Differentially Private Deep Learning

By

Qiuchen Zhang
Doctor of Philosophy

Computer Science and Informatics

Li Xiong, Ph.D.
Advisor

Vaidy Sunderam, Ph.D.
Committee Member

Joyce C. Ho, Ph.D.
Committee Member

Xiaoqian Jiang, Ph.D.
Committee Member

Accepted:

Lisa A. Tedesco, Ph.D.
Dean of the James T. Laney School of Graduate Studies

Date

Differentially Private Deep Learning

By

Qiuchen Zhang
B.S., Xi'an Jiaotong University, 2014
M.S., The Ohio State University, 2015

Advisor: Li Xiong, Ph.D.

An abstract of
A dissertation submitted to the Faculty of the
James T. Laney School of Graduate Studies of Emory University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in Computer Science and Informatics
2021

Abstract

Differentially Private Deep Learning
By Qiuchen Zhang

Deep learning models have achieved great success in many real-world tasks, such as image recognition and machine translation. A large amount of data is needed to train a model, and in many cases, the training data are private. Publishing or sharing a deep learning model trained on private datasets could pose privacy concerns. Differential privacy (DP) has been widely accepted as a strong and provable privacy framework for statistical data analysis. Recent works developed deep learning models with DP, which requires that the statistical model (parameters) learned from a set of data is indistinguishable regardless of the presence or absence of any record in the dataset. Most works on deep learning with DP focus on improving model accuracy given a privacy requirement or enhancing the privacy and utility trade-off. However, existing works become less effective when the model goes deeper, or the privacy requirement is tighter. Meanwhile, there is still little demonstration of how effective DP is in protecting against the existing privacy attacks in practice. Additionally, achieving meaningful differential privacy for graph neural networks that focus on non-Euclidean data is still an open problem. Due to these limitations and problems, we 1) propose new mechanisms to enhance the utility and privacy trade-off in private model training and further improve the practicality to obtain meaningful privacy guarantees when training deep models on sensitive data. Moreover, we 2) propose new DP notions and develop algorithms to provide a quantifiable privacy guarantee against model inversion attacks whose goal is to recover the target class's visual property or features. Finally, we 3) develop new algorithms to achieve node-level differential privacy when training deep learning models on graph data based on differentially private approximate personalized PageRank and differentially private stochastic gradient descent.

Differentially Private Deep Learning

By

Qiuchen Zhang
B.S., Xi'an Jiaotong University, 2014
M.S., The Ohio State University, 2015

Advisor: Li Xiong, Ph.D.

A dissertation submitted to the Faculty of the
James T. Laney School of Graduate Studies of Emory University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in Computer Science and Informatics
2021

Acknowledgments

First of all, I want to thank my advisor, Professor Li Xiong for all her guidance and help over the past several years. I am so lucky to meet such a great mentor in my PhD career. She always encourages me to explore the direction I am interested in and is willing to provide me with full support. I will never forget the time Dr. Xiong has devoted to my research, discussing the problems and methods during meetings, revising and polishing the papers together late at night. I am grateful for these experiences and learned a lot from them.

I would like to thank Professor Vaidy Sunderam and Professor Joyce Ho for serving on my qualifying exam and dissertation committees. Thanks for their insightful comments and suggestions during the process. I would also like to thank Professor Xiaoqian Jiang for attending my dissertation proposal and defense as a committee member, and I want to thank him for his guidance and help during my summer research internship at UTHealth.

I also want to thank my collaborators and friends, Jian Lou, Yonghui Xiao, Jinfei Liu, and Jason Yang, for their help. At the same time, I would like to thank Professor Carl Yang for his help in the work of differentially private graph neural networks. I would like to extend my sincere thanks to Edgar Leon, Yvette Hilaire, and Terry Ingram for their work and help.

Finally, I want to thank my parents for all their support and encouragement. Last but most important, I want to thank my most beloved one, Jing Ma. I will never forget those times that we encouraged each other and worked together. Thank you, and I love you all.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Motivation

Neural networks have achieved great success in many real world tasks from image and speech recognition to precision medicine and self-driving cars [16, 63, 67, 82]. Training neural networks requires large amount of training data and significant computational resources. Cloud systems that offer machine-learning-as-a-service such as Amazon ML [2], Google AI platform [1] and Microsoft Machine Learning Studio [3] have gained increasing popularity. Clients can upload datasets to the cloud server which will train the models, and the clients can obtain access to the model either by downloading the model or performing prediction queries using the model via Application Programming Interfaces (API). Some of these services provide marketplaces where clients can share the trained models with other users with either full description of the model (*white-box*) or API access (*black-box*).

Publishing or sharing the deep learning model trained on private datasets directly could pose serious privacy concerns. The data used for training the model may contain users' sensitive information such as images, voice, medical histories, and location traces. Even though the adversaries do not have access to the original training data, they can use the models to infer or reconstruct (the features of) the training data. Several works demonstrated different attacks aiming to extract information about the private training datasets from the published deep learning models. *Membership inference attacks* [97] attempt to infer whether or not a specific record was in the training dataset given black-box API access to the model. *Model inversion attacks* [34] (MIA) attempt to reconstruct a recognizable face image corresponding to a person (a class) from a face recognition model given the name of the person (the class label) and white-box access to the model. We can see that the purpose of membership inference attack and MIA are different. The former attempts to recover the "existence" information of a target data point, while the latter attempts to recover the statistical property or features of a target class (which can be also private). Therefore, both of them are considered as privacy threats and violations [44, 46, 80, 93, 106].

Differential privacy (DP) [26, 28] has demonstrated itself as a strong and provable privacy framework for statistical data analysis and recently been explored to protect the privacy of training data when training deep learning models [4, 5, 85]. Phan et al. [86] explore the objective function perturbation method and use it to train a deep autoencoder satisfying DP. However, it may not be trivial to generalize to other deep learning models. Shokri and Shmatikov [96] train a deep model across multiple sites collaboratively and protect the DP of each updated parameter. However, the overall privacy budget for the model is huge which leads to a meaningless privacy guarantee.

One widely accepted way to provide rigorous DP guarantee for training neural network models on sensitive data is to use differentially private Stochastic Gradient Descent (DP-SGD) which adds Gaussian noise to the gradients in each iteration during the SGD based optimization process [4]. However, as the model goes deeper, this method becomes less effective [84, 128]. Another promising approach is *Private Aggregation of Teacher Ensembles* (PATE), which trains multiple teacher models on disjoint sensitive data and transfers the knowledge of teacher ensembles to a student model by letting the teachers vote for the label of each record from an unlabeled public dataset [85]. The teachers' votes are aggregated through a differentially private noisy-max mechanism, which is to add DP noise to the number of each label's votes first and then take the label with the majority count as the output. Finally, the student model is trained on the partially labeled public dataset in a semi-supervised fashion and published, while the teacher models are kept private.

Compared to DP-SGD, PATE obtains higher accuracy with a tighter privacy guarantee on the same dataset used in both works [84]. Meanwhile, the PATE mechanism is independent of the learning algorithms and can be applied to different model structures and to datasets with various characteristics. However, the knowledge transferred from teachers to the student, which are noisy-max voted labels, contain a certain proportion of errors or noisy labels, and the proportion has a positive relationship with the level of privacy guarantee that PATE provides and a negative impact on the accuracy of the student model.

On the another hand, since most works on deep learning with DP focus on improving model accuracy given a privacy requirement or enhancing the privacy and utility trade-off, there is still a limited understanding on how effective DP is in protecting against the above mentioned privacy attacks in practice. MA Rahman et al. [89] evaluated DP against membership inference attacks and showed that DP can protect against the attacks successfully only by sacrificing model utility by a considerable margin. This is not surprising as the indistinguishability guarantee of DP with respect to the presence of a record is directly aligned with the goal of preventing the inference of the membership of a record. Injecting noise to the model parameters required by DP naturally degrades the performance of the model. Whether DP or other mechanisms can provide meaningful privacy protection against model inversion attacks without sacrificing model utility is still an open question. While [34] proposed some preliminary defense measures against MIA, it does not provide a rigorous or quantifiable guarantee against the attacks. Intuitively, if we apply the standard record-level DP, the perturbed model may provide some mitigation to MIA due to the perturbed model parameters. However, since there are typically multiple instances (e.g. face images) corresponding to the same class (e.g. person), record-level DP which only protects the presence of one record may not prevent the reconstruction attack since all the records of the same class are encoded in the model.

Finally, the deep learning models we indicated above mainly deal with non-relational and grid-based data such as image, audio, text and gene sequences [30, 42, 69, 87, 102]. Following the success of deep neural networks in these fields, deep learning models have been adopted to process network data with graph structure where each node in the graph can correlate with other nodes through edges. Graph Neural Networks (GNN) [39, 58] has demonstrated outstanding performance in processing graph data and achieved great success in many applications, such as node classification, link prediction, and community detection [47, 50, 101, 111, 116, 125]. In many real-world applications, graphs can be social networks or economic networks where nodes represent the users with private personal information, and edges indicate the relationship or financial interactions among them. GNN models train on sensitive graphs will embed users' confidential information and potentially be explored by attackers [110], which will severely damage the user's privacy and trigger a potential crime. Therefore, achieve meaningful differential privacy for each node when training GNN models on the graph while preserving the model utility is very important. However, despite those successes in training deep learning models that protect the privacy of each data record, there are fewer methods dedicated to preserving the differential privacy for each node in the graph when training GNN models. Yang et al. [118] propose to train a graph generation model using DP-SGD to generate graphs with the edge-DP guarantee that protects the individual link privacy. Sajadmanesh et al. [91] develop a privacy-preserving GNN training algorithm based on local differential privacy (LDP) to protect node features privacy. Zhang et al. [127] use LDP and functional mechanism [124] to enforce privacy guarantee on user's sensitive features when training graph embedding models for recommendation. None of these previous works achieve the goal of providing strict DP with respect to each node in the graph for GNN models.

1.2 Research Contributions

This thesis will mainly contribute from three aspects to solve the limitations and practical issues mentioned above for achieving differential privacy when training deep learning models. First, from the dilemma between privacy and utility point of view, in chapter 3, we enhance the privacy and utility trade-off of DP algorithms in the general deep learning settings. We develop two new mechanisms, PATE+ and PATE++, to train more robust PATE under noisy labels, which achieve better privacy and utility trade-off in learning private models. Second, from the perspective of the defense ability of the existing differentially private deep learning models to MIA, in chapter 4, we broaden the DP definitions and algorithms for non-typical privacy attacks (MIA vs. membership inference attacks). We propose two new DP notions, class-DP and subclass-DP, and develop corresponding algorithms to provide a quantifiable privacy guarantee against MIA. Third, from the perspective of the effectiveness of the existing methods in training DP models on graph data, in chapter 5, we broaden the DP definitions and algorithms for non-typical data (graph data vs. relational data). We present new algorithms to provide node-level differential privacy when training deep learning models on graph data. We summary these contributions as in the following.

Towards Training Robust PATE Under Noisy Labels (Chapter 3). To mitigate the dilemma between providing stronger privacy protection and achieving higher model accuracy in the PATE framework and further improve its effectiveness in privacy-preserving model learning, we propose an enhanced framework PATE++ by incorporating the state-of-the-art noisy label training mechanism into PATE, and a novel noisy label detection mechanism based on the co-teaching(+) [40, 121] framework.

- We modify the student model in the original PATE, a generative adversarial network (GAN) [38] with a semi-supervised training strategy [94], by adding another discriminator to the structure of GAN. The purpose of the second discriminator is to enable co-teaching+ [121] with the first discriminator for robust training with noisy labels.
- We discuss the intrinsic limitations of the "update-by-disagreement" method in the co-teaching+ mechanism and develop a novel noisy label detection mechanism for semi-supervised model training to further improve student model performance when training with noisy labels.
- We evaluate our framework on Fashion-MNIST and SVHN datasets. Empirical results demonstrate that our new PATE structure with additional noisy label detection and switching (from labeled data to unlabeled data) mechanism outperforms the original PATE in privacy-preserving model training.

Provide Quantifiable Privacy Guarantees Against MIA (Chapter 4). We focus on the MIA against deep learning models and aim to understand whether existing DP can provide meaningful defense against MIA. Our works show that while it provides some mitigation, it does not provide effective and quantifiable protection. We subsequently propose new DP notions and mechanisms for more effective and quantifiable protection against MIA.

- We design an enhanced MIA attack with new regularization terms and change-of-variables method to better reconstruct the images. We show that the enhanced attack is more successful both via visual inspection and a formal attack success metric based on the Euclidean distance and structural similarity between the reconstructed images and the original images.

- We propose both class-level DP (class-DP) and subclass-level DP (subclass-DP) for deep neural networks as quantifiable privacy notions against MIA. Subclass-DP can be considered a weaker version of class-DP, allowing better and customizable privacy and utility trade-off when the class number of the privacy dataset is small. We compare and show the relationships between class-DP and subclass-DP with existing DP notions.

- We propose algorithms for training deep learning models with class-DP and subclass-DP. We formally prove the privacy guarantees of the proposed algorithms, and evaluate them on real-world datasets. The results demonstrate that the level of class and subclass-DP directly correlates with the robustness against the MIA and hence can provide a quantifiable measure against the risk.

Achieving Node-Level Differential Privacy for GNN Models (Chapter 5). We aim to ensure rigorous node-level DP for training GNN models. In other words, we want to prevent adversaries from distinguishing two neighboring graphs differing in one node and all of its connecting edges through the model trained on one of the graphs. During the training of GNN models, the graph topological structure will be exploited by the recursive message-passing procedure that propagates information through the graph [39, 101, 112, 117, 129]. The sensitivity of the model parameters due to presence or absence of each node becomes very large due to the correlation from their connecting edges. In order to train a GNN model with good performance while ensuring rigorous node-level DP, we decouple the message-passing process from feature aggregation of GNNs as proposed in [13, 59].

- To the best of our knowledge, we conduct the first formal study of training GNN models with the rigorous node-level DP guarantee. The key idea is to decouple the message-passing process from feature aggregation via DP approximate personalized PageRank and DP-SGD to protect both graph topology and node features while maintaining high model utility.

- We develop two DP approximate personalized PageRank computation algorithms with formal node-level DP guarantee based on the Gaussian mechanism and the exponential mechanism.

- We theoretically analyze the privacy loss caused by the random neighborhood sampling process designated by the PageRank result and calibrate tight Gaussian noise for DP-SGD to provide a rigorous overall privacy guarantee.

- We conduct extensive experiments on real-world graph datasets to demonstrate the effectiveness of the proposed algorithms in achieving satisfying privacy and utility trade-off. We also empirically demonstrate the DP models' ability in concealing the node degree information with respect to varying DP parameters to show the effectiveness of privacy protection of models trained using our proposed algorithms.

Chapter 2

Background

2.1 Differential Privacy

Differential Privacy (DP) ensures the output distributions of an algorithm are indistinguishable with a certain probability when the input datasets are differing in only one record, which is achieved through adding some randomness to the output. Both Laplacian noise and Gaussian noise are widely used to achieve DP, and the scale of the noise is calibrated according to the privacy parameter(s) ϵ (and δ) as well as the sensitivity of the algorithm [28]. In the definition of DP, ϵ and δ are the privacy parameters or privacy budget, which indicate the privacy loss. A smaller ϵ means a higher level of indistinguishability and hence stronger privacy. A smaller δ means a lower probability that the privacy guarantee provided by ϵ will be broken.

Definition 1. *((ϵ, δ)-Differential Privacy) [28]. Let $\mathcal{D}$ and $\mathcal{D}'$ be two neighboring datasets that differ in at most one entry. A randomized algorithm $\mathcal{A}$ satisfies (ϵ, δ)-differential privacy if for all $\mathcal{S} \subseteq Range(\mathcal{A})$:*

$$Pr\left[\mathcal{A}(\mathcal{D}) \in \mathcal{S}\right] \leq e^{\epsilon} Pr\left[\mathcal{A}(\mathcal{D}') \in \mathcal{S}\right] + \delta,$$

where $\mathcal{A}(\mathcal{D})$ represents the output of $\mathcal{A}$ with the input $\mathcal{D}$.

The granularity of DP is dependent on the definition of neighboring datasets. In the original DP definition, two neighboring datasets differ in one record, which can be considered as record-level DP (record-DP). It hides the presence of any record in the input dataset. The standard hamming distance-based DP can be extended depending on other notions of distance between the neighboring datasets under different situations [8,18].

Definition 2. *(Sensitivity) [28]. For two neighboring datasets $\mathcal{D}$ and $\mathcal{D}'$ differing in at most one entry, the sensitivity of an algorithm $\mathcal{A}$ is the maximum change in the norm of the output value of algorithm $\mathcal{A}$ regarding the two neighboring datasets:*

$$\Delta(A) = \sup_{\mathcal{D},\mathcal{D}'} \|\mathcal{A}(\mathcal{D}) - \mathcal{A}(\mathcal{D}')\|$$

where $\|\cdot\|$ denotes L_1 or L_2 norm for the l_1-sensitivity Δ_1 or l_2-sensitivity Δ_2, respectively.

Rényi Differential Privacy (RDP) generalizes (ϵ, 0)-DP in the sense that ϵ-DP is equivalent to (∞, ϵ)-RDP.

Definition 3. *(Rényi Differential Privacy (RDP)) [76]. A randomized mechanism $\mathcal{A}$ is said to guarantee (λ, ϵ)-RDP with $\lambda \geq 1$ if for any neighboring datasets $\mathcal{D}$ and $\mathcal{D}'$,*

$$D_\lambda(\mathcal{A}(\mathcal{D})\|\mathcal{A}(\mathcal{D}')) = \frac{1}{\lambda - 1} \log \mathbb{E}_{x\sim\mathcal{A}(D)} \left[\left(\frac{Pr[\mathcal{A}(D) = x]}{Pr\left[\mathcal{A}\left(D'\right) = x\right]} \right)^{\lambda-1} \right] \leq \epsilon.$$

In the above definition, $D_\lambda(\mathcal{A}(\mathcal{D})\|\mathcal{A}(\mathcal{D}'))$ indicates the Rényi divergence of order λ between $\mathcal{A}(\mathcal{D})$ and $\mathcal{A}(\mathcal{D}')$. RDP satisfies the adaptive sequential composition property of the privacy guarantee as stated in Proposition 1. The self-composition property of two RDP mechanisms can be generalized to the sequence of mechanisms as in Theorem 1.

Proposition 1. *(RDP Composition) [76] Let $f : \mathcal{D} \mapsto \mathcal{R}_1$ be (α, ϵ_1)-RDP and $g : \mathcal{R}_1 \times \mathcal{D} \mapsto \mathcal{R}_2$ be (α, ϵ_2)-RDP, then the mechanism defined as (X, Y), where $X \sim f(D)$ and $Y \sim g(X, D)$, satisfies $(\alpha, \epsilon_1 + \epsilon_2)$-RDP.*

Theorem 1. *(Sequence Composition) [85]. If a mechanism $\mathcal{A}$ consists of a sequence of adaptive mechanisms $\mathcal{A}_1$, ..., $\mathcal{A}_k$ such that for any $i \in [k], \mathcal{A}_i$ guarantees (λ, ε_i)-RDP, then $\mathcal{A}$ guarantees $\left(\lambda, \sum_{i=1}^{k} \varepsilon_i\right)$-RDP.*

Theorem 2. *(From RDP to (ϵ, δ)-DP) [76]. If a mechanism $\mathcal{A}$ guarantees (λ, ϵ)-RDP, then $\mathcal{A}$ guarantees $\left(\epsilon + \frac{\log 1/\delta}{\lambda - 1}, \delta\right)$-DP for any $0 < \delta < 1$.*

Theorem 2 reveals the relationship between (ϵ, δ)-DP and (λ, ϵ)-RDP. Both of them are relaxed from pure ϵ-DP, while RDP equipped with Gaussian noise has better composition property when analyzing the accumulated privacy loss.

The Gaussian mechanism [28] has been used to achieve (ϵ, δ)-DP. For $c^2 > 2\ln(1.25/\delta)$, the Gaussian Mechanism with parameter $\sigma \geq c\Delta_2(A)/\epsilon$ adds Gaussian noise scaled to $\mathcal{N}(0, \sigma^2)$ to each component of the output of algorithm $\mathcal{A}$, resulting in $M(D) \triangleq A(D) + \mathcal{N}(0, \sigma^2)$. The approximating function $M(D)$ satisfies (ϵ, δ)-DP.

Corollary 1. *(Gaussian Mechanism for RDP) [76] Let $f : \mathcal{D} \mapsto \mathcal{R}$ be a real-valued function. If $\mathcal{A}$ has sensitivity 1, then the Gaussian mechanism $G_\sigma \mathcal{A} = f(D) + N\left(0, \sigma^2\right)$ satisfies $(\alpha, \alpha/(2\sigma^2))$-RDP, where $N\left(0, \sigma^2\right)$ is normally distributed random variable with standard deviation σ^2 and mean* 0.

Definition 4. *(The Exponential Mechanism) [28, 73] Given some arbitrary range $\mathcal{R}$, the exponential mechanism relies on a utility function $u : \mathbb{N}^{|\mathcal{X}|} \times \mathcal{R} \rightarrow \mathbb{R}$ that assigns a real valued score to one output $r \in \mathcal{R}$ given a dataset x, where higher scores indicate more desirable outputs. Given the utility function u and the sensitivity of u with respect to any pair of neighboring datasets x and y, which is: $\Delta u \equiv \max_{r \in \mathcal{R}} \max_{x,y:\|x-y\|_1 \leq 1} |u(x, r) - u(y, r)|$, the exponential mechanism $\mathcal{M}_E(x, u, \mathcal{R})$ selects and outputs an element $r \in \mathcal{R}$ with probability proportional to $\exp\left(\frac{\varepsilon u(x,r)}{2\Delta u}\right)$.*

Theorem 3. *The exponential mechanism preserves (ϵ, 0)-differential privacy [28].*

2.2 Deep Learning with Differential Privacy

DP has been applied to deep learning models with DP-SGD algorithms [4,64] in order to protect the privacy of training datasets. In each SGD iteration, DP-SGD clips the Euclidean norm of the gradient and injects calibrated Gaussian noise to the clipped gradient. Each iteration of the DP-SGD becomes a randomized mechanism with a quantifiable *privacy loss*. For a general randomized mechanism $\mathcal{A}$, the privacy loss is defined as follows:

Definition 5. *(Privacy Loss [4]) For neighboring datasets $\mathcal{D}, \mathcal{D}'$, auxiliary input $\boldsymbol{aux}$ and output $\boldsymbol{o} \in Range(\mathcal{A})$, the privacy loss at a particular output $\boldsymbol{o}$ is defined as,*

$$c(\boldsymbol{o}|\mathcal{A}, \boldsymbol{aux}, \mathcal{D}, \mathcal{D}') := \log \frac{\mathbb{P}[\mathcal{A}(\boldsymbol{aux}, \mathcal{D}) = \boldsymbol{o}]}{\mathbb{P}[\mathcal{A}(\boldsymbol{aux}, \mathcal{D}') = \boldsymbol{o}]}. \tag{2.1}$$

For DP-SGD, each iteration incurs a privacy loss, where $\mathcal{A}$ represents one iteration of the DP-SGD update procedure, $\boldsymbol{o}$ is the updated parameter vector, $\boldsymbol{aux}$ is all the parameter sequences obtained before this iteration, $\mathcal{D}$ is the training dataset. Abadi et al. [4] proposed the moments accountant technique with random sampling that provides a tighter privacy loss composition than the advanced composition theorems [29] for the overall privacy loss of DP-SGD over multiple iterations. The moments accountant is defined as follows:

Definition 6. *(Moments Accountant [4]) The moments accountant of a randomized mechanism $\mathcal{A}$ with κ-th moment is defined as follows,*

$$\alpha_{\mathcal{A}}(\kappa) := \arg \max_{\boldsymbol{aux},(\mathcal{D},\mathcal{D}')} \log \mathbb{E}[exp(\kappa c(\boldsymbol{o}|\mathcal{A}, \boldsymbol{aux}, \mathcal{D}, \mathcal{D}'))], \tag{2.2}$$

where the expectation is taken over the output distribution $\boldsymbol{o} \sim \mathcal{A}(\boldsymbol{aux}, \mathcal{D})$ and

$c(\boldsymbol{o}|\mathcal{A}, \boldsymbol{aux}, \mathcal{D}, \mathcal{D}')$ *is the privacy loss.*

Another promising approach for achieving differential privacy in deep learning is *Private Aggregation of Teacher Ensembles*, or PATE, which trains multiple teacher models on disjoint sensitive data and transfers the knowledge of teacher ensembles to a student model by letting the teachers vote for the label of each record from an unlabeled public dataset [84, 85]. The teachers' votes are aggregated through a differentially private noisy-max mechanism, which is to add DP noise to the number of each label's votes first and then take the label with the majority count as the output. Finally, the student model is trained on the partially labeled public dataset in a semi-supervised fashion and published, while the teacher models are kept private. Compared to DP-SGD, PATE achieves higher accuracy with a tighter privacy guarantee. Meanwhile, the PATE method is independent of the learning algorithms and can be applied to different model structures and to datasets with various characteristics. However, the knowledge transferred from teachers to the student, which are noisy-max voted labels, contain a certain proportion of errors or noisy labels, and the proportion has a positive relationship with the level of privacy guarantee that PATE provides and a negative impact on the accuracy of the student model.

Chapter 3

Towards Training Robust PATE Under Noisy Labels

3.1 Introduction

In this chapter, we propose an enhanced framework PATE++ by incorporating the start-of-the-art noisy label training mechanism into PATE to further improve its practical applicability. PATE++ makes several novel contributions. First, we modify the student model in the original PATE, a generative adversarial network (GAN) [38] with a semi-supervised training strategy [94], by adding another discriminator to the structure of GAN. The purpose of the second discriminator is to enable co-teaching [40] with the first discriminator for robust training with noisy labels. Second, to further exploit the benefit of semi-supervised training, we propose a novel noisy label detection mechanism based on the co-teaching framework and move the data with detected noisy labels from labeled dataset to unlabeled dataset instead of excluding them completely from the training process. We evaluate our framework on Fashion-MNIST and SVHN datasets. Empirical results demonstrate that our new PATE structure with additional noisy label detection and switching (from labeled data to unlabeled data) mechanism outperforms the original PATE in privacy-preserving model training. Our work mitigates the dilemma in the PATE framework between providing stronger privacy protection and achieving higher model accuracy, and further improves its effectiveness in privacy-preserving model learning.

3.2 Preliminaries

In this section, we introduce the two essential components of our approach: (1) the PATE framework which was first developed by Papernot et al. in [84] and later improved by Papernot et al. in [85]; (2) the co-teaching mechanism for robust model training with noisy labels and the improved co-teaching+ mechanism [121].

3.2.1 The PATE Framework

Figure 3.1 illustrates the framework of PATE borrowed from [84]. It consists of an ensemble of teacher models and a student model. Each teacher is trained on a disjoint subset of sensitive data that contains user's private information that needs to be protected. Teacher models can be flexibly chosen to fit the data and task. After teachers are trained, the knowledge that teachers learned from sensitive data will be transferred to the student in a private manner. More specifically, at prediction, teachers independently predict labels for the queried data from an unlabeled public dataset. The votes assigned to each class will be counted to form a histogram. To ensure DP, Laplacian or Gaussian noise will be added to each count. The final prediction result for the queried data will be the label with the most votes after adding the noise.

The student model in the PATE framework uses GAN with semi-supervised learning. During student model training, labeled public data are fed into the discriminator D of GAN to form the supervised cross-entropy loss while unlabeled data and gener-

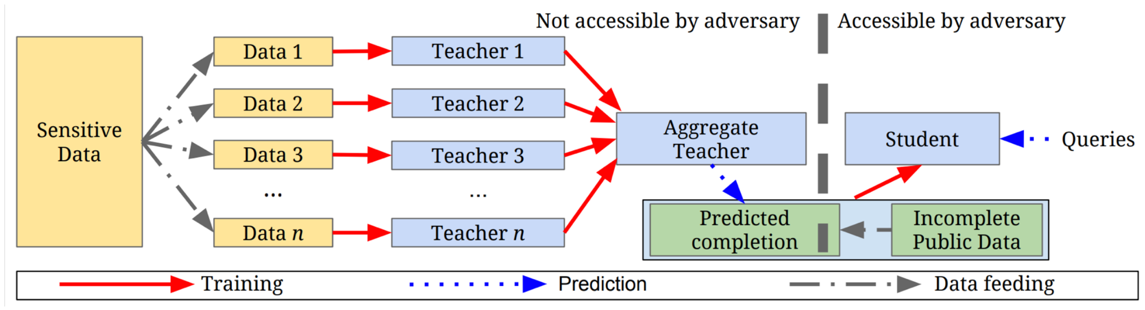


Figure 3.1: Overview of the PATE framework: (1) an ensemble of teachers is trained on disjoint subsets of the sensitive data, (2) a student model is trained on public data labeled using the ensemble.

ated data from generator G (labeled as an additional 'generated' class) are fed into D to form the unsupervised loss. Feature matching is used to increase the stability of GAN by involving a new objective for G, which requires the activations of real data and generated data on an intermediate layer of D to be as similar as possible through gradient-based optimization.

