Releasing Graph Neural Networks with Differential Privacy Guarantees

Anonymous authors
Paper under double-blind review

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 black-box access to the trained model is granted. We propose PrivGnn, a privacy-preserving framework for releasing GNN models. PrivGnn combines the knowledge-distillation framework with the two noise mechanisms, random subsampling, and noisy labeling, to ensure rigorous privacy guarantees. We theoretically analyze our approach in the Rènyi differential privacy framework. Besides, we show the solid experimental performance of our method compared to several baselines adapted for graph-structured data. Our code is available at `https://anonymous.4open.science/r/privGnn-tmlr/`.

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 Hamilton et al. (2017), biology Ktena et al. (2018), medicine Ahmedt-Aristizabal et al. (2021), and recommender systems Fan et al. (2019); Zheng et al. (2021). Specifically, GNNs achieve 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. Hence, it cannot always be made public. Releasing models trained on such data provides an opportunity for using private knowledge beyond company boundaries Ahmedt-Aristizabal et al. (2021). For instance, a social networking company might want to release a content labelling model trained on the private user data without endangering the privacy of the participants Morrow et al. (2022). However, recent works have shown that GNNs are vulnerable to membership inference attacks Olatunji et al. (2021); He et al. (2021); Duddu et al. (2020). Specifically, membership inference attacks aim to identify which data points have been used for training the model. The higher vulnerability of GNNs to such attacks as compared to traditional models has been attributed to their encoding of the graph structure within the model Olatunji et al. (2021). In addition, the current legal data protection policies highlight a compelling need to develop *privacy-preserving GNNs*.

We propose our framework PrivGnn, which builds on the rigid guarantees of *differential privacy* (DP), allowing us to protect sensitive data while releasing the trained GNN model. DP is one of the most popular approaches for releasing data statistics or trained models while concealing the information about individuals present in the dataset Dwork et al. (2006). 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 incorporating sufficient noise into the query result.

The seminal work of Abadi et al. (2016) proposed *differential private stochastic gradient descent* (DP-SGD) algorithm to achieve differential privacy guarantees for deep learning models. Specifically, in each training step, DP-SGD adds appropriate noise to the ℓ_2-clipped gradients during the stochastic gradient descent

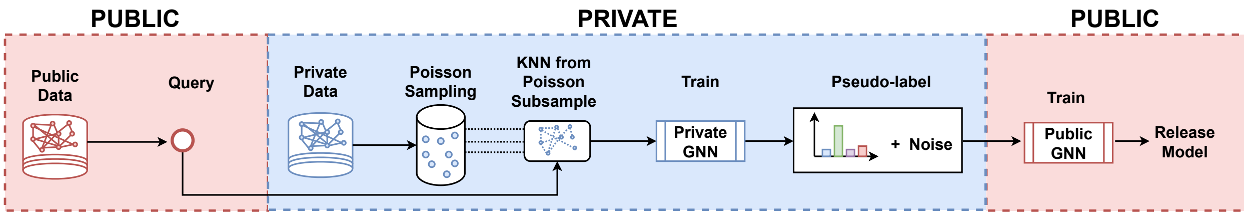


Figure 1: Workflow of PrivGnn. Given *labeled private* and *unlabeled public* datasets, PrivGnn starts by retrieving a subset of the private data using Poisson sampling. We then obtain the K-nearest neighbor nodes based on the features of the public query node. The teacher GNN model is trained on the 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.

optimization. The incurred privacy budget ε for training is computed using the moment's accountant technique. This technique keeps track of the privacy loss across multiple invocations of the noise addition mechanism applied to random subsets of the input dataset.

Besides the slow training process of DP-SGD, the injected noise is proportional to the number of training epochs, which further degrades performance. While DP-SGD is designed for independent and identically distributed data (i.i.d.), the nodes in graph data are related. In fact, GNNs make explicit use of the relational structure by recursive feature propagation over connected nodes. Hence, the privacy guarantee of DP-SGD, which requires a set of i.i.d. examples to form batches and lots, does not trivially hold for GNNs and graph data Igamberdiev & Habernal (2021).

To work around DP-SGD's dependency on the training procedure, such as the number of epochs, Papernot et al. (2016) 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 multiple 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 dataset assumption, 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** include: (i) a *novel privacy-preserving framework* for releasing GNN models (ii) *theoretical analysis* of the framework using the RDP framework (iii) *large scale empirical analysis* establishing the practical utility of the approach. Besides, we show the robustness of our model against two membership inference attacks.

2 Related Works

Graph neural networks (GNNs) Kipf & Welling (2017); Hamilton et al. (2017); Veličković et al. (2018), 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 are highly vulnerable to membership inference attacks Olatunji et al. (2021), thus highlighting the need to develop privacy-preserving GNNs.

Existing works on privacy-preserving GNNs mainly focused on a distributed setting in which the node feature values or/and labels are assumed to be private and distributed among multiple distrusting parties (Sajadmanesh & Gatica-Perez, 2020; Zhou et al., 2020; Wu et al., 2021; Shan et al., 2021). Sajadmanesh & Gatica-Perez (2020) assumed that only the node features are sensitive while the graph structure is publicly available. They developed a local differential privacy mechanism to tackle the problem of node-level privacy.

Wu et al. (2021) proposed a federated framework for privacy-preserving GNN-based recommendation systems while focusing on the task of subgraph level federated learning. Zhou et al. (2020) proposed a privacy-preserving GNN learning paradigm for node classification tasks 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. (2021) 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 types but differ in nodes, edges, and labels.

Igamberdiev & Habernal (2021) adapted differentially-private gradient-based training, DP-SGD Abadi et al. (2016), 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 to distribute the noise effectively, whereas, in GNNs, the nodes exchange information via the message passing framework during training.

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, in our setting, the whole graph with its nodes, features, and labels are 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 users and their networks. While in distributed settings, the adversary might be anyone holding a part of data, in our case, the adversary is the user having white-box 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 differs from the traditional methods of differential privacy analysis of graph statistics or differential private release of graph data. In particular, we are concerned with the differential privacy of the trained GNN model.

3 Our Approach

Setting and Notations. Here, we follow the setting of Papernot et al. (2016) in which, in addition to the private graph (with node features and labels), a public graph (with node features) exists and is available to us. This is a fair assumption, as even for sensitive medical datasets, there exist some publicly available datasets. We also note that companies usually have huge amounts of data but might only release a subset of it for research purposes. A few examples include item-user graphs in recommender systems like the Movielens GroupLensResearch (2021), Netflix dataset and click-through graphs in information retrieval like ORCAS dataset Craswell et al. (2020) which is a document-query graph released by Microsoft. The nodes of the public graph are unlabeled.

We denote a graph by $G = (V, E)$ where V is the node-set, and 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 a node $v \in V$ and the ℓ-hop neighborhood $\mathcal{N}^\ell(v)$ with $\mathcal{N}^0(v) = \{v\}$ which is recursively defined as: $\mathcal{N}^\ell(v) = \{u|(u, w) \in E \text{ and } w \in \mathcal{N}^{\ell-1}(v)\}$.

3.1 PrivGnn Framework

We organize our proposed PrivGnn method into three phases: *private data selection, retrieval of noisy pseudo-labels, and student model training.* The pseudocode for our PrivGnn method is provided in Algorithm 1. We start by randomly sampling a set of query nodes Q from the public dataset (line 1). We will use the set Q together with the noisy labels (extracted from the private data) for training the student GNN as described below.

