Releasing Graph Neural Networks with Differential Privacy Guarantees

Iyiola E. Olatunji
iyiola@l3s.de
L3S Research Center
Hannover, Germany

Thorben Funke
funke@l3s.de
L3S Research Center
Hannover, Germany

Megha Khosla
khosla@l3s.de
L3S Research Center
Hannover, Germany

ABSTRACT

With the increasing popularity of Graph Neural Networks (GNNs) in several sensitive applications like healthcare and medicine, concerns have been raised over the privacy aspects of trained GNNs. More notably, GNNs are vulnerable to privacy attacks, such as membership inference attacks, even if only blackbox access to the trained model is granted. To build defenses, differential privacy has emerged as a mechanism to disguise the sensitive data in training datasets. Following the strategy of Private Aggregation of Teacher Ensembles (PATE), recent methods leverage a large ensemble of teacher models. These teachers are trained on disjoint subsets of private data and are employed to transfer knowledge to a student model, which is then released with privacy guarantees. However, splitting graph data into many disjoint training sets may destroy the structural information and adversely affect accuracy. We propose a new graph-specific scheme of releasing a student GNN, which avoids splitting private training data altogether. The student GNN is trained using public data, partly labeled privately using the teacher GNN models trained exclusively for each query node. We theoretically analyze our approach in the Rènyi differential privacy framework and provide privacy guarantees. Besides, we show the solid experimental performance of our method compared to several baselines, including the PATE baseline adapted for graph-structured data. Our anonymized code is available.

CCS CONCEPTS

• **Security and privacy**; • **Information systems** → *Computing platforms*;

KEYWORDS

Privacy, Graph neural networks, Differential Privacy

ACM Reference Format:
Iyiola E. Olatunji, Thorben Funke, and Megha Khosla. 2022. Releasing Graph Neural Networks with Differential Privacy Guarantees. In *Woodstock '22: ACM Symposium on Neural Gaze Detection, June 03–05, 2022, Woodstock, NY*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

In the past few years, Graph Neural Networks (GNNs) have gained much attention due to their superior performance in a wide range of applications, such as social networks [8], biology [13], medicine [2], and molecular chemistry [15]. Specifically, GNNs achieved state-of-the-art results in various graph-based learning tasks, such as node classification, link prediction, and community detection. Real-world graphs, such as medical and economic networks, are associated with sensitive information about individuals and their activities and cannot always be made public. Releasing pre-trained models provides an opportunity for using the private knowledge beyond company boundaries [2]. However, recent works have shown that GNNs are vulnerable to membership inference attacks [9, 18]. Specifically, membership inference attacks allow to identify which data points have been used for training the model. In general, GNNs are more vulnerable to such attacks as compared to traditional knowledge due to their encoding of the graph structure within the model itself [18]. In addition, the current legal data protection policies to preserve user privacy highlights a compelling need to develop *privacy preserving GNNs*.

In this work, we propose our framework PrivGnn, which builds on the rigid guarantees of *differential privacy* (DP), allowing us to protect the sensitive data while releasing the trained GNN model. Differential privacy [5] is one of the most popular approaches for releasing data statistics or the trained model while concealing the information about individuals present in the dataset. Roughly speaking, the key idea of DP is that if we query a dataset containing N individuals, the query's result will be almost indistinguishable (in a probabilistic sense) from the result of querying a neighboring dataset with one less or one more individual. Hence, each individual's privacy is guaranteed with a specific probability. Such probabilistic indistinguishability is usually achieved by adding a sufficient amount of noise to the query result.

The seminal work of Abadi et al. [1] proposed *differential private stochastic gradient descent* (DP-SGD) algorithm to achieve differential privacy guarantees for deep learning models. Specifically, in each step of the training, DP-SGD adds appropriate noise to the ℓ_2-clipped gradients during the stochastic gradient descent optimization. The incurred privacy budget ε for training is computed using the moment's accountant technique that keeps track of the privacy loss across multiple invocations of the noise addition mechanism applied to random subsets of the input dataset [1].

Besides the slow training process of DP-SGD, the injected noise is proportional to the number of training epochs, which further degrades performance. More importantly, the privacy guarantee for DP-SGD does not trivially hold for graph data and GNN models [11]. While DP-SGD is designed for independent and identically distributed data (i.i.d.), the nodes in graph data are related. GNNs use a message-passing algorithm to exchange information among connected nodes [8]. Therefore, the privacy guarantee of DP-SGD,

Woodstock '22, June 03–05, 2022, Woodstock, NY

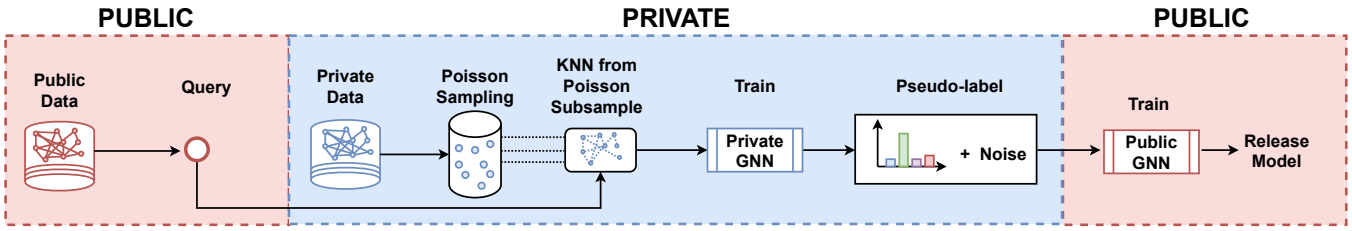


Figure 1: Workflow of PrivGnn. We are given two corresponding datasets, *labeled private data* and *unlabeled public data*. PrivGnn starts by sampling the private data using Poisson sampling, Def. 7, to retrieve a subset of the private data. We then obtain the K-nearest neighbor nodes based on the features of the public query node. The teacher GNN model is trained on the graph induced on K-nearest neighbors. We obtain a pseudo-label for the query node by adding independent noise to the output posterior. The pseudo-label and data from the public graph are used in training the student model, which is then released.

which requires a set of i.i.d. examples to form batches and lots, does not hold for GNNs and graph data [11].

To work around DP-SGD's dependency of the training procedure, such as the number of epochs, Papernot et al. [19] proposed *Private Aggregation of Teacher Ensembles* (PATE). PATE leverages a large ensemble of teacher models trained on disjoint subsets of private data to transfer knowledge to a student model, which is then released with privacy guarantees. However, splitting graph data into many disjoint training sets destroys the structural information and adversely affects accuracy.