The initial PATE uses Laplacian noise for the perturbation and moments accountant [4] to compose the total privacy cost for multiple predictions. The improved PATE uses Gaussian noise based on RDP. Additionally, they proposed a *selective* aggregation mechanism called the confident Gaussian NoisyMax aggregator (Confident-GNMax) as in Algorithm 2. Teacher ensembles will only answer the queries if their votes have strong consensus, which is checked privately. This mechanism benefits both privacy and utility. The privacy cost is small when most teachers agree on one vote. Meanwhile, when most teachers agree, the prediction result is more likely to be correct. However, even with the Confident-GNMax mechanism, the voted labels still contain a certain ratio of errors due to the noisy aggregation. Additionally, in order to achieve a tighter privacy guarantee, larger noise is needed for perturbing the votes, thus causing more noise in the student training dataset, which severely affects the utility of the trained student model.

3.2.2 Co-teaching and Co-teaching+ Mechanisms

Deep learning models have enough capacity to remember all training instances even with noisy labels, which leads to bad generalization ability [122]. Han et al. [40] propose a simple but effective mechanism called co-teaching for training deep models with the existence of noisy labels. Their method is based on the observation that during the training, models would first memorize or fit training data with clean labels and then those with noisy labels [9]. Co-teaching maintains two networks with the same structure but independent initialization. In each mini-batch of data, each network selects a ratio of small-loss instances as useful knowledge and teaches its peer network with such useful instances for updating the parameters. Intuitively, small-loss instances are more likely to be the ones with correct labels, thus training the network in each mini-batch using only small-loss instances is more robust to noisy labels.

In the early stage of co-teaching, due to independent and random parameter initialization, two networks have different abilities to filter out different types of error using the small-loss trick. However, this divergence between two networks will gradually diminish with the increase of training epochs, which decreases the ability to select clean data and increases the accumulated error. To solve this issue, Yu et al. introduce the "Update by Disagreement" strategy to co-teaching and name the improved mechanism co-teaching+ [121]. Similar to co-teaching, co-teaching+ maintains two networks simultaneously. In each mini-batch of training, two networks feed forward and predict the same batch of data independently first, and then a ratio of small-loss instances will be chosen by each network only from those data with the disagreed predictions between two networks and fed to each other for parameter updating. This disagreement-update step keeps the constant divergence between two networks and promotes the ability of them to select clean data.

3.3 Improved Training Mechanism For PATE

Inspired by co-teaching mechanism and its improved version co-teaching+, we modify the PATE framework to improve the student model's robustness when training with noisy labels provided by teachers.

3.3.1 PATE+: Student Model with Co-teaching+

The student model of PATE is a GAN trained under semi-supervised learning with both supervised and unsupervised losses while co-teaching(+) is originally used in the supervised model training. To utilize co-teaching(+) in the student model, our main idea is to add an additional discriminator in the GAN used in the student model, as shown in Figure 3.2. We do not use two GANs with both generator and discriminator as the peers for co-teaching(+) because the small-loss trick plays its role only in the supervised part, while the generator is involved in the unsupervised loss of GAN as well as the feature matching loss [94], which are both unsupervised and not associated with labels.

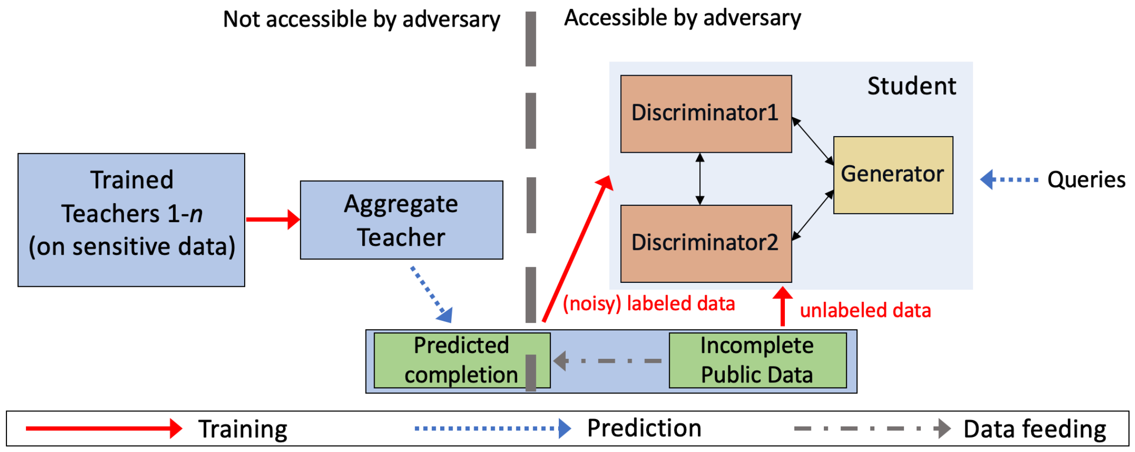


Figure 3.2: Overview of the PATE+ framework. (1) an ensemble of teachers is trained on disjoint subsets of the sensitive data, (2) a semi-supervised GAN student model with one generator and two discriminators co-teaching+ with each other is trained on public data labeled using the ensemble.

Suppose there exist K possible classes in sensitive data as well as the labeled public data that the student model will be trained on. In the semi-supervised learning using GANs, the data generated by generator G are labeled with a new "generated" class $y = K + 1$. The discriminator D takes in a data sample $\boldsymbol{x}$ as input and outputs class probabilities distribution $p_D(y|\boldsymbol{x}, j < K + 1)$. For labeled data $\boldsymbol{x}$, the cross-entropy between the observed label and the predicted distribution $p_D(y|\boldsymbol{x}, j < K+1)$ forms the supervised loss. For generated data, $p_D(y = K + 1|\boldsymbol{x})$ is used to supply the probability that x is not real. For those unlabeled data, since we know they come from one of the K classes of real data, we can learn from them by maximizing $\log p_D(y \in \{1, \ldots, K\}|\boldsymbol{x})$ [94].

For the student model in Figure 3.2, there are two discriminators and one generator. The supervised loss and unsupervised loss for `Discriminator1` and `Discriminator2` (D_1 and D_2) are expressed as:

$$\begin{aligned} L^{D_i}_{\text{supervised}} &= -\{\mathbb{E}_{\boldsymbol{x},y\sim p_{\text{data}}\,(\boldsymbol{x},y)} \log p_{D_i}(y|\boldsymbol{x}, y < K + 1)\}; \\ L^{D_i}_{\text{unsupervised}} &= -\,\{\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}\,(\boldsymbol{x})} \log [1 - p_{D_i}(y = K + 1|\boldsymbol{x})] \\ &\qquad + \mathbb{E}_{\boldsymbol{x}\sim G} \log [p_{D_i}(y = K + 1|\boldsymbol{x})]\}. \end{aligned}$$

where $i = 1, 2$ and p_{data} indicates the real data distribution. Feature matching loss in the semi-supervised GANs training is defined as: $\left\|\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}} \mathbf{f}(\boldsymbol{x}) - \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\mathbf{f}(G(\boldsymbol{z}))\right\|_2^2$, where $p_{\boldsymbol{z}}(\boldsymbol{z})$ indicates the random distribution and $\mathbf{f}(\boldsymbol{x})$ is the activation output of an intermediate layer of the discriminator. In the structure of student model as shown in Figure 3.2, the generator takes the activations from two discriminators which are expressed as $\mathbf{f}_{D_1}(\boldsymbol{x})$ and $\mathbf{f}_{D_2}(\boldsymbol{x})$ respectively. We use the average of two feature losses associated with two discriminators as the objective for the generator. Therefore, the feature matching loss of the generator in the student model is defined as:

$$\begin{aligned} L_{\text{fm}}^{G} = & \frac{1}{2}\Big(\left\|\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}} \mathbf{f}_{D_1}(\boldsymbol{x}) - \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\mathbf{f}_{D_1}(G(\boldsymbol{z}))\right\|_2^2 \\ & + \left\|\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}} \mathbf{f}_{D_2}(\boldsymbol{x}) - \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\mathbf{f}_{D_2}(G(\boldsymbol{z}))\right\|_2^2 \Big). \end{aligned}$$

The main steps for training student model with the "update by disagreement" strategy are illustrated in Algorithm 1.

Algorithm 1: PATE+: Training Student Model in PATE with Discriminators Co-teaching+

```
1 Input: $D_1$, $D_2$, $G$, labeled public data $M_l$ from private teachers aggregation,
    unlabeled data $M_u$, batch size $B$, learning rate $\eta$, epoch $E$, ratio $R$.

   1: Duplicate $M_l$ or $M_u$ to make them have the same size.
   2: for $e = 1, ..., E$ do
   3:    Shuffle $M_l$, $M_u$ into $\frac{|M_l|}{B}$ mini-batches respectively.
   4:    for $b = 1, ..., \frac{|M_l|}{B}$ do
   5:       Fetch $b$-th mini-batch $m_l$ ($m_u$) from $M_l$ ($M_u$);
   6:       Generate B fake samples $m_g$ from $G$;
   7:       Select samples with the different predicted results between $D_1$ and $D_2$ in $m_l$ as
            $\hat{m_l}$
   8:       for $i = 1, 2$ do
   9:          Fetch the $R\%$ smallest-loss samples $\hat{m_l}^{(i)}$ of $D_i$:
               $\hat{m_l}^{(i)} = \text{argmin}_{\hat{m_l}':|\hat{m_l}'| \geq R|\hat{m_l}|} L_{\text{supervised}}^{D_i}(\hat{m_l}'; D_i)$
  10:       end for
  11:       Update $D_1 = D_1 - \eta\nabla(L_{\text{supervised}}^{D_1}(\hat{m_l}^{(2)}; D_1) + L_{\text{unsupervised}}^{D_1}(m_u, m_g; D_1))$ // $D_1$
            indicates parameters of Discriminator1 here
  12:       Update $D_2 = D_2 - \eta\nabla(L_{\text{supervised}}^{D_2}(\hat{m_l}^{(1)}; D_2) + L_{\text{unsupervised}}^{D_2}(m_u, m_g; D_2))$ // $D_2$
            indicates parameters of Discriminator2 here
  13:       Update $G = G - \eta\nabla L_{\text{fm}}^{G}(m_u, m_g; G)$ // $G$ indicates parameters of Generator
            here
  14:    end for
  15: end for
  Output: Trained $D_1$, $D_2$ and $G$, where $D_1$ and $D_2$ satisfy rigorous DP guarantee.
```

3.3.2 Privacy Guarantee of PATE+

The privacy guarantee of Algorithm 1 is inherited from the privacy guarantee of the labeled public dataset M_l by the post-processing property of DP [28]. We analyze the

RDP guarantee of generating M_l by the Confident-GNMax aggregator which is used in the scalable PATE framework [85]. To start with, we recall the two steps of the Confident-GNMax aggregator. Given a query sample x belonging to one of the classes from 1 to m, let $n_i(x)$ denote the vote count for the i-th class of x. Confident-GNMax aggregator first privately checks if there is enough consensus among teachers (line 1 in Algorithm 2). $\mathcal{N}(0, \sigma_1^2)$ is the Gaussian distribution with mean 0 and variance σ_1^2. If the check is passed, Confident-GNMax aggregator will output the class label with noisy plurality after adding Gaussian noise ($\mathcal{N}(0, \sigma_2^2)$) to each vote count (line 2 in Algorithm 2), while discarding this query without labeling it if the pass is failed. The sensitivity of private consensus check (line 1) is 1 because the private training data is divided without overlapping, and one data sample will only affect one teacher model which will change the maximum vote count ($\max_i \{n_j(x)\}$) by at most 1. Therefore, line 1 in Algorithm 2 guarantees $(\lambda, \lambda/2\sigma_1^2)$-RDP for all $\lambda > 1$ by corollary 1. Line 2 in Algorithm 2 is the GNMax mechanism in [85]. By the data-dependent privacy guarantee in Proposition 8 of [85], line 2 satisfies $(\lambda, \lambda/\sigma_2^2)$-RDP for all $\lambda > 1$. By using the composition property of RDP in Proposition 1, we can conclude the privacy guarantee for the Confident-GNMax Aggregator as in Theorem 4.

Algorithm 2: Confident-GNMax Aggregator [85]

```
1 Input: input $x$, threshold $T$, noise parameters $\sigma_1$ and $\sigma_2$.

   1: if $\max_i \{n_j(x)\} + \mathcal{N}\left(0, \sigma_1^2\right) \geq T$ then
   2:     return $argmax_j \left\{n_j(x) + \mathcal{N}\left(0, \sigma_2^2\right)\right\}$
   3: else
   4:     return $\perp$
   5: end if
```

Theorem 4. *For any $\lambda > 1$, the Confident-GNMax Aggregator in Algorithm 2 satisfies (λ, β)-RDP where $\beta = \lambda/2\sigma_1^2 + \lambda/\sigma_2^2$ if the private consensus check in line 1 of Algorithm 2 is passed, or $\beta = \lambda/2\sigma_1^2$ if the check is failed.*

By using the privacy guarantee of Confident-GNMax aggregator in Theorem 4, we derive the privacy guarantee of the PATE+ algorithm.

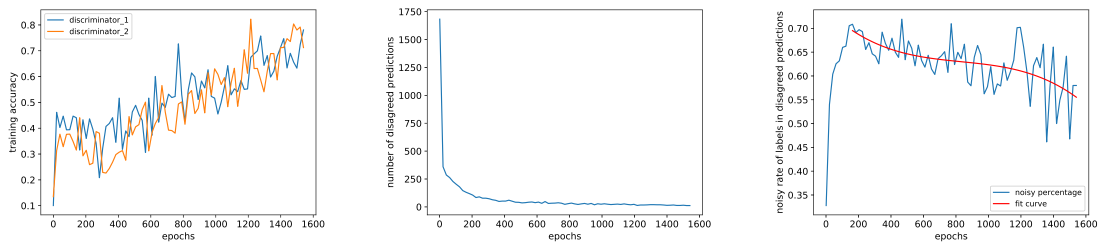
Proposition 2. *If querying the teacher ensembles with a public dataset M, and the teacher ensembles label M using Confident-GNMax aggregator in Algorithm 2 to generate a labeled dataset M_l, then the student model trained on M_l using PATE+ algorithm in Algorithm 1 satisfies (ϵ, δ)-DP for any $0 < \delta < 1$ and $\epsilon = \lambda\left(\frac{|M|}{2\sigma_1^2} + \frac{|M_l|}{\sigma_2^2}\right) + \frac{\log 1/\delta}{\lambda-1}$, where $\lambda > 1$.*

Proof. Suppose the number of data samples in public dataset M and in labeled dataset M_l is $|M|$ and $|M_l|$ respectively. Therefore, the number of data samples that are discarded (without labeling) during Confident-GNMax aggregation is $|M| - |M_l|$. We use Theorem 4 in conjunction with Theorem 1 to derive the total (λ, β)-RDP privacy guarantee for answering M and generating M_l, where $\lambda > 1$ and $\beta = (|M| - |M_l|) * \frac{\lambda}{2\sigma_1^2} + |M_l| * (\frac{\lambda}{2\sigma_1^2} + \frac{\lambda}{\sigma_2^2}) = \frac{|M|\lambda}{2\sigma_1^2} + \frac{|M_l|\lambda}{\sigma_2^2}$. By Theorem 2, we can transfer $(\lambda, \frac{|M|\lambda}{2\sigma_1^2} + \frac{|M_l|\lambda}{\sigma_2^2})$-RDP into $\left(\lambda\left(\frac{|M|}{2\sigma_1^2} + \frac{|M_l|}{\sigma_2^2}\right) + \frac{\log 1/\delta}{\lambda-1}, \delta\right)$ -DP for any $0 < \delta < 1$. By the post-processing property of DP, the student model trained on M_l will satisfies (ϵ, δ)-DP where $\epsilon = \lambda\left(\frac{|M|}{2\sigma_1^2} + \frac{|M_l|}{\sigma_2^2}\right) + \frac{\log 1/\delta}{\lambda-1}$ and $0 < \delta < 1$ since it has no access to the private training data of teacher ensembles and therefore, can not obtain additional knowledge about the private dataset. □

Notice that training PATE with co-teaching satisfies the same privacy guarantee with PATE+ because the discrepancy between co-teaching and co-teaching+, which is the "update by disagreement" strategy, is independent with the private training data of teacher ensembles and the privacy analysis.

3.3.3 PATE++: PATE+ with Noisy Label Cleansing

Potential Drawbacks in PATE+. "Update by disagreement" strategy actually has two potential drawbacks. First, in the late stage of training, two discriminators are going to achieve a similar capacity and consensus on the predictions with most



(a) training accuracy

(b) number of disagreed predictions

(c) noisy rate of labels in disagreed predictions

Figure 3.3: A student model trained on 2,200 labeled (726 are noisy labels) and 6,800 unlabeled data from Fashion-MNIST dataset using PATE+ algorithm. (a) The training accuracy of two discriminators in the student model vs epochs (b) The number of labeled samples with different predictions by two discriminators vs epochs (c) The noisy rate of labels in disagreed predictions vs epochs.

data. Therefore, the number of "disagreed" data in each mini-batch is limited, which restricts the models from learning since they can only learn from limited data. Second, the proportion of noisy labels within the "disagreement" in the mini-batch is increasing with the epoch, and models' utility is sacrificed by learning from data with more noisy labels. According to the learning pattern of deep models [9], after the models have learned to fit easy (clean) data, they are more likely to agree on the predictions of clean data while disagreeing on noisy data because the predictions on noisy data have more randomness and errors before models fit them.

We demonstrate our hypothesis using an example. We train a student model using Algorithm 1. Each discriminator is a convolutional neural network (see Experiments on Fashion-MNIST for details). A total of 2,200 data labeled by Confident-GNMax aggregator and 6,800 unlabeled data is used as the training dataset, where 726 of 2,200 labeled samples are noisy data (the labels of them are different from their ground truth labels). Fig 3.3(a) shows the training accuracy of two discriminators in the student model. We can see that they follow different learning paths. Fig 3.3(b) shows the number of training samples with disagreed predictions by two discriminators. The number is decreased to a small value during the training process. This observation is consistent with our first hypothesis of the potential drawbacks of the "update by disagreement" strategy. When two discriminators gradually acquire a similar capacity, the number of "disagreed" data in each mini-batch is few (less than 50 out of 2,200 after 400 epochs). Therefore, discriminators can only learn from very few data in the late period, which seriously affects their learning capacity. We calculate the percentage of noisy labels within the "disagreement" in each epoch as shown by the blue line in Fig 3.3(c). The noisy label rate in all labeled data is 0.33, while the noisy label rate in the "disagreement" is much higher. This observation reflects our second hypothesis that the model's utility will be sacrificed by learning from the "disagreed" data which contains more noisy labels.

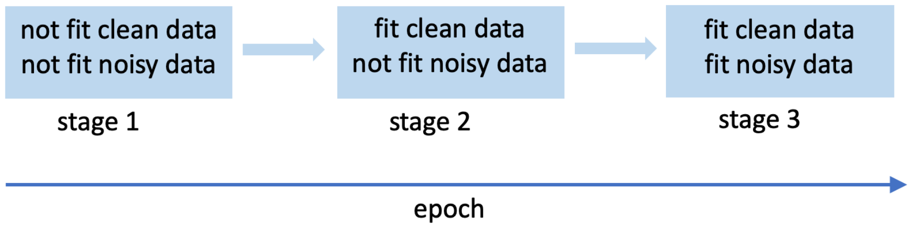


Figure 3.4: Illustration of the three stages of model training process with the existence of noisy data.

Model Training Stages. We roughly divide the model learning process into three stages based on the observations in [9]. In the early stage which is indicated as stage 1 in Figure 3.4, models have not fit either clean or noisy data. The disagreement on predictions between two peer models is mainly caused by randomness. The percentage of noisy labels within the "disagreement" roughly equals the percentage of noisy labels in the entire training dataset. In stage 2, models have fit the clean data (except for "hard examples") but not the noisy data. The peer models are more likely to have the same (and correct) predictions for clean data. For those noisy data, since the models have not fitted them, the predictions of them are more random and with more errors. Therefore, prediction disagreements are more likely to happen on the data with noisy labels during this stage. In stage 3, which is the late stage of training, due to the memorization effort, the models have learned to fit both the clean and noisy data. The peer models begin to be more consistent in the prediction of both types of data. Thus the ratio of noisy labels in the disagreed predictions decreases. We can observe this phenomenon from Fig 3.3(c). We fit part of the blue line which is the noisy label rate in the "disagreement" as the function of epochs using smoothing spline fit [103] to observe the general trend of the curve more clearly, which is shown in the red line. We can see the noisy percentage in the "disagreement" increases in the early stage of model training while decreases in the later stage.

Noisy Label Cleansing. Based on our analysis and observations, we hypothesize that the noisy label ratio is the highest within the “disagreement” during stage 2. We propose to filter out noisy labels using this criterion, i.e., the data that has different prediction results by the two peer models during stage 2. However, there is a critical situation that we need to consider. Two peer models do not always have the same learning speed, and they follow different paths during the optimization (as shown in Figure 3.3(a)). Therefore, it could happen that one model already fits the clean data while the other does not. In this situation, suppose there is a data record with the true (clean) label, the first model gives the correct prediction with high probability, while the second model with the weaker capability predicts it as other labels incorrectly and causes the variation in predictions. Thus clean data could also be chosen by the “disagreement” criterion. To avoid this situation, we further refine our criterion. Notice that in the above-mentioned circumstance, the “disagreement” happens when the first model with the stronger capability predicts the true label for the clean data (the predicted label is the same as the observed label) while the second model with the weaker capability predicts a wrong label (the predicted label is different from the observed label). Therefore, we further filter out noisy data whose predicted labels by peer models are both different from the observed label from the “disagreement” in stage 2. That is, our noisy label cleansing mechanism has two criteria: 1) peer

Algorithm 3: PATE++: PATE+ with Noisy Label Cleansing

```
1 Input: $D_1$, $D_2$, $G$, labeled public data $M_l$ from private teachers aggregation,
    unlabeled data $M_u$, batch size $B$, learning rate $\eta$, epoch $E$, ratio $R$, removal
    percentage $\tau$, decay factor $\alpha$.
   1: Step 1: Filter out noisy label in $M_l$ based on PATE+ framework
   2: Duplicate $M_l$ or $M_u$ to make them have the same size.
   3: Initialize the filtered out noisy dataset $M_n$ as $\emptyset$.
   4: Initialize a count table $T$ for each data in $M_l$ to be 0.
   5: for $e = 1, ..., E$ do
   6:   Shuffle $M_l$, $M_u$ into $\frac{|M_l|}{B}$ mini-batches respectively.
   7:   for $b = 1, ..., \frac{|M_l|}{B}$ do
   8:     Fetch $b$-th mini-batch $m_l$ ($m_u$) from $M_l$ ($M_u$);
   9:     Generate B fake samples $m_g$ from $G$;
  10:     Select samples with the different predicted results between $D_1$ and $D_2$ in $m_l$ as
          $\hat{m_l}$
  11:     Select samples in $\hat{m_l}$ whose prediction results by $D_1$ and $D_2$ are both different
          with its observed label as $\overline{m_l}$.
  12:     Set the count of data in $\overline{m_l}$ to 1.
  13:     Fetch the $R$% smallest-loss samples $\hat{m_l}^{(1)}$ ($\hat{m_l}^{(2)}$) of $D_1$ ($D_2$) as in line 8-10 in
          Algorithm 1
  14:     Update $D_1, D_2$, $G$ as in line 11-13 in Algorithm 1
  15:   end for
  16:   Multiply the count of each labeled data in this epoch with $\alpha$ and add to the count
        table $T$.
  17: end for

  18: Step 2: Remove filtered out noisy labels
  19: Remove $\tau$% data with the most count from $M_l$ to form $M_l^{san}$.
  20: Add those removed data to the unlabeled dataset to form $M_u^{san}$.

  21: Step 3: Retrain the PATE+ on sanitized datasets $M_l^{san}$ and $M_u^{san}$ using
      Algorithm 1
  Output: Trained $D_1$, $D_2$ and $G$, where $D_1$ and $D_2$ satisfy rigorous DP guarantee.
```

models disagree on the predictions for this data, and 2) the prediction results by two peer models are both different from the observed label of the data.

The last question is, how can we know when the models change from stage 1(2) to stage 2(3). One possible solution is to use the validation utility to help us decide. In stage 1, models have very low utility since they fit neither clean nor noisy data. In stage 2, the utility of models increases as the models have learned useful knowledge from clean and easy-to-fit data. In stage 3, models' utility can still increase but with relatively slower speed compared to stage 2, since noisy labels are hard to fit. However, due to the uncertainty of the gradient-based optimization process, it is not efficient to separate these stages using the validation utility. We solve this problem using the weighted decay count. We count the number of epochs for each data when it satisfies the previously mentioned two criteria. Clean data tend to satisfy those two criteria during stage 1, while noisy data tend to satisfy those two criteria in both stage 1 and stage 2. Therefore, data with more counts at the end of training are determined as the data with noisy labels. To further reduce the effects of stage 1, we multiply a weight (smaller than 1) to the counts at the end of each epoch before adding them to the new counts of the next epoch. Weighted decay count smooths the decision process and makes the criteria more robust to the randomness caused by the gradient-based optimization process.

PATE++. In Algorithm 3, we present the complete PATE++ framework for training more robust PATE by filtering out noisy labels based on the PATE+ framework first, and then retraining PATE+ on the sanitized dataset, which is formed by removing the top $\tau\%$ data with the most count as introduced above. τ indicates the removal percentage. The privacy analysis for Algorithm 3 follows Proposition 2 by the post-processing property of DP. Notice the noisy label cleansing procedure does not involve additional privacy leakage since it does not depend on the private training dataset of teacher ensembles.

3.4 Experiments

We performed experiments on Fashion-MNIST and SVHN to demonstrate the efficiency of our proposed PATE+ and PATE++ frameworks compared to the original PATE for training the student model on noisy data provided by private teachers aggregation.

3.4.1 Fashion-MNIST

Fashion-MNIST dataset [113] consists of 10 classes with 60,000 training examples and 10,000 testing examples. Similar to in the original PATE, we use 60,000 training examples to train the teachers and 10,000 testing examples as the public dataset for training the student. We divide the 60,000 training examples randomly into 250 disjoint subsets equally. Each subset is used to train one teacher model, which is a convolutional neural network with seven convolutional layers followed by two fully connected layers and an output layer (same as the deep model in the original PATE). After 250 teachers are trained, we use Confident-GNMax aggregator to label 2,200 data from the public dataset twice. For the first time, we use the smaller noise which leads to (5.04, 10^{-5})-DP guarantee. For the second time, we use the larger noise which leads to (4.05, 10^{-5})-DP guarantee. Adding the larger noise during the private teacher aggregation leads to a tighter privacy guarantee (smaller ϵ), while the trade-off is that there will be more noisy labels within the labeled dataset. The structure of the discriminators in the student model is the same as the structure of teachers. The generator of the student model is a three-layer fully connected neural network. The 10,000 testing examples are further divided into the first 9,000 (where 2,200 are labeled by teachers as labeled data and 6,800 are used as unlabeled data) for training and the last 1,000 for testing. We compare the test accuracy of the student models trained by 1) the original PATE (traditional semi-supervised training); 2)

PATE with co-teaching between two peer discriminators; 3) PATE+ (PATE with co-teaching+ between two peer discriminators); and 4) PATE++ (PATE+ with noisy label cleansing). We train student models with batch size 100 using Adam optimizer with the learning rate set to 0.01. In PATE++, the decay factor α is set to 0.9 by grid search. Table 3.1 shows the experimental results.

From Table 3.1, we can observe that PATE++ achieves the best performance on training the student model. The improvement is even higher (4.8% vs. 0.8%) when the privacy budget is tight (4.05 vs. 5.04). This further motivates our proposed mechanism PATE++, since there is an inevitable trade-off between utility and privacy in the PATE framework, The stronger privacy requires adding larger noise during the private teacher aggregation which leads to a higher noise ratio in the student training data. PATE++ mitigates this by making the student model more robust when trained with noisy labels.