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; data x_i, size m

Hyperparam.: K the number of training data points for training $\Phi^\dagger$; β the scale of Laplacian noise; γ the subsampling ratio

Output: Student GNN, Φ

1 Sample public query node-set $Q \subset V$

2 **for** *query* $v \in Q$ **do**

3   Select random subset, $\hat{V}^\dagger \subset V^\dagger$, using Poisson sampling:
      For $\sigma_i \sim Ber(\gamma), v_i \in V^\dagger$ select $\hat{V}^\dagger = \{v_i | \sigma_i = 1, v_i \in V^\dagger\}$

4   Retrieve the K-nearest-neighbors of v in $\hat{V}^\dagger$ $V^\dagger_{\text{KNN}}(\text{v}) = \text{argmin}_K\{\text{d}(X(v), X^\dagger(u)), u \in \hat{V}^\dagger\}$

5   Construct the induced subgraph H of the node-set $V^\dagger_{\text{KNN}(v)}$ from the graph $G^\dagger$

6   Initialize and train the GNN $\Phi^\dagger$ on subgraph H using the private labels $Y^\dagger_{|H}$

7   Compute pseudo-label $\tilde{y}_v$ using noisy posterior of $\Phi^\dagger$:
      $\tilde{y}_v = \text{argmax}\{\Phi^\dagger(v) + \{\eta_1, \eta_2 \ldots, \eta_c\}\}, \eta_i \sim \text{Lap}(0, \beta)$

8 Train GNN Φ on G using the training set Q with the pseudo-labels $\tilde{Y} = \{\tilde{y}_v | v \in Q\}$

9 **return** *Trained student GNN model* Φ

3.1.1 Private data selection (Lines 4-5)

Corresponding to a query, we first obtain a random subsample of the private data using the Poisson subsampling mechanism with sampling ratio γ defined as follows (see Def. 1). We denote the retrieved subset of private nodes by $\hat{V}^\dagger$. We will see in our privacy analysis that such a data sampling mechanism leads to amplification in privacy.

Definition 1 (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.

Generating Query-specific Private Subgraph Our next step is to generate a private subgraph to train a query-specific teacher GNN. We will then use the trained GNN to generate a pseudo-label for the corresponding query.

To generate a query-specific subgraph, we extract the K- nearest neighbors of the query node from the retrieved 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). We use this selected private data to train a query-specific GNN.

3.1.2 Retrieving noisy pseudo-labels (Lines 6-7)

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. To preserve privacy, we add independent Laplacian noise to each coordinate of the posterior with noise scale $\beta = 1/\lambda$ to obtain the noisy pseudo-label $\tilde{y}_v$ of the query node v

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

We note that the teacher GNN is applied in an inductive setting. That is, to infer labels on a public query node that was not seen during training. Therefore, the GNN model most effective in an inductive setting like GraphSAGE Hamilton et al. (2017) should be used as the teacher GNN.

3.1.3 Student model (transductive) training

Our private models $\Phi^\dagger$ only answer the selected number of queries from the public graph. This is because each time a model $\Phi^\dagger$ is queried, it utilizes the model trained on private data, which will lead to increased privacy costs. The privately (pseudo-)labeled query nodes and the unlabeled public data are used to train a student model Φ in a transductive setting. We then release the public student model Φ and note that the public model Φ is differentially private. The privacy guarantee is a result of the two applied noise mechanisms, which lead to the plausible deniability of the retrieved pseudo-labels and by the post-processing property of differential privacy.

3.1.4 Privacy analysis

We start by describing DP and the related privacy mechanism, followed by a more generalized notion of DP, i.e., RDP. Specifically, we utilize the bound on subsampled RDP and the advanced composition theorem for RDP to derive practical privacy guarantees for our approach. For a complete derivation of our privacy guarantee, see Appendix B.

Differential Privacy Dwork et al. (2006) DP is the most common notion of privacy for algorithms on statistical databases. Informally, DP bounds the change in the output distribution of a mechanism when there is a small change in its input. Concretely, 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 2 (ε-DP). *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 output to an arbitrary query with values in $\mathbb{R}^n$. The mechanism is defined as

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

where Lap is the Laplace distribution and Δ_1 is the ℓ_1 sensitivity of the query f. Here the second parameter, Δ_1/ε is also referred to as the scale parameter of the Laplacian distribution. Our final guarantees are expressed using (ε, δ)-DP which is a relaxation of ε-DP and is defined as follows.

Definition 3 ((ε, δ)-DP). *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:*

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

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

Definition 4 (RDP Mironov (2017)). *A mechanism $\mathcal{M}$ is (α, ε)-RDP with order $\alpha \in (1, \infty)$ if for all neighboring datasets X, X' the following holds*

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

As $\alpha \to \infty$, RDP converges to the pure ε-DP. In its functional form, 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 $\mathcal{M}$. In this work, we will use the following RDP formula corresponding to the Laplacian mechanism

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

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

Lemma 5 (From RDP to (ε, δ)-DP). *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 $\delta \in (0, 1)$.*

For a composed mechanism $\mathcal{M} = (\mathcal{M}_1, \ldots, \mathcal{M}_t)$, we employ Lemma 5 to provide an (ε, δ)-DP guarantee

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

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

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

Privacy Amplification by Subsampling. A commonly used approach in privacy is *subsampling* in which the DP mechanism is applied to the randomly selected sample from the data. Subsampling offers a 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. See Appendix B, Theorem 8 for details.

3.2 Privacy Guarantees of PrivGnn

Theorem 7. *For any $\delta > 0$, Algorithm 1 is (ε, δ)-DP with*

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

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

Proof Sketch. First, using the Laplacian mechanism, the computation of the noisy label for each public query node is $1/\beta$-DP. We perform the transformation of the privacy variables using the RDP formula for the Laplacian mechanism. As 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 $\varepsilon_{\text{LAP} \circ \text{POIS}}(\alpha)$. To obtain a bound corresponding to Q queries, we further use the advanced composition theorem of RDP. The final expression is obtained by appropriate substitution of α and finally invoking Lemma 5 for the composed mechanisms. The detailed proof is provided in Appendix B.2. ☐

Remark. We remark that Equation equation 2 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, following Mironov (2017), we numerically compute the privacy budget with $\delta < 1/|V^{\dagger}|$ while varying $\alpha \in \{2, \ldots, 32\}$ and report the corresponding best ε.

4 Experimental Evaluation

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

RQ 1. *How do the performance and privacy guarantees of PrivGnn compare to that of the 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 the sampling ratios on the privacy-utility trade-off for PrivGnn?*

RQ 5. *How do various design choices of PrivGnn affect its performance?*

Besides, we conduct additional experiments to investigate (i) the robustness of PrivGnn towards two membership inference attacks (c.f. Section C), (ii) the sensitivity of PrivGnn (c.f. Appendix D.2) to the size of private/public datasets and (iii) the effect of pre-training on the accuracy of PrivGnn (c.f. Appendix D.3). We also discuss the time and space requirements of PrivGnn in Appendix G and the behavior of PrivGNN and PATE baselines when the features are more informative than the graph structure in Appendix D.1.

4.1 Experimental Setup

Datasets. We perform our experiments on four representative datasets, namely Amazon Shchur et al. (2018), Amazon2M Chiang et al. (2019), Reddit Hamilton et al. (2017) and Facebook Traud et al. (2012). We also perform additional experiments on ArXiv Hu et al. (2020) and Credit defaulter graph Agarwal et al. (2021) with highly informative features. The detailed description of the datasets is provided in Appendix E and their statistics in Table 9.

Baselines. We compare our PrivGnn approach with three baselines. These are (i) *non-private inductive baseline (B1)* in which we train a GNN model on all private data and test the performance of the model on the public test set, (ii) *non-private transductive baseline (B2)* in which we train the GNN model using the public train split with the ground truth labels. The performance on the public test split estimates the "best possible" performance on the public data (iii) *two variants of PATE baseline* namely PateM and PateG, which use MLP and GNN respectively as their teacher models. A detailed description of the baselines is provided in Appendix F.