Since existing DP methods are not directly applicable to GNNs, we propose a privacy-preserving framework, PrivGnn, for releasing GNN models with differential privacy guarantees. Similar to PATE's assumptions, we are given two graphs: a labeled private graph and an unlabeled public graph. PrivGnn leverages the paradigm of knowledge distillation. The knowledge of the teacher model trained on the private graph is transferred to the student model trained only on the public graph in a differential privacy manner. PrivGnn achieves practical privacy guarantees by combining the student-teacher training with two noise mechanisms: **random subsampling using Poisson sampling** and **noisy labeling mechanism** to obtain pseudo-labels for public nodes. In particular, we release the student GNN model, which is trained using a small number of public nodes labeled using the teacher GNN models developed exclusively for each public query node. We present a *Rènyi differential privacy* (RDP) analysis of our approach and provide tight bounds on incurred privacy budget or privacy loss. Figure 1 shows an overview of our PrivGnn approach.

To summarize, **our key contributions** are as follows.

(1) We propose PrivGnn, a novel privacy-preserving framework for releasing GNN models via differential privacy. By leveraging the student-teacher training paradigm, PrivGnn is robust to attacks on GNN models, including MI attack and model stealing attacks.
(2) We derive tight privacy guarantees employing the theoretical results of RDP for Poisson subsampled mechanisms and advanced composition theorem for RDP. Ours is the first work utilizing the RDP framework for analyzing the privacy of GNNs.
(3) We experimentally show that PrivGnn achieves close to optimal accuracy of the non-private version with practical privacy guarantees (single-digit ε).

2 RELATED WORKS

Graph neural networks (GNNs) [8, 12, 23] mainly popularized by graph convolution networks and their variants compute node representations by recursive aggregation and transformation of feature representations of its neighbors. While they encode the graph directly into the model via the aggregation scheme, GNNs have been shown to be highly vulnerable to membership inference attacks [18], thus highlighting the need to develop privacy-preserving GNNs.

Existing works on privacy-preserving GNNs mainly focussed on a distributed setting in which the node feature values or/and labels are assumed to be private and distributed among multiple distrusting parties [20, 21, 25, 27]. Sajadmanesh and Gatica-Perez [20] assumed that only the node features are sensitive while the graph structure is publicly available and developed a local differential privacy mechanism to tackle the problem of node-level privacy.

Wu et al. [25] proposed a federated framework for privacy-preserving GNN-based recommendation systems while focussing on the task of subgraph level federated learning. Other works [4, 26] in non-distributed settings also focussed mainly on preserving the privacy of user attributes in recommender systems.

Zhou et al. [27] proposed a privacy-preserving GNN learning paradigm for node classification task among distrusting parties such that each party has access to the same set of nodes. However, the features and edges are split among them. Shan et al. [21] proposed a server-aided privacy-preserving GNN for the node level task on a horizontally partitioned cross-silo scenario via a secure pooling aggregation mechanism. They assumed that all data holders have the same feature domains and edge type but differ in nodes, edges, and labels.

Igamberdiev and Habernal [11] adapted differentially-private gradient-based training (DP-SGD) [1] to GCNs for natural language processing task. However, as noted by the authors, the privacy guarantees of their approach might not hold because (DP-SGD) requires a set of i.i.d. examples to form batches and lots in order to distribute the noise effectively, whereas, in GNNs, the nodes exchange information via the message passing framework during training. To the best of our knowledge, this is the only previous

work that assumes a non-distributive setting and the sensitivity of the whole private graph, including the other data like node features and labels.

Differences with Existing Works. Our work is different from existing works in the following ways. **First**, contrary to the distributed setting where either the graph structure or a set of nodes are assumed to be non-private, we assume that the whole graph with its nodes, features, and labels is private and has a single owner. Such a scenario can, for example, arise for a company owning a social network that wants to publish a trained GNN model without compromising the privacy of any user and her network. While in distributed settings, the adversary might be any owner party holding a part of data, the adversary in our case is the user having Whitebox access to the trained model. **Second**, we avoid the requirement of i.i.d. data as needed to train a differential private model with DP-SGD. **Finally**, our setting is different from the traditional methods for different privacy analysis of graph statistics or differential private release of graph data. In particular, we are concerned with differential privacy for the trained GNN model.

3 THEORETICAL COMPONENTS

We review the necessary technical and theoretical components that we employ for the privacy analysis of our approach. We start by describing differential privacy (DP) and the related privacy mechanism followed by a more generalized notion of DP, i.e., Rènyi differential privacy (RDP). More specifically, we utilize the bound on subsampled RDP and the advanced composition theorem for RDP to derive practical privacy guarantees for our approach.

3.1 Differential Privacy

Differential privacy [5] is the most common notion of privacy for algorithms on statistical databases. Informally, DP bounds the change in output distribution of a mechanism when there is a small change in its input. Specifically, ε-DP puts a multiplicative upper bound on the worst-case change in output distribution when the input differs by exactly one data point.

Definition 1 (ε-DP [5]). *A mechanism $\mathcal{M}\colon \mathcal{X} \to \Theta$ is ε-DP if for every pair of neighboring datasets $X, X' \in \mathcal{X}$, and every possible (measurable) output set $E \subseteq \Theta$ the following inequality holds:*

$$Pr[\mathcal{M}(X) \in E] \leq e^{\varepsilon} Pr[\mathcal{M}(X') \in E].$$

An example of an ε-DP algorithm is the **Laplace Mechanism** which allows releasing a noisy answer/output to an arbitrary query with values in $\mathbb{R}^n$. The mechanism is defined as

$$\mathbb{L}_{\varepsilon} f(x) \triangleq f(x) + \text{Lap}(0, \Delta_1/\varepsilon), \tag{1}$$

where Lap is the Laplace distribution and Δ_1 is the ℓ_1 sensitivity of the query f defined as

$$\Delta_1 \triangleq \max_{X,X'} ||f(X) - f(X')||_1.$$

The Laplace distribution $\text{Lap}(0, \beta)$ is centered around 0 and has with the scale parameter $\beta > 0$ the following probability density function

$$\text{Lap}(x|0, \beta) = \frac{1}{2\beta} \exp\left(-\frac{|x|}{\beta}\right).$$

Since many DP algorithms combine multiple randomization mechanisms, we require the following composition result.

Theorem 2 (Composition and Post-Processing [16]). *Let D be any dataset, $\mathcal{M}_1, \mathcal{M}_2, \cdots \mathcal{M}_n$ be n mechanism that satisfy $\epsilon_1, \epsilon_2, \cdots \epsilon_n$-DP respectively. Then the following properties hold for $\mathcal{M}_1, \mathcal{M}_2, \cdots \mathcal{M}_n$:*

1. *By* **sequential composition**, *releasing the output of $\mathcal{M}_1(D)$, $\mathcal{M}_2(D), \cdots \mathcal{M}_n(D)$ satisfies $(\epsilon_1 + \epsilon_2 + \cdots \epsilon_n)$-DP.*
2. *Given* disjoint *datasets $(D_1, D_2, \ldots, D_n)$, releasing each of the $\mathcal{M}_i(D_i)$, satisfies $\max(\epsilon_1, \epsilon_2, \ldots, \epsilon_n)$-DP by* **parallel composition**.
3. *By the* **post-processing immunity** *property, any post-processed output of $\mathcal{M}_i(D)$ still satisfies ϵ_i-DP .*

A commonly used relaxation of ε-DP is (ε, δ)-DP. (ε, δ)-DP is useful for expressing privacy guarantees of a variety of differentially private algorithms, for example, the ones based on the Gaussian additive noise mechanism and, more importantly, those whose analysis follows from composition theorems.