Table 3.1: Test accuracy of the students under different frameworks trained on Fashion-MNIST dataset.

	Student Accuracy			
Privacy budget (ϵ, δ)	**Original PATE**	**PATE with co-teaching**	**PATE+ (Alg.1)**	**PATE++ (Alg.3)**
$(4.05,\ 10^{-5})$	74.8%	77.3%	76.5%	**79.6%**
$(5.04,\ 10^{-5})$	82.1%	82.5%	82.7%	**82.9%**

Selection of R and τ. As suggested in [40], the ratio of small-loss instances R should be chosen increasingly during the training since when the number of epochs goes large, the model will gradually overfit on noisy labels. Thus, more instances can be kept in the mini-batch at the start while less should be in the end. We use their proposed scheduling: $R(e) = 1 - \beta \min\left\{\frac{e}{15}, 1\right\}$ where e is the epoch and β is the estimated noise rate which can be determined by manually verifying a small sampled subset. We report the student accuracy of 1) PATE with co-teaching, and 2) PATE+ (the same as PATE++ with τ=0) under the different noise ratio estimation values in

Table 3.2. We show the setting with $(4.05, 10^{-5})$-DP guarantee. We can see that the estimated noise rate for the scheduling has an effect on student performance. How to best estimate the noise rate and set the optimal scheduling function is still an unsolved problem in the co-teaching and co-teaching+ works [40, 121].

Table 3.2: Test accuracy of the student models with varying R (bold results coincide with Table 3.1).

Estimated Noise Ratio $\boldsymbol{\beta}$	0.1	0.2	0.3	0.4
PATE with co-teaching	76.2%	**77.3%**	77%	76.2%
PATE+ (Alg.1)	76.4%	**76.5%**	77.3%	77.4%

Table 3.3: Test accuracy of the student models with varying τ (bold result coincides with Table 3.1).

Removal Ratio $\boldsymbol{\tau}$	0.091	0.182	0.227	0.273	0.318	0.364
PATE++ ($\beta = 0.2$)	78.1%	78.6%	79%	79.5%	**79.6%**	78.2%

We fix $\beta = 0.2$ and report the student accuracy of PATE++ with different τ values for the noisy label cleansing ratio in Table 3.3. Increasing the removal ratio τ will increase the chance to move more noisy labels from the labeled dataset to the unlabeled dataset and lead to better student performance because the student model is trained on the dataset with less noisy labels. However, the trade-off is that with the higher removal ratio, less labeled data will be left as well as data with clean labels that the student can learn useful knowledge from. In practice, we choose the removal ratio by grid search.

3.4.2 SVHN

SVHN [78] contains 10 classes with 73,257 training examples and 26,032 testing examples. We use the same structure for the student model as in Fashion-MNIST experiments. The 26,032 testing examples are divided into 10,000 for student training and 16,032 for student testing. We use the clean teacher votes made available online by the authors of PATE to do the Confident-GNMax aggregation for labeling student's training data. 3,000 data are labeled privately using the smaller noise corresponding to (4.93, 10^{-6})-DP guarantee and the larger noise corresponding to (3.96, 10^{-6})-DP guarantee. The student models are trained the batch size 100 inputs using the Adam optimizer with the learning rate set to 0.003 and the decay factor α set to 0.9 in PATE++. Table 3.4 shows the experimental results on SVHN with the estimated noise rate $\beta = 0.2$ and the removal percentage $\tau = 0.4$.

Table 3.4: Test accuracy of the students under different frameworks trained on SVHN dataset.

	Student Accuracy			
Privacy budget (ϵ, δ)	**Original PATE**	**PATE with co-teaching**	**PATE+ (Alg.1)**	**PATE++ (Alg.3)**
(3.96, 10^{-6})	80.5%	86.1%	79.8%	**91.5%**
(4.93, 10^{-6})	91.7%	92.8%	91.6%	**93.7%**

We can observe in Table 3.4 that PATE++ significantly outperforms the original PATE, especially under the tight privacy budget. The student accuracy of PATE+ is shy when compared with PATE with co-teaching. The reason could be the drawbacks of the "update by disagreement" strategy that we mentioned previously.

3.5 Related Work

[104] proposed to transfer the knowledge learned from a publicly available non-private dataset to the teachers in order to alleviate the problem that the training data assigned for each individual teacher maybe not enough to achieve an ideal performance for some complex datasets and tasks. [100] exploited knowledge distillation [45] to further transfer the knowledge from teacher ensembles to the student model privately through the representations from intermediate layers of teacher models. [11] developed a new semi-supervised learning algorithm called MixMatch, which achieves state-of-the-art performance in several benchmark datasets by combining several dominant

approaches for semi-supervised learning together into a unified framework. They demonstrate that MixMatch improves the performance of PATE with respect to the accuracy-privacy trade-off, which is unsurprising because PATE is a general framework with the student model trained by the semi-supervised learning paradigm in order to reduce the total privacy cost induced by each individual query. Any improved semi-supervised learning algorithm is expected to improve the original PATE framework. Different from these previous works, our work improves PATE from another perspective by incorporating the novel noisy label training and cleansing mechanism under the semi-supervised learning framework to improve the student model accuracy without additional privacy cost.

Learning with noisy examples has a long research history [88]. Currently, training deep learning models with noisy labels has received increasing attention [40, 49, 54, 65, 72, 121]. A comprehensive review of all the works within this area is beyond the scope of this chapter. Our proposed mechanisms incorporate the co-teaching and co-teaching+ methods into the PATE framework to better train the student model with noisy labels and achieve promising results. Investigation of other noisy label training methods to further enhance the performance will be an interesting research direction.

3.6 Conclusion

We proposed the PATE+ mechanism for robust training of the student model in PATE, and PATE++ mechanism based on PATE+ which combines co-teaching+ between two discriminators within the structure of GAN and noisy label cleansing. Experimental results demonstrate the advantage of our mechanisms compared to the original PATE, especially when the privacy budget is tight. Our proposed mechanisms enhance the utility and privacy trade-off in private model training and further improve the practicality to achieve meaningful privacy guarantees when training deep models on sensitive data. We leave applying PATE++ to other applications such as sequence-based models and graph models as future work.

Chapter 4

Provide Quantifiable Privacy Guarantees Against MIA

4.1 Introduction

Most works on deep learning with DP focus on improving model accuracy given a privacy requirement or enhancing the privacy and utility trade-off. There is still a limited demonstration of how effective DP is in protecting against the above mentioned attacks in practice. [89] evaluated DP against membership inference attacks and showed that DP can protect against the attacks successfully only by sacrificing model utility by a considerable margin. This is not surprising as the indistinguishability guarantee of DP with respect to the presence of a record is directly aligned with the goal of preventing the inference of the membership of a record. Injecting noise to the model parameters required by DP naturally degrades the performance of the model.

Whether DP or other mechanisms can provide meaningful privacy protection against model inversion attacks without sacrificing model utility is still an open question. While [34] proposed some preliminary defense measures against MIA, it does not provide a rigorous or quantifiable guarantee against the attacks. Intuitively, if we apply the standard record-level DP, the perturbed model may provide some mitigation to MIA due to the perturbed model parameters. However, since there are typically multiple instances (e.g. face images) corresponding to the same class (e.g. person), record-level DP which only protects the presence of one record may not prevent the reconstruction attack since all the records of the same class are encoded in the model. Another potential solution is to use group-DP [28] to protect the presence of all records corresponding to one class as a group. However, this will lead to amplified perturbation by the group size which can be determined by the largest class size. Such an application may yield unacceptable model accuracy due to the significantly amplified perturbation while overprotecting certain data since different classes may have varying numbers of records.

In this chapter, we focus on the MIA against deep learning models and aim to understand whether existing DP can provide meaningful defense against MIA. Our results show that while it provides some mitigation, it does not provide effective and quantifiable protection. We subsequently propose new DP notions and mechanisms for more effective and quantifiable protection against MIA.

4.2 Preliminaries

We consider the setting where a model provider trains a neural network classification model $f(\mathbf{x})$ using a private training set D, where $\mathbf{x} \in \mathbb{R}^d$ is an input record in d-dimensional space. The output of $f(\mathbf{x})$ is the prediction vector $\mathbf{y} \in R^k$ where each dimension corresponds to one predefined label or class. The model provider shares the trained model with other parties without sharing the data. We study model inversion attacks where an adversary abuses the shared model by attempting to reconstruct the original (features of) training data corresponding to a target class. Our goal is to develop privacy notions and algorithms that allow a model provider to build a model that is robust against model inversion attacks.

Threat Model. We assume a white-box attack in which an adversary has access to the published model including model structure and parameters, but has no access to the training data, nor back door access [98] to the training process.

4.2.1 Model Inversion Attack

Model inversion attack [34] is a reverse engineering attack that attempts to "reconstruct" the training data from a trained neural network model. Given the model parameters and a target *label*, the goal is to find a data point $\mathbf{x}$ corresponding to the label following the same distribution with data points in $\mathcal{D}$ that maximizes $f_{label}(\mathbf{x})$, which is equivalent to minimize the following objective function:

$$c(\mathbf{x}) = 1 - f_{label}(\mathbf{x}), \tag{4.1}$$

where $f_{label}(\mathbf{x})$ is the confidence score of the target class.

While the reconstructed data point may not correspond to a specific data point in the dataset, it leaks the statistical property or general features of the target class. For example, a face image generated by a successful MIA reveals how the person with the target name (the class label) looks like [34].

4.3 Improved Model Inversion Attack

While the original MIA has gained success on simple neural networks such as Softmax regression and Multilayer perceptron network (MLP) [34], it has limited success on deep neural networks only with auxiliary training data [46] or with adversarial training [74]. For more complex models, MIA tends to produce images that look unrealistic even with the denoising and sharpening filter [34]. In this section, we propose new regularization terms to enhance the optimization used in MIA to produce more recognizable images. We demonstrate that the enhanced MIA can be effective against deep learning models with more complex network structures.

ℓ_1-Norm Regularization. ℓ_1-norm regularization can be used to enforce sparsity on the solution vector, or reconstructed image $\mathbf{x}$. The sparsity constraint reduces and limits the intensity of pixels which are not important in leading the model to output the target class label. Therefore, it can help with removing noise and enhancing the contrast of the output image $\mathbf{x}$, especially with black and white images. The loss function of MIA with ℓ_1-norm regularization on image $\mathbf{x}$ becomes:

$$c(\mathbf{x}) = 1 - f_{label}(\mathbf{x}) + \lambda \|\mathbf{x}\|_1 \tag{4.2}$$

where the coefficient λ controls the penalty effect caused by ℓ_1-norm regularization.

BTV Regularization. While ℓ_1-norm regularizer may achieve noise removal and contrast enhancement on black and white images with a clear contrast, this benefit may be limited on gray scale images. For such images, we propose to use the bilateral total variation (BTV) regularization [32]:

$$R_{BTV} = \sum_{l=-p}^{p} \sum_{\substack{m=0 \\ m+l\geq 0}}^{p} \alpha^{m+l} \left\|\mathbf{x} - S_x^l S_y^m \mathbf{x}\right\|_1 \tag{4.3}$$

The BTV regularizer is essentially the accumulation of differences between central pixels and their neighborhoods within the spatial window size measured by p. It helps to maintain the main image features and preserve sharp edges when performing the super-resolution reconstruction task where the goal is to recover a single high-resolution image from a set of low-resolution images [32, 61]. The loss function of MIA with BTV regularization becomes:

$$c(\mathbf{x}) = 1 - f_{label}(\mathbf{x}) + \lambda R_{BTV} \tag{4.4}$$

Enhanced MIA Algorithm. Algorithms 4 outlines our enhanced MIA with the new regularizers. Line 4 uses an optional change-of-variable for the optimization by introducing a "box constraint" [17] to ensure that the value of each pixel in the reconstructed image stays in the range $[0,1]$: $\mathbf{x} = \frac{1}{2}(\tanh(\mathbf{w}) + 1)$. The optimization is then implemented over $\mathbf{w}$. If no change-of-variable is used, we can directly set $\mathbf{x} = \mathbf{w}$. We use Adam optimizer instead of SGD used in the original MIA [34], which uses the moving average of the first and second moments of gradients (line 6 and 7) to scale the learning rate adaptively.

Algorithm 4: Improved MIA Algorithm

```
   Input: label, $T$, $\beta_1$, $\beta_2$, $\tau$, $\eta$, $\lambda$, the target model $f$.
1  Initialize variables $\mathbf{w}$, $\mathbf{m}$, and $\mathbf{v}$ to be zeros with the same size as training
     images of $f$.
2  Define $c(\mathbf{x})$ using eq.(4.2) or (4.4)
3  for $t = 1 \cdots T$ do
4      $\mathbf{x}_{t-1} = \frac{1}{2}(\tanh(\mathbf{w}_{t-1}) + 1)$
5      $\mathbf{g}_t = \nabla c(\mathbf{x}_{t-1})$
6      $\mathbf{m}_t = \beta_1 \mathbf{m}_{t-1} + (1 - \beta_1)\mathbf{g}_t$
7      $\mathbf{v}_t = \beta_2 \mathbf{v}_{t-1} + (1 - \beta_2)\mathbf{g}_t^2$
8      $\mathbf{w}_t = \mathbf{w}_{t-1} - \eta \frac{\mathbf{m}_t}{\sqrt{\mathbf{v}_t + \tau}}$
9  end
10 return $\mathbf{x}_T = \frac{1}{2}(\tanh(\mathbf{w}_T) + 1)$
```

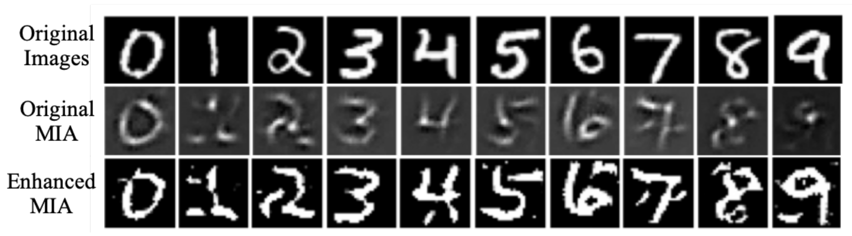


Figure 4.1: MIA on MNIST dataset.

Visual Results. Figure 4.1 shows the reconstructed images of the original and enhanced MIA (using ℓ_1 norm and change-of-variable) against a CNN model trained on the MNIST dataset in comparison to sample original images. We set the parameter values as $T = 5000$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\tau = 10^{-8}$, $\eta = 0.1$, and $\lambda = 0.05$. We can

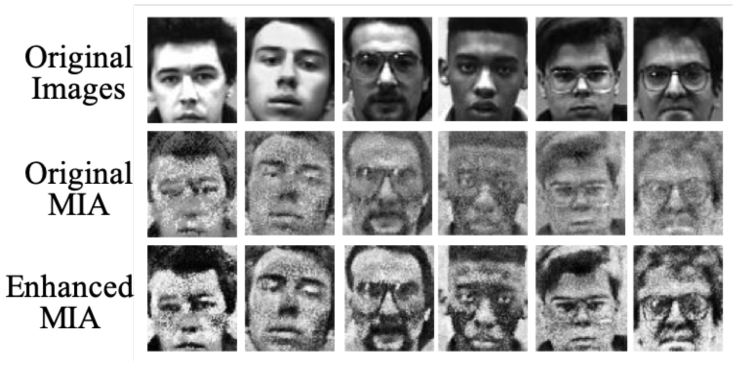


Figure 4.2: MIA on Faces94 dataset.

observe that the enhanced MIA generates more realistic and similar images to the original images than the original MIA. Figure 4.2 shows the reconstructed images of the original and enhanced MIA (using BTV regularization) against a softmax classifier trained on the Faces94 dataset (see Section 4.5 for details). We set $p = 2$, $\alpha = 0.9$, and $\lambda = 0.001$ for the BTV regularization. We observe that the reconstructed face images by enhanced MIA preserve sharper edges and corners, and less blur compared to original MIA.

Table 4.1: MIA distance for MNIST dataset

Class (Digit)	0	1	2	3	4	5	6	7	8	9
Original MIA	8.02	8.36	9.14	9.50	8.35	9.76	9.64	8.38	8.79	8.93
Enhanced MIA	**5.93**	**7.89**	**8.51**	**7.92**	**8.2**	**8.41**	**8.51**	**7.76**	**8.06**	**8.76**

Table 4.2: MIA distance for Faces94 dataset

Class (Person)	1	2	3	4	5	6
Original MIA	0.707	0.663	0.64	0.681	0.668	0.707
Enhanced MIA	**0.691**	**0.643**	**0.626**	**0.664**	**0.65**	**0.692**

Attack Success Metric. To quantify the results of reconstruction besides visual inspection, we define an attack success metric, *MIA distance*, as the minimum distance between the reconstructed image and all the training images in the target class. In contrast to the average distance which measures the distance between the reconstructed image and the "average" image of the target class, we use minimum distance because it represents the worst case scenario. A smaller distance indicates the recovered image is more similar to the original training data, suggesting a more successful attack. A larger distance means that the attack is less successful and the model is more robust. We will also use this metric to evaluate the model's robustness against MIA.

Different distance metrics can be used depending on the data. For example, for the MNIST dataset which includes simple white and black images, we can use Euclidean distance which is shown in Table 4.1. We observe that the reconstructed images from enhanced MIA have a consistently smaller distance than the original MIA. For the gray-scale face images, we adopt the structural similarity index (SSIM) [108], which is more suitable for measuring the perceptual similarity between two face images than Euclidean distance [57]. The value range of SSIM is [0, 1] where 1 indicates the most similar. Table 4.2 shows the minimum distance (1-SSIM) between the reconstructed images and training images in the target class for both original MIA and enhanced MIA. We can observe that enhanced MIA achieves better results.

4.4 Class and Subclass Differential Privacy

In this section, we propose class-DP and subclass-DP as quantifiable privacy notions against MIA and corresponding privacy algorithms to achieve them.

4.4.1 Class-Level Differential Privacy

Definition of Class-Level DP. Our main goal is to provide a rigorous and strong privacy notion that can quantify the protection against MIA which targets the statistical property of a given class corresponding to a set of records in the training data. Intuitively, our secret to be protected is the statistical properties or features of a target class. Motivated by this, we propose class-level DP which defines the neighboring databases as two datasets differing in one class (i.e. all records that belong to the same class). Class-level DP guarantees that the resulting models are indistinguishable even if all the records in any one class are substituted. Therefore, MIA can not reconstruct a representative image of any target class.

Definition 7. *(Class Neighboring Datasets). Let $\mathcal{D}$ denote a dataset with K classes of records. The class neighboring datasets to $\mathcal{D}$ are the datasets $\mathcal{D}'$ that can be obtained from $\mathcal{D}$ by replacing all the records in an arbitrary class $k \in 1, ..., K$.*

Compared to the definition of neighboring datasets [28] in record-DP, a pair of class neighboring datasets differ in one class of data, which indicates they have the same number of classes and all of those classes are the same (same data and labels) except one. For example, let $\mathcal{D}$ be a hand-written digit dataset containing images of digits from 0 to 9, the class number of $\mathcal{D}$ is 10. Replacing all images of digit 0 with images of letter a in $\mathcal{D}$ forms a class neighboring dataset $\mathcal{D}'$.

Definition 8. *(Class-Level Differential Privacy). A randomized algorithm $\mathcal{A}$ with domain $\mathbb{N}^{|\mathcal{X}|}$ satisfies class-level (ϵ, δ)-differential privacy if for all $\mathcal{S} \subseteq \text{Range}(\mathcal{A})$ and for all class neighboring datasets $\mathcal{D}$, $\mathcal{D}' \in \mathbb{N}^{|\mathcal{X}|}$:*

$$\Pr[\mathcal{A}(\mathcal{D}) \in \mathcal{S}] \leq e^{\epsilon} \Pr[\mathcal{A}(\mathcal{D}') \in \mathcal{S}] + \delta.$$

Considering record-DP can be unbounded (neighboring datasets are formed by adding or removing one record) or bounded (replacing one record or removing and then adding one record), our class-DP definition here is bounded, i.e. the class neighboring datasets are formed by replacing an entire class. We can also define the unbounded class-DP which requires the indistinguishability of the resulting model regardless of whether an entire class is present or not in the training data. We will show in our privacy analysis later all proofs can be derived similarly with these two versions of DP with the only difference being a constant factor.

Comparison with Record and Group-DP. Class-DP is a strong privacy guarantee. It protects not only one data record in private datasets like in record-DP [28] but also all other records which share common patterns or follow the same distribution with that record in the same class. Class-DP is different from group-DP [28] which ensures the indistinguishability of the statistical output regardless of the presence or absence of any group of a given size of m. While bearing some similarities, class-DP is not equivalent to group-DP even if we assume all classes have the same size m. This is because the neighboring pairs in class-DP differ in one class, and the classes are only a subset of all possible groups of size m. We can consider class-DP (with the same class size m) as a weaker version of group-DP, but specifically designed to protect against MIA. In addition, class-DP allows groups of different sizes which are determined by the size of each class and hence provide more precise protection against MIA.

We can potentially adopt group-DP to protect against MIA by setting the group size as the largest class size. However, doing so will lead to amplified perturbation by the group size and hence unacceptable model accuracy. In fact, any (ϵ, δ)-DP mechanism M satisfies $(m\epsilon, m\delta)$-group-DP for group size m with no necessary change to the private training process. This amplifying factor m can be very large and will render the model not useful with a meaningful privacy guarantee.

4.4.2 Algorithm for Class-DP

Algorithm 5 outlines the steps to achieve class-DP for deep learning models based on class-based sampling. Suppose the training dataset $\mathcal{D} = \{C_1, \cdots, C_K\}$ contains K classes of data. During each step of the SGD, each class is sampled with probability q (line 3). The data of all selected classes will be used in the current step of SGD for calculating gradients and updating parameters. Dividing the noisy sum of the clipped gradient by the number of selected classes for the current SGD step approximates the average update of all classes while preventing the information of a single class from leakage.

Algorithm 5: Class-level differentially private SGD

```
Input: Training dataset $\mathcal{D} = \{x_1, \cdots, x_N\}$, loss function
       $\mathcal{L}(\theta) = \frac{1}{N}\sum_i \mathcal{L}(\theta, x_i)$, learning rate $\eta_t$, noise scale $\sigma$, gradient norm
       bound $S$, sampling ratio $q$.
1  Initialize $\theta_0$ randomly.
2  for $t = 1 \cdots T$ do
3      Sample each class with probability $q$.
4      for each selected class $C_i (i = 1, \cdots, k_t)$ do
5          For each $x_j \in C_i$, compute $\mathbf{g}_t(x_j) \leftarrow \nabla_{\theta_t}\mathcal{L}(\theta_t, x_j)$
6          Average gradients within class $C_i$, $g^{(i)} \leftarrow \frac{1}{|C_i|}\sum_{j=1}^{|C_i|} g_t(x_j)$
            %%Compute gradient
7          $\overline{g}^{(i)} \leftarrow g^{(i)} / \max\left(1, \frac{\|g^{(i)}\|_2}{S}\right)$ %%Clip gradient
8      end
9      $\tilde{g}_t \leftarrow \frac{1}{qK}\left(\sum_{i=1}^{k_t}\overline{g}^{(i)} + \mathcal{N}(0, \sigma^2 S^2 \mathbf{I})\right)$ %%Add noise
10     $\theta_{t+1} \leftarrow \theta_t - \eta_t \tilde{\mathbf{g}}_t$ %%Update
11 end
12 return $\theta_T$ and overall privacy budget $(\epsilon, \delta)$ computed by moments
     accountant with sampling.
```

Privacy Analysis. We analyze the privacy of Algorithm 5 by extending the moments accountant technique in Definition 6 to the class-DP setting. There are two key differences between the setting in [4] and ours: 1) [4] considers record-level DP while ours is class-level; 2) the neighboring concept in [4] is random "in or out" of a record while ours is a random substitution of a class. Due to such discrepancy, our analysis deviates from theirs. To begin with, we recall the following two lemmas from [4]. Lemma 1 facilitates the composition of the moments accountant of an iterative algorithm. Lemma 2 provides the translation of moments accountant to (ϵ, δ)-DP.

Lemma 1. *(Composibility [4]) Let mechanism $\mathcal{A}$ be a composition of a sequence of adaptive mechanisms $\mathcal{A}_1, ..., \mathcal{A}_T$, where $\mathcal{A}_t : \prod_{i=1}^{t-1} Range(\mathcal{A}_i) \times \mathcal{D} \rightarrow Range(\mathcal{A}_t)$. For any κ, it gives $\alpha_A(\kappa) \leq \sum_{t=1}^{T}\alpha_{\mathcal{A}_t}(\kappa)$.*

Lemma 2. *(Tail Bound [4]) For any $\epsilon > 0$, the mechanism $\mathcal{A}$ is (ϵ, δ)-DP for $\delta = \min_{\kappa} exp(\alpha_{\mathcal{A}} - \kappa\epsilon)$.*

We also develop the following Lemma 3 that will be used in our main DP result in Theorem 5. It adapts Theorem 2 in [4], where the main difference is to replace $\boldsymbol{\mu}_{\Gamma}30257 \sim \mathcal{N}(1, \sigma^2)$ there to $\boldsymbol{\mu}_{\Gamma}30258 \sim \mathcal{N}(2, \sigma^2)$ here, and quantify the new $\alpha_{\boldsymbol{\mu}_0,\boldsymbol{\mu}}(\kappa)$ accordingly. This is because for class-DP and subclass-DP, we prefer the neighboring dataset notion to be a random substitution of class/subclass rather than "in or out" of an arbitrary record.