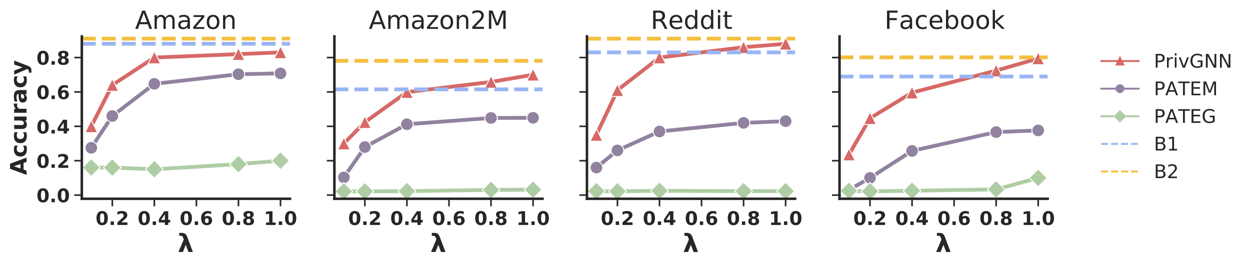
Model and Hyperparameter Setup. For the private and public GNN models, we employ a two-layer GraphSAGE model Hamilton et al. (2017) with a hidden dimension of 64 and ReLU activation function with batch normalization. For the MLP used in PateM, we used three fully connected layers and ReLU activation layers. All methods are compared using the same hyperparameter settings described in Appendix F.1.

Privacy Parameters. 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} for Amazon, Facebook, Credit, and Reddit, and 10^{-5} for ArXiv and Amazon2M. We fix the subsampling ratio γ to 0.3 for all datasets except for Amazon2M, which is set to 0.1. All our experiments were conducted for 10 different instantiations, and we report the mean values across the runs in the main paper. The detailed results with the standard deviation are provided in Appendix A.

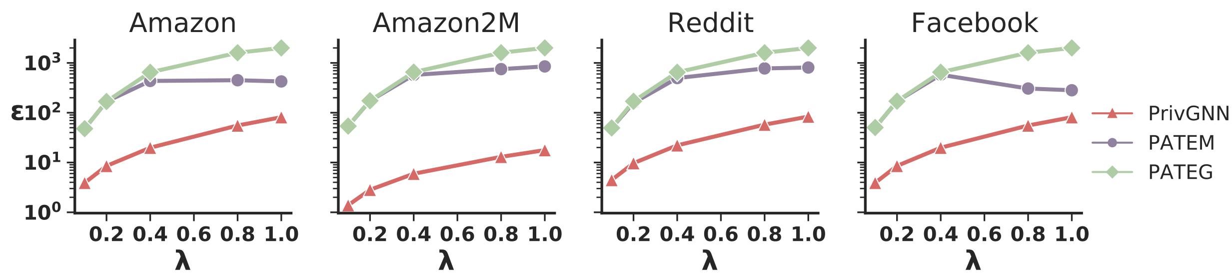
5 Results

We now discuss the results of our five research questions presented in Section 4.

5.1 Privacy-utility Trade-off (RQ 1)



(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 0.1 for Amazon2M and 0.3 for other datasets.

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 (c.f. Figure 2a) and the final incurred privacy budget (c.f. Figure 2b) with respect to $\lambda = 1/\beta$ which is inversely proportional to the injected noise for each query. Here, we set $|Q|= 1000$ for all three private methods.

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 part of the public data, achieves slightly higher results than B1. For Amazon2M, Facebook and Reddit, PrivGnn 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. We observe that for datasets with a high average degree (Amazon2M, Facebook, and Reddit), PrivGnn achieved higher improvements of up to 134% and 2672% in accuracy when compared to the PATE methods. Moreover, as we will discuss next, PrivGnn incurs significantly lower privacy costs as compared to the two private baselines.

Thirdly, we plot the incurred privacy budget ε corresponding to different λ in Figure 2b. PrivGnn achieves a relatively small ε of 2.83 on the Amazon2M dataset, while on the Amazon, Reddit, and Facebook dataset, a ε of 8.53 at $\lambda = 0.2$. The corresponding value of ε on PateM and PateG for the datasets are 167.82 on Amazon, 173.82 on Amazon2M, 157.90 for Reddit, and 170.41 for Facebook. *In fact, PrivGnn achieves the performance of non-private baseline at ε equal to 5.94 and 19.81 for Amazon2M and Reddit datasets respectively.*

It is not surprising that both PateM and PateG have similar ε on all datasets except for Reddit. This is because PATE methods depend on agreement or consensus among the teacher models, which further reduces ε. This indicates that there is no consensus among the teachers.

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 subsampling mechanism, whereas PATE uses the complete private data. **Second**, in the 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. On the other hand, the worse privacy and accuracy of PATEindicates a larger disagreement among 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 specific data portions, hence the resulting disagreement.

5.2 Effect of K on the Accuracy (RQ 2)

To quantify the effect of the size of the random subset 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 to $\{300, 750\}$, and for the Amazon2M, Reddit, and Facebook 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 1, a general observation is that the higher the K, the higher the accuracy. For instance, sampling
$K = 750$ nodes achieve over 50% improvements on the Amazon dataset than using $K = 300$ private nodes for training. On the Amazon2M, Reddit, and Facebook datasets, a value of K above 1000 only increases the accuracy marginally. 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 (Amazon), while for graphs with a higher average degree (Amazon2M, Reddit, and Facebook), a larger K leads to better performance. We further quantify the effect of K in PrivGnn by removing KNN and directly training with the entire private graph in our ablation studies(c.f. Section 5.5).

Table 1: Accuracy for varying K and $|Q|$= 1000.

	Amazon		**Amazon2M**			**Reddit**			**Facebook**		
λ/K	300	750	750	1000	3000	750	1000	3000	750	1000	3000
0.1	0.17	0.40	0.22	0.26	0.30	0.30	0.33	0.35	0.15	0.23	0.24
0.2	0.34	0.64	0.36	0.41	0.42	0.48	0.53	0.61	0.26	0.41	0.45
0.4	0.53	0.80	0.41	0.57	0.60	0.65	0.72	0.80	0.33	0.55	0.60
0.8	0.56	0.82	0.44	0.59	0.66	0.69	0.76	0.86	0.38	0.69	0.72
1.0	0.59	0.83	0.47	0.64	0.70	0.71	0.77	0.88	0.40	0.76	0.80

5.3 Effect of $|Q|$ on the 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 in accuracy when 1000 query nodes are pseudo-labeled and when only 500 queries are employed across all datasets at different noise levels except on Amazon2M.

Further looking at detailed results in Table 2, for the different ranges of λ on the Amazon, Reddit, and Facebook datasets, we observe up to 46% decrease in privacy budget of PrivGnn for answering 500 queries over answering 1000 queries. On the Amazon2M dataset, we observe up to 45% decrease in privacy budget. This implies that smaller $|Q|$ is desirable, which PrivGnn offers.

In Table 2, we also compare PrivGnn with PATE variants with respect to the incurred privacy budget ε for answering the different number of queries. Our method offers a significantly better privacy guarantee

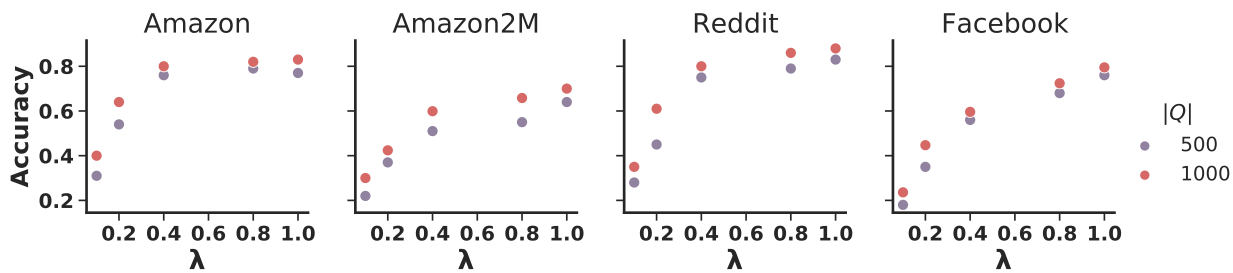


Figure 3: Accuracy for the different number of queries answered by the teacher model of PrivGnn

(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 budget for PrivGnn. In particular, the privacy budget (see Table 2), ε, of PrivGnn is reduced by 50% on the Amazon, Reddit, and Facebook datasets and by 40% on the Amazon2M dataset with only a slight reduction in accuracy.