Definition 3 ((ε, δ)-DP [6]). *A mechanism $\mathcal{M}\colon \mathcal{X} \to \Theta$ is (ε, δ)-DP if for every pair of neighboring datasets $X, X' \in \mathcal{X}$, and every possible (measurable) output set $E \subseteq \Theta$ the following inequality holds:*

$$Pr[\mathcal{M}(X) \in E] \leq e^{\varepsilon} Pr[\mathcal{M}(X') \in E] + \delta.$$

3.2 Rènyi Differential Privacy

Rènyi Differential Privacy (RDP) is a generalization of DP based on the concept of Rènyi divergence. It is well-suited for expressing guarantees of the composition of heterogeneous mechanisms, especially those applied to the data subsamples.

Definition 4 (RDP [17]). *A mechanism $\mathcal{M}$ is (α, ε)-RDP with order $\alpha \in (1, \infty)$ if for all neighboring datasets X, X'*

$$D_{\alpha}(\mathcal{M}(X)||\mathcal{M}(X')) = \frac{1}{\alpha - 1} \log E_{\theta \sim \mathcal{X}} \left[\left(\frac{p\mathcal{M}(X)(\theta)}{p\mathcal{M}(X')(\theta)} \right)^{\alpha} \right] \leq \varepsilon.$$

We note that as $\alpha \to \infty$, RDP converges to the pure ε-DP. For convenience, we consider RDP in its function form. Hence, we denote $\varepsilon_{\mathcal{M}}(\alpha)$ as the RDP ε of $\mathcal{M}$ at order α. The function $\varepsilon_{\mathcal{M}}(\cdot)$ provides a clear characterization of the privacy guarantee associated with $\mathcal{M}$. Specifically, for popular base mechanisms $\mathcal{M}$ which can be Gaussian or Laplace mechanism, their RDP formulas are known analytically [17]. In this work, we will use the following RDP formula corresponding to the Laplacian mechanism

$$\varepsilon_{\text{LAP}}(\alpha) = \frac{1}{\alpha - 1} \log \left(\left(\frac{\alpha}{2\alpha - 1} \right) e^{\frac{\alpha - 1}{\beta}} + \left(\frac{\alpha - 1}{2\alpha - 1} \right) e^{\frac{-\alpha}{\beta}} \right) \tag{2}$$

for $\alpha > 1$, where β is the scale parameter of the Laplace distribution. More generally, we can convert RDP to the standard (ε, δ)-DP for any $\delta > 0$ using the following result.

Lemma 5 (From RDP to (ε, δ)-DP [17]). *If a mechanism $\mathcal{M}_1$ satisfies (α, ε)-RDP, then $\mathcal{M}_1$ also satisfies $(\varepsilon + \frac{\log 1/\delta}{\alpha - 1}, \delta)$-DP for any $\delta \in (0, 1)$.*

In general, for a composed mechanism $\mathcal{M} = (\mathcal{M}_1, \ldots, \mathcal{M}_t)$, Lemma 5 allows us to provide an (ε, δ)-DP guarantee as follows.

$$\delta \Rightarrow \varepsilon : \varepsilon(\delta) = \min_{\alpha>1} \frac{\log(1/\delta)}{\alpha - 1} + \varepsilon_{\mathcal{M}}(\alpha - 1), \tag{3}$$

$$\varepsilon \Rightarrow \delta : \delta(\varepsilon) = \min_{\alpha>1} e^{(\alpha-1)(\varepsilon_{\mathcal{M}}(\alpha-1)-\varepsilon)}. \tag{4}$$

In our work, we use (3) to obtain a (ε, δ) guarantee. Another notable advantage of RDP over (ε, δ)-DP is that it composes very naturally.

Lemma 6 (Composition with RDP). *Let $\mathcal{M} = (\mathcal{M}_1, \ldots, \mathcal{M}_t)$ be mechanisms where $\mathcal{M}_i$ can potentially depend on the outputs of $\mathcal{M}_1, \ldots, \mathcal{M}_{i-1}$. Then $\mathcal{M}$ obeys RDP with $\varepsilon_M(\cdot) = \sum_{i=1}^{t} \varepsilon_{M_i}(\cdot)$.*

Lemma 6 implies that the privacy loss by the composition of two mechanisms $\mathcal{M}_1$ and $\mathcal{M}_2$ is

$$\varepsilon_{\mathcal{M}_1 \times \mathcal{M}_2}(\cdot) = [\varepsilon_{\mathcal{M}_1} + \varepsilon_{\mathcal{M}_2}](\cdot).$$

3.3 Privacy Amplification by Subsampling

A commonly used approach in privacy is *subsampling* in which the DP mechanism is applied on the randomly selected sample from the data. Subsampling offers stronger privacy guarantee in that the one data point that differs between two neighboring datasets has a decreased probability of appearing in the smaller sample. That is, when we apply an (ε, δ)-DP mechanism to a random γ-subset of the data, the entire procedure satisfies $(O(\gamma\varepsilon), \gamma\delta)$-DP. The intuitive notion of amplifying the privacy by subsampling is that the privacy guarantees of the DP mechanism can be amplified by applying it to a small random subsample of records from a given dataset [3]. This is also referred to as the *subsampling lemma* in the literature [3]. Under some restrictions on α, we can represent the combination of the subsampling lemma and the tight advanced composition of RDP [24, 28] as:

$$\varepsilon_{MoSample_\gamma}(\alpha) \leq O(\gamma^2 \varepsilon_{\mathcal{M}}(\alpha)).$$

In this paper, we apply the Poisson subsampled RDP-amplification bound from [28].

Definition 7 (PoissonSample). *Given a dataset X, the mechanism PoissonSample outputs a subset of the data $\{x_i | \sigma_i = 1, i \in [n]\}$ by sampling $\sigma_i \sim Ber(\gamma)$ independently for $i = 1, \ldots, n$.*

The mechanism is equivalent to the "sampling without replacement" scheme with $m \sim \text{Binomial}(\gamma, n)$. As $n \to \infty, \gamma \to 0$ while $\gamma n \to \zeta$, the Binomial distribution converges to a Poisson distribution with parameter ζ. Here, we provide the tight privacy amplification bound for PoissonSample, which we later use to analyze our approach.

