. Each method learns node representations for both node classification and link prediction. Specifically, in these methods, the input layer to the second-to-last layer are used for learning node representations. Suppose node u's representation is $\mathbf{z}_u$. When performing node classification, all these methods train a (Euclidean) softmax classifier g_ψ in the last layer, i.e., $g_\psi(\mathbf{z}_u) = \text{softmax}(\mathbf{z}_u^T \cdot \psi)$, where T is a transpose and $\text{softmax}(\mathbf{x})_i = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$. When performing link prediction, GCN and GAT train a parameterized bilinear link predictor h_ϕ, i.e., $h_\phi(\mathbf{z}_u, \mathbf{z}_v) = \mathbf{z}_u^T \cdot \phi \cdot \mathbf{z}_v$; HGCN trains a Fermi-Dirac decoder [15, 19] as the link predictor.

Note that these methods only focus on learning node representations for solving the primary learning task and do not consider protecting the privacy for the other task. Furthermore, we apply our framework to learn privacy-preserving node representations.

Training set, validation set, and testing set. Following existing works [4, 14, 39], in each graph dataset, for node classification, we randomly sample 20 nodes per class to form the training set, randomly sample 500 nodes in total as the validation set, and randomly sample 1,000 nodes in total as the testing set. For link prediction, we randomly sample 85% positive links and 50% negative links for training, sample 5% positive links and an equal number of negative links for validation, and use the remaining 10% positive links and sample an equal number of negative links for testing.

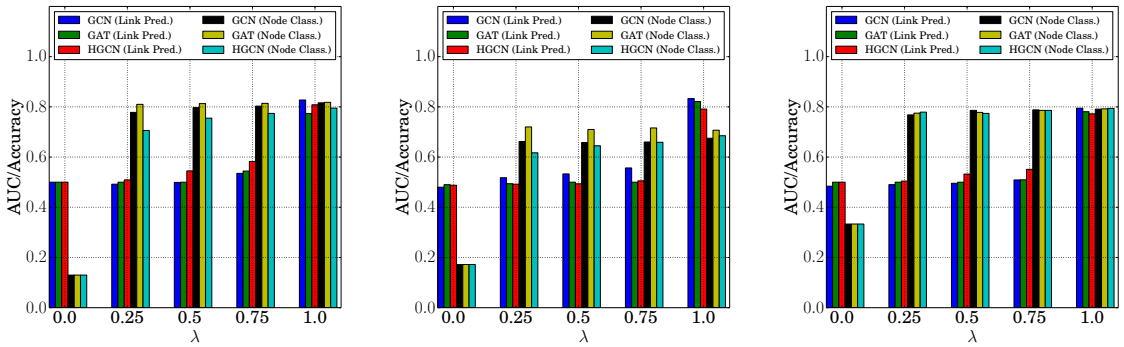
Parameter setting. We train our framework on the training set and tune hyperparameters to select the model with the minimal error on the validation set. Then, we use the selected model to evaluate the testing set. By default, we set the trade-off factor λ to be 0.5 during training. We also study the impact of λ in our experiments. We train all graph neural networks using the publicly available source code. We implement our framework in PyTorch.

Evaluation metric. Following previous works [4, 13, 14, 39], we use *AUC* to evaluation the link prediction performance and use *accuracy* to evaluate the node classification performance.



(a) Cora (b) Citeseer (c) Pubmed

Figure 2: Impact of λ. Link prediction with node privacy protection.



(a) Cora (b) Citeseer (c) Pubmed

Figure 3: Impact of λ. Node classification with link privacy protection.

5.2 Experimental Results

5.2.1 Link prediction without/with node privacy protection. In this experiment, we consider link prediction as the primary learning task, and node classification as the privacy protection task. Table 2(a) shows the performance of the two tasks without node privacy protection by existing methods. Specifically, we use the three graph neural networks, i.e., GCN, GAT, and HGCN, to learn node representations, and use them to train a link predictor for link prediction. Next, we also leverage these node representations to train a node classifier. We have the following observations: 1) All these methods achieve very high AUCs on the three graphs, i.e., almost all AUCs are above 90%, demonstrating their effectiveness for link prediction. 2) Although these node representations are not specially learnt for node classification, they can be used by the node classifier to accurately infer the node labels, thus leaking node privacy. For instance, all the methods obtain the accuracies around/above 70% and they perform significantly better than random guessing.