Lemma 3. *Let $\boldsymbol{\mu}_{\Gamma}30256$ and $\boldsymbol{\mu}_{\Gamma}30258$ denote the probability density function of $\mathcal{N}(0, \sigma^2)$ and $\mathcal{N}(2, \sigma^2)$ respectively. Let $\boldsymbol{\mu}$ be the mixture of $\boldsymbol{\mu}_{\Gamma}30256$ and $\boldsymbol{\mu}_{\Gamma}30258$: $\boldsymbol{\mu} = (1-q)\boldsymbol{\mu}_{\Gamma}30256 + q\boldsymbol{\mu}_{\Gamma}30258$. Let $\alpha_{\boldsymbol{\mu}_0,\boldsymbol{\mu}}(\kappa) = \log\max(E_1, E_2)$, where $E_1 = \mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{\mu}_{\Gamma}30256}[(\frac{\boldsymbol{\mu}_{\Gamma}30256(\boldsymbol{z})}{\boldsymbol{\mu}(\boldsymbol{z})})^{\kappa}]$ and $E_2 = \mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{\mu}}[(\frac{\boldsymbol{\mu}(\boldsymbol{z})}{\boldsymbol{\mu}_{\Gamma}30256(\boldsymbol{z})})^{\kappa}]$. Suppose $q < \frac{1}{16\sigma}$ and $\kappa \leq \sigma^2 \ln\frac{1}{q\sigma}$, then it gives $\alpha_{\boldsymbol{\mu}_0,\boldsymbol{\mu}}(\kappa) \leq \frac{4q^2\kappa(\kappa+1)}{(1-q)\sigma^2} + O(q^3/\sigma^3)$.*

We follow the proof of Theorem 2 in [4] with an emphasize on the difference part with [4]. Let $\alpha = \frac{4q^2\kappa(\kappa+1)}{(1-q)\sigma^2} + O(q^3/\sigma^3)$. To prove $\alpha_{\boldsymbol{\mu}_0,\boldsymbol{\mu}}(\kappa) = \log\max(E_1, E_2) \leq \alpha$, we need to prove $E_1 = \mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{\mu}_{\Gamma}30256}[(\frac{\boldsymbol{\mu}_{\Gamma}30256(\boldsymbol{z})}{\boldsymbol{\mu}(\boldsymbol{z})})^{\kappa}] \leq 1+\alpha$; $E_2 = \mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{\mu}}[(\frac{\boldsymbol{\mu}(\boldsymbol{z})}{\boldsymbol{\mu}_{\Gamma}30256(\boldsymbol{z})})^{\kappa}] \leq 1+\alpha$, so that $\alpha_{\boldsymbol{\mu}_0,\boldsymbol{\mu}}(\kappa) \leq \log(1+\alpha) \leq \alpha$. Following [4], both inequalities can be proved by the same method. For any distributions ν_a and ν_b, $\mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{\nu}_{\Gamma}30561}[(\frac{\boldsymbol{\nu}_{\Gamma}30561(\boldsymbol{z})}{\boldsymbol{\nu}_{\Gamma}30562(\boldsymbol{z})})^{\kappa}] = \mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{\nu}_{\Gamma}30562}[(\frac{\boldsymbol{\nu}_{\Gamma}30561(\boldsymbol{z})}{\boldsymbol{\nu}_{\Gamma}30562(\boldsymbol{z})})^{\kappa+1}]$, where the latter can be expanded using binomial expansion,

$$\mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{\nu}_{\Gamma}30562}[(\frac{\boldsymbol{\nu}_{\Gamma}30561(\boldsymbol{z})}{\boldsymbol{\nu}_{\Gamma}30562(\boldsymbol{z})})^{\kappa+1}] = \sum_{i=0}^{\kappa+1}\binom{\kappa+1}{i}\mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{\nu}_{\Gamma}30562}[(\frac{\boldsymbol{\nu}_{\Gamma}30561(\boldsymbol{z}) - \boldsymbol{\nu}_{\Gamma}30562(\boldsymbol{z})}{\boldsymbol{\nu}_{\Gamma}30562(\boldsymbol{z})})^{i}].$$

Substituting $(\nu_a, \nu_b) = (\mu, \mu_0)$ and $(\nu_a, \nu_b) = (\mu_0, \mu)$ in, when $i = 0$, the first term is 1; when $i = 1$, the second term is 0. In the following, we calculate the third term with the more difficult case $(\nu_a, \nu_b) = (\mu_0, \mu)$, i.e., $i = 2$, which starts to deviate from [4]:

$$\mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{\mu}}[(\frac{\boldsymbol{\mu}_0(\boldsymbol{z}) - \boldsymbol{\mu}(\boldsymbol{z})}{\boldsymbol{\mu}(\boldsymbol{z})})^2] = q^2\mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{\mu}}[(\frac{\boldsymbol{\mu}_0(\boldsymbol{z}) - \boldsymbol{\mu}_\Gamma 30258(\boldsymbol{z})}{\boldsymbol{\mu}(\boldsymbol{z})})^2] \overset{(i)}{\leq} \frac{q^2}{1-q}$$
$$\int_{-\infty}^{+\infty} \frac{(\boldsymbol{\mu}_0(\boldsymbol{z}) - \boldsymbol{\mu}_2(\boldsymbol{z}))^2}{\boldsymbol{\mu}_0(\boldsymbol{z})} d\boldsymbol{z} = \frac{q^2}{1-q}\mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{\mu}_0}[\frac{(\boldsymbol{\mu}_0(\boldsymbol{z}) - \boldsymbol{\mu}_2(\boldsymbol{z}))^2}{\boldsymbol{\mu}_0(\boldsymbol{z})}],$$

where (i) is by $\boldsymbol{\mu} \geq (1-q)\boldsymbol{\mu}_0$ and the above can be further bounded by calculating the last expectation:

$$\begin{aligned}&\mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{\mu}_0}[\frac{(\boldsymbol{\mu}_0(\boldsymbol{z}) - \boldsymbol{\mu}_2(\boldsymbol{z}))^2}{\boldsymbol{\mu}_0(\boldsymbol{z})}] = \mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{\mu}_0}[(1 - exp(\frac{4\boldsymbol{z} - 4}{2\sigma^2}))^2] \\ &= exp(\frac{4}{\sigma^2}) - 1 \leq \frac{4}{\sigma^2}.\end{aligned} \tag{4.5}$$

The third term can be bounded as

$$\binom{\kappa+1}{2}\mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{\mu}}[(\frac{\boldsymbol{\mu}_0(\boldsymbol{z}) - \boldsymbol{\mu}(\boldsymbol{z})}{\boldsymbol{\mu}(\boldsymbol{z})})^2] \leq \frac{4q^2\kappa(\kappa+1)}{(1-q)\sigma^2}. \tag{4.6}$$

In the following, we show the terms from $i = 3, ...$ are dominated by $i = 3$ term which is of order $O(\frac{q^3\kappa^3}{\sigma^3})$.

$$\mathbb{E}_{\boldsymbol{z}\sim\boldsymbol{\mu}}[(\frac{\boldsymbol{\mu}_0(\boldsymbol{z}) - \boldsymbol{\mu}(\boldsymbol{z})}{\boldsymbol{\mu}(\boldsymbol{z})})^i] \leq \overbrace{\int_{-\infty}^{0} \boldsymbol{\mu}(\boldsymbol{z})|(\frac{\boldsymbol{\mu}_0(\boldsymbol{z}) - \boldsymbol{\mu}(\boldsymbol{z})}{\boldsymbol{\mu}(\boldsymbol{z})})^i| d\boldsymbol{z}}^{(I)} +$$
$$\overbrace{\int_{0}^{2} \boldsymbol{\mu}(\boldsymbol{z})|\frac{(\boldsymbol{\mu}_0(\boldsymbol{z}) - \boldsymbol{\mu}(\boldsymbol{z})}{\boldsymbol{\mu}(\boldsymbol{z})})^i| d\boldsymbol{z}}^{(II)} + \overbrace{\int_{2}^{+\infty} \boldsymbol{\mu}(\boldsymbol{z})|(\frac{\boldsymbol{\mu}_0(\boldsymbol{z}) - \boldsymbol{\mu}(\boldsymbol{z})}{\boldsymbol{\mu}(\boldsymbol{z})})^i| d\boldsymbol{z}}^{(III)}.$$

$$(I) \leq \frac{2^i q^i}{(1-q)^{i-1}\sigma^{2i}} \int_{-\infty}^{0} \boldsymbol{\mu}_0(\boldsymbol{z})|\boldsymbol{z} - 1|^i d\boldsymbol{z} \leq \frac{(4q)^i (i-1)!!}{2(1-q)^{i-1}\sigma^i}.$$

$$(II) \leq \frac{q^i}{(1-q)^i} \int_{0}^{2} \boldsymbol{\mu}(\boldsymbol{z}) \frac{4^i}{\sigma^{2i}} d\boldsymbol{z} \leq \frac{(4q)^i}{(1-q)^i\sigma^{2i}}$$

$$(III) \leq \frac{q^i}{(1-q)^{i-1}\sigma^{2i}} \int_{2}^{+\infty} \boldsymbol{\mu}_0(\boldsymbol{z})(\frac{2\boldsymbol{\mu}_2(\boldsymbol{z})}{\boldsymbol{\mu}_0(\boldsymbol{z})})^i d\boldsymbol{z},$$

which is 2^i factor larger the estimation in [4]. Together, the $i \geq 3$ terms are dominated by $i = 3$ term with order $O(\frac{q^3\kappa^3}{\sigma^3})$. In sum, we have proved that $\alpha_{\boldsymbol{\mu}_0,\boldsymbol{\mu}}(\kappa) \leq \frac{4q^2\kappa(\kappa+1)}{(1-q)\sigma^2} + O(q^3/\sigma^3)$.

Theorem 5. *Let $\sigma^2 = \frac{16q^2T\ln(\frac{1}{\delta})}{\epsilon^2}$ and $q < \sqrt{\frac{\epsilon}{64\sqrt{T\ln(1/\delta)}}}$. Algorithm 5 satisfies (ϵ, δ)-class-DP.*

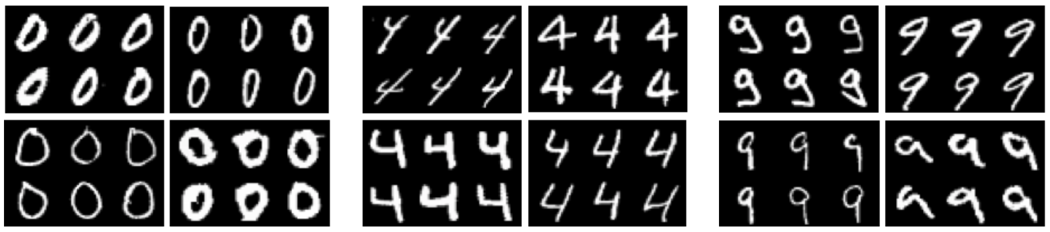
Let $\mathcal{A}_t(\mathcal{D}) := \sum_{i\in[k_t]} \frac{1}{qK}\left(\sum_{i=1}^{k_t} \overline{g}^{(i)} + \mathcal{N}(0, \sigma^2 S^2 \mathbf{I})\right)$, where each $\overline{g}^{(i)}$ is the S-clipped gradient computed based on the sampled class C_i and satisfies $\|\overline{g}^{(i)}\|_2 \leq S$. First, we upper bound $\alpha_{\mathcal{A}_t}(\kappa)$. For class neighboring datasets $(\mathcal{D}, \mathcal{D}')$, Without loss of generality, let $\mathcal{D} = \{C_1, ..., C_{K-1}, C_K\}$ and $\mathcal{D}' = \{C_1, ..., C_{K-1}, C'_K\}$, where each $C_k,\ k = 1, ..., K$, denotes all the data (records and label) in class k. The distribution of $\mathcal{A}_t(\mathcal{D}') \sim \mathcal{N}(\frac{1}{qK}\sum_{i=1}^{k_t} \overline{g}^{(i)}_{|\mathcal{D}'}, \frac{1}{(qK)^2}\sum_{i=1}^{k_t} \sigma^2 S^2 \boldsymbol{I})$, where $\overline{g}^{(i)}_{|\mathcal{D}'}$ denotes the stochastic gradient computed on $\mathcal{D}'$. It is equivalent to $\mathcal{A}_t(\mathcal{D}') \sim \sum_{i=1}^{k_t} \frac{1}{qK}\left(\overline{g}^{(i)}_{|\mathcal{D}'} + S \cdot \boldsymbol{\mu}_0\right)$, with $\boldsymbol{\mu}_0 \sim \mathcal{N}(0, \sigma^2)$, where $\overline{g}^{(i)}_{|\mathcal{D}'}$ is the clipped gradient computed based on $\mathcal{D}'$. For $\mathcal{A}_t(\mathcal{D})$, depending on whether the K-th class is sampled or not, the mean of $\mathcal{A}_t(\mathcal{D})$ is $\{(1-q)\sum_{i=1}^{k_t} \overline{g}^{(i)}_{|\mathcal{D}'}\} + \{q(\sum_{i=1}^{k_t} \overline{g}^{(i)}_{|\mathcal{D}'} - \overline{g}^{(K)}_{|C'_K} + \overline{g}^{(K)}_{|C_K})\}$. The arg max in eq.(2.2) is achieved when $\|\overline{g}^{(K)}_{|C^{K'}} - \overline{g}^{(K)}_{|C^K}\|_2 = 2S$, which gives $\mathcal{A}_t(\mathcal{D}) \sim \frac{1}{qK}\sum_{i=1}^{k_t} \overline{g}^{(i)}_{|\mathcal{D}'} + S \cdot ((1-q)\boldsymbol{\mu}_0 + q\boldsymbol{\mu}_2)$, with $\boldsymbol{\mu}_2 \sim \mathcal{N}(2, \sigma^2)$. Thus, to bound $\alpha_{\mathcal{A}_t}(\kappa)$, it suffices to estimate $\alpha_{\boldsymbol{\mu}_0,\boldsymbol{\mu}}(\kappa)$ which is given in Lemma 3. With the composition property in Lemma 1, we have $\alpha_{\mathcal{A}}(\kappa) \leq \sum_{t=1}^{T} \alpha_{\mathcal{A}_t}(\kappa) \leq \frac{4Tq^2\kappa^2}{\sigma^2}$. By Lemma 2, to ensure (ϵ, δ)-class-DP, it suffices to ensure $\frac{4Tq^2\kappa^2}{\sigma^2} \leq \frac{\kappa\epsilon}{2}$, $exp(-\frac{\kappa\epsilon}{2}) \leq \delta$. In addition, since having used Lemma 3, we need to satisfy its constraints: $q < \frac{1}{16\sigma}$, $\kappa \leq \sigma^2 \log(\frac{1}{q\sigma})$. With our choice of q and σ, we can verify that the above constraints hold. Finally, Algorithm 5 is (ϵ, δ)-class-DP.

Remark 1. *For unbounded class-DP (i.e. random deletion of a class), we can still provide similar privacy guarantee by following similar procedure as the proof for the bounded class-DP case above: we ensure that Algorithm 5 is (ϵ, δ)-class DP if $\sigma^2 = \frac{4q^2T\ln(\frac{1}{\delta})}{\epsilon^2}$ and $q < \sqrt{\frac{\epsilon}{32\sqrt{T\ln(1/\delta)}}}$.*

4.4.3 Subclass-Level Differential Privacy

While class-DP provides strong protection against MIA, it may require a large amount of noise when the number of classes is small, i.e. class size is large. Recall that in our privacy analysis we utilize moments accountant to provide tight privacy loss analysis for our class-DP-SGD algorithm. Moments accountant itself depends heavily on the privacy amplification via random sampling with a sampling ratio of q. The smaller the q, the better the amplification and the smaller the privacy loss. For datasets where the number of classes is small comparing to the number of data records, i.e. q will be large, achieving meaningful class-DP while preserving model accuracy may not be feasible.

Definition of Subclass-DP. To address this, we propose subclass-DP that defines the neighboring databases based on a subclass, a predefined subset of records within a single class. In many practical applications, there exist natural subclasses within a large class. Subclass-DP ensures the indistinguishability of the output model with respect to any subclass. It can be considered as a weaker version of class-DP. We show that it will allow better and customizable privacy and utility trade-off.



(a) Class of digit 0 (b) Class of digit 4 (c) Class of digit 9

Figure 4.3: Examples of Subclasses from the MNIST dataset.

Consider the image classification tasks that we focus on in this chapter, images with the same label in the dataset often exhibit different sub-patterns. For example, as shown in Figure 4.3 where images are from the MNIST dataset, each class of digit images can be naturally divided into different groups, and images within a group are more similar to each other than those from other groups.

Definition 9. *(Subclass Neighboring Datasets). Let $\mathcal{D}$ denote a dataset with K subclasses of records. The subclass neighboring datasets to $\mathcal{D}$ are datasets $\mathcal{D}'$ that can be obtained from $\mathcal{D}$ by replacing all the records in an arbitrary subclass $k \in 1, ..., K$.*

Definition 10. *(Subclass-Level Differential Privacy). A randomized algorithm $\mathcal{A}$ with domain $\mathbb{N}^{|\mathcal{X}|}$ satisfies subclass level (ϵ, δ)-differential privacy if for all $\mathcal{S} \subseteq \mathrm{Range}(\mathcal{A})$ and for all subclass neighboring datasets $\mathcal{D}$, $\mathcal{D}' \in \mathbb{N}^{|\mathcal{X}|}$:*

$$\Pr[\mathcal{A}(\mathcal{D}) \in \mathcal{S}] \leq e^{\epsilon} \Pr[\mathcal{A}(\mathcal{D}') \in \mathcal{S}] + \delta.$$

Subclass-DP can be considered as a generalized notion of record-DP and class-DP. When the number of subclasses in each class k_{sub} is 1, it is equivalent to class-DP. When k_{sub} is the number of records in each class, it is equivalent to record-DP. The relationship between subclass-DP and group-DP is the same as that between class-DP and group-DP, except that subclass-DP corresponds to smaller group size.

Algorithm for Subclass-DP. The algorithm for Subclass-DP is the same as the class-DP algorithm (Algorithm 5) except that we sample random subclasses instead of random class (line 3) and the average gradient is computed within each sampled subclass (line 7). This additional subclass-based sampling provides additional privacy amplification which promises better privacy and utility trade-off. In this chapter, we form subclasses using k-means clustering algorithm with a predefined number of clusters k_{sub} to mimic the natural subclasses.

Privacy Analysis. The privacy analysis for the subclass-DP algorithm is inherited from Theorem 5 and it's proof by switching the class-level related notion to the subclass-level ones. We summarize the subclass-DP guarantee in the following corollary while omitting its detailed proof.

Corollary 2. *Let* $\sigma^2 = \frac{16q^2 T \ln(\frac{1}{\delta})}{\epsilon^2}$ *and* $q < \sqrt{\frac{\epsilon}{64\sqrt{T\ln(1/\delta)}}}$. *Algorithm 5 with subclass sampling is* (ϵ, δ)*-subclass differentially private.*

4.5 Experiments

We evaluate the proposed class and subclass DP-SGD algorithms on MNIST [62] and Faces94[1] to demonstrate their effectiveness in defending against MIA while preserving good model utility. MNIST contains 60,000/10,000 training/test examples which are gray-scale handwritten digit images with the size 28 $\times$ 28. Faces94 is a facial image dataset with 153 individuals and each has 20 color facial images with the size 180 $\times$ 200. We convert color images into grayscale and rescale them to 60 $\times$ 70. We use image augmentation techniques[2] to create additional facial images such that each individual has 220 images. We then randomly divide the training/test set into 190/30. We use a vanilla model without privacy protection and a model with record-DP as baseline comparisons.

4.5.1 MNIST

A convolutional neural network (CNN) with two convolution layers followed by two fully connected layers is used. We train it without DP protection as the **vanilla model** and the test accuracy reaches 98.9%. We train the same CNN models with record-DP using the DP-SGD method proposed in [4] as the **record-DP model**. We use three choices of noise scale for the Gaussian noise which are $\sigma = 0.65, 1.0, 1.8$, and obtain three models with test accuracy of 96%, 93%, and 91% and corresponding privacy loss of $(6, 10^{-5})$, $(1.6, 10^{-5})$, and $(0.5, 10^{-5})$-DP respectively. Finally, we train the same CNN model using the subclass-DP-SGD algorithm (Algorithm 5 with random subclass sampling) as the **subclass-DP model**. The reason we use subclass-

[1]https://cswww.essex.ac.uk/mv/allfaces/faces94.html
[2]https://github.com/aleju/imgaug

DP instead of class-DP is that the number of classes is small for the MNIST dataset which will make class-DP not meaningful. We will evaluate class-DP on the Faces94 dataset later in the section. Each class of digits in the MNIST dataset is divided into 50 subclasses using the k-means clustering algorithm (k=50). We set the subclass sampling ratio of q to be 0.2. We also use three choices of noise scale which are $\sigma = 1.6, 3.0, 3.65$. We choose a fixed gradient norm bound 3.0. We obtain three models with test accuracy of 96%, 93%, and 91% and corresponding privacy loss of $(18.6, 10^{-3})$, $(8.2, 10^{-3})$, and $(7, 10^{-3})$-subclass-DP respectively. Note that our criteria for the three subclass-DP models are to have matching accuracy with the three record-DP models. This way, we can have a fair comparison for each pair of record-level DP model and subclass-DP model at the same level of model accuracy in terms of their robustness to MIA.

MIA. We implement the improved MIA in Algorithm 1 with the same parameter setting and evaluate it against all models. The parameters of MIA are set as: $T = 5000$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\tau = 10^{-8}$, $\eta = 0.1$, and $\lambda = 0.05$. Figure 4.4 and 4.5 demonstrate the MIA results on the record-DP models and subclass-DP models respectively. The first row of each figure shows the ground truth training image samples of digit 0 to 9, and the second row shows the reconstructed images of MIA on the vanilla model. The third/fourth/fifth row of Figure 4.4 and 4.5 show the reconstructed images of MIA on the record-DP model and subclass-DP model for model accuracy at 96%, 93%, and 91% respectively. Notice that the record-DP model and the subclass-DP model at the same row in Figure 4.4 and 4.5 have the same model accuracy so we can have a fair comparison of their robustness. Comparing Figure 4.4 and Figure 4.5, we can see that record-DP can not defend against MIA. Even with small ϵ (row 5) which corresponds to $(0.5, 10^{-5})$-DP, MIA can still reconstruct the corresponding digits. On the other hand, subclass-DP provides strong protection against MIA which fails to reconstruct original training data representatives.

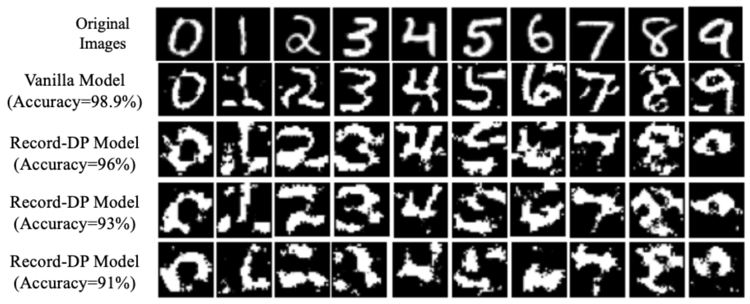

Original Images
Vanilla Model (Accuracy=98.9%)
Record-DP Model (Accuracy=96%)
Record-DP Model (Accuracy=93%)
Record-DP Model (Accuracy=91%)

Figure 4.4: MIA results on vanilla model and record-DP models trained on MNIST.

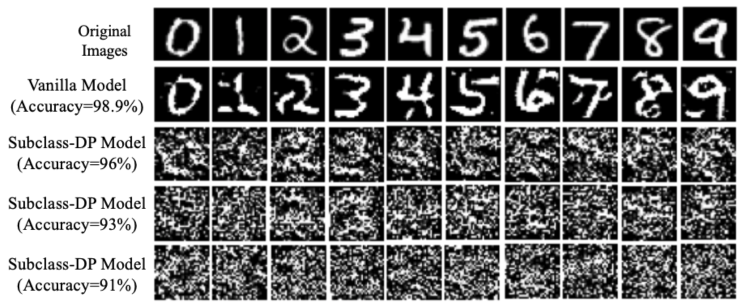
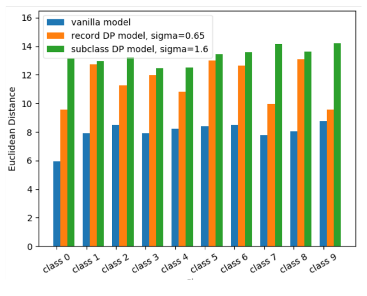
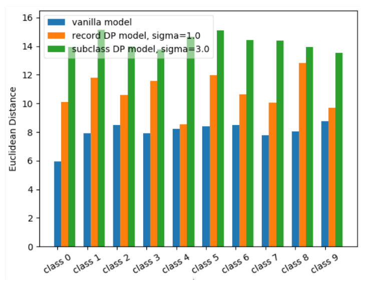

Original Images
Vanilla Model (Accuracy=98.9%)
Subclass-DP Model (Accuracy=96%)
Subclass-DP Model (Accuracy=93%)
Subclass-DP Model (Accuracy=91%)

Figure 4.5: MIA results on vanilla model and subclass-DP models trained on MNIST.

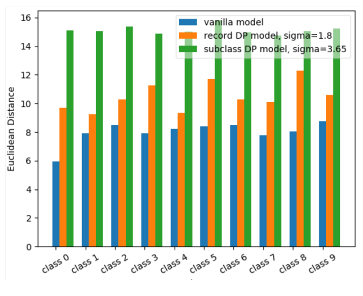
MIA Robustness. Figure 4.6 shows MIA robustness of subclass DP in comparison with vanilla model and record-level DP model in terms of the MIA distance (minimum Euclidean distance between the reconstructed image and training images in the target class) as defined in Section 3. We can see that record-DP models provide some protection against MIA compared to the vanilla model. Comparing the three figures, we observe that the subclass-DP models have stronger MIA robustness



(a) test accuracy=96%



(b) test accuracy=93%



(c) test accuracy=91%

Figure 4.6: MIA Robustness of record-DP and subclass-DP models trained on MNIST with different model utility.

(larger distance) than the record-DP models at the same level of model accuracy for all classes, providing more effective protection against MIA (a better MIA robustness and accuracy trade-off).

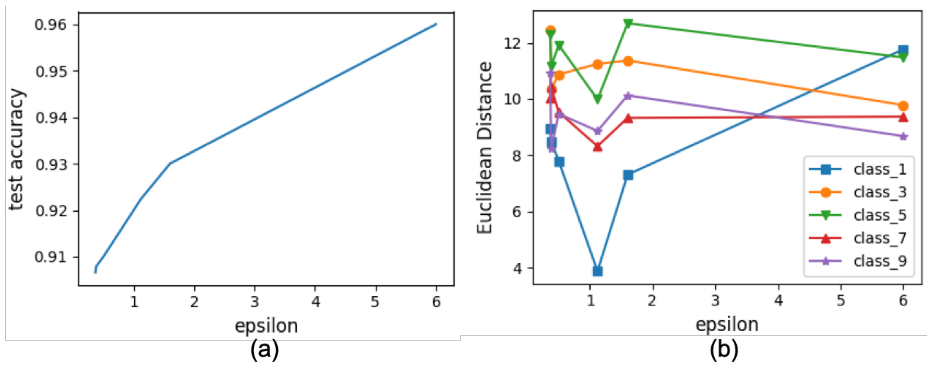


Figure 4.7: Record-DP models trained on MNIST: (a) ϵ vs test accuracy (b) ϵ vs MIA robustness

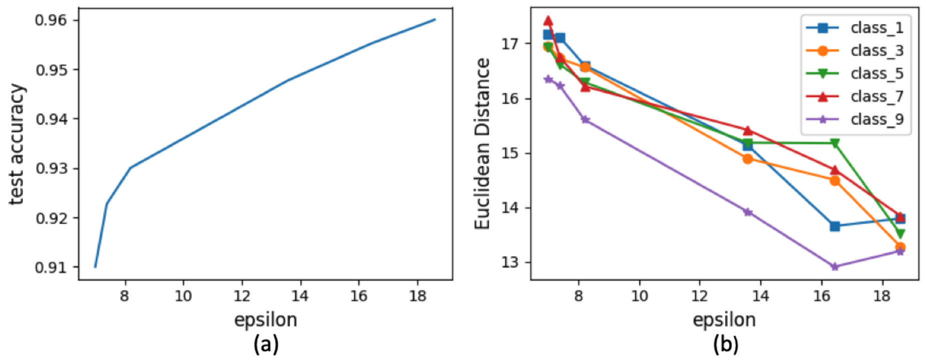


Figure 4.8: Subclass-DP models trained on MNIST: (a) ϵ vs test accuracy (b) ϵ vs MIA robustness

Figure 4.7 (4.8) shows (a) the relationship between ϵ and model utility measured by test accuracy, and (b) the relationship between ϵ and MIA robustness for record-DP (subclass-DP) models trained on MNIST with different noise scales and all other hyperparameters fixed. We note that the absolute value of epsilon and their comparison between record-DP and subclass-DP are not very meaningful. Instead, our goal is to adjust the epsilon for the two models to achieve the same range of accuracy so we can have a fair comparison of their MIA robustness (i.e. the trade-off of accuracy and

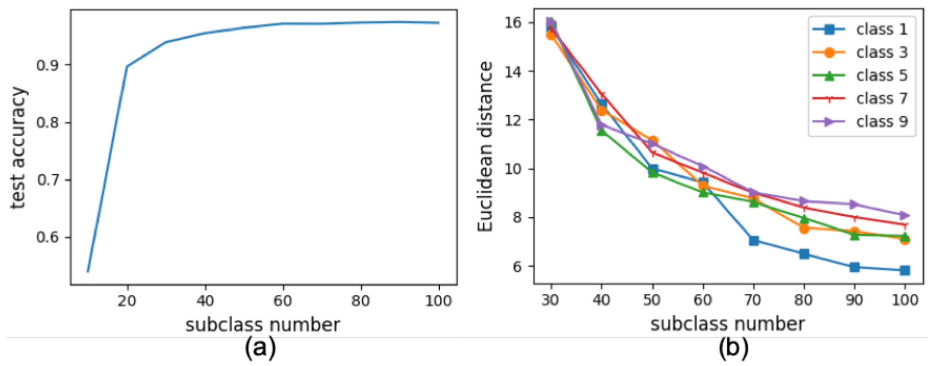


Figure 4.9: (a) Model test accuracy vs number of subclasses k_{sub}. (b) MIA robustness vs number of subclasses k_{sub}

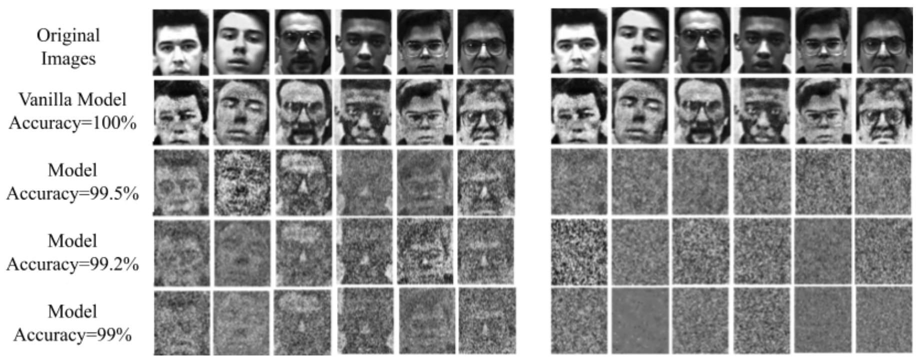
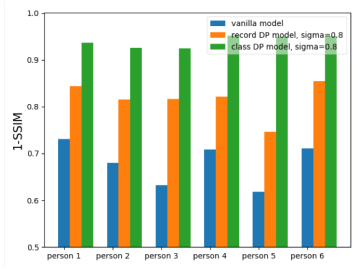
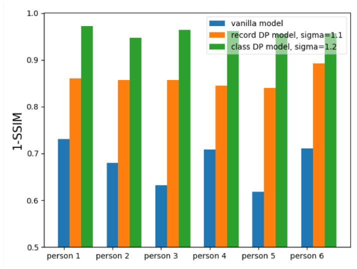


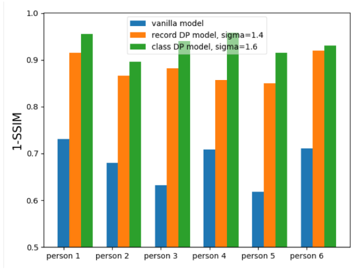
Figure 4.10: MIA results on vanilla model and record-DP (left) and class-DP (right) models trained on Faces94 dataset.