Theorem 8 (General upper bound [28]). *Let $\mathcal{M}$ be any mechanism that obeys $(\alpha, \varepsilon(\alpha))$-RDP. Let γ be the sub-sampling probability and then we have for integer $\alpha \geq 2$,*

$$\varepsilon_{MoPoissonSample}(\alpha) \leq \frac{1}{\alpha - 1} \log \Bigg\{ (1-\gamma)^{\alpha-1}(\alpha\gamma - \gamma + 1) + \binom{\alpha}{2} \gamma^2 (1-\gamma)^{\alpha-2} e^{\varepsilon(2)} + 3 \sum_{\ell=3}^{\alpha} \binom{\alpha}{\ell} (1-\gamma)^{\alpha-\ell} \gamma^{\ell} e^{(\ell-1)\varepsilon(\ell)} \Bigg\}. \tag{5}$$

4 OUR APPROACH

4.1 Setting and Notations

We assume that in addition to the private graph (with node features and labels), a non-overlapping public graph (with node features) exists and is available to us. This is a fair assumption [19] as even for sensitive medical datasets there exist some publicly available datasets. The nodes of the public graph are unlabeled.

We denote a graph by $G = (V, E)$ where V is the node-set, E represents the edges among the nodes. Let X denote the feature matrix for the node-set V, such that $X(i)$ corresponds to the feature vector for node i. We use additionally the superscript † to denote private data, such as the private graph $G^\dagger = (V^\dagger, E^\dagger)$ with node feature matrix $X^\dagger$ and labels $Y^\dagger$. For simplicity of notation, we denote public elements without any superscript, such as the public (student) GNN Φ or the set of query nodes $Q \subset V$. In addition, we denote for a node $v \in V$ with $\mathcal{N}^\ell(v)$ the ℓ-hop neighborhood, that is with $\mathcal{N}^0(v) = \{v\}$ recursively defined as

$$\mathcal{N}^\ell(v) = \{u | (u, w) \in E \text{ and } w \in \mathcal{N}^{\ell-1}(v)\}. \tag{6}$$

4.2 Algorithm of PrivGnn

We organize our proposed PrivGnn method into three phases: private data selection, retrieval of noisy pseudo-labels, and student model training. The algorithm of our PrivGnn method is shown in Algorithm 1. We start by sampling a set of query nodes Q randomly from the public dataset.

Algorithm 1: PrivGnn

Input: Private graph $G^\dagger = (V^\dagger, E^\dagger)$ with private node feature matrix $X^\dagger$ and private labels $Y^\dagger$; Unlabeled public graph $G = (V, E)$ with public features X;
Hyperparam.: K the number of training data points for training $\Phi^\dagger$; β the scale of Laplacian noise; γ the subsampling ratio
Output: Privately trained student model Φ

Sample public query node set $Q \subset V$
Select random subset, $\hat{V}^\dagger \subset V^\dagger$, using Poisson sampling:
  For $\sigma_i \sim Ber(\gamma), v_i \in V^\dagger$ select $\hat{V}^\dagger = \{v_i | \sigma_i = 1, v_i \in V^\dagger\}$
for *query* $v \in Q$ **do**
  $V^\dagger_{\text{KNN}}(\text{v}) = \text{argmin}_K\{\text{d}(X(v), X^\dagger(u)), u \in \hat{V}^\dagger\}$
    `// Retrieve the K-nearest-neighbors of v`
  Construct the induced subgraph H of the node set $V^\dagger_{\text{KNN}(v)}$ from the graph $G^\dagger$
  Initialize and train the GNN $\Phi^\dagger$ on subgraph H using the private labels $Y^\dagger_{|H}$
  Compute pseudo-label $\tilde{y}_v$ using noisy posterior of $\Phi^\dagger$: $\tilde{y}_v = \text{argmax}\{\Phi^\dagger(v) + \{\eta_1, \eta_2 \ldots, \eta_c\}\}, \eta_i \sim \text{Lap}(0, \beta)$
Train GNN Φ on G using the training set Q with the pseudo-labels $\tilde{Y} = \{\tilde{y}_v | v \in Q\}$
return *Trained student GNN model* Φ

4.2.1 Private data selection (Lines 2-5). In order to benefit from the privacy amplification via subsampling, we first apply PoissonSample (see Def. 7) to the private data with sampling ratio γ. Then we use K-nearest neighbor (KNN) on the retrieved subset $\hat{V}^\dagger$ by measuring the distance between the feature space of the query and the nodes in the subset $\hat{V}^\dagger$. In other words, for a query node $v \in Q$ we retrieve the node-set $V^\dagger_{\text{KNN}(v)}$ with

$$V^\dagger_{\text{KNN}}(v) = \text{argmin}_K\{\text{d}(X(v), X^\dagger(u)), u \in \hat{V}^\dagger\},$$

where $\text{d}(\cdot,\cdot)$ denotes a distance function, such as Euclidean distance or cosine distance, and $X(u)$ is the feature vector of the node u. The subgraph H induced by the node set $V^\dagger_{\text{KNN}}(v)$ of the private graph $G^\dagger$ with the subset of features $X^\dagger_{|H}$ and the subset of labels $Y^\dagger_{|H}$ constitute the selected private data (for the query node v).

4.2.2 Retrieval of noisy pseudo-labels (Lines 5-7). As the next step, we want to retrieve a pseudo-label for the public query node v. We train the private teacher GNN $\Phi^\dagger$ using the selected private data: the subgraph H with features $X^\dagger_{|H}$ and labels $Y^\dagger_{|H}$. Then, we retrieve the prediction $\Phi^\dagger(v)$ using the ℓ-hop neighborhood $\mathcal{N}^\ell(v)$ (corresponding to ℓ-layer GNN) of the query node v and the respective subset of feature vectors from the public data. Afterward, we add independent Laplacian noise to each coordinate of the posterior with noise scale $\beta = \frac{1}{\lambda}$ to obtain the noisy pseudo-label $\tilde{y}_v$ of the query node v

$$\tilde{y}_v = \text{argmax}\left\{\Phi^\dagger(v) + \{\eta_1, \eta_2 \ldots, \eta_c\}\right\}, \eta_i \sim \text{Lap}(0, \beta).$$

Note that we build separate teacher GNN models corresponding to each query node.

Inductive teacher GNN. We note that the teacher GNN is applied in an inductive setting. In particular, we use our teacher GNN to infer labels on a public query node that was not seen so far during training. Therefore, the GNN most effective in inductive settings like GraphSage [8] should be used as the teacher GNN.