Table 2(b) shows the performance of the two tasks with node privacy protection. Specifically, we use our framework to learn node representations, the link predictor, and the node classifier, simultaneously. We observe that our framework achieves an utility-privacy tradeoff. In particular, our framework has a tolerable AUC drop, compared with AUCs in Table 2(b). However, our framework obtains much lower accuracies than those in Table 2(b). In some cases, the accuracies are close to random guessing, demonstrating a nearly perfect node privacy protection. The above results validate that our framework is effective for link prediction, as well as for protecting node privacy.

5.2.2 Node classification with link privacy protection. In this experiment, we consider node classification as the primary learning task, and link prediction as the privacy protection task. Table 3(a) shows the performance of the two tasks without link privacy protection by existing methods. Similarly, we use the three graph neural networks to learn node representations, and use them to train a node classifier for node classification. We have similar observations as results shown in Table 2(a). First, all methods achieve promising accuracies on the three graphs, i.e., close to the results shown in [4, 14, 32]. Next, we leverage these node representations to train a link predictor to infer link status. We observe that these methods obtain AUCs significantly larger than those obtained by random guessing, thus leaking link privacy seriously.

Table 3(b) shows the performance of the two tasks with link privacy protection. Our framework achieves an utility-privacy tradeoff, similar to results in Table 3(b). First, our framework has slightly accuracies degradation (around 1%-4%), compared with accuracies in Table 2(b). Second, our framework obtains much lower AUCs and

almost all of these AUCs are close to random guessing. The results again validate that our framework is effective for node classification, as well as for protecting link privacy.

5.2.3 Impact of the trade-off factor λ. In this experiment, we study the impact of the trade-off factor λ in our framework. Figure 2 and Figure 3 show the performance on the three graphs vs. different λ for protecting node privacy and protecting link privacy, respectively. We have the following key observations: 1) When $\lambda = 0$, our framework only considers protecting node/link privacy, and achieves the lowest performance (i.e., close to random guessing) for inferring the node label/link status. However, the performance for the primary learning task is also the worst (i.e., random guessing). 2) When $\lambda = 1$, our framework only considers primary task learning and achieves the highest performance for the link prediction/node classification. However, it also obtains the highest performance for inferring the node label/link status, thus leaking the most information of nodes/link status. 3) When $0 < \lambda < 1$, our framework considers both primary learning and privacy protection. We note that our framework is not sensitive to λ's value in this range. That is, the performance of graph neural networks based on our framework for primary learning and privacy protection are relatively stable across all λ's in this range.

6 CONCLUSION

We propose the first framework for privacy-preserving representation learning on graphs from the mutual information perspective. Our framework includes a primary learning task and a privacy protection task. The goal is to learn node representations such that they can be used to achieve high performance for the primary learning task, while obtaining low performance for the privacy protection task (e.g., close to random guessing). We formally formulate our goal via mutual information objectives. However, mutual information is challenging to compute in practice. Motivated by mutual information neural estimation, we derive tractable variational bounds for the mutual information, and parameterize each bound via a neural network. Next, we train these neural networks to approximate the true mutual information and learn privacy-preserving node representations. We evaluate our framework on various graph datasets and show that our framework is effective for learning privacy-preserving node representations on graphs.

Acknowledgements. We thank the anonymous reviewers for their constructive comments. This work is supported by the Amazon Research Award. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding agencies.