(a) test accuracy=99.5%



(b) test accuracy=99.2%



(c) test accuracy=99%

Figure 4.11: MIA Robustness of record-DP and class-DP models trained on Faces94 dataset.

MIA robustness). In addition, what is important is whether the epsilon value correlates with the MIA robustness (i.e. provides quantifiable protection against MIA). By comparing Figure 4.7(b) and Figure 4.8(b), we make two observations. First, subclass-DP has a much larger MIA distance than record-DP at the same accuracy

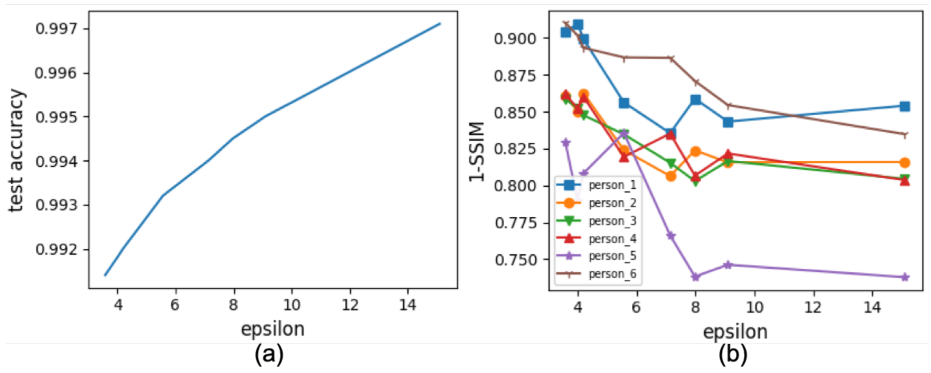


Figure 4.12: Record-DP models trained on Faces94 dataset: (a) ϵ vs test accuracy (b) ϵ vs MIA robustness

level, indicating a much stronger MIA robustness and accuracy trade-off. Second, the level of ϵ in record-DP models does not have any significant correlation with MIA robustness. On the other hand, the ϵ of subclass-DP models directly correlates with their MIA robustness, i.e. a smaller epsilon corresponds to more robustness (larger distance). Hence it validates our hypothesis that subclass-DP can provide a more effective and quantifiable measure against the model inversion risk.

Varying Number of Subclasses k_{sub}. Next, we study the impact of the number of subclasses k_{sub} within each class on subclass-DP models in terms of model utility and model robustness against MIA. Figure 4.9(a) shows the model test accuracy under different k_{sub} and (b) shows the MIA robustness under different k_{sub}. For both figures, the noise scale is fixed as 1.6 and the subclass sampling ratio is fixed as 0.2. We can observe that under the same noise scale and subclass sampling ratio, increasing k_{sub} will increase the model accuracy (becomes flat after k_{sub} is large enough), and decrease the MIA robustness. This utility and robustness trade-off is consistent with our definition of subclass-DP. When the k_{sub} is large enough, it will degrade to record-DP (when k_{sub} equals to the class size), and the model will be under higher risk of MIA.

4.5.2 Faces94

We use the softmax regression model as in [34] and train the **vanilla model** without privacy protecting with 100.0% test accuracy. We train **record-DP models** with the same structure using the DP-SGD algorithm [4]. We set three noise scales which are $\sigma = 0.8/1.1/1.4$ and obtain three models with 99.5%, 99.2%, and 99% test accuracy and corresponding $(9.1, 10^{-4})$, $(4.2, 10^{-4})$, and $(2.7, 10^{-4})$-DP respectively. Finally, we train **class-DP models** with the same architecture as the vanilla model using the class-DP-SGD method in Algorithm 5. The class sampling rate q is set to be 0.33. The gradient norm bound is 10. We choose three noise scales which are $\sigma = 0.8/1.2/1.6$ and obtain three models with 99.5%, 99.2%, and 99% test accuracy and corresponding $(62, 10^{-2})$, $(45.5, 10^{-2})$, and $(40.5, 10^{-2})$-class DP respectively.

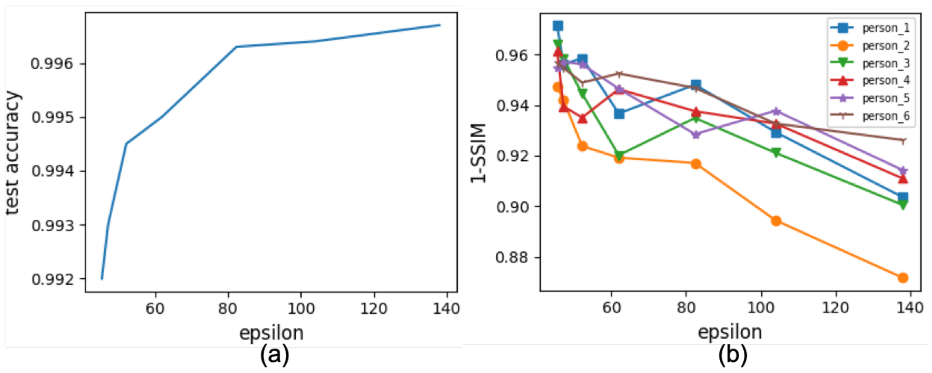


Figure 4.13: Class-DP models trained on Faces94 dataset: (a) ϵ vs test accuracy (b) ϵ vs MIA robustness

MIA. We evaluate the improved MIA using Algorithm 1 with the loss function (6) where $\alpha = 0.9$ and $p = 2$. The parameter settings are the same for all the models to recover face images of each class (person). The parameters of MIA are set as: $T = 100$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\tau = 10^{-8}$, $\eta = 0.05$, and $\lambda = 0.001$. Figure 4.10 demonstrates the MIA results on the record-DP models and class-DP models respectively. Again, the record-DP model and the class-DP model in the same row have the same model accuracy so we can have a fair comparison. We can draw a similar conclusion to the

MNIST dataset that the record-DP model can not prevent the reconstruction of the images but class-DP does while achieving the same model accuracy.

MIA Robustness. Figure 4.11 shows MIA robustness of class DP in comparison with the vanilla model and record-level DP model in terms of the MIA distance. We observe that the class-DP models have stronger MIA robustness than the record-DP models at the same level of model accuracy for all classes, providing more effective protection against MIA.

Figure 4.12 and 4.13 show model accuracy and MIA robustness with respect to varying ϵ for record-DP and class-DP respectively. Again, we emphasize that the ϵ value for class-DP may seem significantly large, the absolute value is not very meaningful. What is important is that at the same model accuracy level, class-DP achieves significantly larger MIA distance than record-DP, which means a much more effective MIA protection. Similar to MNIST, we also observe that the level of ϵ in record-DP models do not have a significant correlation with the robustness of the model against MIA, while the ϵ of class-DP models directly correlates with their robustness against MIA.

4.6 Related Work

[34] first studied MIA targeting neural network models to recover recognizable facial images of individual's portrait by their names and white-box access to the model parameters. [119] trained an inversion model using an auxiliary dataset composed of the adversary's background knowledge to recover the private dataset. Their attack is different from our improved model inversion attack since they require an auxiliary dataset in order to train an additional model to implement the inversion attack. Our MIA requires no additional datasets and models. However, how to use auxiliary datasets to further improve MIA to generate more recognizable images close to the

original one is an interesting topic for further work. There are also works exploring MIA under the distributing setting [46, 92, 107, 130] which are orthogonal to our improved MIA in a centralized setting. [105] proposed to protect MIA by introducing a regularizer into the training loss to mitigate the mutual information between the model input and the prediction. However, their method does not provide a rigorous or quantifiable guarantee against MIA as we do in this chapter.

As for broadened DP notions, [14] protected "user-level" DP for user-partitioned data when training an LSTM language model with a strong DP guarantee. [36] proposed "client-level" DP, which can be achieved in the federated setting along with good model utility when the number of clients is large enough. The class-level DP coincides with user-level and client-level DP [14,36] when one class corresponds to one user (client). However, class and user are two orthogonal concepts, e.g. one class may not directly correspond to one user and each user can have data of multiple classes. In this sense, the class-level DP and user-level DP are not the same for most of the cases. In terms of the targeted problems, our purpose of defending against MIA is very different from [14, 36], which considers the privacy issues for federated training. An important contribution of our work is to show that DP-based techniques can be applied against MIA with proper development, in spite of the previous unsuccessful attempt [46] which applied record-level DP straightforwardly.

4.7 Conclusion

We study the problem of protecting deep learning models against MIA. We show that traditional record-DP for building private deep learning models does not provide effective and quantifiable protection against MIA. Further, we propose two new DP notions, class-DP and subclass-DP, and algorithms for protecting deep learning models against MIA. Experiments show that class or subclass-DP can effectively defend against MIA while preserving good model utility. While we focus on the centralized setting and neural networks in this chapter, the class-DP and subclass-DP notions are generally applicable to other machine learning settings (e.g. collaborative setting) and models (e.g. decision trees) to protect against MIA, and we leave the evaluation of them as future works.

Chapter 5

Achieving Node-Level Differential Privacy for GNN Models

5.1 Introduction

Neural networks have achieved great success in many real-world tasks [16, 63, 67, 82]. Recently, there is a growing interest in transferring the success of neural networks on image and text domain to the graph-structured data such as knowledge graphs, social networks, biological networks and molecular structures. Among these efforts, Graph Neural Networks (GNNs) demonstrate the superior performance in mining graph data and learning graph representations for downstream inference tasks including node classification, link predication, community detection and graph classification [13, 39, 58, 68, 112, 129]. Similar to the privacy concerns that neural network models trained on private datasets could expose sensitive information of the training data, GNN models trained on graph datasets that embed both the node features and graph topology information are also subject to different types of privacy attacks [43, 81].

In this chapter, we aim to ensure rigorous node-level DP for training GNN models. In other words, we want to prevent adversaries from distinguishing two neighboring graphs differing in one node and all of its connecting edges through the model trained on one of the graphs. Unlike grid-based data such as image, audio, and text, the graph data contains both feature vectors for each node and the topological structure which is usually indicated by the adjacency matrix. During the training of GNN models, the topological structure will be exploited by the recursive message-passing procedure that propagates information through the graph [39,101,112,117,129]. Each node receives hidden representations or features from its neighbors when generating its representation. This correlation between nodes makes the existing methods for training DP neural networks [4, 84–86, 120] ineffective in achieving the node-level DP in GNN models. The challenge is that the assumption in those works that instances in the training data are independent does not hold for graph data. The sensitivity of the model parameters due to presence or absence of each node becomes very large due to the correlation from their connecting edges, making the required DP noise for the same level of privacy guarantee to be too large to retain the model utility.

In order to train a GNN model with good performance while ensuring rigorous node-level DP, we decouple the message-passing process from feature aggregation of GNNs as proposed in [13, 59]. We develop two DP approximate personalized PageRank (DP-APPR) algorithms to obscure the graph topology information and decorrelate the nodes during the model training, which limits the sensitivity due to each node and facilitates DP-SGD to play its role in protecting nodes' existence information exposed by features. More specifically, we pre-compute the approximate personalized PageRank (APPR) [7,48,109] for each training node and exploit the PageRank vector to aggregate information from other nodes when generating the node representation. The graph structure is embedded into the PageRank matrix, and the message-passing process based on the graph is replaced by the feature transformation weighted by the PageRank vectors [13, 59]. We propose two algorithms for achieving DP-APPR to hide the graph structure information of each node embedded in the PageRank vector. The first algorithm uses a PageRank vector clipping strategy to bound the global sensitivity and adds Gaussian noise to each bounded PageRank vector using the Gaussian mechanism [28]. We use the optimal composition theorem [52, 53] to bound the total privacy loss for the PageRank matrix. Second, we point out that the DP-APPR computing process can be regarded as the DP top-K selection problem [25] because each node only needs to aggregate information from top-K most relevant neighbors, which are pointed by the top-K largest elements in the PageRank vector. Typically, the exponential mechanism [28] is the algorithm used to solve the DP top-K selection problem [25, 28]. Consequently, we propose the second algorithm of DP-APPR using the exponential mechanism with Gumbel distributed noise to first more accurately select the top-K elements without the actual values, and then report the corresponding noisy values with additional privacy costs. After we get the DP APPR matrix for training nodes, we utilize DP-SGD [4] during model training to provide further privacy protections for nodes' features whose information will be embedded into the model parameters after the model is trained. We formally analysis the privacy loss caused by the sampling process dependent on the DP PageRank results during private model training using DP-SGD. Correspondingly, we calibrate tighter Gaussian noise for the clipped gradients to provide a rigorous overall privacy guarantee for learning GNN models.

5.2 Preliminaries

This section briefly introduces the background knowledge of Graph Neural Networks, GNN models based on approximate personalized PageRank, and differential privacy.

5.2.1 Graph Neural Networks

Given a graph $G = (\mathrm{V}, \mathrm{E}, \mathrm{X})$, where V and E denote the set of vertices and edges, respectively, and $\mathrm{X} \in \mathbb{R}^{|\mathrm{V}| \times d}$ represents the feature matrix where each row corresponds to the associated feature vector $X_v \in \mathbb{R}^d$ $(v = 1, ..., |\mathrm{V}|)$ of the node v. A GNN model learns a representation function f that generates the node embedding h_v for each node $v \in \mathrm{V}$ based on the features of itself as well as all its neighbors [112, 129]. We use $N(v) = \{v\} \cup \{u \in \mathrm{V} \mid (v, u) \in \mathrm{E}\}$ to denote the node set containing node v and all its immediate (1-hop) neighbors. The hidden representation of node v learned by the k-th $(k = 1, ..., K)$ layer of a GNN is denoted as $h_v^{(k)}$, where $h_v^{(0)}$ indicates the node feature X_v. The neighborhood aggregation procedure (or message passing) of GNN models can generically be expressed as:

$$h_v^{(k)} = \sigma \left(W_k \cdot \text{ AGGREGATE } \left(\left\{h_u^{(k-1)}, \forall u \in N(v)\right\}\right)\right) \tag{5.1}$$

where W_k is the trainable linear transfer matrix of the GNN model on the k-th layer, σ is a non-linear activation function such as ReLU, and AGGREGATE is an aggregation function defined by the specific model [39, 58, 101, 116, 117] to integrate the representations of a node and its neighbors from the previous layer to generate the representation of the node in the current layer. After k layers of aggregation, the representation of a node captures the feature information of its k-hop neighbors based on graph structure.

5.2.2 Decoupling GNNs with Personalized PageRank

GNN models use the recursive message-passing procedure to spread information through a graph, which couples the neighborhood aggregation and feature transformation for node representation learning. In the original design of GNN models based on message-passing, each additional neighborhood propagation step requires adding an extra layer to the model. This coupling pattern can cause some potential issues in model training, including neighbors explosion and over-smoothing [13,20,23,59,66,68, 111]. Recent works propose to decouple the neighborhood aggregation process from feature transformation and achieve superior performance [13,20,23,59,68]. Bojchevski et al. [13,59] show that neighborhood aggregation/propagation based on personalized PageRank [37] can maintain the influence score of "neighboring" (relevant) nodes in infinitely many hops away from the source node, without explicit message-passing procedure. They pre-compute a sparse matrix $\mathbf{\Pi}$ and use it to aggregate node representations generated using a neural network (e.g., a multilayer perceptron) to get final predictions, which is expressed as follows:

$$\mathbf{Z} = \operatorname{softmax}\left(\mathbf{\Pi}\boldsymbol{H}\right), \quad \boldsymbol{H}_{v,:} = f_\theta\left(X_v\right) \tag{5.2}$$

where $\boldsymbol{H}_{v,:}$ is the node representation generated by a neural network f_θ using the node feature vector X_v of each node v independently. They use an approximate personalized PageRank (APPR) calculation algorithm proposed by Andersen et al, [7] to compute an APPR matrix $\mathbf{\Pi}^{ppr}$ where each row $\boldsymbol{\pi}(v)$ is equal to the APPR vector of one node v. $\mathbf{\Pi}^{ppr}$ encodes the graph structure information, where each entry (v, u) indicates the relevance or importance of node u corresponding to the source node v. They further truncate $\mathbf{\Pi}^{ppr}$ by keeping only the top k largest entries of each row and setting all other small elements to zero to get a sparse matrix $\mathbf{\Pi}$. The predictions of node v with using $\mathbf{\Pi}$ for aggregating information from "neighbors" (most relevant nodes) can be expressed as:

$$z_i = \operatorname{softmax}\left(\sum_{u \in \mathcal{N}^k(v)} \boldsymbol{\pi}'(v)_u H_{u,:}\right) \tag{5.3}$$

where $\mathcal{N}^k(v)$ enumerates indices of the k non-zero entries in $\boldsymbol{\pi}'(v)$ which is the v-th row of $\mathbf{\Pi}$ corresponding to the node v's sparse approximated personalized PageRank vector.

5.3 Differentially Private Graph Neural Network

In this section, we first highlight the weakness of applying DP-SGD directly in training GNN models with node-level DP protection due to the correlation between nodes. Then we present the details of our approach for training DP GNN models using DP approximate personalized PageRank. The purpose of using approximate personalized PageRank is to decouple the message passing from feature aggregation and to enable each node to generate its representation independently, which facilitates DP-SGD to play its role in providing DP protection. The overall privacy budget will be split into two parts. One is spent on injecting noise during DP-SGD process to conceal nodes' existence information exposed by nodes' features. The other is used to compute DP approximate personalized PageRank vectors to hide nodes' existence information disclosed by nodes' edges. Note that only the graph structure information (edges) is used during the calculation of PageRank vectors.

5.3.1 Differentially Private SGD

DP-SGD [4,99] first computes the gradient $\mathbf{g}\left(x_i\right)$ for each example x_i in the randomly sampled batch with size equals to B, and then clips the l_2 norm of each gradient with a clipping threshold C to bound the sensitivity of $\mathbf{g}\left(x_i\right)$. After that, the clipped gradient $\overline{\mathbf{g}}\left(x_i\right)$ of each example will be summed together and added with the Gaussian noise $\mathcal{N}\left(0, \sigma^2 C^2 \mathbf{I}\right)$ to protect privacy. Finally, the average of the noisy accumulated gradient $\tilde{\mathbf{g}}$ will be used to update the model parameters for this step. We express $\tilde{\mathbf{g}}$

as:

$$\tilde{\mathbf{g}} \leftarrow \frac{1}{B}\left(\sum_{i=1}^{B} \overline{\mathbf{g}}\left(x_i\right) + \mathcal{N}\left(0, \sigma^2 C^2 \mathbf{I}\right)\right) \tag{5.4}$$

In DP-SGD, at each step, each example individually calculates its gradient, e.g., only the features of x_i will be used to compute the gradient $\mathbf{g}\left(x_i\right)$ for the example x_i. Note that we do not consider the features of other examples embedded previously in the model parameters since noise is added at each training step. In Equation 5.4, according to the Gaussian mechanism [28], the Gaussian noise is calibrated as $\mathcal{N}\left(0, \sigma^2 C^2 \mathbf{I}\right)$ where the sensitivity is C, and each step is (ϵ, δ)-DP with respect to the batch if $\sigma = \sqrt{2 \log \frac{1.25}{\delta}}/\varepsilon$. Considering two neighboring datasets that differ in one example, since one example will only affect at most one gradient in $\sum_{i=1}^{B} \overline{\mathbf{g}}\left(x_i\right)$, and each gradient is clipped to have the maximum l_2 norm equals to C, therefore, the l_2 sensitivity of $\sum_{i=1}^{B} \overline{\mathbf{g}}\left(x_i\right)$ is C.

However, when considering training GNN models, nodes are no longer independent, and one node's feature will affect the gradients of other nodes. This correlation between nodes during gradient computation and model updating is introduced by the neighborhood aggregation mechanism as indicated in Equation 5.1. In a GNN model with K layers, one node has the chance to utilize additional features from all its neighbors up to K-hop when calculating its gradient. The neighborhood aggregation or message-passing process enables each node to consider both the local and global information during the representation generation, and empowers GNNs to encode graph structure information into the trained model parameters. Rethinking Equation 5.4, when training GNN models, at each step, the sensitivity of $\sum_{i=1}^{B} \overline{\mathbf{g}}\left(x_i\right)$ becomes $B \times C$ since changing one node in the graph could potentially change the value of all gradients in $\sum_{i=1}^{B} \overline{\mathbf{g}}\left(x_i\right)$ and each gradient is clipped the l_2 norm to C.

Substitute $B \times C$ for C in Equation 5.4 and we get the following equation:

$$\tilde{\mathbf{g}'} \leftarrow \frac{1}{B}\left(\sum_{i=1}^{B} \overline{\mathbf{g}}\left(x_i\right) + \mathcal{N}\left(0, \sigma^2 B^2 C^2 \mathbf{I}\right)\right) \tag{5.5}$$

If we choose σ in Equation 5.5 to be $\sqrt{2 \log \frac{1.25}{\delta}}$, then each step is still (ϵ, δ)-DP with respect to the batch by the Gaussian mechanism [28]. Comparing Equation 5.5 to Equation 5.4, we can see that to achieve the same level of privacy protection at each step during DP-SGD, the standard deviation of the Gaussian noise added to the gradients is scaled up by a factor of the batch size B, which will seriously affect the accuracy of the final model.

To facilitate DP-SGD to play its role during training GNN models, we replace the message-passing process based on the graph topology with the feature transformation weighted by the PageRank vector [13, 59]. The idea is to add DP noise to the PageRank vector to obscure the graph topology information and decorrelate the nodes during the model training, which limits the sensitivity due to each node on the model parameters.

5.3.2 Differentially Private Approximate Personalized PageRank

PageRank algorithm was initially developed by Brin and Page [15,83] for ranking the importance of website pages and used by the Google Search engine. Consider a graph with the adjacency matrix A and the degree matrix D, PageRank is equivalent to solving Equation 5.6 as in the following:

$$\mathbf{p} = \alpha \mathbf{s} + (1 - \alpha) W \mathbf{p} \tag{5.6}$$

where $\alpha \in (0,1]$ is called the *teleportation constant*, $\mathbf{s}$ is called the *preference vector* which is a given distribution over the nodes, and W is the random walk matrix $W = AD^{-1}$ [83] or the lazy random walk matrix $W = \left(I + AD^{-1}\right)/2$ [7,33]. Personalized PageRank [37] uses a non-uniform probability distribution vector as $\mathbf{s}$ (such as the one-hot vector associated with the index of the source node) to compute the personalized PageRank vector for each node. Elements in a personalized PageRank vector reflect other nodes' relevance to the source node, where higher values indicate more importance to the source node. Andersen et al. [7] propose the first approximate personalized PageRank (APPR) which is adopted in [13, 59] to replace the explicit message-passing procedure for GNNs. Most recently, Fountoulakis et al. in their work [33] demonstrate that the APPR algorithm can be characterized as an l_1-regularized optimization problem, and propose an iterative shrinkage-thresholding algorithm (ISTA) to solve it with a running time independent of the size of the graph. This algorithm serves as a basis for our DP approximate personalized PageRank algorithms.

We provide the pseudo-code of ISTA in Algorithm 6, which is adopted from Algorithm 3 in [33] with substituting $D^{-1/2}\mathbf{p}_k$ for $\mathbf{q}_k$ to update $\mathbf{p}_k$ directly during each step k of optimization, for the convenience of DP design. The output $\mathbf{p}_k$ is the computed APPR vector for the source node v (the v-th element in the one-hot indicator vector $\mathbf{s}$ equals 1). In Algorithm 6, $\mathbf{e}$ is the vector of all ones, $i \sim j$ means node i is a neighbor of node j, $i \sim S$ means node i is a neighbor of at least one node in the node set S, d_i indicates the degree of node i, A_{ij} indicates the (i,j)-th element in the adjacency matrix A, $[\mathbf{p}]_i$ indicates the i-th element in the vector $\mathbf{p}$, and $I_S \in \mathbb{R}^{n\times|S|}$ is a matrix with the columns indexed by S are taken from the identity matrix $\mathbb{R}^{n\times n}$ where n is the number of nodes in the graph.

We develop our DP approximate personalized PageRank algorithms based on the ISTA in Algorithm 6. Our goal is to compute the APPR vector for each node while

Algorithm 6: ISTA (Algorithm 3 in [33])

```
1  Initialize: $\gamma \in (0,1), \alpha \in (0,1], \rho \in (0,1), \mathbf{p}_0 = 0, \mathbf{s}$ such that $\mathbf{e}^T\mathbf{s} = 1$ and
     $\mathbf{s} \geq \mathbf{0}$, set $\nabla f(\mathbf{p}_0) = -\alpha D^{-1}\mathbf{s}$.
2  while $\|\nabla f(\mathbf{p}_k)\|_\infty > (1+\gamma)\rho\alpha$ do
3      Set $S_k := \{i \in [n] \mid \mathbf{p}_k(i) - \nabla_i f(\mathbf{p}_k) \geq \rho\alpha\}$;
4      $\Delta\mathbf{p}_k := -(\nabla_{S_k} f(\mathbf{p}_k) + \rho\alpha)$ and $\mathbf{p}_{k+1}(S_k) = \mathbf{p}_k(S_k) + \Delta\mathbf{p}_k$;
5      for each $i \in S_k$ set do
6          $\nabla_i f(\mathbf{p}_{k+1}) = (1 - d_i^{-1})\nabla_i f(\mathbf{p}_k) - \rho\alpha d_i^{-1} - \frac{1-\alpha}{2}[I_{S_k}\Delta\mathbf{p}_k]_i d_i^{-1} -$
           $\frac{1-\alpha}{2d_i}\sum_{l\sim i, l\in S_k}\frac{A_{il}[I_{S_k}\Delta\mathbf{p}_k]_l}{d_l}$;
7      end
8      for each $j \notin S_k$ such that $j \sim S_k$ set do
9          $\nabla_j f(\mathbf{p}_{k+1}) = \nabla_j f(\mathbf{p}_k) - \frac{1-\alpha}{2d_j}\sum_{l\sim j, l\in S_k}\frac{A_{jl}[I_{S_k}\Delta\mathbf{p}_k]_l}{d_l}$;
10     end
11     for each $j \notin S_k$ such that $j \nsim S_k$ set do
12         $\nabla_j f(\mathbf{p}_{k+1}) = \nabla_j f(\mathbf{p}_k)$;
13     end
14     $k = k + 1$;
15 end
16 return $\mathbf{p}_k$
```

preserving the node-level DP corresponding to the graph. Given Algorithm 6, one possible method to achieve the goal is to inject DP noise to the gradient of $\mathbf{p}_k$ at each iteration and bound the accumulated privacy loss for the optimization process. There are two challenges. First, the total number of iterations is unknown in advance, and the iteration numbers may vary for each node. Therefore, given a fixed total privacy budget in advance, it is difficult to assign a budget to each node and each iteration in order to calibrate the corresponding DP noise. Second, it is hard to bound the influence of each node on $\Delta\mathbf{p}_k$. Even though the ISTA works in a localized manner and only accesses a small portion of nodes in the graph, the DP noise would still be too large according to the worse case analysis to preserve the algorithm's effectiveness in calculating useful APPR vectors. To tackle these challenges, we use output perturbation (perturbing $\mathbf{p}_k$ directly as versus its gradient at each iteration) to avoid noise calibration for each iteration and the norm clipping strategy to bound the sensitivity. We propose two methods based on the Gaussian mechanism and the exponential mechanism for output perturbation.