4.2.3 Student model (transductive) training (Line 8). Our private models $\Phi^\dagger$ only answer the selected number of queries from the public graph. This is because each time the model $\Phi^\dagger$ is queried, it utilizes the model trained on private data, which will lead to increased privacy costs. Therefore, all the selected answered queries are used as pseudo-labels together with the unlabeled data to train a student model Φ in a transductive setting. We then release the public student model Φ which is trained using the noisy pseudo-labels. We note that public model Φ when released is differential private (more details on privacy guarantees in Section 4.3) based on the postprocessing property of differential privacy from the plausible deniability of the pseudo-labels.

4.3 Privacy Analysis

Theorem 9. *For any $\delta > 0$, Algorithm 1 is (ε, δ)-DP with*

$$\varepsilon \leq \log\left(\frac{1}{\sqrt{\delta}}\right) + |Q| \log\left(1 + \gamma^2\left(\frac{2}{3}e^{1/\beta} + \frac{1}{3}e^{-2/\beta} - 1\right)\right), \quad (7)$$

where Q is the set of query nodes chosen from the public dataset, β is the scale of the Laplace mechanism.

Proof. Our algorithm is composed of a (i) sampling mechanism in which a small sample of private data is used to train the private GNN model (ii) Laplacian mechanism (with scale parameter β), which is used to generate a noisy label for the public node (queried on corresponding private GNN).

First, note that as the L_1 norm of the posterior is bounded by 1, we add independent Laplacian noise to each posterior element with scale β (Note that each posterior element is also bounded by 1). The computation of noisy label for each public query node is, therefore, $1/\beta$-DP. The sequential composition (c.f. Theorem 2) over N queries will result in a crude bound over the DP guarantee of our approach, namely N/β. To obtain a tighter bound which takes into account taking the effect of the private data subsampling, we perform the transformation of the privacy variables using the RDP formula for Laplacian mechanism as given in Eq. (2) for $\alpha > 1$

$$\varepsilon_{\text{Lap}}(\alpha) = \frac{1}{\alpha - 1}\log\left(\left(\frac{\alpha}{2\alpha - 1}\right)e^{\frac{\alpha-1}{\beta}} + \left(\frac{\alpha - 1}{2\alpha - 1}\right)e^{\frac{-\alpha}{\beta}}\right). \quad (8)$$

Moreover, the model uses only a random sample of the data to select the nodes for training the private model. We, therefore, apply the tight advanced composition of Theorem 8 to obtain $\varepsilon_{\text{LapoPois}}(\alpha)$ as given in Eq. (5). To simplify the expression, we set $\alpha = 2$ in Eq. (5) and obtain

$$\varepsilon_{\text{LapoPois}}(2) \leq \log\left(1 - \gamma^2 + \gamma^2 e^{\varepsilon_{\text{Lap}}(2)}\right). \quad (9)$$

Substituting $\varepsilon_{\text{Lap}}(2)$ in (9), we obtain

$$\varepsilon_{\text{LapoPois}}(2) = \log\left(1 - \gamma^2 + \gamma^2\left(\frac{2}{3}\cdot e^{1/\beta} + \frac{1}{3}\cdot e^{-2/\beta}\right)\right). \quad (10)$$

Now applying the advanced composition for RDP (Lemma 6) for N queries, we get the upper bound on total privacy loss of our approach at $\alpha = 2$

$$\varepsilon_{\text{PrivGnn}}(2) \leq |Q| \log\left(1 - \gamma^2 + \gamma^2\left(\frac{2}{3}\cdot e^{1/\beta} + \frac{1}{3}\cdot e^{-2/\beta}\right)\right)$$

For any given $\delta > 0$, we use Lemma 5 and Eq. (3) to obtain the (ε, δ)-DP guarantee. Specifically for any $\delta > 0$ we obtain

$$\varepsilon(\delta) = \min_{\alpha>1}\frac{\log(1/\delta)}{\alpha - 1} + \varepsilon_{\text{PrivGnn}}(\alpha - 1), \quad (11)$$

Substituting $\alpha = 3$ in the above we obtain the stated upper bound, i.e.,

$$\varepsilon(\delta) \leq \log\left(\frac{1}{\sqrt{\delta}}\right) + |Q| \log\left(1 + \gamma^2\left(\frac{2}{3}e^{1/\beta} + \frac{1}{3}e^{-2/\beta} - 1\right)\right),$$

thereby completing the proof. □

Note that the above expression provides only a rough upper bound. It is not suggested to use it for manual computation of ε as it would give a much larger estimate. We provide it here for simplicity and to show the effect of the number of queries and sampling ratio on the final privacy guarantee. For our experiments, we compute a tighter bound (for the final $\varepsilon(\delta)$ with $\delta < 1/|V^\dagger|$) using numerical methods with $\alpha \in \{2, \ldots, 32\}$ and report the corresponding best ε.

5 EXPERIMENTAL EVALUATION

In this section, we demonstrate the effectiveness of our PrivGnn method for the node classification task. We show that our model performs better in terms of both accuracy and privacy cost as compared to state-of-the-art methods such as PATE [19] where the data splitting is the bottleneck.

Some parameters are specific to privacy, such as the scale of noise, β, injected for each query. We set the privacy parameter $\lambda = 1/\beta$. To demonstrate the effects of this parameter, we vary λ and report the corresponding privacy budget for all queries. We set the reference values as follows $\{0.1, 0.2, 0.4, 0.8, 1\}$. We set δ to $10^{-4}, 10^{-5}$, and 10^{-5} for Amazon, ArXiv and Reddit, respectively. All our experiments were conducted for 10 different instantiations, and we report the mean values across all runs.

With our experiments, we aim to answer the following research questions:

RQ 1. *How do the performance and privacy guarantees of PrivGnn compare to the non-private and other private baselines?*

RQ 2. *What is the effect of sampling different K-nearest neighbors on the performance of private PrivGnn method?*

RQ 3. *How does the final privacy budget of different approaches compare with an increase in the number of query nodes ($|Q|$)?*

RQ 4. *What is the effect of varying sampling ratio on the privacy-utility tradeoff for PrivGnn?*

5.1 Baselines

We compare our PrivGnn approach with three baselines.

Non-Private Inductive Baseline (B1). In the non-private inductive baseline, we train a single GNN model on all private data and test the performance of the model on the test set of the public data. This corresponds to releasing the model trained on complete private data without any privacy guarantees.

Non-Private Transductive Baseline (B2). This non-private baseline estimates the "best possible" performance on the public data. Specifically, a GNN model is trained using the training set (50% of the public data for the Amazon and Reddit, and 60% for the ArXiv dataset) and their corresponding ground truth labels in a transductive setting. We then test the model on the test set (the remaining 50% of the public data for Amazon and Reddit, and 40% for the ArXiv dataset).