REFERENCES

[1] Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. 2017. Deep variational information bottleneck. In *ICLR*.
[2] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. 2018. Mutual information neural estimation. In *ICML*.
[3] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2015. Grarep: Learning graph representations with global structural information. In *CIKM*.
[4] Ines Chami, Zhitao Ying, Christopher Ré, and Jure Leskovec. 2019. Hyperbolic graph convolutional neural networks. In *NeurIPS*.
[5] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. 2016. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *NIPS*.
[6] Pengyu Cheng, Weituo Hao, Shuyang Dai, Jiachang Liu, Zhe Gan, and Lawrence Carin. 2020. CLUB: A Contrastive Log-ratio Upper Bound of Mutual Information. In *ICML*.
[7] Ganqu Cui, Jie Zhou, Cheng Yang, and Zhiyuan Liu. 2020. Adaptive Graph Encoder for Attributed Graph Embedding. In *KDD*.
[8] Alberto Garcia Duran and Mathias Niepert. 2017. Learning graph representations with embedding propagation. In *NIPS*.
[9] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *NIPS*.
[10] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *SIGKDD*.
[11] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *NIPS*.
[12] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. 2019. Learning deep representations by mutual information estimation and maximization. In *ICLR*.
[13] Thomas N Kipf and Max Welling. 2016. Variational graph auto-encoders. In *NIPS Workshop*.
[14] Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. *ICLR* (2017).
[15] Dmitri Krioukov, Fragkiskos Papadopoulos, Maksim Kitsak, Amin Vahdat, and Marián Boguná. 2010. Hyperbolic geometry of complex networks. *Physical Review E* (2010).
[16] Ang Li, Yixiao Duan, Huanrui Yang, Yiran Chen, and Jianlei Yang. 2020. TIPRDC: task-independent privacy-respecting data crowdsourcing framework for deep learning with anonymized intermediate representations. In *KDD*.
[17] Qi Liu, Maximilian Nickel, and Douwe Kiela. 2019. Hyperbolic graph neural networks. In *NeurIPS*.
[18] Yao Ma, Suhang Wang, Charu C Aggarwal, and Jiliang Tang. 2019. Graph convolutional networks with eigenpooling. In *KDD*.
[19] Maximillian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In *NIPS*.
[20] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv* (2018).
[21] Zhen Peng, Wenbing Huang, Minnan Luo, Qinghua Zheng, Yu Rong, Tingyang Xu, and Junzhou Huang. 2020. Graph Representation Learning via Graphical Mutual Information Maximization. In *WWW*.
[22] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *SIGKDD*.
[23] Ben Poole, Sherjil Ozair, Aaron van den Oord, Alexander A Alemi, and George Tucker. 2019. On variational bounds of mutual information. In *ICML*.
[24] Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. 2018. Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In *WSDM*.
[25] Meng Qu, Yoshua Bengio, and Jian Tang. 2019. GMNN: Graph Markov Neural Networks. In *ICML*.
[26] Leonardo FR Ribeiro, Pedro HP Saverese, and Daniel R Figueiredo. 2017. struc2vec: Learning node representations from structural identity. In *KDD*.
[27] Ryan A Rossi, Rong Zhou, and Nesreen K Ahmed. 2018. Deep inductive network representation learning. In *WWW*.
[28] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, and et al. 2008. Collective classification in network data. *AI magazine* (2008).
[29] Fan-Yun Sun, Jordan Hoffmann, Vikas Verma, and Jian Tang. 2020. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. In *ICLR*.
[30] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *WWW*.
[31] Cunchao Tu, Weicheng Zhang, Zhiyuan Liu, and Maosong Sun. 2016. Max-margin DeepWalk: discriminative learning of network representation. In *IJCAI*.
[32] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. In *ICLR*.
[33] Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2019. Deep Graph Infomax.. In *ICLR*.
[34] Felix Wu, Tianyi Zhang, Amauri Holanda de Souza Jr, Christopher Fifty, Tao Yu, and Kilian Q Weinberger. 2019. Simplifying graph convolutional networks. In *ICML*.
[35] Jun Wu, Jingrui He, and Jiejun Xu. 2019. Demo-Net: Degree-specific graph neural networks for node and graph classification. In *KDD*.
[36] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How powerful are graph neural networks?. In *ICLR*.
[37] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. 2018. Representation learning on graphs with jumping knowledge networks. In *ICML*.
[38] Zhilin Yang, William Cohen, and Ruslan Salakhudinov. 2016. Revisiting semi-supervised learning with graph embeddings. In *ICML*.
[39] Muhan Zhang and Yixin Chen. 2018. Link prediction based on graph neural networks. In *NIPS*.