Gaussian Mechanism

For each node v, we first compute its APPR vector $\mathbf{p}_{(v)}$ using ISTA, then we clip the l_2 norm of $\mathbf{p}_{(v)}$ to be bounded by a constant value C_1 in order to bound the influence of each node on $\mathbf{p}_{(v)}$. We add the Gaussian noise calibrated as $\mathcal{N}(0, \sigma^2)$ where $\sigma = \sqrt{2\ln(1.25/\delta)}C_1/\epsilon$ to each element in the clipped $\mathbf{p}_{(v)}$ and get a noisy APPR vector $\tilde{\mathbf{p}}_{(v)}$. $\tilde{\mathbf{p}}_{(v)}$ satisfies (ϵ, δ)-DP by standard Gaussian mechanism property [28]. We further select the top K largest entries in $\tilde{\mathbf{p}}_{(v)}$ by setting all other entries with small values to zero and get a sparse vector $\tilde{\mathbf{p}}'_{(v)}$. We substitute $\tilde{\mathbf{p}}'_{(v)}$ for $\boldsymbol{\pi}'(v)$ in Equation 5.3 to aggregate features/representations from most relevant nodes of node v when generating node v's predictions during the GNN model's training. Furthermore, we need to consider the accumulated privacy loss when generating the APPR matrix $\mathbf{\Pi}$ with M rows corresponding to the APPR vectors of M nodes. We propose to use the optimal composition theorem [52, 53] to provide a tight overall privacy guarantee, as it yields the state-of-the-art bound for serial composition of privacy loss. We provide our DP approximate personalized PageRank algorithm using the Gaussian mechanism in Algorithm 7.

Theorem 6. *Let $\epsilon > 0$ and $\delta \in (0, 1]$, Algorithm 7 is $(\epsilon_g, 2M\delta)$-differentially private where $\epsilon = \epsilon_g / \left(2\sqrt{M \ln\left(e + \epsilon_g/2M\delta\right)}\right)$.*

Proof. Our proof is mainly based on the optimal composition theorem in [53] which argues that for k sub-mechanisms, each with an (ϵ, δ)-DP guarantee, the overall privacy guarantee is (ϵ_g, δ_g)-DP, where $\epsilon = \epsilon_g/(2\sqrt{k\ln(e + \epsilon_g/\delta_g)})$ and $\delta = \delta_g/2k$. In Algorithm 7, the noisy APPR vector for each node satisfies (ϵ, δ)-DP by the Gaussian mechanism independently. Since the returned APPR matrix contains the noisy

Algorithm 7: DP-APPR using the Gaussian Mechanism (DP-APPR-GM)

```
Input: ISTA hyperparameters: $\gamma, \alpha, \rho$; privacy parameters: $\epsilon$, $\delta$; clip bound
       $C_1$, a graph $(V, E)$ where $V = \{v_1, ..., v_N\}$, an integer $K > 0$ and an
       integer $M \in [1, N]$.
1  Initialize the APPR matrix $\mathbf{\Pi} \in \mathbb{R}^{M \times N}$ with all zeros.
2  for $i = 1, ..., M$ do
3      Compute APPR Vector:
4      Compute the APPR vector $\mathbf{p}_{(v_i)}$ for node $v_i$ using Algorithm 6;
5      Clip Norm:
6      $\hat{\mathbf{p}}_{(v_i)} \leftarrow \mathbf{p}_{(v_i)} / \max\left(1, \frac{\|\mathbf{p}_{(v_i)}\|_2}{C_1}\right)$ ;
7      Add Noise:
8      $\tilde{\mathbf{p}}_{(v_i)} \leftarrow \hat{\mathbf{p}}_{(v_i)} + \mathcal{N}(0, \sigma^2 \mathbf{I})$, where $\sigma = \sqrt{2\ln(1.25/\delta)} C_1/\epsilon$;
9      Sparsification:
10     $\tilde{\mathbf{p}}'_{(v_i)} \leftarrow$: select the top $K$ largest entries in $\tilde{\mathbf{p}}_{(v_i)}$ by setting all other
         entries with small values to zero.
11     Replace the $i$-th row of $\mathbf{\Pi}$ with $\tilde{\mathbf{p}}'_{(v_i)}$.
12 end
13 return $\mathbf{\Pi}$ and compute the overall privacy cost using the optimal
     composition theorem.
```

APPR vectors of M nodes, thus the number of components for composition is M. We substitute M for k and $2M\delta$ for δ_g, which can conclude the proof. □

Exponential Mechanism

Note that $\tilde{\mathbf{p}}'_{(v)}$ in Algorithm 7 is a sparse vector which only contains the top K largest entries in the noisy vector $\tilde{\mathbf{p}}_{(v)}$ satisfying (ϵ, δ)-DP. Therefore, we can also consider the DP sparse APPR vector computation as a DP top-K selection problem [25]. Recall the purpose of calculating APPR vectors is to utilize them to aggregate representations from relevant nodes for the source node during model training. The index of each entry in an APPR vector indicates the index of the same node in the graph, and the value of each entry reflects the importance or relevance of this node to the source node. By reserving the top K largest entries for each APPR vector, it is equivalent to compute a weighted average of the representations of the K most relevant nodes to the source node. The graph structure information is encoded in both the indexes and values of non-zero entries in each sparse APPR vector. Therefore, in order to provide DP protection for the graph structure information during computing APPR vectors, we use a two-stage process with the exponential mechanism[1] to first select the top K largest entries of each APPR vector with DP, followed by an optional Laplacian mechanism [28] to get the noisy values for the top K entries with DP.

Algorithm 8: DP-APPR using the Exponential Mechanism (DP-APPR-EM)

```
Input: ISTA hyperparameters: $\gamma, \alpha, \rho$; privacy parameters: $\epsilon$, $\epsilon_2$, $\delta$; clip
       bound $C_2$, a graph $(V, E)$ where $V = \{v_1, ..., v_N\}$, an integer $K > 0$
       and an integer $M \in [1, N]$.
1  Initialize the APPR matrix $\mathbf{\Pi} \in \mathbb{R}^{M \times N}$ with all zeros.
2  for $i = 1, ..., M$ do
3      Compute APPR:
4      Compute the APPR vector $\mathbf{p}_{(v_i)}$ for node $v_i$ using Algorithm 6;
5      Clip Norm:
6      $\hat{\mathbf{p}}_{(v_i)} \leftarrow$: for each entry $\mathbf{p}_{(v_i)}[j], j \in [1, ..., N],$ in $\mathbf{p}_{(v_i)}$, set $\mathbf{p}_{(v_i)}[j] =$
         $\max\left(1, \frac{\|\mathbf{p}_{(v_i)}[j]\|_1}{C_2}\right)$
7      Add Noise:
8      $\tilde{\mathbf{p}}_{(v_i)} \leftarrow \hat{\mathbf{p}}_{(v_i)} + Gumbel(\beta \mathbf{I})$, where $\beta = C_2/\epsilon$;
9      Report Noisy Indexes:
10     $\mathbf{N}_K \leftarrow$: select the indexes of the top $K$ entries with the largest values in
         $\tilde{\mathbf{p}}_{(v_i)}$;
11     Report Noisy Values:
12     option 1: $\tilde{\mathbf{p}}'_{(v_i)} \leftarrow$: set $\hat{\mathbf{p}}_{(v_i)}[j]$, $j \in \mathbf{N}_K$, to be $1/K$, and other entries to be
         0;
13     option 2: $\tilde{\mathbf{p}}'_{(v_i)} \leftarrow$: set $\hat{\mathbf{p}}_{(v_i)}[j]$, $j \in \mathbf{N}_K$, to be $\hat{\mathbf{p}}_{(v_i)}[j] + Laplace(KC_2/\epsilon_2)$,
         and other entries to be 0;
14     Replace the $i$-th row of $\mathbf{\Pi}$ with $\tilde{\mathbf{p}}'_{(v_i)}$.
15 end
16 return $\mathbf{\Pi}$ and compute the overall privacy cost using the optimal
     composition theorem.
```

We provide the DP approximate personalized PageRank algorithm using the exponential mechanism in Algorithm 8. For each node v, we first compute its APPR

[1]The Sparse Vector Technique (SVT) [27] can also be used in our case to select top K largest entries in each APPR vector with DP. We choose to use the exponential mechanism in our work since as suggested in [71], it performs better than the SVT for DP top-K selection under the non-interactive setting. Moreover, the SVT needs to choose a threshold T during the noisy checking process, which makes it less efficient, and a bad value of T would affect the utility of final results.

vector $\mathbf{p}_{(v)}$ using ISTA, then we clip the value of each entry in $\mathbf{p}_{(v)}$ to have the maximum value bounded by C_2 in order to restrict the influence of each node on each entry of one APPR vector. We use each entry's clipped value as its utility score since the magnitude of each entry indicates its importance (utility) and is used as the weight when aggregating the representation of the node indicated by the entry's index. We simulate the exponential mechanism by injecting a one-shot Gumbel noise to the clipped vector $\hat{\mathbf{p}}_{(v)}$ and then select the indexes of top K largest noisy entries [25]. After we get the indexes of the selected top K entries, we can either: option 1) set the values of all top K entries to be $1/K$, which means we consider the top K entries equally important to the source node, or option 2) spend some additional privacy budget to report the noisy values of the top K entries with DP. Given the same total privacy budget, option 1 has better chance to output indexes of the actual top K entries while losing the opportunity to reserve useful importance scores. In contrast, option 2 saves some privacy budget during selecting the indexes of actual top K entries and uses the saved budget for reporting the corresponding weights.

[25, 28] remarks that the exponential mechanism [28, 73] can be simulated by adding the Gumbel distributed noise $\mathit{Gumbel}(\Delta u/\epsilon)$ to each utility score $u(x, r)$ and then reporting the outcome with the largest noisy value, which satisfies 2ϵ-DP. The authors further present the privacy guarantee of the algorithm $\mathcal{M}^k_{\text{Gumbel}}(u)$ for reporting the indices of top k largest noisy values using the exponential mechanism and Gumbel noise, as in Corollary 3, which will be used in our proof for Theorem 7.

Corollary 3. *[25] $\mathcal{M}^k_{Gumbel}(u)$ adds the one-shot $Gumbel(\Delta(u)/\epsilon)$ noise to each utility score $u(x, r)$ and outputs the k indices with the largest noisy values. For any $\delta \geq 0$, $\mathcal{M}^k_{Gumbel}(u)$ is (ε', δ)-DP where*

$$\epsilon' = 2 \cdot \min\left\{k\epsilon, k\epsilon\left(\frac{e^{2\epsilon}-1}{e^{2\epsilon}+1}\right) + \epsilon\sqrt{2k\ln(1/\delta)}\right\}$$

Theorem 7. *For any $\epsilon > 0$, $\epsilon_2 > 0$ and $\delta \in (0, 1]$, let $\epsilon_1 =$
$2 \cdot \min\left\{K\epsilon, K\epsilon\left(\frac{e^{2\epsilon}-1}{e^{2\epsilon}+1}\right) + \epsilon\sqrt{2K\ln(1/\delta)}\right\}$, Algorithm 8 is $(\epsilon_{g_1}, 2M\delta)$-differentially*

private for option 1, and $(\epsilon_{g_2}, 2M\delta)$*-differentially private for option 2, where* $\epsilon_1 = \epsilon_{g_1}/\left(2\sqrt{M \ln(e + \epsilon_{g_1}/2M\delta)}\right)$ *and* $\epsilon_1 + \epsilon_2 = \epsilon_{g_2}/\left(2\sqrt{M \ln(e + \epsilon_{g_2}/2M\delta)}\right)$.

Proof. We first consider the privacy loss of outputting the noisy APPR vector $\tilde{\mathbf{p}}'_{(v_i)}$ for the node v_i in Algorithm 8. For each element in the APPR vector, we use its value as its utility score. Since each element is nonnegative and clipped by the constant C_2, the l_1 sensitivity $\Delta(u)$ of each element is equal to C_2. By adding the one-shot Gumbel noise *Gumbel*$(\beta\mathbf{I})$ where $\beta = C_2/\epsilon$ to the clipped APPR vector $\tilde{\mathbf{p}}(v_i)$, option 1 selects K indices with the largest noisy values and satisfies (ϵ_1, δ)-DP where $\epsilon_1 = 2 \cdot \min\left\{K\epsilon, K\epsilon\left(\frac{e^{2\epsilon}-1}{e^{2\epsilon}+1}\right) + \epsilon\sqrt{2K\ln(1/\delta)}\right\}$ according to Corollary 3. Option 2 uses Laplace mechanism [28] to report K selected noisy values. By adding Laplace noise *Laplace*$(\mathrm{KC_2/ffl_2})$ to each clipped element, option 2 costs an additional ϵ_2 privacy budget [28] since the l_1 sensitivity of each element is C_2, and satisfies $(\epsilon_1 + \epsilon_2, \delta)$-DP.

Now we consider the privacy loss of Algorithm 8 which outputs M noisy APPR vectors. We use the optimal composition theorem in [53] which argues that for k sub-mechanisms, each with an (ϵ, δ)-differential privacy guarantee, the overall privacy guarantee is (ϵ_g, δ_g), where $\epsilon = \epsilon_g/(2\sqrt{k\ln(e + \epsilon_g/\delta_g)})$ and $\delta = \delta_g/2k$. By substituting M for k and ϵ_1 / $\epsilon_1 + \epsilon_2$ (option 1/option 2) for ϵ, we get the privacy loss of Algorithm 8 with option 1 is $(\epsilon_{g_1}, 2M\delta)$, where $\epsilon_1 = \epsilon_{g_1}/\left(2\sqrt{M \ln(e + \epsilon_{g_1}/2M\delta)}\right)$, and the privacy loss of Algorithm 8 with option 2 is $(\epsilon_{g_2}, 2M\delta)$, where $\epsilon_1 + \epsilon_2 = \epsilon_{g_2}/\left(2\sqrt{M \ln(e + \epsilon_{g_2}/2M\delta)}\right)$. This completes the proof. □

5.3.3 Differentially Private GNNs

We show our overall approach for training a DP GNN model in Algorithm 9. The DP APPR matrix $\mathbf{\Pi}$ is pre-computed using Algorithm 7 or Algorithm 8. The loss function $\mathcal{L}(\theta, v_i)$ is the cross-entropy between the one hot vector of node v_i's true label and its prediction vector generated by Equation 5.3. Each column of $\mathbf{\Pi}$ is clipped to have the maximum l_1 norm to be τ, to bound the privacy loss of the algorithm (See Theorem 8 for reasoning). In experiments, we use a constant $\tau = 1$. During each step, we randomly sample a batch B from nodes having PageRank vectors. The features of at most $B \cdot K$ nodes will be loaded into memory for gradients computing. We compute the gradient for each node in B and clip it to have the maximum l_2 norm equals C. We add the Gaussian noise with sensitivity C to the summed gradient and update model parameters using the average noisy gradient to satisfy DP. Compare with Equation 5.5, by utilizing the DP APPR vectors to aggregate information, the sensitivity due to each node is scaled down by a factor of the batch size.

Algorithm 9: Differentially Private GNNs

```
Input: A training graph dataset $(V, E, X)$ where $V = \{v_1, ..., v_N\}$, a subset
       $V_M \subseteq V$ with size $M$, learning rate $\eta_t$, batch size $B$, training steps $T$,
       noise scale $\sigma$, gradient norm bound $C$, clip bound $\tau$, the DP APPR
       matrix $\mathbf{\Pi} \in \mathbb{R}^{M \times N}$ of $V_M$ satisfying $(\epsilon_{pr}, \delta_{pr})$-DP.
1  Initialize $\theta_0$ randomly.
2  for $j = 1, ..., N$ do
3      $\mathbf{\Pi}_{:,j} \leftarrow \mathbf{\Pi}_{:,j} / \max\left(1, \frac{\|\mathbf{\Pi}_{:,j}\|_1}{\tau}\right)$
4  end
5  for $t = 1, ..., T$ do
6      Take a random sampled batch $B$ from $V_M$.
7      Compute Gradient:
8      For each $i \in B_t$, compute $\mathbf{g}_t(v_i) \leftarrow \nabla_{\theta_t} \mathcal{L}(\theta_t, v_i)$.
9      Clip Gradient:
10     $\overline{\mathbf{g}}_t(v_i) \leftarrow \mathbf{g}_t(v_i) / \max\left(1, \frac{\|\mathbf{g}_t(v_i)\|_2}{C}\right)$.
11     Add Noise:
12     $\tilde{\mathbf{g}}_t \leftarrow \frac{1}{B}\left(\sum_i \overline{\mathbf{g}}_t(v_i) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})\right)$.
13     Update Parameters:
14     $\theta_{t+1} \leftarrow \theta_t - \eta_t \tilde{\mathbf{g}}_t$.
15 end
16 return $\theta_T$ and compute the overall privacy cost using the moments
     accountant.
```

The following theorem presents the DP analysis of Algorithm 9. An essential distinction between our algorithm and the original DP-SGD is that our neighborhood sampling returns a correlated batch of nodes features for stochastic gradient computation (i.e., the computation of $\mathbf{g}_t(v_i)$ requires the features of the neighboring nodes of node v_i, and node v_i accesses the fixed K nodes based on the PageRank vector), while the original DP-SGD uses the much simpler Poisson sampling. As a result, the privacy analysis of our algorithm is more involved, especially in terms of quantifying the privacy amplification ratio. We prove that the privacy amplification ratio is proportional to the maximum of the column-wise ℓ_1 norm of the PageRank matrix.

Theorem 8. *There exist constants c_1 and c_2 so that given probability $q = B/N$ and the number of steps T, for any $\epsilon_{sgd} < c_1 q^2 T$, Algorithm 9 is $(\epsilon_{sgd} + \epsilon_{pr}, \delta_{sgd} + \delta_{pr})$ -differentially private for any $\delta_{sgd} > 0$ if we choose*

$$\sigma \geq c_2 \frac{q\tau\sqrt{T \log(1/\delta_{sgd})}}{\epsilon_{sgd}}$$

Proof. Denote μ_0 the Gaussian distribution with mean 0 and variance 1. Assume $\mathbb{D}'$ is the neighboring feature dataset of $\mathbb{D}$, which differs at $i^\dagger$ such that $\mathbf{x}'_{i^\dagger} \neq \mathbf{x}_{i^\dagger}$. Without loss of generality, we assume $\nabla f(\mathbf{x}_i) = \mathbf{0}$, for any $\mathbf{x}_i \in \mathbb{D}$, while $\nabla f(\mathbf{x}'_{i^\dagger}) = \boldsymbol{e}_1$. Recall that the PageRank matrix is $\mathbf{\Pi}$, where $\mathbf{\Pi}_{i:}$ is the i-th row and the PageRank vector for node i, while $\mathbf{\Pi}_{:j}$ is the j-th column of $\mathbf{\Pi}$. In addition, we assume that $\|\mathbf{\Pi}_{:j}\|_1 \leq \tau$, for all $j = 1, ..., n$, and denote μ_τ the Gaussian distribution with mean τ and variance 1.

$$\begin{aligned}
\mathbb{E}[\mathcal{G}(\mathbb{D})] = [\frac{|\mathcal{B}|}{n} \sum_{j\neq i^\dagger, j\notin\mathcal{N}\left(i^\dagger\right)} G_j] + [\frac{|\mathcal{B}|}{n} \sum_{j\neq i^\dagger, j\in\mathcal{N}\left(i^\dagger\right)} G_j] + [\frac{|\mathcal{B}|}{n} G_i] \\
= [\frac{|\mathcal{B}|}{n} \sum_{j\neq i^\dagger, j\notin\mathcal{N}\left(i^\dagger\right)} \sum_{k\in\mathcal{N}(j)} \mathbf{\Pi}_{jk} \nabla f\left(\mathbf{x}_k\right)] \\
+ [\frac{|\mathcal{B}|}{n} \sum_{j\neq i^\dagger, j\in\mathcal{N}\left(i^\dagger\right)} \left(\sum_{k\in\mathcal{N}(j)\backslash i^\dagger} \mathbf{\Pi}_{jk} \nabla f\left(\mathbf{x}_k\right) + \mathbf{\Pi}_{ji^\dagger} \nabla f\left(\mathbf{x}_{i^\dagger}\right) \right)] \\
+ [\frac{|\mathcal{B}|}{n} \left(\sum_{k\in\mathcal{N}\left(i^\dagger\right)\backslash i^\dagger} \mathbf{\Pi}_{i^\dagger k} \nabla f\left(\mathbf{x}_k\right) + \mathbf{\Pi}_{i^\dagger i^\dagger} \nabla f\left(\mathbf{x}_{i^\dagger}\right) \right)],
\end{aligned} \tag{5.7}$$

which indicates $\mathcal{G}(\mathbb{D}) \sim \mu_0$.

$$\begin{aligned}
\mathbb{E}\left[\mathcal{G}\left(\mathbb{D}'\right)\right] &= [\frac{|\mathcal{B}|}{n}\sum_{j\neq i^\dagger, j\notin\mathcal{N}\left(i^\dagger\right)} G_j] + [\frac{|\mathcal{B}|}{n}\sum_{j\neq i^\dagger, j\in\mathcal{N}\left(i^\dagger\right)} G_j'] + [\frac{|\mathcal{B}|}{n}G_i'] \\
&= [\frac{|\mathcal{B}|}{n}\sum_{j\neq i^\dagger, j\notin\mathcal{N}\left(i^\dagger\right)}\sum_{k\in\mathcal{N}(j)} \mathbf{\Pi}_{jk}\nabla f\left(\mathbf{x}_k\right)] \\
&+[\frac{|\mathcal{B}|}{n}\sum_{j\neq i^\dagger, j\in\mathcal{N}\left(i^\dagger\right)}\left(\sum_{k\in\mathcal{N}(j)\backslash i^\dagger}\mathbf{\Pi}_{jk}\nabla f\left(\mathbf{x}_k\right) + \mathbf{\Pi}_{ji}\nabla f\left(\mathbf{x}_{i^\dagger}'\right)\right)] \\
&+[\frac{|\mathcal{B}|}{n}\left(\sum_{k\in\mathcal{N}\left(i^\dagger\right)\backslash i^\dagger}\mathbf{\Pi}_{i^\dagger k}\nabla f\left(\mathbf{x}_k\right) + \mathbf{\Pi}_{i^\dagger i^\dagger}\nabla f\left(\mathbf{x}_{i^\dagger}'\right)\right)] \\
&= \mathbb{E}[\mathcal{G}(\mathbb{D})] + \frac{|\mathcal{B}|}{n}\sum_{j=1}^{n}\mathbf{\Pi}_{ji^\dagger}\left(f\left(\mathbf{x}_{i^\dagger}'\right) - f\left(\mathbf{x}_{i^\dagger}\right)\right) \\
&= \mathbb{E}[\mathcal{G}(\mathbb{D})] + \frac{|\mathcal{B}|}{n}\left\|\mathbf{\Pi}_{:i^\dagger}\right\|_1 \\
&\leq \mathbb{E}[\mathcal{G}(\mathbb{D})] + \frac{|\mathcal{B}|}{n}\tau,
\end{aligned} \tag{5.8}$$

which indicates $\mathcal{G}\left(\mathbb{D}'\right) \sim \mu_0 + \frac{|\mathcal{B}|}{n}\mu_\tau$.

In the following, we quantify the divergence between $\mathcal{G}$ and $\mathcal{G}'$ by following the moments accountant paper, where we show that

$$\mathbb{E}\left[\left(\frac{\mu(z)}{\mu_0(z)}\right)^\lambda\right] \leq \alpha, \tag{5.9}$$

$$\text{and } \mathbb{E}\left[\left(\frac{\mu_0(z)}{\mu(z)}\right)^\lambda\right] \leq \alpha, \tag{5.10}$$

for some explicit α. To do so, the following is to be bounded for v_0 and v_1.

$$\mathbb{E}_{z\sim v_0}\left[\left(\frac{v_0(z)}{v_1(z)}\right)^\lambda\right] = \mathbb{E}_{z\sim v_1}\left[\left(\frac{v_1(z)}{v_0(z)}\right)^{\lambda+1}\right]. \tag{5.11}$$

Following [4], the above can be expanded with binomial expansion, which gives

$$\begin{aligned}&\mathbb{E}_{z\sim v_1}\left[\left(\tfrac{v_1(z)}{v_0(z)}\right)^{\lambda+1}\right]=\textstyle\sum_{t=0}^{\lambda+1}(\lambda+1)\mathbb{E}_{z\sim v_1}\left[\left(\tfrac{v_0-v_1(z)}{v_1(z)}\right)^{t}\right]\\&=1+0+T_3+T4+\dots\end{aligned}\tag{5.12}$$

Next, we bound T_3 by substituting the pairs of $v_0=\mu_0, v_1=\mu$ and $v_0=\mu, v_1=\mu_0$ in, and upper bound them, respectively.

For T_3, with $v_0=\mu_0, v_1=\mu$, we have

$$\begin{aligned}T_3&=\frac{(\lambda+1)\lambda}{2}\mathbb{E}_{z\sim\mu}\left[\left(\frac{\mu_0(z)-\mu(z)}{\mu(z)}\right)^2\right]\\&=\frac{(\lambda+1)\lambda}{2}\mathbb{E}_{z\sim\mu}\left[\left(\frac{q\mu_\tau(z)}{\mu(z)}\right)^2\right]\\&=\frac{q^2(\lambda+1)\lambda}{2}\int_{-\infty}^{+\infty}\frac{(\mu_\tau(z))^2}{\mu_0(z)+q\mu_\tau(z)}dz\\&\le\frac{q^2(\lambda+1)\lambda}{2}\int_{-\infty}^{+\infty}\frac{(\mu_\tau(z))^2}{\mu_0(z)}dz\\&=\frac{q^2(\lambda+1)\lambda}{2}\mathbb{E}_{z\sim\mu_0}\left[\left(\frac{\mu_\tau(z)}{\mu_0(z)}\right)^2\right]\\&=\frac{q^2(\lambda+1)\lambda}{2}\exp\left(\frac{\tau^2}{\sigma^2}\right)\\&\le\frac{q^2(\lambda+1)\lambda}{2}\left(\frac{\tau^2}{\sigma^2}+1\right)\\&\le\frac{q^2\tau^2(\lambda+1)\lambda}{\sigma^2},\end{aligned}\tag{5.13}$$

where in the last inequality, we assume $\frac{\tau^2}{\sigma^2}+1\le 2\frac{\tau^2}{\sigma^2}$, i.e., $\frac{\tau^2}{\sigma^2}\ge 1$. Thus, it requires $\sigma\le\tau$.

As a result,

$$\alpha_{\mathcal{G}}(\lambda)\le\frac{q^2\tau^2(\lambda+1)\lambda}{\sigma^2}+O\left(q^3\lambda^3/\sigma^3\right).\tag{5.14}$$

To satisfy

$$T\frac{q^2\tau^2\lambda^2}{\sigma^2}\le\frac{\lambda\epsilon_{sgd}}{2},\tag{5.15}$$

$$\exp\left(-\frac{\lambda\epsilon_{sgd}}{2}\right)\le\delta_{sgd},\tag{5.16}$$

we set

$$\epsilon_{sgd} = c_1 q^2 \tau^2 T, \tag{5.17}$$

$$\sigma = c_2 \frac{q\tau\sqrt{T\log(1/\delta_{sgd})}}{\epsilon_{sgd}}. \tag{5.18}$$

Given the input DP APPR matrix costs additional $(\epsilon_{pr}, \delta_{pr})$ privacy budget, we can conclude the proof by using the standard composition theorem of DP.

□

5.4 Experimental Results

We evaluate the proposed DP GNNs training method on three commonly used graph datasets: Cora-ML [12], PubMed [77], and the Microsoft Academic graph [95]. The characteristics of them are provided in Table 5.1.

Table 5.1: Dataset statistics

Dataset	Type	Classes	Features	Nodes	Edges
Cora-ML	Citation	7	2879	2995	8416
PubMed	Citation	3	500	19717	44324
MS Academic	Co-author	15	6805	18333	81894

Setup. To simulate the real-world situations where training nodes are assumed to be private and not publicly available, for each graph dataset, we split the nodes into a training set (80%) and a test set (20%) that are entirely separate from each other, which means the edges between nodes that belong to different sets will be deleted, and the nodes in the training set are not accessible during the inference phase. This is different from the common experimental settings in most GNNs papers [12, 13, 39, 59, 101, 117] where the edges between training nodes and test nodes are preserved, and information of training nodes (e.g., features) is assumed to be visible and can be utilized during testing. To increase the training speed in our experiments, we use a sampling rate q' to randomly sample nodes from the training sets to form a graph containing only the sampled nodes and their connected edges, which is used for training. This sampling process also brings a privacy amplification effect in our privacy guarantee by multiplying a factor of q' [10, 55]. Experiments are conducted on the server equipped with an Nvidia K80 GPU, a 6-cores Intel CPU, and 56 GiB RAM. All experiments are run for 10 independent trials and the mean values are reported.