Private Aggregation of Teacher Ensembles (PATE). We adopt the PATE framework [19] where the private node set is partitioned into n disjoint subsets of nodes. Then we train GNN models (teacher models) on each dataset (the corresponding node induced graph) separately. We then query each of the teacher models with nodes from the public domain and aggregate the prediction of all teacher models based on their label counts. Then independent random noise is added to each of the vote counts. The obtained noisy label is then used in training the student model, which is later released. We call this PateG. We also used multilayer perceptron (MLP) instead of GNN models for training on the disjoint node subset of the private graph. We denote the MLP approach as PateM. Note that only the features (no edge information) will be used for training teacher models in PateM. Our choice of n was experimentally chosen. We observe that $n > 20$ for Amazon dataset and $n > 50$ for the ArXiv and Reddit datasets leads to extremely small data in each partition and low accuracy. Therefore, we use $n = \{20, 50, 50\}$ for the Amazon, ArXiv and Reddit datasets respectively.

5.2 Datasets

Table 1: Data Statistics. $|V|$ and $|E|$ denote the number of vertices and edges in the corresponding graph dataset. deg is the average degree of the graph. We select 50% of the Amazon and Reddit, and 40% of the ArXiv dataset as test set from the public graph.

			Private			Public		
	#class	$\lvert X\rvert$	$\lvert V^{\dagger}\rvert$	$\lvert E^{\dagger}\rvert$	deg	$\lvert V\rvert$	$\lvert E\rvert$	deg
Amazon	10	767	2500	11595	9.28	6000	37171	6.20
ArXiv	40	128	90941	187419	2.06	78402	107900	1.37
Reddit	41	602	12300	266148	21.64	30000	876846	29.23

We perform our experiments on three representative datasets, namely Amazon, ArXiv, and Reddit. The statistics of the dataset are shown in Table 1. We briefly discuss each dataset below.

Amazon. We used the Amazon Computers [22] which is a subset of Amazon co-purchase graph [14]. The nodes represent products, and the node features are product reviews represented as bag-of-words. The edges indicate that two products are frequently bought together, while class labels are the product categories. We created the private graph on 2500 nodes and used 3000 nodes each for the public trainset and test set. The task for this dataset is to assign products to their respective product category.

ArXiv. The ArXiv dataset [10] is a citation network where each node represents an arXiv paper and edges represent that a paper cites the other. The node feature is a 128-dimensional feature vector obtained by averaging the embeddings of words in the title and abstract of each paper. Our private graph is created from 90941 papers published until 2017. We used 48603 papers published in 2018 and 29799 papers published since 2019 for our public train nodes and test nodes, respectively.

Reddit. The Reddit dataset [8] represents the post-to-post interactions of a user. An edge between two posts indicates that the same user commented on both posts. The labels correspond to the community of a post. We randomly sample 300 nodes from each class for the private graph and selected 15000 nodes each for the public train and public test nodes, respectively.

5.3 Experimental setup

For each of the private (teacher) ($\Phi^{\dagger}$) and public (student) (Φ) GNN models, we used a two-layer GraphSAGE model [8] with hidden dimension of 64 and RELU activation function. We applied batch normalization on the output of the first layer. We applied a dropout of 0.5 and a learning rate of 0.01. For the multilayer perceptron model (MLP) used in PateM, we used three fully connected layers and RELU activation layers. We trained all models for 500 epochs

using the negative log-likelihood loss function for node classification with the Adam optimizer. We fix the subsampling ratio γ to 0.3 unless otherwise stated.

All experiments were conducted using PyTorch Geometric library [7] and Python3 on 11GB GeForce GTX 1080 Ti GPU. Our anonymized code is available.

6 RESULTS

Having our experimental setup established, we now state and discuss the results of our four research questions presented in Section 5.

6.1 Privacy-utility Tradeoff (RQ 1)

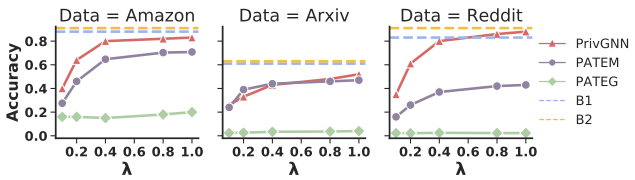
Figure 2 shows the performance of our privacy-preserving PrivGnn method as compared to two private and two non-private baselines. Since the achieved privacy guarantees depend on the injected noise level, we report the performance and the final incurred privacy budget with respect to $\lambda = 1/\beta$ which is inversely proportional to noise injected corresponding to each query.

First, the performance of our PrivGnn approach converges quite fast to the non-private method, B1, as the noise level decreases. Note that B1 used all the private data for training and has no privacy guarantees. The second non-private baseline B2, trained using over 50% of the public data, achieves slightly higher results than B1. For Reddit, PrivGnn even slightly outperforms B1 for $\lambda > 0.4$.

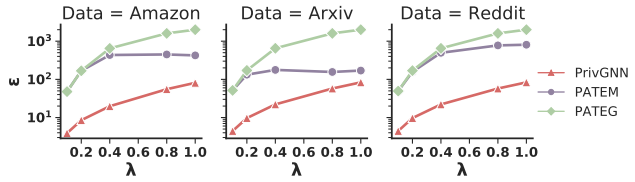
Secondly, compared with the private methods (PateM and PateG), PrivGnn achieves significantly better performance. For instance, at a smaller $\lambda = 0.2$, we achieved 40% and 300% improvement in accuracy (with respect to private baselines) respectively on the Amazon dataset. On the ArXiv dataset, we achieved 15% decrease over PateM. Note that the teacher models of PateM are MLPs and do not utilize graph structure. The observed result is probably because the node degree of ArXiv is very small (≤ 2). Thus, GNNs, which uses the aggregation of the neighbors of a node to make predictions might not benefit from such low degree graphs. However, against PateG, we achieved 1169% increment. We observe that for a larger graph with a high average degree, we achieved an even higher accuracy of 134% and 2672% improvement when compared to the PATE methods. Nevertheless, as we will discuss next, PrivGnn incurs significantly lower privacy costs as compared the the two private baselines.

Thirdly, we plot the incurred privacy budget ε corresponding to different λ in Figure 2b. Here we set $|Q| = 1000$ for all three private methods. PrivGnn achieves a relatively small ε of 8.53 while PateM and PateG achieves 167.82 on the Amazon dataset for $\lambda = 0.2$. On the ArXiv dataset, although PateM performs slightly better in terms of accuracy than PrivGnn, the privacy cost of PateM is significantly large which renders it useless. For instance, PrivGnn achieves ε of 4.45 while PateM has a corresponding ε of 51.51. We observe that the privacy budget increases as the λ gets larger. This phenomenon is expected since larger λ implies low privacy and higher risk (high ε).