A ALGORITHMIC PSEUDO CODE

Algorithm 1 Link prediction with node privacy protection

Input: Graph G, $\mathcal{V}_L = \{\mathbf{x}_u, y_u\}$, $\mathcal{E}_p = \{\mathbf{x}_u, \mathbf{x}_v, A_{uv} = 1\}$, $\mathcal{E}_n = \{\mathbf{x}_u, \mathbf{x}_v, A_{uv} = 0\}$, trade-off factor λ, #gradient steps I, #rounds T.

Output: Network parameters: θ^*, ϕ^*, ψ^*.

Initialize θ, ϕ, ψ for the embedding network f_θ, link predictor h_ϕ, and node classifier g_ψ;
Initialize learning rates lr_1, lr_2, lr_3;
Initialize $t \leftarrow 0$;
for $t < T$ **do**
  $L_1 = \sum_{(u,v)\in\mathcal{E}_p\cup\mathcal{E}_n} CE(h_\phi(f_\theta(\mathbf{x}_u), f_\theta(\mathbf{x}_v)), A_{uv})$;
  $L_2 = \sum_{v\in\mathcal{V}_L} CE(g_\psi(f_\theta(\mathbf{x}_v)), y_v)$;
  Initialize $i \leftarrow 0$;
  for $i < I$ **do**
    $\phi \leftarrow \phi - lr_1 \cdot \frac{\partial L_1}{\partial \phi}$;
    $\psi \leftarrow \psi + lr_2 \cdot \frac{\partial L_2}{\partial \psi}$;
    $\theta \leftarrow \theta - lr_3 \cdot \frac{\partial(\lambda L_1 - (1-\lambda) L2)}{\partial \theta}$;
    $i \leftarrow i + 1$;
  end for
  $t \leftarrow t + 1$
end for
return θ, ϕ, ψ.

Algorithm 2 Node classification with link privacy protection

Input: Graph G, $\mathcal{V}_L = \{\mathbf{x}_u, y_u\}$, $\mathcal{E}_p = \{\mathbf{x}_u, \mathbf{x}_v, A_{uv} = 1\}$, $\mathcal{E}_n = \{\mathbf{x}_u, \mathbf{x}_v, A_{uv} = 0\}$, trade-off factor λ, #gradient steps I, #rounds T.

Output: Network parameters: θ^*, ϕ^*, ψ^*.

Initialize θ, ϕ, ψ for the embedding network f_θ, link predictor h_ϕ, and node classifier g_ψ;
Initialize learning rates lr_1, lr_2, lr_3;
Initialize $t \leftarrow 0$.
for $t < T$ **do**
  $L_1 = \sum_{v\in\mathcal{V}_L} CE(g_\psi(f_\theta(\mathbf{x}_v)), y_v)$;
  $L_2 = \sum_{(u,v)\in\mathcal{E}_p\cup\mathcal{E}_n} CE(h_\phi(f_\theta(\mathbf{x}_u), f_\theta(\mathbf{x}_v)), A_{uv})$;
  Initialize $i \leftarrow 0$;
  for $i < I$ **do**
    $\psi \leftarrow \psi - lr_1 \cdot \frac{\partial L_1}{\partial \psi}$;
    $\phi \leftarrow \phi + lr_2 \cdot \frac{\partial L_2}{\partial \phi}$;
    $\theta \leftarrow \theta - lr_3 \cdot \frac{\partial(\lambda L_1 - (1-\lambda) L2)}{\partial \theta}$;
    $i \leftarrow i + 1$;
  end for
  $t \leftarrow t + 1$
end for
return θ, ϕ, ψ.