Model and Parameter Settings. We use the same neural network model as in [13] for all datasets. The model is a two-layer feed-forward neural network with a hidden layer size equal to 32. We fix the training epochs to 200, the learning rate to 0.005, and the batch size to 60. The hyperparameters for ISTA are chosen as $\alpha = 0.25$, $\rho = 10^{-4}$ and $\gamma = 10^{-4}$ by a grid search. We report experimental results under different K values, which control the size of the effective neighborhood of a node. The sampling rates are set to $q' = 9\%/1\%/3\%$ for Cora-ML/PubMed/MS Academic datasets. Moreover, we set $M = 70/60/150$ for Cora-ML/PubMed/MS Academic datasets.

We use **GM**, **EM-v0**, and **EM-v1** to denote our proposed method in Algorithm 9 corresponding to different DP APPR algorithms, Algorithm 7, Algorithm 8 with option 1, and Algorithm 8 with option 2, respectively.

Baselines. We compare our proposed algorithms with three baseline methods. **DPSGD** denotes using ISTA to calculate the APPR matrix (without privacy cost), and then applying the DP-SGD with the noise calibration rule shown in Equation 5.5 to train the node-level DP model. **Features** indicates a baseline method that only uses each node feature as an independent input to train the GNN model and utilizes the original DP-SGD to achieve node-level DP. Note that **Features** is equal to the case where we use the one-hot label indicator vector as each node's APPR vector (i.e., no correlation with other nodes is used) in Algorithm 9. **Random** means we generate an APPR matrix of training nodes randomly (without privacy cost) and use it in Algorithm 9 to train the DP model. Besides, we use **Vanilla** to denote the model without privacy protection (e.g., computing the APPR matrix by ISTA and then using SGD to train the model) as a reference.

Inference Phase. As suggested in [13], instead of computing the APPR vectors for all testing nodes and generating predictions based on their APPR vectors, we use power iteration during inference, e.g.,

$$Q^{(0)} = H, \quad Q^{(p)} = (1-\alpha)D^{-1}AQ^{(p-1)} + \alpha H, p \in [1, ..., P]. \tag{5.19}$$

In Equation 5.19, H is the representation matrix of testing nodes generated by the trained private model, with the input being the feature matrix of testing nodes. D and A are the degree matrix and adjacency matrix of the graph containing only testing nodes, respectively. The final output of power iteration $Q^{(P)}$ will be input into a softmax layer to generate the predictions for testing nodes. In our experiments, we set $P = 2$ and the teleportation constant $\alpha = 0.25$ as suggested in [13].

5.4.1 Privacy vs. Accuracy Trade-off

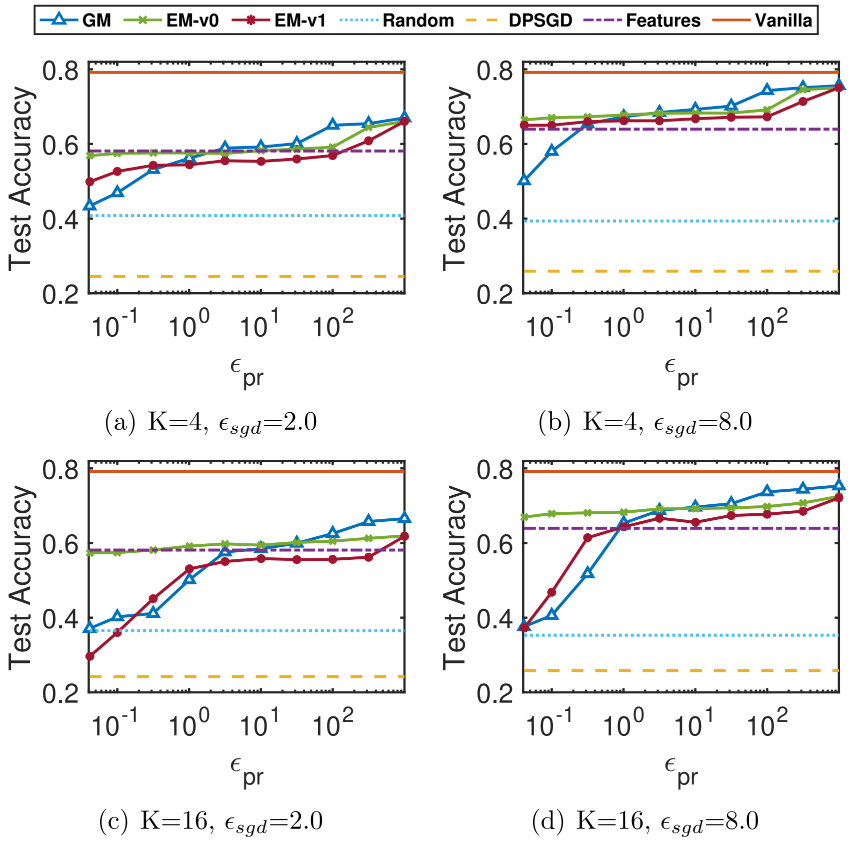
We use the value of privacy budget ϵ (with fixed δ) to represent the level of privacy protection each model guarantees, and use the test accuracy for node classification to indicate the model's utility. Note that the total privacy budget ϵ is the sum of ϵ_{pr} used in DP-APPR and ϵ_{sgd} used in DP-SGD. Figure 5.1/5.2/5.3 show the comparison results between our proposed methods and baselines, in Cora-ML/PubMed/MS Academic datasets, respectively. In each subfigure, we fix ϵ_{sgd} and use ϵ_{pr} as the x-axis to highlight the effect of privacy budget used in DP-APPR on the private model's accuracy.

We first discuss and compare the three alternative algorithms we proposed. For **GM** and **EM-v1**, the higher the ϵ_{pr}, the less noise is added when calculating the

APPR vector for each training node, which indicates that each node has a better chance to aggregate representations from more important nodes using more precise weights (importance scores). This is reflected in the higher model test accuracy compared to **EM-v0**. While for **EM-v0**, noise in DP-APPR will only affect the output of the indexes of the top K most relevant nodes corresponding to the source node, but not their importance scores. As we can see, **EM-v0** achieves better performance than **GM** and **EM-v1** when the privacy budget ϵ_{pr} is small, this is because **EM-v0** uses $1/K$ as the importance score for all nodes (considering nodes equally important), which diminishes the negative effect of less important or irrelevant nodes which have high importance scores due to the noise in **GM** and **EM-v1**. Both **EM-v0** and **EM-v1** are based on the exponential mechanism designed for identifying the index of the noisy max, which enables them to perform better than **GM** in finding the indexes of the actual top K. Therefore, when the privacy budget is small, **EM-v0** and **EM-v1** have the higher model test accuracy than **GM**. However, when the privacy budget is large, they all have a good chance to find the indexes of the actual top K, and **GM** becomes gradually better than **EM-v0** and **EM-v1**, as the Gaussian noise has better privacy loss composition property.

Compared with the baseline methods, our proposed methods outperform **DPSGD** in all datasets because of the weakness of applying DP-SGD directly to train node-level DP GNNs, as discussed in section 5.3.1. Comparing our methods with **Features**, we can find that when the privacy budget is small, the noisy indexes of the top K and their importance scores will negatively affect the model performance, which makes **Features** have the higher test accuracy than the proposed methods in some cases (esp. for PubMed dataset that the graph topology information plays a less important role than node features in the inferences), as it does not rely on the information from other nodes.

By comparing two subfigures in each row, we can observe the relationship between test accuracy and ϵ_{sgd}. We can see that the effect of ϵ_{sgd} on test accuracy is larger than the effect of ϵ_{pr} (i.e., test accuracy will change more significantly when we use different ϵ_{sgd} values). This is because ϵ_{sgd} affects the noise added to the model parameters directly, while ϵ_{pr} affects the model performance indirectly by affecting the utility of the DP APPR matrix. K controls the number of neighbors that each node will access. When ϵ_{pr} is small, choosing a larger K value leads to more fake important nodes with noisy importance scores being involved, which affects the model performance negatively. By comparing two subfigures in each column, we can see that when ϵ_{pr} becomes smaller, test accuracy drops more quickly when $K = 16$ than $K = 4$.



(a) K=4, ϵ_{sgd}=2.0 (b) K=4, ϵ_{sgd}=8.0

(c) K=16, ϵ_{sgd}=2.0 (d) K=16, ϵ_{sgd}=8.0

Figure 5.1: Relationship between privacy budget ϵ (fixed $\delta = 2 \times 10^{-3}$) and test accuracy on Cora-ML dataset.

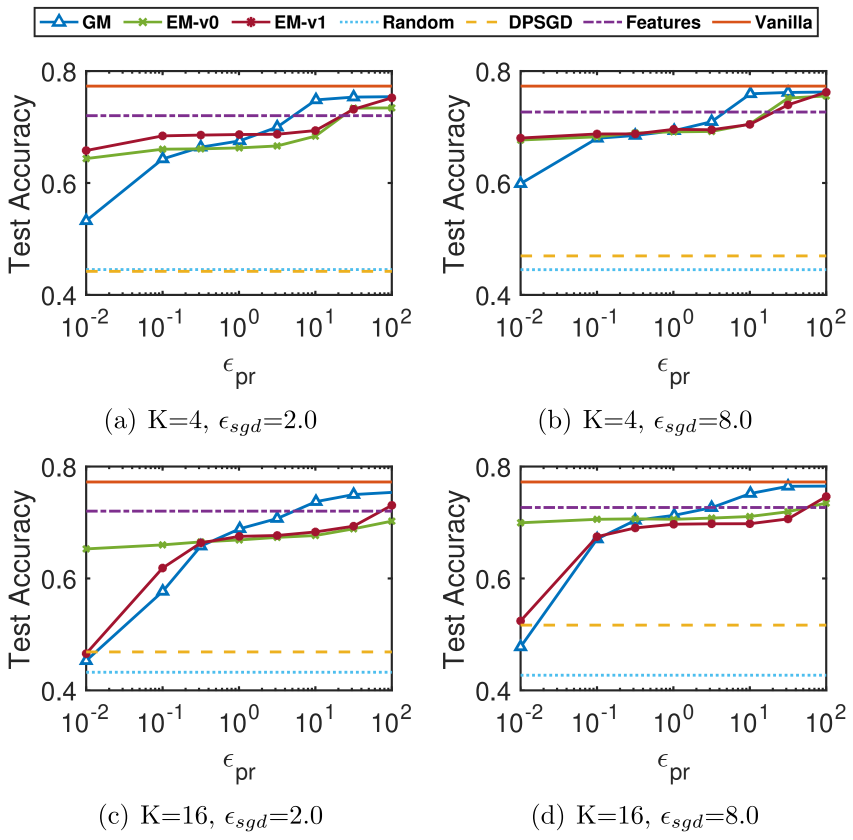


Figure 5.2: Relationship between privacy budget ϵ (fixed $\delta = 2 \times 10^{-4}$) and test accuracy on Pubmed dataset.

5.4.2 Demonstration of Privacy Protection Effectiveness

Node Degree Prediction. While DP gives a theoretical privacy guarantee of our methods (as shown in Theorem 8), in this section, we use a simple node degree inference attack as an example to provide an empirical understanding towards the privacy protection effectiveness of our methods. We use the embeddings of training nodes generated by the trained private model to predict (infer) the degree of the training nodes. We use the root-mean-square error (RMSE) between training nodes' true and predicted degrees to indicate the node degree prediction performance. Lower RMSE indicates that it is harder to predict the training nodes' precise degree information using their embeddings output by the private model, which means the private model is more successful in hiding each node's existence information (including its edges).

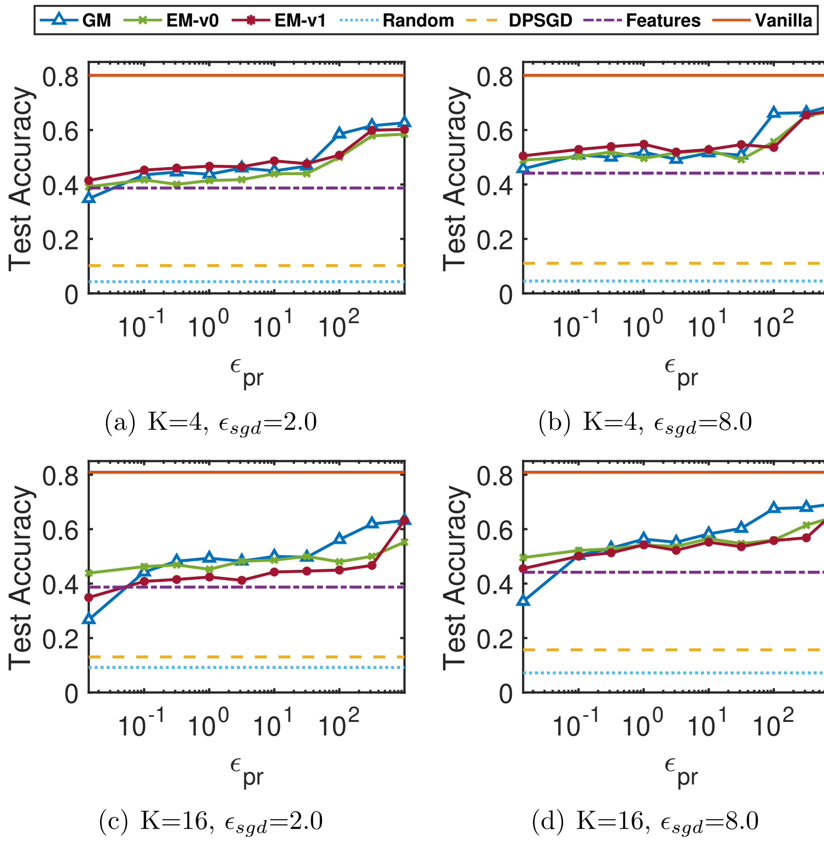
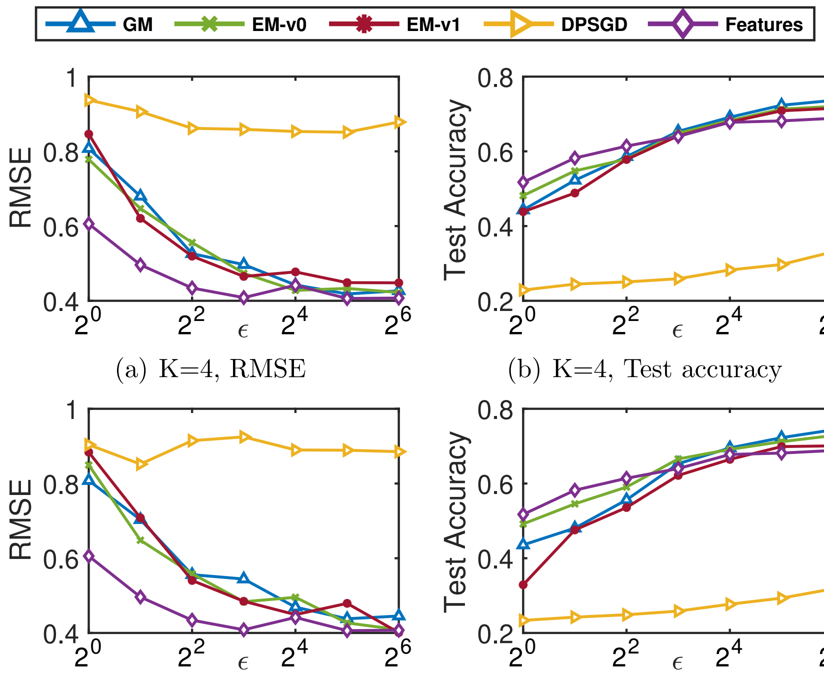


Figure 5.3: Relationship between privacy budget ϵ (fixed $\delta = 2 \times 10^{-4}$) and test accuracy on MS Academic dataset.

Figure 5.4 and Figure 5.5 show the node degree prediction RMSE under different privacy protection levels ϵ (with fixed δ) for the private models trained using the proposed methods and baseline methods on Cora-ML and PubMed datasets. For the proposed methods, the total privacy budget ϵ is divided into two equal parts for the DP-APPR and DP-SGD, respectively. We also report the corresponding test accuracy of the private models under each privacy budget. We can see that **GM**, **EM-v0**, and **EM-v1** have larger node degree prediction RMSE than **Features** in all settings, which indicates our proposed methods are more effective in protecting the node degree information while maintaining comparable model utility (test accuracy). Note the **DPSGD** has the largest RMSE, but also an unacceptably low test accuracy. Node embeddings generated by the private model from **DPSGD** are too noisy to be



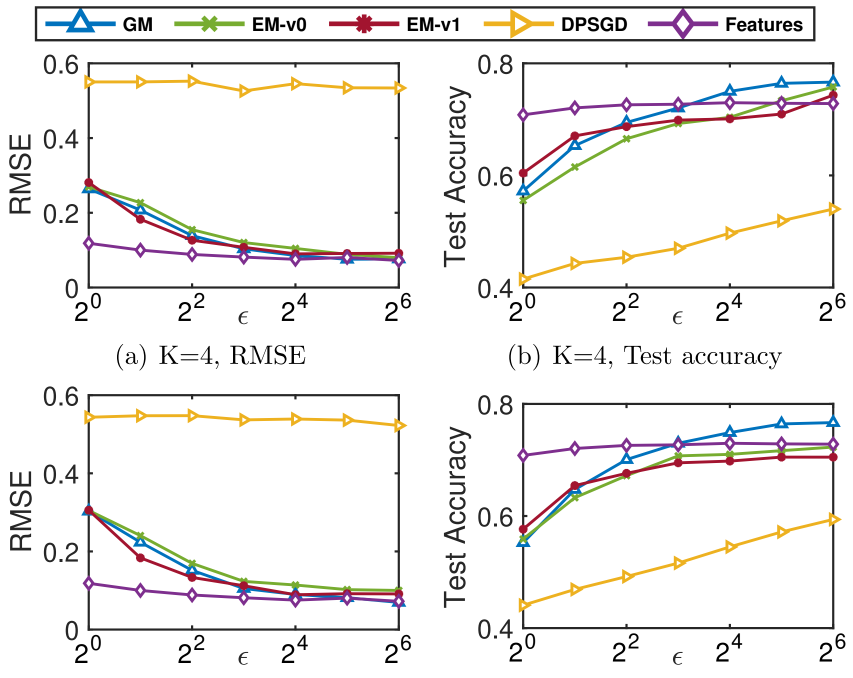
(a) K=4, RMSE (b) K=4, Test accuracy (c) K=16, RMSE (d) K=16, Test accuracy

Figure 5.4: Cora-ML. Subfigures in the left column are the node degree prediction RMSE under different privacy protection levels ϵ (fixed $\delta = 2 \times 10^{-3}$). Subfigures in the right column indicates the corresponding models' test accuracy.

used in predicting node labels (and not surprisingly also node degrees). In contrast, our methods achieve a better trade-off in preserving the model utility and providing effective privacy protection.

Node Embedding Clustering.

We visualize the t-SNE clustering of training nodes' embeddings generated by the private models in Figure 5.6 and Figure 5.7 for Cora-ML and PubMed datasets, respectively. The color of each node corresponds to the label of the node. We can observe that when the privacy budget is small ($\epsilon = 1$), which indicates a strong privacy protection, the training nodes that belong to different classes are hard to be distinguished from each other by using their embeddings generated by the private model. Meanwhile, when the privacy guarantee becomes weak (ϵ becomes larger), embeddings of nodes with the same class label will gradually move into the same



(a) K=4, RMSE (b) K=4, Test accuracy

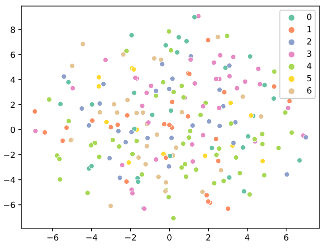
(c) K=16, RMSE (d) K=16, Test accuracy

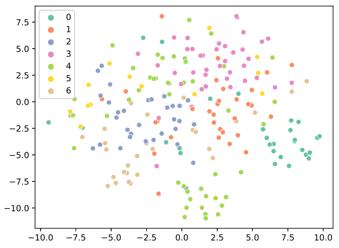
Figure 5.5: PubMed. Subfigures in the left column are the node degree prediction RMSE under different privacy protection levels ϵ (fixed $\delta = 2 \times 10^{-4}$). Subfigures in the right column indicates the corresponding models' test accuracy.

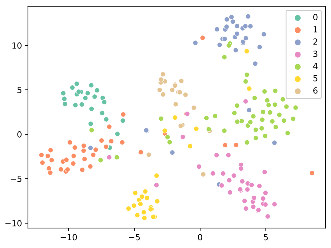
cluster. This observation demonstrates that the privacy budget used in our proposed methods is correlated with the model's ability to generate meaningful node embeddings, and therefore also associated with the privacy protection effectiveness since some adversaries will utilize the embeddings generated from the model to carry out privacy attacks [24, 34].

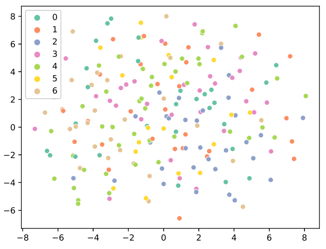
5.4.3 Effects of Privacy Parameters

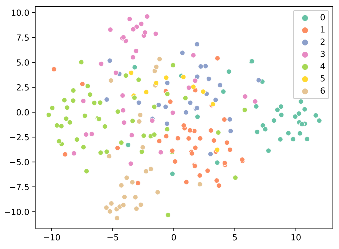
We demonstrate the effects of the parameters specific to privacy, including the batch size, the clipping bound in DP-APPR, and the clipping bound in DP-SGD. We set the batch size to 60, the clipping bound C_1 in DP-APPR-GM to 0.01, the clipping bound C_2 in DP-APPR-EM to 0.001, and the gradient norm clipping bound C for DP-SGD to 1. We analyze them individually with keeping the rest constant as the


(a) **GM**, $\epsilon = 1$


(b) **GM**, $\epsilon = 4$


(c) **GM**, $\epsilon = 16$


(d) **EM-v1**, $\epsilon = 1$


(e) **EM-v1**, $\epsilon = 4$

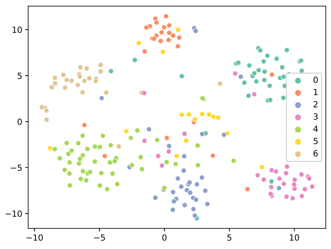
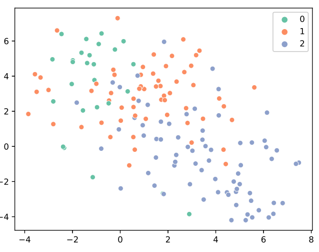

(f) **EM-v1**, $\epsilon = 16$

Figure 5.6: Cora-ML. Clustering of training nodes' embeddings generated by private models with different privacy guarantees ϵ (fixed $\delta = 2 \times 10^{-3}$). **GM**(**EM-v1**) indicates the private model is trained using the **GM** (**EM-v1**) method.

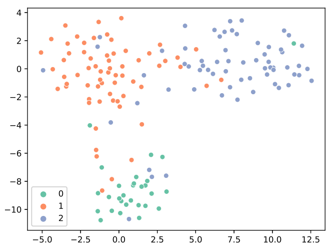
above reference values. The total privacy budget under each parameter setting is fixed as $(\epsilon, \delta) = (8, 2 \times 10^{-3})$ for Cora-ML and $(8, 2 \times 10^{-4})$ for PubMed.

Batch Size. According to Theorem 8, given the fixed total privacy budget and epochs, the standard deviation of the Gaussian noise is proportional to the square root of the batch size. Therefore, in **GM**, **EM-v0**, and **EM-v1**, a large batch size can diminish the effect of noise added to each batch's sum of gradients. In contrast, in **DPSGD**, according to Equation 5.5, the effect of noise added to the batch's sum gradient will increase with the increase of the batch size since there is an additional factor batch size in the standard deviation of the noise. Figure 5.8 and Figure 5.9 show the effect of batch size on the model's test accuracy on the Cora-ML and PubMed datasets. We can observe the opposing effects of batch size on our methods and **DPSGD**.

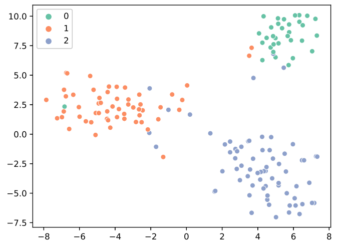
Clipping Bound in DP-APPR. In Algorithm 7 and Algorithm 8, the clipping



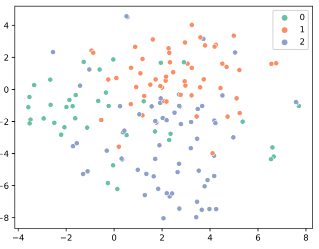
(a) **GM**, $\epsilon = 1$



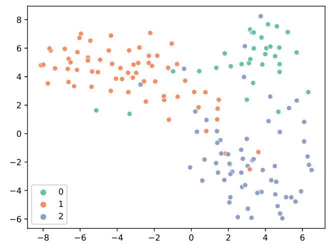
(b) **GM**, $\epsilon = 4$



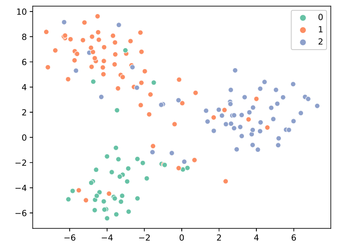
(c) **GM**, $\epsilon = 16$



(d) **EM-v1**, $\epsilon = 1$

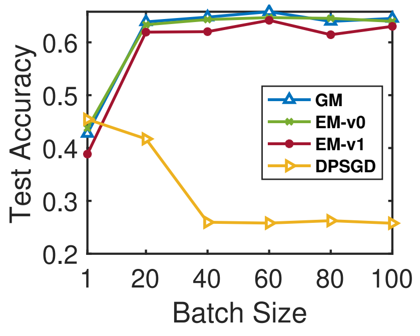


(e) **EM-v1**, $\epsilon = 4$

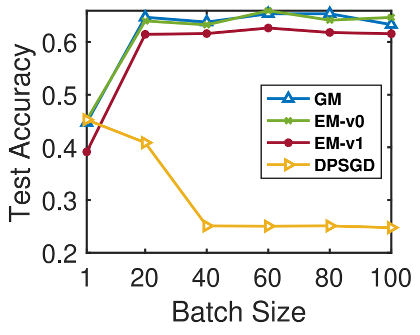


(f) **EM-v1**, $\epsilon = 16$

Figure 5.7: PubMed. Clustering of training nodes' embeddings generated by private models with different privacy guarantees ϵ (fixed $\delta = 2 \times 10^{-4}$). **GM** (**EM-v1**) indicates the private model is trained using the **GM** (**EM-v1**) method.



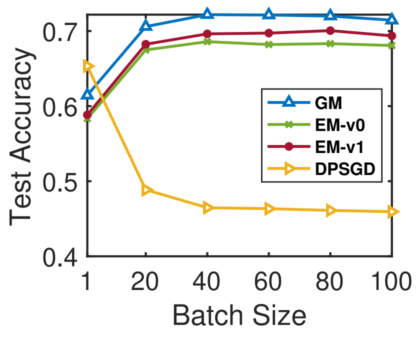
(a) K=4



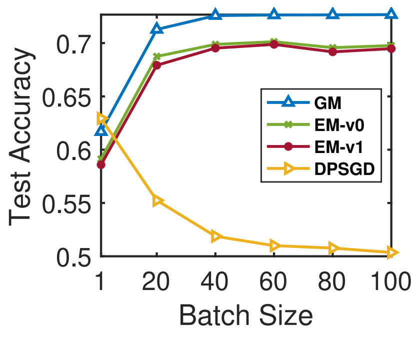
(b) K=16

Figure 5.8: Cora-ML. Relationship between batch size and model test accuracy. Fix total privacy budget $(\epsilon, \delta) = (8,\, 2 \times 10^{-3})$.

bound C_1 and the clipping bound C_2 are used to bound the sensitivity. Given a constant total privacy budget, the standard deviation of the noise added to the APPR vectors is proportional to the clipping bound. Choosing a smaller clipping bound value can avoid adding too much redundant noise. Figure 5.10 and Figure 5.11 show the effect of clipping bound in DP-APPR on the model's test accuracy on the Cora-ML



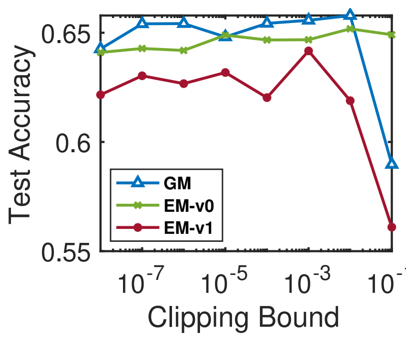
(a) K=4



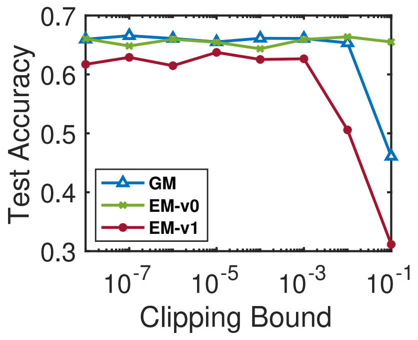
(b) K=16

Figure 5.9: PubMed. Relationship between batch size and model test accuracy. Fix total privacy budget $(\epsilon, \delta) = (8,\ 2 \times 10^{-4})$.

and PubMed datasets. In experiments, we set C_1 to be 0.01 and C_2 to be 0.001 for all datasets.