Summary. PrivGnn outperforms both variants of PATE by incurring a lower privacy budget and achieving better accuracy. We attribute this observation to the following reasons. **First**, our privacy budget is primarily reduced due to the use of subsampling mechanism, whereas PATE uses the complete private data. **Second**, in case of PrivGnn, we train a personalized GNN for each query node using nodes closest to the query node and the induced relations among them. We believe that this leads to more accurate labels used to train the student model, resulting in better performance. The worse privacy and accuracy of PATE, on the other hand, indicates a larger disagreement between the teachers. The random data partitioning in PATE destroys the graph structure. Moreover, teachers trained on disjoint and disconnected portions of the data might overfit to specific data portions, hence the resulting disagreement.



(a) Accuracy vs. Noise ($\propto 1/\lambda$) injected to each query.



(b) Privacy budget vs. Noise ($\propto 1/\lambda$) injected to each query.

Figure 2: Privacy-utility analysis. Here, $|Q|$ is set to 1000. For PrivGnn, γ is set to 0.3.

6.2 Effect of K on Accuracy (RQ 2)

To quantify the effect of the number of private nodes on the accuracy of PrivGnn, we vary the number of neighbors K used in K-nearest neighbors. Note that the computed privacy guarantee is independent of K. For the Amazon dataset, we set $K = \{300, 750\}$ and for the ArXiv and Reddit dataset, we set K to {750, 1000, 3000}. We select smaller K for Amazon due to the small size of the private dataset.

As shown in Table 2, on the Amazon dataset, sampling $K = 750$ nodes achieves over 50% improvement than using $K = 300$ private nodes for training. On the ArXiv dataset, the difference in performance between the using 750 nodes and 3000 nodes for training is quite marginal (about 5%). On the Reddit dataset, using larger K improved the accuracy by up to 25%. From the above results, we conclude that the value of the hyperparameter K should be chosen based on the average degree of the graph. A small K suffices for sparse graphs (e.g. ArXiv) while for graphs with higher average degree (e.g. Reddit) a larger K leads to better performance.

6.3 Effect of $|Q|$ on accuracy and privacy (RQ 3)

Since the privacy budget is highly dependent on the number of queries answered by the teacher GNN model, we compare the performance and relative privacy budget incurred for answering different numbers of queries. In Figure 3, we observe a negligible difference between when 1000 query nodes are pseudo-labeled and

Table 2: Accuracy for varying K and $|Q| = 1000$.

	Amazon		ArXiv			Reddit		
λ/K	300	750	750	1000	3000	750	1000	3000
0.1	0.17	0.40	0.21	0.24	0.26	0.30	0.33	0.35
0.2	0.34	0.64	0.33	0.34	0.33	0.48	0.53	0.61
0.4	0.53	0.80	0.46	0.46	0.43	0.65	0.72	0.80
0.8	0.56	0.82	0.48	0.47	0.48	0.69	0.76	0.86
1.0	0.59	0.83	0.50	0.48	0.52	0.71	0.77	0.88

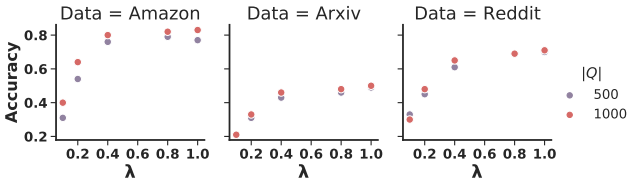


Figure 3: Accuracy for different number of queries answered by the teacher model of PrivGnn

when only 500 queries are employed across all datasets at different noise levels.

Further looking at detailed results in Table 3, for the different ranges of λ on the Amazon dataset, we observe up to 46% decrease in privacy budget of PrivGnn for answering 500 queries over answering 1000 queries. On the ArXiv and Reddit dataset, we observe a 31% decrease and up to 45% decrease in privacy budget. This implies that smaller $|Q|$ is desirable which PrivGnn offers.

In Table 3, we also compare PrivGnn with PATE variants with respect to the incurred privacy budget ε for answering the different number of queries. We observe that our method offers a significantly better privacy guarantee (over 20 times reduction in ε) than PateM and PateG across all datasets. While decreasing the number of queries shows a negligible change in accuracy, it greatly reduces the incurred privacy budge for PrivGnn. In particular, the privacy budget (see Table 3) , ε, of PrivGnn is reduced by 50% on the Amazon dataset and by 40% on the ArXiv and Reddit dataset with only a negligible reduction in accuracy.

Active Learning for Query Selection. In order to further improve the privacy guarantee while maintaining a good accuracy, we experimented with various active learning strategies for intelligent selection of query nodes to be labeled. In particular, we employed structure-based approaches such as clustering and ranking based on centrality measures such as PageRank and degree centrality. However, we did not observe any significant impact on model performance compared to when queries are sampled randomly.

6.4 Varying subsampling ratio γ (RQ 4)

A smaller sampling ratio will reduce the amount of private data used for training which will, in turn, reduce the privacy budget ε. Therefore, to validate the effectiveness of the sampling ratio, we decrease γ from 0.3 to 0.1. This allows us to further benefit from the privacy amplification by subsampling as explained in Section 3.3. Since γ affects the amount of available data to train the teacher

Table 3: Privacy budget (ε) for varying number of queries $|Q|$ answered by the teacher.

	$\|Q\|$	Method	$\lambda = 0.1$	$\lambda = 0.2$	$\lambda = 0.4$	$\lambda = 0.8$	$\lambda = 1.0$
Amazon	500	PrivGnn	2.67	5.69	13.15	31.10	44.07
		PateM	27.82	87.82	223.08	233.46	220.57
		PateG	27.82	87.82	327.82	800.97	1000.97
	1000	PrivGnn	3.90	8.53	19.81	55.30	81.23
		PateM	47.82	167.82	434.69	449.36	424.78
		PateG	47.82	167.82	647.82	1600.97	1967.58
ArXiv	500	PrivGnn	3.05	6.45	14.31	33.40	46.37
		PateM	31.51	72.21	72.83	65.12	67.61
		PateG	31.51	91.51	331.51	801.43	1001.43
	1000	PrivGnn	4.45	9.69	22.11	57.60	83.53
		PateM	51.51	132.86	177.04	156.39	170.14
		PateG	51.51	171.51	651.51	1601.43	2001.43
Reddit	500	PrivGnn	3.05	6.45	14.31	33.40	46.37
		PateM	29.43	83.73	243.45	392.23	415.35
		PateG	29.43	89.43	329.43	801.17	1001.17
	1000	PrivGnn	4.45	9.69	22.11	57.60	83.53
		PateM	49.43	157.90	496.61	775.95	808.43
		PateG	49.43	169.43	649.43	1601.17	2001.17

model, we only perform this experiment on large datasets (ArXiv and Reddit). We exclude Amazon because having low γ on the Amazon dataset will render the data useless, and no meaningful graph can be reconstructed. As shown in Table 4, the accuracy of using the sampling ratio $\gamma = 0.3$ on the ArXiv dataset is comparable to that of $\gamma = 0.1$. On the Reddit dataset, we observe up to a 14% decrease in the accuracy when $\gamma = 0.1$. Nonetheless, we observe a 318% reduction in the privacy budget when $\gamma = 0.1$. This implies that smaller γ offers better ε on the both datasets at low cost to accuracy. We conclude that PrivGnn offers an overall good priavcy-utlity tradeoff with respect to γ. In other words for larger datasets, significant reduction in privacy costs can be achieved (when using smaller γ) with relatively lower degradation in accuracy.