We remark that we also experimented with more intelligent strategies for query selection which did not show any performance gains. In particular, we employed structure-based approaches such as clustering and ranking based on centrality measures such as PageRank and degree centrality to choose the most representative query nodes but observed no performance gains over random selection.

Table 2: Privacy budget (ε) (lower the better) for varying number of queries $|Q|$ answered by the teacher.

	$\|Q\|$	$\lambda \rightarrow$	0.1	0.2	0.4	0.8	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
Amazon2M	500	PrivGnn	0.97	1.96	4.03	8.73	11.17
		PateM	33.82	93.82	294.89	379.92	416.65
		PateG	33.82	93.82	333.82	801.73	1000.73
	1000	PrivGnn	1.39	2.83	5.94	12.98	17.73
		PateM	53.82	173.82	573.80	751.09	850.63
		PateG	53.82	173.82	653.82	1601.73	2001.73
Reddit	500	PrivGnn	2.67	5.69	13.15	31.10	44.07
		PateM	29.43	83.73	243.45	392.23	415.35
		PateG	29.43	89.43	329.43	801.17	1001.17
	1000	PrivGnn	3.90	8.53	19.81	55.30	81.23
		PateM	49.43	157.90	496.61	775.95	808.43
		PateG	49.43	169.43	649.43	1601.17	2001.17
Facebook	500	PrivGnn	2.67	5.96	13.15	31.10	44.07
		PateM	30.41	90.41	293.44	151.02	136.41
		PateG	30.41	90.41	330.41	801.30	1000.30
	1000	PrivGnn	3.90	8.53	19.81	55.30	81.23
		PateM	50.41	170.41	579.15	306.12	282.63
		PateG	50.41	170.41	650.41	1601.30	2001.30

5.4 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 vary γ from 0.1 to 0.3. This allows us to further benefit from the privacy amplification by subsampling as explained in Section B.1. Since γ affects the amount of the available data used in training the teacher model, we only perform this experiment on representative datasets (Amazon2M and Reddit). As shown in Table 3, the accuracy of using the sampling ratio $\gamma = 0.3$ on the Amazon2M dataset is slightly higher with only 6% decrease when compared to that of $\gamma = 0.1$ for $\lambda = 0.4$. On the Reddit dataset, we observe a 35% decrease in the accuracy when $\gamma = 0.1$ at the same λ. Nonetheless, we observe a 315% reduction in the privacy budget when $\gamma = 0.1$. This implies that smaller γ offers better ε on both datasets at a relatively lower drop in accuracy. We conclude that PrivGnn provides an overall good privacy-utility trade-off with respect to γ. In other words, for larger datasets, a significant reduction in privacy costs can be achieved (when using smaller γ) with a relatively lower degradation in accuracy.

Table 3: Performance and the respective privacy budget for different sampling ratios γ with $|Q|= 1000$ for Amazon2M and Reddit. The non-private baseline B1 achieves an accuracy of 0.62 and 0.78 for Amazon2M and Reddit, respectively. PrivGnn already achieves comparable or better accuracy with ε of 5.94 and 19.81, respectively (marked in bold).

	$\gamma = 0.1$				$\gamma = 0.3$			
	Reddit		Amazon2M		Reddit		Amazon2M	
λ	ε	Accuracy	ε	Accuracy	ε	Accuracy	ε	Accuracy
0.1	1.20	0.19	1.39	0.30	3.90	0.35	4.45	0.35
0.2	2.47	0.37	2.83	0.42	8.53	0.61	9.69	0.51
0.4	5.20	0.52	**5.94**	**0.60**	**19.81**	**0.80**	22.11	0.64
0.8	11.82	0.63	12.98	0.66	55.30	0.86	57.60	0.69
1.0	15.44	0.65	17.17	0.70	81.23	0.88	83.53	0.77

Table 4: Ablation studies. Here, we present the accuracy of different methods with varying λ and $|Q|= 1000$. We show for representative datasets Amazon2M and Reddit with $\gamma = 0.1$ and 0.3 respectively.

	$\lambda \rightarrow$	0.1	0.2	0.4	0.8	1.0
Reddit	PrivGnn	0.35	0.61	0.80	0.86	0.88
	KNNRem	0.34	0.58	0.75	0.79	0.81
	SingleGNN	0.28	0.54	0.72	0.77	0.79
Amazon2M	PrivGnn	0.30	0.42	0.60	0.66	0.70
	KNNRem	0.28	0.39	0.49	0.56	0.59
	SingleGNN	0.25	0.32	0.43	0.54	0.56

5.5 Ablation Studies (RQ 5)

To answer RQ 5, we conducted ablation studies by varying different components of our proposed model. First, we removed the KNN component (Line 4 in Algorithm 1) and constructed the induced subgraph on the entire subsampled dataset for each query. We refer to this as KNNRem. Secondly, besides removing the KNN component, we only sample once from the private data instead of sampling for each query. Hence, we only train a single teacher model using the graph induced by subsampling. We refer to this as SingleGNN. We present our results in Table 4. We remark that, theoretically, the same privacy guarantees as our method holds for these two ablations. In practice, on the other hand, PrivGnn would offer better privacy because only a smaller portion of the private sample is used for teacher training compared to the ablations in which the KNN component is removed.

We make the following observations from our results. First, we see up to 18% and 8% drop in accuracy with KNNRem on the Amazon2M and Reddit datasets, respectively. This indicates the importance of our KNN approach since only the most informative node for the query node is used in training and predicting the pseudo-label.

Secondly, considering the performance of SingleGNN, the advantage of the multiple subsampling and the KNN components is prominent in both datasets. At a higher noise level (when $\lambda = 0.2$), we observe a performance drop of 23% and 12% on the Amazon2M and Reddit dataset, respectively, when SingleGNN is used as compared to PrivGnn.

Additional Experiments. We performed several additional experiments. This includes two membership inference attacks to evaluate the privacy leakage of the released model. We observe that PrivGnn reduces the attacks to a random guess (Appendix C). We analyze the impact of informative node features on the performance of PrivGnn in Appendix D.1. The result indicates that PrivGnn in the presence of highly informative node features performs well than both PateM and PateG baselines. To better understand the influence of the private-public graph size on the performance of PrivGnn, we perform a sensitivity analysis by flipping the graphs. The result shows that PrivGnn is not sensitive to the size of the private-public graphs (Appendix D.2). We also perform an experiment that utilizes a pre-training technique to improve the performance of PrivGnn. The result shows a slight improvement in the performance of PrivGnn (Appendix D.3). Finally, we analyze the runtime and space requirements of PrivGnn in Appendix G and highlight the limitations and future works (Appendix H).

6 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 the knowledge of the *teacher* models trained on private data to the *student* model. Privacy is intuitively and theoretically guaranteed due to private data subsampling as well as the 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. Results from the ablation studies further support our design choices. We present the privacy analysis of our approach by leveraging the Rènyi differential privacy framework. Our experiment results on six real-world datasets show the effectiveness of our approach in obtaining good accuracy under practical privacy guarantees.

References

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

Chirag Agarwal, Himabindu Lakkaraju, and Marinka Zitnik. Towards a unified framework for fair and stable graph representation learning. In *Uncertainty in Artificial Intelligence*, pp. 2114–2124. PMLR, 2021.

David Ahmedt-Aristizabal, Mohammad Ali Armin, Simon Denman, Clinton Fookes, and Lars Petersson. Graph-based deep learning for medical diagnosis and analysis: Past, present and future. *Sensors (Basel, Switzerland)*, 21(14), 2021.

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

Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 257–266, 2019.

Nick Craswell, Daniel Campos, Bhaskar Mitra, Emine Yilmaz, and Bodo Billerbeck. Orcas: 18 million clicked query-document pairs for analyzing search. *arXiv preprint arXiv:2006.05324*, 2020. URL `https://microsoft.github.io/msmarco/ORCAS.html`.

Vasisht Duddu, Antoine Boutet, and Virat Shejwalkar. Quantifying privacy leakage in graph embedding. *arXiv preprint arXiv:2010.00906*, 2020.

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

Dumitru Erhan, Aaron Courville, Yoshua Bengio, and Pascal Vincent. Why does unsupervised pre-training help deep learning? In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 201–208. JMLR Workshop and Conference Proceedings, 2010.

Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. Graph neural networks for social recommendation. In *The World Wide Web Conference*, pp. 417–426, 2019.

Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric. *arXiv preprint arXiv:1903.02428*, 2019.

GroupLensResearch. Movielens, Dec 2021. URL `https://grouplens.org/datasets/movielens/`.

William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NIPS*, 2017.

Xinlei He, Rui Wen, Yixin Wu, Michael Backes, Yun Shen, and Yang Zhang. Node-level membership inference attacks against graph neural networks. *arXiv preprint arXiv:2102.05429*, 2021.

Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. *arXiv preprint arXiv:1905.12265*, 2019.

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

Timour Igamberdiev and Ivan Habernal. Privacy-preserving graph convolutional networks for text classification. *arXiv preprint arXiv:2102.09604*, 2021.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

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

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 43–52, 2015.

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

Garrett Morrow, Briony Swire-Thompson, Jessica Montgomery Polny, Matthew Kopec, and John P Wihbey. The emerging science of content labeling: Contextualizing social media content moderation. *Journal of the Association for Information Science and Technology*, 73(10):1365–1386, 2022.

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

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

Nicolas Papernot, Shuang Song, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Úlfar Erlingsson. Scalable private learning with pate. *arXiv preprint arXiv:1802.08908*, 2018.

Sina Sajadmanesh and Daniel Gatica-Perez. Locally private graph neural networks. *arXiv preprint arXiv:2006.05535*, 2020.

Chuanqiang Shan, Huiyun Jiao, and Jie Fu. Towards representation identical privacy-preserving graph neural network via split learning. *arXiv preprint arXiv:2107.05917*, 2021.

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

Amanda L Traud, Peter J Mucha, and Mason A Porter. Social structure of facebook networks. *Physica A: Statistical Mechanics and its Applications*, 391(16):4165–4180, 2012.

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

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

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

Yu Zheng, Chen Gao, Liang Chen, Depeng Jin, and Yong Li. Dgcn: Diversified recommendation with graph convolutional networks. In *Proceedings of the Web Conference 2021*, pp. 401–412, 2021.

Jun Zhou, Chaochao Chen, Longfei Zheng, Huiwen Wu, Jia Wu, Xiaolin Zheng, Bingzhe Wu, Ziqi Liu, and Li Wang. Vertically federated graph neural network for privacy-preserving node classification. *arXiv preprint arXiv:2005.11903*, 2020.

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

Appendix

Organization. The Appendix is organized as follows. We begin by providing the detailed accuracy scores of different methods in Section A. Missing theoretical details and proof are provided in Section B. We launch two black-box membership inference attacks in Section C followed by the analysis of the influence of highly informative features (Section D.1), the effect of the size of the public graph (Section D.2) and the use of pre-training to improve performance (Section D.3). Section E provides a detailed description of datasets used followed by details on baselines, hyperparameters, and training in Section F. The runtime and space requirement of PrivGnn is discussed in Section G. Lastly, the limitations of the work are highlighted in Section H.

A Detailed Results

In Tables 5 and 6, we present the mean accuracy of the different methods as plotted in Figure 2 and 3 respectively.

B Detailed Proof

Here, we provide missing theoretical details and the full proof of Theorem 7. In the following, we motivate the use of private data subsampling to achieve a reduction in the privacy budget. We use Theorem 8 in our analysis to bound the privacy budget (in the RDP framework) of our Poisson subsampled Laplacian mechanism.

Table 5: Performance of each method on all datasets. Note that the non-private baselines B1 and B2 are not changing with different λ.

	Method	$\lambda = 0.1$	$\lambda = 0.2$	$\lambda = 0.4$	$\lambda = 0.8$	$\lambda = 1$
Amazon	PrivGnn	0.40 ± 0.08	0.64 ± 0.07	0.80 ± 0.13	0.82 ± 0.17	0.83 ± 0.10
	PateM	0.28 ± 0.04	0.46 ± 0.03	0.65 ± 0.04	0.70 ± 0.03	0.71 ± 0.03
	PateG	0.16 ± 0.05	0.16 ± 0.07	0.15 ± 0.08	0.18 ± 0.11	0.20 ± 0.09
	B1	0.88 ± 0.02	0.88 ± 0.02	0.88 ± 0.02	0.88 ± 0.02	0.88 ± 0.02
	B2	0.91 ± 0.02	0.91 ± 0.02	0.91 ± 0.02	0.91 ± 0.02	0.91 ± 0.02
Amazon2M	PrivGnn	0.30 ± 0.05	0.42 ± 0.10	0.60 ± 0.18	0.66 ± 0.08	0.70 ± 0.19
	PateM	0.10 ± 0.04	0.28 ± 0.02	0.41 ± 0.02	0.44 ± 0.01	0.47 ± 0.01
	PateG	0.02 ± 0.01	0.02 ± 0.01	0.03 ± 0.03	0.04 ± 0.02	0.04 ± 0.03
	B1	0.62 ± 0.01	0.62 ± 0.01	0.62 ± 0.01	0.62 ± 0.01	0.62 ± 0.01
	B2	0.78 ± 0.01	0.78 ± 0.01	0.78 ± 0.01	0.78 ± 0.01	0.78 ± 0.01
Reddit	PrivGnn	0.35 ± 0.09	0.61 ± 0.18	0.80 ± 0.10	0.86 ± 0.13	0.88 ± 0.07
	PateM	0.16 ± 0.01	0.26 ± 0.01	0.37 ± 0.01	0.42 ± 0.03	0.43 ± 0.05
	PateG	0.03 ± 0.03	0.03 ± 0.03	0.06 ± 0.02	0.04 ± 0.05	0.08 ± 0.04
	B1	0.83 ± 0.01	0.83 ± 0.01	0.83 ± 0.01	0.83 ± 0.01	0.83 ± 0.01
	B2	0.92 ± 0.01	0.92 ± 0.01	0.92 ± 0.01	0.92 ± 0.01	0.92 ± 0.01
Facebook	PrivGnn	0.24 ± 0.08	0.45 ± 0.07	0.60 ± 0.15	0.72 ± 0.11	0.80 ± 0.11
	PateM	0.03 ± 0.04	0.10 ± 0.05	0.25 ± 0.08	0.37 ± 0.03	0.38 ± 0.03
	PateG	0.02 ± 0.03	0.02 ± 0.04	0.03 ± 0.04	0.04 ± 0.03	0.10 ± 0.03
	B1	0.69 ± 0.01	0.69 ± 0.01	0.69 ± 0.01	0.69 ± 0.01	0.69 ± 0.01
	B2	0.80 ± 0.01	0.80 ± 0.01	0.80 ± 0.01	0.80 ± 0.01	0.80 ± 0.01

Table 6: Mean accuracy of varying the number of queries $|Q|$.