Table 4: Performance and the respective privacy budget for different sampling ratio γ with $|Q| = 500$ and $K = 3000$.

	$\gamma = 0.1$			$\gamma = 0.3$		
λ	ε	**ArXiv**	**Reddit**	ε	**ArXiv**	**Reddit**
0.1	0.96	0.19	0.29	3.05	0.26	0.38
0.2	1.96	0.29	0.43	6.45	0.29	0.60
0.4	4.03	0.43	0.58	14.31	0.42	0.78
0.8	8.72	0.45	0.65	33.40	0.47	0.83
1.0	11.16	0.46	0.68	46.37	0.50	0.84

7 CONCLUSION

We propose PrivGnn, a novel privacy-preserving framework for releasing GNN models with differential privacy guarantees. Our approach leverages the knowledge distillation framework, which

transfers to a *student* model the knowledge of *teacher* models trained on private data. Privacy is intuitively guaranteed due to private data sub-sampling as well as noisy labeling of public data. Moreover, as we build *personalized* teacher models by using the K-nearest neighbors of the corresponding query nodes, the trained teacher model is more confident in its prediction. We present the privacy analysis of our approach by leveraging the Rènyi differential privacy framework. Our experiment results on three real-world datasets show the effectiveness of our approach to obtain good accuracy under practical privacy guarantees.

REFERENCES

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 308–318.

[2] David Ahmedt-Aristizabal, Mohammad Ali Armin, Simon Denman, Clinton Fookes, and Lars Petersson. 2021. Graph-Based Deep Learning for Medical Diagnosis and Analysis: Past, Present and Future. *Sensors (Basel, Switzerland)* 21, 14 (July 2021). https://doi.org/10.3390/s21144758

[3] Borja Balle, Gilles Barthe, and Marco Gaboardi. 2018. Privacy amplification by subsampling: Tight analyses via couplings and divergences. *arXiv preprint arXiv:1807.01647* (2018).

[4] Ghazaleh Beigi, Ahmadreza Mosallanezhad, Ruocheng Guo, Hamidreza Alvari, Alexander Nou, and Huan Liu. 2020. Privacy-aware recommendation with private-attribute protection using adversarial learning. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 34–42.

[5] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*. Springer, 265–284.

[6] Cynthia Dwork, Aaron Roth, et al. 2014. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science* 9, 3-4 (2014), 211–407.

[7] Matthias Fey and Jan Eric Lenssen. 2019. Fast graph representation learning with PyTorch Geometric. *arXiv preprint arXiv:1903.02428* (2019).

[8] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *NIPS*.

[9] Xinlei He, Rui Wen, Yixin Wu, Michael Backes, Yun Shen, and Yang Zhang. 2021. Node-Level Membership Inference Attacks Against Graph Neural Networks. *arXiv preprint arXiv:2102.05429* (2021).

[10] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020. Open graph benchmark: Datasets for machine learning on graphs. *arXiv preprint arXiv:2005.00687* (2020).

[11] Timour Igamberdiev and Ivan Habernal. 2021. Privacy-Preserving Graph Convolutional Networks for Text Classification. *arXiv preprint arXiv:2102.09604* (2021).

[12] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR)*.

[13] Sofia Ira Ktena, Sarah Parisot, Enzo Ferrante, Martin Rajchl, Matthew Lee, Ben Glocker, and Daniel Rueckert. 2018. Metric learning with spectral graph convolutions on brain connectivity networks. *NeuroImage* 169 (2018), 431–442.

[14] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-Based Recommendations on Styles and Substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Santiago, Chile) *(SIGIR '15)*. Association for Computing Machinery, New York, NY, USA, 43–52. https://doi.org/10.1145/2766462.2767755

[15] Kevin McCloskey, Ankur Taly, Federico Monti, Michael P Brenner, and Lucy J Colwell. 2019. Using attribution to decode binding mechanism in neural network models for chemistry. *Proceedings of the National Academy of Sciences* 116, 24 (2019), 11624–11629.

[16] Frank D McSherry. 2009. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*. 19–30.

[17] Ilya Mironov. 2017. Rényi differential privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*. IEEE, 263–275.

[18] Iyiola E Olatunji, Wolfgang Nejdl, and Megha Khosla. 2021. Membership inference attack on graph neural networks. *arXiv preprint arXiv:2101.06570* (2021).

[19] Nicolas Papernot, Martín Abadi, Ulfar Erlingsson, Ian Goodfellow, and Kunal Talwar. 2016. Semi-supervised knowledge transfer for deep learning from private training data. *arXiv preprint arXiv:1610.05755* (2016).

[20] Sina Sajadmanesh and Daniel Gatica-Perez. 2020. Locally Private Graph Neural Networks. *arXiv preprint arXiv:2006.05535* (2020).

[21] Chuanqiang Shan, Huiyun Jiao, and Jie Fu. 2021. Towards Representation Identical Privacy-Preserving Graph Neural Network via Split Learning. *arXiv preprint arXiv:2107.05917* (2021).

[22] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. 2018. Pitfalls of graph neural network evaluation. *arXiv preprint arXiv:1811.05868* (2018).

[23] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. *International Conference on Learning Representations* (2018).

[24] Yu-Xiang Wang, Borja Balle, and Shiva Prasad Kasiviswanathan. 2019. Subsampled Rényi differential privacy and analytical moments accountant. In *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 1226–1235.

[25] Chuhan Wu, Fangzhao Wu, Yang Cao, Yongfeng Huang, and Xing Xie. 2021. Fedgnn: Federated graph neural network for privacy-preserving recommendation. *arXiv preprint arXiv:2102.04925* (2021).

[26] Shijie Zhang, Hongzhi Yin, Tong Chen, Zi Huang, Lizhen Cui, and Xiangliang Zhang. 2021. Graph Embedding for Recommendation against Attribute Inference Attacks. In *Proceedings of the Web Conference 2021*. 3002–3014.

[27] Jun Zhou, Chaochao Chen, Longfei Zheng, Huiwen Wu, Jia Wu, Xiaolin Zheng, Bingzhe Wu, Ziqi Liu, and Li Wang. 2020. Vertically Federated Graph Neural Network for Privacy-Preserving Node Classification. *arXiv preprint arXiv:2005.11903* (2020).

[28] Yuqing Zhu and Yu-Xiang Wang. 2019. Poission subsampled rényi differential privacy. In *International Conference on Machine Learning*. PMLR, 7634–7642.