	Amazon		**Amazon2M**		**Reddit**		**Facebook**	
$\lambda/_{\lvert Q\rvert}$	500	1000	500	1000	500	1000	500	1000
0.1	0.31	0.40	0.22	0.30	0.28	0.35	0.18	0.24
0.2	0.54	0.64	0.37	0.42	0.45	0.61	0.35	0.45
0.4	0.76	0.80	0.51	0.60	0.75	0.80	0.56	0.60
0.8	0.79	0.82	0.55	0.66	0.79	0.86	0.68	0.72
1.0	0.77	0.83	0.64	0.70	0.83	0.88	0.76	0.80

B.1 Privacy Amplification by Subsampling

A commonly used approach in privacy is *subsampling* in which the DP mechanism is applied to the randomly selected sample from the data. Subsampling offers a 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 privacy by subsampling is that the privacy guarantee of the DP mechanism is tighter by applying it to a small random subsample of records from a given dataset. This is also referred to as the *subsampling lemma* in the literature Balle et al. (2018). Under some restrictions on α, we can represent the combination of the subsampling lemma, and the tight advanced composition of RDP Wang et al. (2019); Zhu & Wang (2019) as: $\varepsilon_{\mathcal{M}oSample_\gamma}(\alpha) \leq O(\gamma^2 \varepsilon_{\mathcal{M}}(\alpha))$. In this paper, we apply the Poisson subsampled RDP-amplification bound from Zhu & Wang (2019).

Theorem 8 (General upper bound Zhu & Wang (2019)). *Let $\mathcal{M}$ be any mechanism that obeys $(\alpha, \varepsilon(\alpha))$-RDP. Let γ be the subsampling probability, then for integer $\alpha \geq 2$, the privacy budget is*

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

B.2 Proof of our Privacy Guarantee

Proof of Theorem 7. Our algorithm is composed of (i) a sampling mechanism in which a small sample of private data is used to train the private GNN model and (ii) a 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 Dwork et al. (2006) over N queries will result in a crude bound over the DP guarantee of our approach, namely N/β. To obtain a tighter bound that takes the effect of the private data subsampling into account, we perform the transformation of the privacy variables using the RDP formula for the Laplacian mechanism for $\alpha > 1$

$$\varepsilon_{\text{LAP}}(\alpha) = \frac{1}{\alpha-1} \log \left(\left(\frac{\alpha}{2\alpha-1} \right) e^{\frac{\alpha-1}{\beta}} + \left(\frac{\alpha-1}{2\alpha-1} \right) e^{\frac{-\alpha}{\beta}} \right). \tag{3}$$

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 an upper bound of $\varepsilon_{\text{LAP}o\text{POIS}}(\alpha)$. To simplify the expression, we set $\alpha = 2$ in the corresponding bound and obtain

$$\varepsilon_{\text{LAP}o\text{POIS}}(2) \leq \log \left(1 - \gamma^2 + \gamma^2 e^{\varepsilon_{\text{LAP}}(2)} \right). \tag{4}$$

Substituting $\varepsilon_{\text{LAP}}(2)$ in equation 4, we obtain

$$\varepsilon_{\text{LAP}o\text{POIS}}(2) = \log \left(1 - \gamma^2 + \gamma^2 \left(\frac{2}{3} \cdot e^{1/\beta} + \frac{1}{3} \cdot e^{-2/\beta} \right) \right). \tag{5}$$

Now applying the advanced composition for RDP (Lemma 6) for $|Q|$ queries, we get the upper bound on the 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. equation 1 to obtain the (ε, δ)-DP guarantee. Specifically for any $\delta > 0$ we obtain

$$\varepsilon(\delta) = \min_{\alpha > 1} \frac{\log(1/\delta)}{\alpha - 1} + \varepsilon_{\textsc{PrivGnn}}(\alpha - 1). \tag{6}$$

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. □

C PrivGnn and Membership Inference Attacks

In addition to the theoretical guarantee of differential privacy, here, we empirically test the robustness of PrivGnn against membership inference (MI) attacks. In particular, we apply the attack of Olatunji et al. (2021) in two different settings. In the first setting (Attack-1), the goal of the adversary is to distinguish private nodes from public nodes. In contrast, in the second setting (Attack-2), the goal of the adversary is to distinguish the public nodes which were privately labeled from the unlabeled public nodes.

Overall our two MI attacks consist of three phases: the training of a shadow model, the training of an attack model, and the membership inference phase, which involves using the trained attack model to infer the membership status of any given query from the target (attacked) model. In summary, we first generate posteriors for the shadow dataset using the target. This generated data is then used to train the attack model, which is a binary classifier that distinguishes between the training and non-training nodes. For more details about the MI attack, we refer the reader to Olatunji et al. (2021) and their publicly available implementation.

In the following, we describe our two attack variants and the corresponding results: Attack-1 aims at determining the private nodes (the original training data), and Attack-2 aims to determine the pseudo-labeled public nodes (the noisy training set for the public student model). For all the attacks, we select the student model with the largest $\lambda = 1$, which corresponds to a lower privacy guarantee and the highest information leakage based on the achieved ϵ.

C.0.1 Identifying Private Nodes via MI (Attack-1)

(a) MI Attack-1.

(b) MI Attack-2.

Figure 4: Performance of MI Attack-1 and Attack-2 on PrivGnn. B1 refers to the MI attack on the non-private GNN model.

Here, we assume that the attacker does not know the training paradigm of the released student model but is aware of the architecture of the released model (target model). She wants to infer the private nodes used for training from the dataset available to her (for example, by some means, she was able to get hold of the Facebook social network). After training her attack model, the attacker queries the target model with nodes of interest and inputs her posteriors into her trained attack model to infer membership.

We compare the performance of the attack on a non-private model (B1) and PrivGnn. As shown in Figure 4a, in the non-private model, the attacker can accurately identify private nodes which were used in training the target GNN model with high AUROC (> 0.75) on all datasets. However, the performance of that attack on PrivGnn is reduced to a random guess (≤ 0.54).

C.0.2 Identifying Private Nodes by Proxy via MI (Attack-2)

For this attack, we assume that the attacker knows that the released student model (target model) is trained using the knowledge distillation paradigm. Therefore, for such a strong attacker, her goal is to determine the public nodes that were privately labeled and used by PrivGnn in training the target model from the other public nodes. The rationale is that if an attacker can confidently determine the public nodes that are privately (noisily) labeled, then it might reveal some information about the private dataset. For instance, consider an attacker that is aware that PrivGnn utilizes *KNN* in selecting the private nodes used in labeling a public node. If she determines such a public node that was privately labeled, she might infer private information of up to K private nodes assuming that she has access to the entire graph but does not know which is private or public. Therefore, the private nodes are said to be identified by "proxy".

As shown in Figure 4b, our PrivGnn method reduces the MI attack to a random guess achieving an AUROC of 0.40, 0.51, and 0.54 on the Amazon, ArXiv, and Reddit datasets, respectively, as compared to the non-private MI attack (corresponding to the MI attack on Baseline B1) with AUROC of 0.82, 0.75, and 0.87. This is because of the two noise mechanisms that PrivGnn adopts. Specifically, the random subsampling limits the private data used for the KNN procedure. Moreover, the knowledge transferred to the student model is limited due to noisy pseudo-labels.

D Additional Experiments

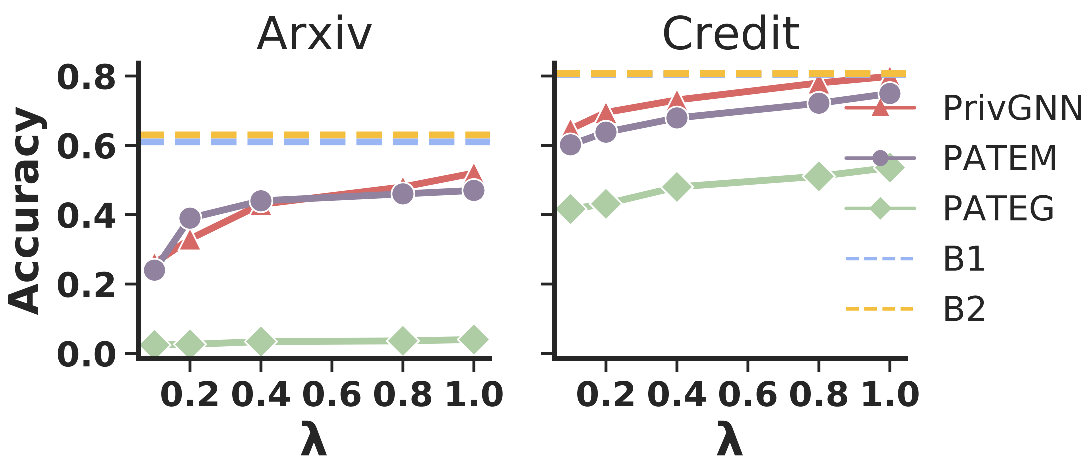
D.1 Impact of Highly Informative Node Features

Here, we use two datasets, ArxiV and Credit, whose features are already highly informative for the task. To verify this, we first train the datasets on a three-layer MLP using only the features, and we obtained relatively high accuracy as compared to B2, which in addition uses the graph structure, as shown in Table 7.

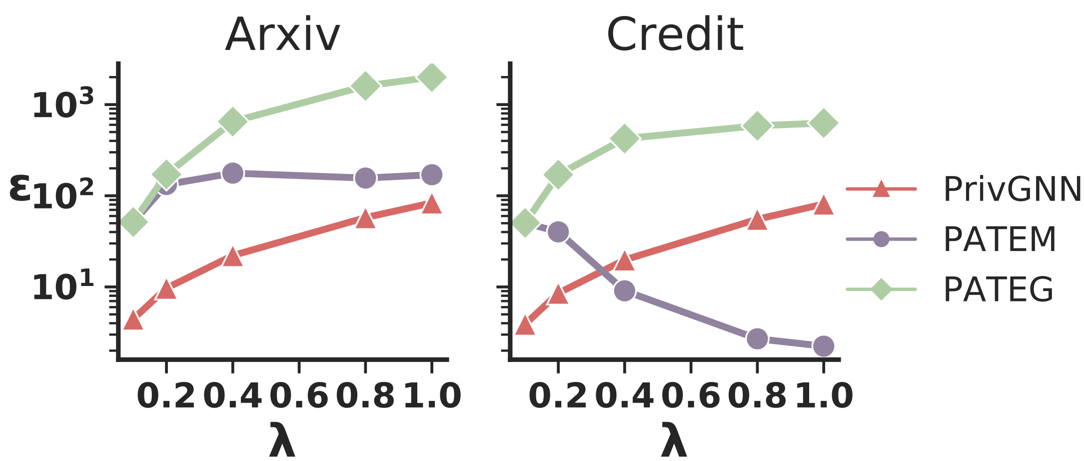
Table 7: MLP accuracy of Credit and ArXiv. B2 utilizes the GNN model

	Credit	**ArXiv**
MLP	0.78	0.55
B2	0.81	0.63

As shown in Figure 5, on both datasets, the performance of PateM is on par with PrivGnn. On the ArXiv dataset, we achieved a 15% decrease over PateM (when $\lambda = 0.2$). Note that the teacher models of PateM are MLPs and do not utilize graph structure. The observed result is probably because the average node degree of ArXiv is very small (≤ 2). Thus, GNNs, which use the aggregation of the neighbors of a node to make predictions, might not benefit from such low-degree graphs. However, against PateG, we achieved a 1169% improvement in performance which uses graph structure. Recall that PateG teacher models are queried on graph data when they are trained like MLP (due to the low connectivity among training nodes), which explains its worse performance than PateM. Moreover, on the Credit dataset, the privacy budget is very low, indicating high agreement among the teacher models in PateM. We also observe that PateG performs better on this dataset which further supports our argument that the feature is highly informative as compared to the performance on other datasets in Figure 2. We will like to emphasize that the drop in the privacy budget of the Credit dataset as less noise is added is not surprising. This is due to the data-dependent analysis that PateM utilizes (see Papernot et al. (2016)).



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



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

Figure 5: Privacy-utility analysis of the impact of highly informative node features. Here, $|Q|$ is set to 1000. For PrivGnn, γ is set to 0.3.

D.2 Effect of the Size of the Public Dataset

To investigate the effect of the size of the public dataset on the performance of PrivGnn, we flip the public and private portions corresponding to different datasets. For instance, instead of having 2500 nodes in the private dataset as shown for Amazon in Table 9, we have 6000 nodes for the private dataset and 2500 for the public dataset. As shown in Figure 6, there are no significant differences in the results of our model when there is a larger public dataset available (c.f Figure 2a). For a fair comparison, we run all baselines, including PateM and PateG, on the flipped data. PrivGnn outperforms all baselines on the flipped datasets. As expected, the result of the non-private baselines (B1 and B2) are interchanged. We observe that our model is not sensitive to the size of the private or public datasets. Therefore, our method is suitable in settings where the available private or public data is large and in settings with a limited private or public dataset.

D.3 Improving Performance via Pre-training

As shown in Section D.2, PrivGnn is not sensitive to the size of the public or private data. However, in settings where a larger public dataset is available, we ask whether the student model can benefit from

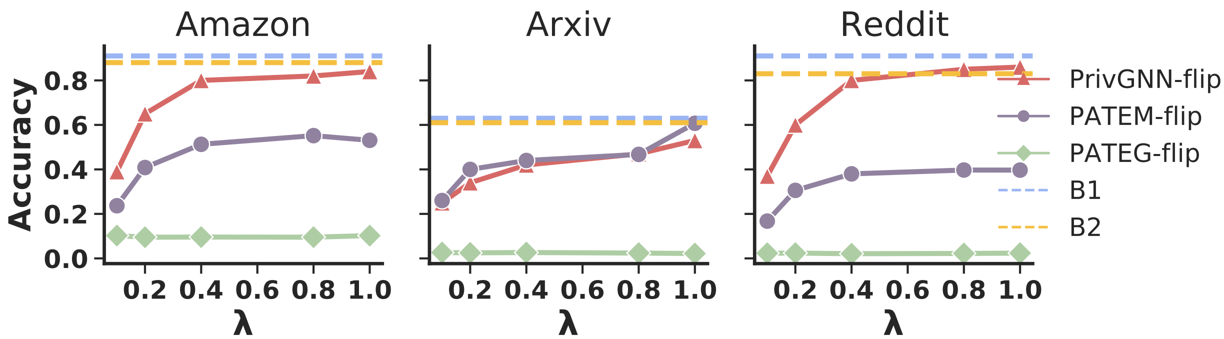


Figure 6: Performance of different models after flipping the public and private datasets.

such large "unlabeled" data. Therefore, we propose to leverage the abundance of the unlabeled dataset by pre-training the student GNN model using unsupervised objectives on the public dataset. The unsupervised objective is based on the graph reconstruction objective Hamilton et al. (2017) which assigns similar embedding to connected nodes. After pre-training, we initialize the student model with the pre-trained model and fine-tune the student model using the privately labeled nodes. We emphasize that the pre-training does not constitute any privacy risks since the data is public. One major advantage of unsupervised pre-training is that it acts as a regularizer and provides better generalization than randomly initializing the weights Erhan et al. (2010). We conducted a preliminary experiment on the Amazon2M and ArXiv datasets. As shown in Table 8, on a smaller number of queries ($|Q|$=300), we obtain an increment of up to 35% on Amazon2M and 17% on ArXiv over the randomly initialized PrivGnn. We also observe up to 4% and 9% increase on higher number of queries ($|Q|$=1000) on Amazon2M and ArXiv, respectively.

As pointed out by Hu et al. (2019), pre-training GNNs is still relatively difficult and may require domain expertise to carefully select examples that are well correlated to the downstream task to avoid "negative" transfer (hurt generalization). Therefore, we leave an in-depth study of how pre-training can help the better performance of PrivGnn on more datasets as future work.

E Datasets

In the following, we describe the datasets used in this work.

Amazon. We use the Amazon Computers dataset Shchur et al. (2018) which is a subset of the Amazon co-purchase graph McAuley et al. (2015). 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 train and test sets. The task for this dataset is to assign products to their respective product category.

Amazon2M. The Amazon2M dataset Chiang et al. (2019) is the largest Amazon co-purchasing network where each node is a product, and the features are extracted via bag-of-words on the product description. Edges represent whether two products are purchased together. It consists of over 2 million nodes. Hence, Amazon2M. We created a private graph on 1 million nodes and a public graph on the remaining nodes.

Reddit. The Reddit dataset Hamilton et al. (2017) 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 sampled 300 nodes from each class for the private graph and selected 15000 nodes each for the public train and public test nodes, respectively.

Facebook. The Facebook dataset Traud et al. (2012) is an anonymized Facebook social network of users from 100 American institutions. We utilized the social network among UIUC students where the task is to

Table 8: Performance of pre-training the student model on Amazon2M and ArXiv dataset. Δ indicates the % difference between PrivGnn without pre-training and the pre-trained PrivGnn (Pre-PrivGnn).

	$\lvert Q \rvert$	$\lambda \rightarrow$	0.1	0.2	0.4	0.8	1.0
Amazon2M	1000	PrivGnn	0.30	0.42	0.60	0.66	0.70
		Pre-PrivGnn	0.31	0.44	0.61	0.68	0.72
		Δ	3%	4%	2%	3%	3%
	500	PrivGnn	0.22	0.37	0.51	0.55	0.64
		Pre-PrivGnn	0.24	0.40	0.53	0.58	0.67
		Δ	9%	8%	4%	5%	5%
	300	PrivGnn	0.12	0.19	0.28	0.34	0.36
		Pre-PrivGnn	0.17	0.22	0.30	0.36	0.43
		Δ	35%	15%	7%	6%	18%
ArXiv	1000	PrivGnn	0.26	0.33	0.43	0.48	0.52
		Pre-PrivGnn	0.26	0.33	0.43	0.48	0.52
		Δ	7%	9%	9%	6%	4%
	500	PrivGnn	0.26	0.29	0.42	0.47	0.50
		Pre-PrivGnn	0.25	0.31	0.45	0.49	0.51
		Δ	4%	7%	7%	4%	2%
	300	PrivGnn	0.11	0.21	0.32	0.37	0.41
		Pre-PrivGnn	0.13	0.25	0.36	0.40	0.46
		Δ	17%	17%	12%	8%	11%

predict their class year. The nodes represent Facebook users(students), and the edges indicate friendship. The private graph is constructed over half of the nodes and the public graph on the other half.

ArXiv. The ArXiv dataset Hu et al. (2020) 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 embedding 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.

Credit. The Credit defaulter graph Agarwal et al. (2021) is a financial network where nodes are individuals and edges exist between individuals based on the similarity of their spending and payment pattern. The task is to determine whether an individual will default on their credit card payment. The private and public graphs consist of 10000 and 15000 nodes, respectively.

We remark that irrespective of the size of the public data, we only pseudo-label a very small number of query nodes (in our experiments, the maximum query size was 1000) of the public graph to train the student model.

F 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 Amazon, Amazon2M, Reddit, and Credit, 60% of ArXiv, and 80% of the Facebook 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, Amazon2M, Reddit, and Credit, and 40% of ArXiv and 20% of the Facebook dataset).

Table 9: 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 Amazon, Amazon2M, Reddit and Credit, 40% of ArXiv, and 20% of the Facebook dataset as the test set from the public graph.

			Private			**Public**		
	#class	$\|X\|$	$\|V^\dagger\|$	$\|E^\dagger\|$	deg	$\|V\|$	$\|E\|$	deg
Amazon	10	767	2500	11595	4.64	6000	37171	6.20
Amazon2M	47	100	1**M**	12.7**M**	12.73	1.4**M**	21.8**M**	15.08
Reddit	41	602	12300	266148	21.64	30000	876846	29.23
Facebook	34	501	13401	270992	20.22	13402	266141	24.82
ArXiv	40	128	90941	187419	2.06	78402	107900	1.38
Credit	2	13	10818	185916	17.19	15000	361093	24.07

PATE. We adopt the PATE framework Papernot et al. (2016) 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 induced graph) separately. We 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 the Amazon, Facebook, Credit, and Reddit datasets and $n > 50$ for the ArXiv lead to extremely small data in each partition and low accuracy. We also set $n = 50$ for Amazon2M.

Why are Sajadmanesh & Gatica-Perez (2020) and Zhou et al. (2020) not compared? We again emphasize that Sajadmanesh & Gatica-Perez (2020) and Zhou et al. (2020) follow a completely different setting from ours in which the multiple parties are either holding a part of the data or the central aggregator is distrusted. Their goal is to train a GNN model wherein each data holder wants to preserve the privacy of his/her data from other data holders or the distrusted central aggregator. We refer the interested reader to Section 2 for more details about these methods and their differences from our work.

F.1 Model and Hyperparameter Setup

For each of the private (teacher) ($\Phi^\dagger$) and public (student) (Φ) GNN models, we used a two-layer GraphSAGE model Hamilton et al. (2017) with hidden dimension of 64 and ReLU activation function. We applied batch normalization to 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 200 epochs using the negative log-likelihood loss function for node classification with the Adam optimizer. All our experiments were conducted for 10 different instantiations using PyTorch Geometric library Fey & Lenssen (2019) and Python3 on 11GB GeForce GTX 1080 Ti GPU.

G The Runtime and Space Requirement of PrivGnn

We remark that the runtime and space requirements of our proposed PrivGnn are on par or even less than that of other private models like PATE. Importantly, since the user is only exposed to the public model, we point out that the time and space requirements of the released model are the same as that of the corresponding non-private GNN models.

For the training time, we train Q models corresponding to Q queries. As shown in Figure 3, we can decrease Q to 500 (equivalent to training 500 teacher models). The runtime of PrivGnn can be further reduced by parallel training of models. Note that the original PATE trains a sufficiently large number of teacher models to achieve a reasonable privacy-accuracy trade-off. For instance, on the Glyph dataset (See Table 1 of

Papernot et al. (2018)), PATE trained 5000 teacher models. With our requirement of a much smaller number of teacher models, PrivGnn is faster or better than PATE in terms of time and space requirements.

The space requirement of PrivGnn is the same as the space requirements of one GNN model because we need not save the trained teacher model after querying them once with the query nodes. Therefore, they can be discarded after training which leads to no significant space overhead. For PATE, on the other hand, all teacher models are required to answer each query leading to its higher space requirements.

H Limitations and Future Works

Query reduction. As observed in the experiments, fewer queries significantly reduced the overall privacy budget ϵ. However, this has a relative effect on the accuracy of the released model. Hence, a more sophisticated active learning approach or unsupervised methods for query reduction that can achieve a high accuracy are worth exploring.

Analysing PrivGnn's behavior. The use of explanation methods to analyze the properties of the learned representation of the PrivGnn is desirable. This analysis will help understand the internal workings of private models and compare the representations of private and non-private models.