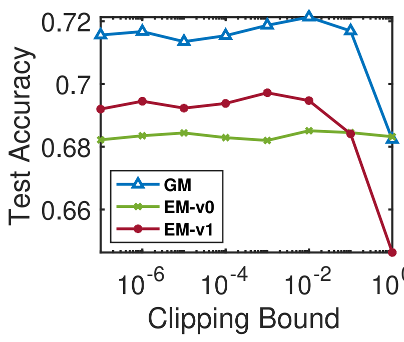


(a) K=4

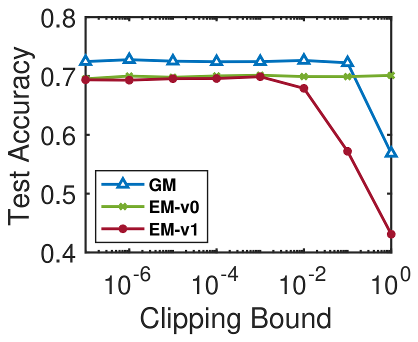


(b) K=16

Figure 5.10: Cora-ML. Relationship between clipping bound of DP-APPR and model test accuracy. Fix total privacy budget $(\epsilon, \delta) = (8,\ 2 \times 10^{-3})$.



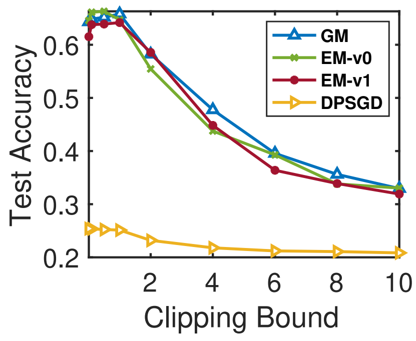
(a) K=4



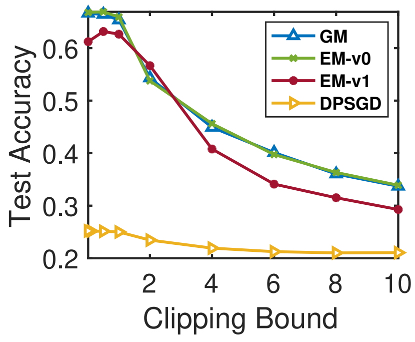
(b) K=16

Figure 5.11: PubMed. Relationship between clipping bound of DP-APPR and model test accuracy. Fix total privacy budget $(\epsilon, \delta) = (8,\ 2 \times 10^{-4})$.

Clipping Bound in DP-SGD. The gradient norm clipping bound in DP-SGD affects the noise scale added to the gradients (linearly) as well as the optimization direction of model parameters. A large clipping bound may involve too much noise to the gradients, while a small clipping bound may undermine gradients' ability for unbiased estimation. Figure 5.12 and Figure 5.13 show the effect of gradient norm clipping bound in DP-SGD on the model's test accuracy on the Cora-ML and PubMed datasets. We choose the gradient norm clipping bound to be 1 for all datasets in our experiments.

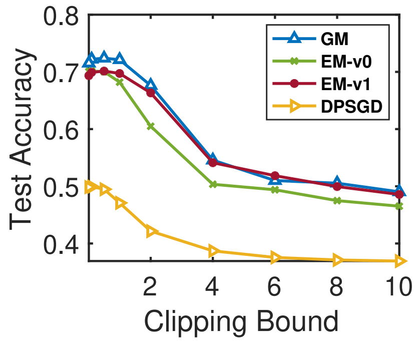


(a) K=4

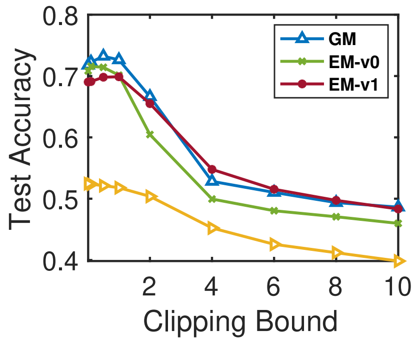


(b) K=16

Figure 5.12: Cora-ML. Relationship between clipping bound of DP-SGD and model test accuracy. Fix total privacy budget $(\epsilon, \delta) = (8,\, 2 \times 10^{-3})$.



(a) K=4



(b) K=16

Figure 5.13: PubMed. Relationship between clipping bound of DP-SGD and model test accuracy. Fix total privacy budget $(\epsilon, \delta) = (8,\, 2 \times 10^{-4})$.

5.5 Related Work

Differentially Private Graph Publishing. Works on privacy-preserving graph data publishing aim to release the entire graph [35, 51, 79, 114], or the statistics or properties of the original graph [6, 21, 22, 56, 70, 123], with the DP guarantee. Different from those works, our work in this chapter focuses on training GNN models on private graph datasets and publishing the model that satisfies a formal node-level DP guarantee.

Differentially Private Network Embedding. Xu et al. [115] model the network embedding as matrix factorization and apply the objective perturbation mechanism [19] on the loss function to achieve edge-level DP. Zhang et al. [126] propose a Lipschitz condition on the objective function of the matrix factorization based graph embedding model and a gradient clipping strategy to ensure link privacy. The objective of works in this area is to ensure that the sharing of the embeddings generated from sensitive graph datasets satisfy the DP guarantee. In contrast, our privacy-protection target is the model parameters. In other words, we want the model parameters trained on the private graph dataset to satisfy DP.

Differentially Private Graph Neural Networks. Yang et al. [118] propose to train a graph generation model using DP-SGD to generate graphs with the edge-DP guarantee that protects the individual link privacy. Sajadmanesh et al. [91] develop a privacy-preserving GNN training algorithm based on local differential privacy (LDP) to protect node features privacy. Zhang et al. [127] use LDP and functional mechanism [124] to enforce privacy guarantee on user's sensitive features when training graph embedding models for recommendation. None of these previous works achieve the goal of providing strict DP with respect to each node in the graph for GNN models.

5.6 Conclusion

In this chapter, we studied the problem of private learning for GNN models. To the best of our knowledge, we are the first to propose the method for training GNN models under rigorous node-level DP guarantees, which consider both the node edges and node features. Our method is based on DP approximate personalized PageRank and DP-SGD. We developed two algorithms using the Gaussian mechanism and the exponential mechanism for achieving DP APPR matrix calculation. DP-APPR not only protects nodes' edge information but also decorrelates the nodes to limit the sensitivity of each node during the model training using DP-SGD, which facilitates DP-SGD to play its role in protecting nodes' features information. Experimental results on real-world graph datasets demonstrate the effectiveness of our proposed methods in achieving good privacy and utility trade-off. We leave developing better DP-APPR algorithms with tighter privacy guarantee and adaptive privacy budget allocation strategy (e.g., for each node) as future work.

Chapter 6

Conclusion and Future Work

In this dissertation, we research the practical issues of preserving data privacy during training deep learning models. Chapter 3 focuses on addressing the dilemma about the privacy and utility trade-off of DP algorithms in the general deep learning settings. We develop two new mechanisms, PATE+ and PATE++, to train more robust PATE under noisy labels, which obtain a better utility and privacy trade-off in private model training and further improve the practicality to achieve meaningful privacy guarantees when training deep models on sensitive data. Chapter 4 focuses on broadening the DP definitions and algorithms for non-typical privacy attacks. We study the problem of protecting deep learning models against MIA. We show that traditional record-DP for building private deep learning models does not provide effective and quantifiable protection against MIA. Then, we propose two new DP notions, class-DP and subclass-DP, and algorithms for protecting deep learning models against MIA. Experimental results show that class or subclass-DP can effectively defend against MIA while preserving good model utility. Chapter 5 focuses on broadening the DP definitions and algorithms for non-typical data. We conduct the first formal study of training GNN models on graph data with the rigorous node-level DP guarantee. The key idea is to decouple the message-passing process from feature aggregation via DP approximate personalized PageRank and DP-SGD to protect both graph topology and node features while maintaining high model utility. Further, we develop two DP approximate personalized PageRank computation algorithms with formal node-level DP guarantee based on the Gaussian mechanism and the exponential mechanism. Experiments on real-world graph datasets demonstrate the effectiveness of the proposed algorithms in achieving satisfying privacy and utility trade-off.

Future work has the following potential directions:

Federated Graph Neural Networks with Differential Privacy. It is interesting to extend the proposed node-level differentially private graph neural network training methods into the federated setting where each local site has its graph dataset that is private and cannot be shared. The problems that we need to consider under the federated setting include but are not limited to the non-I.I.D and unbalanced data distribution and the communication cost bottleneck. Meanwhile, we may also need to consider different types of federated graph neural networks [41] such as graph-level federated GNNs and node-level federated GNNs, etc. The privacy definitions under varying levels of federated GNNs may be changed in order to achieve more meaningful privacy protection under different settings. For example, for graph-level federated GNNs where each local site or user has its isolated graph, we may consider the graph-level differential privacy to provide the DP protection for each local graph. While for node-level federated GNNs, each local site has a set of nodes belonging to a graph containing nodes from all sites, and each node corresponds to a user. In this case, we need to consider the node-level differential privacy as in our work in chapter 5, and apply our proposed methods to achieve the node-level DP protection.

Personalized Privacy Budget Allocation Mechanisms. Developing personalized privacy budget allocation mechanisms to assign the varying amount of privacy budget for each class/subclass in Class/Subclass-DP or each node in differentially private GNNs is interesting and meaningful. First, in Class/Subclass-DP, classes or subclasses may have a varying number of data records, and the degree of leakage of information to model parameters is different for each class/subclass. Therefore, we may assign more privacy budget for those classes/subclasses which contain more data records and more vulnerable to MIA. Second, in the differentially private GNNs, different nodes have different degrees, and their impact on the model parameters and model utility is also different. Hence, we want to assign the different privacy budgets to each node during training the node-level differentially private GNNs.

Extend the Applications to Healthcare Data. Finally, it will be exciting to apply our proposed methods and algorithms to achieve a better privacy and utility trade-off or meaningful privacy protection against MIA when training deep learning models on healthcare data like EHR or medical images [31, 75]. Meanwhile, we can apply our node-level differentially private GNNs algorithms to contact tracing [60] and disease spread modeling [90] problems, where each individual corresponds to a node in the interaction graph, to protect the differential privacy of each participant.

Bibliography

[1] Google ai platform. `https://cloud.google.com/ai-platform/`, 2019.

[2] Machine learning on aws. `https://aws.amazon.com/machine-learning/`, 2019.

[3] Microsoft machine learning studio. `https://azure.microsoft.com/en-us/services/machine-learning-studio/`, 2019.

[4] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *ACM SIGSAC CCS*, 2016.

[5] Martín Abadi, Ulfar Erlingsson, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Nicolas Papernot, Kunal Talwar, and Li Zhang. On the protection of private information in machine learning systems: Two recent approches. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pages 1–6. IEEE, 2017.

[6] Faraz Ahmed, Alex X Liu, and Rong Jin. Publishing social network graph eigenspectrum with privacy guarantees. *IEEE Transactions on Network Science and Engineering*, 7(2):892–906, 2019.

[7] Reid Andersen, Fan Chung, and Kevin Lang. Local graph partitioning using pagerank vectors. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 475–486. IEEE, 2006.

[8] Miguel E Andrés, Nicolás E Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. Geo-indistinguishability: Differential privacy for location-based systems. In *Proceedings of the 2013 ACM SIGSAC CCS*, 2013.

[9] Devansh Arpit, Stanisław Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 233–242. JMLR. org, 2017.

[10] Amos Beimel, Shiva Prasad Kasiviswanathan, and Kobbi Nissim. Bounds on the sample complexity for private learning and private data release. In *Theory of Cryptography Conference*, pages 437–454. Springer, 2010.

[11] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems*, pages 5050–5060, 2019.

[12] Aleksandar Bojchevski and Stephan Günnemann. Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. *ICLR*, 2018.

[13] Aleksandar Bojchevski, Johannes Klicpera, Bryan Perozzi, Amol Kapoor, Martin Blais, Benedek Rózemberczki, Michal Lukasik, and Stephan Günnemann. Scaling graph neural networks with approximate pagerank. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2464–2473, 2020.

[14] MH Brendan, D Ramage, K Talwar, and L Zhang. Learning differentially private recurrent language models. In *International conference on learning representations, Vancouver, BC, Canada*, volume 30, 2018.

[15] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1-7):107–117, 1998.

[16] Chensi Cao, Feng Liu, Hai Tan, Deshou Song, Wenjie Shu, Weizhong Li, Yiming Zhou, Xiaochen Bo, and Zhi Xie. Deep learning and its applications in biomedicine. *Genomics, proteomics & bioinformatics*, 16(1):17–32, 2018.

[17] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy (SP)*, 2017.

[18] Konstantinos Chatzikokolakis, Miguel E Andrés, Nicolás Emilio Bordenabe, and Catuscia Palamidessi. Broadening the scope of differential privacy using metrics. In *International Symposium on Privacy Enhancing Technologies Symposium*, 2013.

[19] Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. Differentially private empirical risk minimization. *JMLR*, 2011.

[20] Ming Chen, Zhewei Wei, Bolin Ding, Yaliang Li, Ye Yuan, Xiaoyong Du, and Ji-Rong Wen. Scalable graph neural networks via bidirectional propagation. *NeurIPS*, 2020.

[21] Rui Chen, Benjamin CM Fung, S Yu Philip, and Bipin C Desai. Correlated network data publication via differential privacy. *The VLDB Journal*, 23(4):653–676, 2014.

[22] Wei-Yen Day, Ninghui Li, and Min Lyu. Publishing graph degree distribution with node differential privacy. In *Proceedings of the 2016 International Conference on Management of Data*, pages 123–138, 2016.

[23] Hande Dong, Jiawei Chen, Fuli Feng, Xiangnan He, Shuxian Bi, Zhaolin Ding, and Peng Cui. On the equivalence of decoupled graph convolution network and label propagation. *The World Wide Web Conference*, 2021.

[24] Vasisht Duddu, Antoine Boutet, and Virat Shejwalkar. Quantifying privacy leakage in graph embedding. In *Mobiquitous 2020-17th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, pages 1–11, 2020.

[25] David Durfee and Ryan M Rogers. Practical differentially private top-k selection with pay-what-you-get composition. *Advances in Neural Information Processing Systems*, 32:3532–3542, 2019.

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

[27] Cynthia Dwork, Moni Naor, Omer Reingold, Guy N Rothblum, and Salil Vadhan. On the complexity of differentially private data release: efficient algorithms and hardness results. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 381–390, 2009.

[28] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 2014.

[29] Cynthia Dwork, Guy N Rothblum, and Salil Vadhan. Boosting and differential privacy. In *FOCS*, 2010.

[30] Gökcen Eraslan, Žiga Avsec, Julien Gagneur, and Fabian J Theis. Deep learning: new computational modelling techniques for genomics. *Nature Reviews Genetics*, 20(7):389–403, 2019.

[31] Andre Esteva, Alexandre Robicquet, Bharath Ramsundar, Volodymyr Kuleshov, Mark DePristo, Katherine Chou, Claire Cui, Greg Corrado, Sebastian Thrun, and Jeff Dean. A guide to deep learning in healthcare. *Nature medicine*, 25(1):24–29, 2019.

[32] Sina Farsiu, M Dirk Robinson, Michael Elad, and Peyman Milanfar. Fast and robust multiframe super resolution. *IEEE Transactions on Image Processing*, 2004.

[33] Kimon Fountoulakis, Farbod Roosta-Khorasani, Julian Shun, Xiang Cheng, and Michael W Mahoney. Variational perspective on local graph clustering. *Mathematical Programming*, 174(1):553–573, 2019.

[34] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *ACM SIGSAC CCS*, 2015.

[35] Tianchong Gao and Feng Li. Sharing social networks using a novel differentially private graph model. In *2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, pages 1–4. IEEE, 2019.

[36] Robin C Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. In *NIPS Workshop*, 2017.

[37] David F Gleich. Pagerank beyond the web. *siam REVIEW*, 57(3):321–363, 2015.

[38] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, pages 2672–2680, 2014.

[39] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in neural information processing systems*, pages 1024–1034, 2017.

[40] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in neural information processing systems*, pages 8527–8537, 2018.

[41] Chaoyang He, Keshav Balasubramanian, Emir Ceyani, Yu Rong, Peilin Zhao, Junzhou Huang, Murali Annavaram, and Salman Avestimehr. Fedgraphnn: A federated learning system and benchmark for graph neural networks. *arXiv preprint arXiv:2104.07145*, 2021.

[42] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015.

[43] Xinlei He, Jinyuan Jia, Michael Backes, Neil Zhenqiang Gong, and Yang Zhang. Stealing links from graph neural networks. In *30th USENIX Security Symposium (USENIX Security 21)*, 2021.

[44] Zecheng He, Tianwei Zhang, and Ruby B Lee. Model inversion attacks against collaborative inference. 2019.

[45] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *NIPS Deep Learning Workshop*, 2014.

[46] Briland Hitaj, Giuseppe Ateniese, and Fernando Pérez-Cruz. Deep models under the gan: information leakage from collaborative deep learning. In *ACM SIGSAC CCS*, 2017.

[47] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. *ICLR*, 2020.

[48] Glen Jeh and Jennifer Widom. Scaling personalized web search. In *Proceedings of the 12th international conference on World Wide Web*, pages 271–279, 2003.

[49] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *ICML*, pages 2304–2313, 2018.

[50] Di Jin, Ziyang Liu, Weihao Li, Dongxiao He, and Weixiong Zhang. Graph convolutional networks meet markov random fields: Semi-supervised community detection in attribute networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 152–159, 2019.

[51] Zach Jorgensen, Ting Yu, and Graham Cormode. Publishing attributed social graphs with formal privacy guarantees. In *Proceedings of the 2016 international conference on management of data*, pages 107–122, 2016.

[52] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. The composition theorem for differential privacy. In *International conference on machine learning*, pages 1376–1385. PMLR, 2015.

[53] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. The composition theorem for differential privacy. *IEEE Transactions on Information Theory*, 63(6):4037–4049, 2017.

[54] Davood Karimi, Haoran Dou, Simon K Warfield, and Ali Gholipour. Deep learning with noisy labels: exploring techniques and remedies in medical image analysis. *Medical Image Analysis*, page 101759, 2020.

[55] Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? *SIAM Journal on Computing*, 2011.

[56] Shiva Prasad Kasiviswanathan, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. Analyzing graphs with node differential privacy. In *Theory of Cryptography Conference*, pages 457–476. Springer, 2013.

[57] Younghwan Kim, Jang-Hee Yoo, and Kyoungho Choi. A motion and similarity-based fake detection method for biometric face recognition systems. *IEEE Transactions on Consumer Electronics*, 2011.

[58] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *ICLR*, 2016.

[59] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. *ICLR*, 2019.

[60] Valerio La Gatta, Vincenzo Moscato, Marco Postiglione, and Giancarlo Sperli. An epidemiological neural network exploiting dynamic graph structured data applied to the covid-19 outbreak. *IEEE Transactions on Big Data*, 2020.

[61] Amine Laghrib, Abdelilah Hakim, and Said Raghay. A combined total variation and bilateral filter approach for image robust super resolution. *EURASIP Journal on Image and Video Processing*, 2015.

[62] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

[63] Der-Hau Lee, Kuan-Lin Chen, Kuan-Han Liou, Chang-Lun Liu, and Jinn-Liang Liu. Deep learning and control algorithms of direct perception for autonomous driving. *Applied Intelligence*, 51(1):237–247, 2021.

[64] Jaewoo Lee and Daniel Kifer. Concentrated differentially private gradient descent with adaptive per-iteration privacy budget. In *KDD*, 2018.

[65] Junnan Li, Richard Socher, and Steven CH Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. *ICLR*, 2020.

[66] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. *AAAI*, 2018.

[67] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep learning for generic object detection: A survey. *International journal of computer vision*, 2020.

[68] Meng Liu, Hongyang Gao, and Shuiwang Ji. Towards deeper graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 338–348, 2020.

[69] Shangbang Long, Xin He, and Cong Yao. Scene text detection and recognition: The deep learning era. *International Journal of Computer Vision*, pages 1–24, 2020.

[70] Wentian Lu and Gerome Miklau. Exponential random graph estimation under differential privacy. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 921–930, 2014.

[71] Min Lyu, Dong Su, and Ninghui Li. Understanding the sparse vector technique for differential privacy. *Proceedings of the VLDB Endowment*, 10(6), 2017.

[72] Eran Malach and Shai Shalev-Shwartz. Decoupling" when to update" from" how to update". In *Advances in Neural Information Processing Systems*, pages 960–970, 2017.

[73] Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pages 94–103. IEEE, 2007.

[74] Felipe A Mejia, Paul Gamble, Zigfried Hampel-Arias, Michael Lomnitz, Nina Lopatina, Lucas Tindall, and Maria Alejandra Barrios. Robust or private? adversarial training makes models more vulnerable to privacy attacks. *arXiv preprint arXiv:1906.06449*, 2019.

[75] Riccardo Miotto, Fei Wang, Shuang Wang, Xiaoqian Jiang, and Joel T Dudley. Deep learning for healthcare: review, opportunities and challenges. *Briefings in bioinformatics*, 19(6):1236–1246, 2018.

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

[77] Galileo Namata, Ben London, Lise Getoor, Bert Huang, and UMD EDU. Query-driven active surveying for collective classification. In *10th International Workshop on Mining and Learning with Graphs*, volume 8, 2012.

[78] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. 2011.

[79] Hiep H Nguyen, Abdessamad Imine, and Michaël Rusinowitch. Differentially private publication of social graphs at linear cost. In *2015 IEEE/ACM In-*

ternational Conference on Advances in Social Networks Analysis and Mining (ASONAM), pages 596–599. IEEE, 2015.

[80] Hyunseok Oh and Youngki Lee. Exploring image reconstruction attack in deep learning computation offloading. In *International Workshop on Deep Learning for Mobile Systems and Applications*, 2019.

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

[82] Daniel W Otter, Julian R Medina, and Jugal K Kalita. A survey of the usages of deep learning for natural language processing. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[83] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[84] Nicolas Papernot, Martín Abadi, Úlfar Erlingsson, Ian Goodfellow, and Kunal Talwar. Semi-supervised knowledge transfer for deep learning from private training data. *ICLR*, 2017.

[85] Nicolas Papernot, Shuang Song, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Úlfar Erlingsson. Scalable private learning with pate. In *ICLR*, 2018.

[86] NhatHai Phan, Yue Wang, Xintao Wu, and Dejing Dou. Differential privacy preservation for deep auto-encoders: an application of human behavior prediction. In *AAAI*, 2016.

[87] Hendrik Purwins, Bo Li, Tuomas Virtanen, Jan Schlüter, Shuo-Yiin Chang, and Tara Sainath. Deep learning for audio signal processing. *IEEE Journal of Selected Topics in Signal Processing*, 13(2):206–219, 2019.

[88] JR Quinlan. Learning from noisy data. In *Proc. of the International Machine Learning Workshop*, pages 58–64. Citeseer, 1983.

[89] Md Atiqur Rahman, Tanzila Rahman, Robert Laganière, Noman Mohammed, and Yang Wang. Membership inference attack against differentially private deep learning model. *Trans. Data Priv.*, 11(1):61–79, 2018.

[90] Adam Sadilek, Henry Kautz, and Vincent Silenzio. Modeling spread of disease from social interactions. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 6, 2012.

[91] Sina Sajadmanesh and Daniel Gatica-Perez. Locally private graph neural networks. *CoRR abs/2006.05535*, 12, 2020.

[92] Ahmed Salem, Apratim Bhattacharya, Michael Backes, Mario Fritz, and Yang Zhang. Updates-leak: Data set inference and reconstruction attacks in online learning. In *USENIX Security Symposium, 2020.*

[93] Ahmed Salem, Apratim Bhattacharya, Michael Backes, Mario Fritz, and Yang Zhang. Updates-leak: Data set inference and reconstruction attacks in online learning. *arXiv preprint arXiv:1904.01067*, 2019.

[94] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in neural information processing systems*, pages 2234–2242, 2016.

[95] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. *Relational Representation Learning Workshop (R2L 2018), NeurIPS*, 2018.

[96] Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In *ACM SIGSAC CCS*, 2015.

[97] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *IEEE Symposium on Security and Privacy (SP)*, 2017.

[98] Congzheng Song, Thomas Ristenpart, and Vitaly Shmatikov. Machine learning models that remember too much. In *ACM SIGSAC CCS*, 2017.

[99] Shuang Song, Kamalika Chaudhuri, and Anand D Sarwate. Stochastic gradient descent with differentially private updates. In *IEEE Global Conference on Signal and Information Processing*, 2013.

[100] Lichao Sun, Yingbo Zhou, Ji Wang, Jia Li, Richard Sochar, Philip S Yu, and Caiming Xiong. Private deep learning with teacher ensembles. *arXiv preprint arXiv:1906.02303*, 2019.

[101] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *ICLR*, 2018.

[102] Oriol Vinyals, Łukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. Grammar as a foreign language. In *NeurIPS*, 2015.

[103] Matt P Wand. A comparison of regression spline smoothing procedures. *Computational Statistics*, 15(4):443–462, 2000.

[104] Lulu Wang, Junxiang Zheng, Yongzhi Cao, and Hanpin Wang. Enhance pate on complex tasks with knowledge transferred from non-private data. *IEEE Access*, 7:50081–50094, 2019.

[105] Tianhao Wang, Yuheng Zhang, and Ruoxi Jia. Improving robustness to model inversion attacks via mutual information regularization. *arXiv preprint arXiv:2009.05241*, 2020.

[106] Zhibo Wang, Mengkai Song, Zhifei Zhang, Yang Song, Qian Wang, and Hairong Qi. Beyond inferring class representatives: User-level privacy leakage from federated learning. *arXiv preprint arXiv:1812.00535*, 2018.

[107] Zhibo Wang, Mengkai Song, Zhifei Zhang, Yang Song, Qian Wang, and Hairong Qi. Beyond inferring class representatives: User-level privacy leakage from federated learning. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pages 2512–2520. IEEE, 2019.

[108] Zhou Wang, Alan C Bovik, Hamid R Sheikh, Eero P Simoncelli, et al. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 2004.

[109] Zhewei Wei, Xiaodong He, Xiaokui Xiao, Sibo Wang, Shuo Shang, and Ji-Rong Wen. Topppr: top-k personalized pagerank queries with precision guarantees on large graphs. In *Proceedings of the 2018 International Conference on Management of Data*, pages 441–456, 2018.

[110] Bang Wu, Xiangwen Yang, Shirui Pan, and Xingliang Yuan. Model extraction attacks on graph neural networks: Taxonomy and realization. *arXiv preprint arXiv:2010.12751*, 2020.

[111] Felix Wu, Tianyi Zhang, Amauri Holanda de Souza Jr, Christopher Fifty, Tao Yu, and Kilian Q Weinberger. Simplifying graph convolutional networks. *ICML*, 2019.

[112] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 2020.

[113] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[114] Qian Xiao, Rui Chen, and Kian-Lee Tan. Differentially private network data release via structural inference. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 911–920, 2014.

[115] Depeng Xu, Shuhan Yuan, Xintao Wu, and HaiNhat Phan. Dpne: Differentially private network embedding. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 235–246. Springer, 2018.

[116] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *ICLR*, 2019.

[117] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *International Conference on Machine Learning*, pages 5453–5462. PMLR, 2018.

[118] Carl Yang, Haonan Wang, Lichao Sun, and Bo Li. Secure network release with link privacy. *arXiv preprint arXiv:2005.00455*, 2020.

[119] Ziqi Yang, Ee-Chien Chang, and Zhenkai Liang. Adversarial neural network inversion via auxiliary knowledge alignment. *arXiv preprint arXiv:1902.08552*, 2019.

[120] Lei Yu, Ling Liu, Calton Pu, Mehmet Emre Gursoy, and Stacey Truex. Differentially private model publishing for deep learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 332–349. IEEE, 2019.

[121] Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor W Tsang, and Masashi Sugiyama. How does disagreement benefit co-teaching? *ICLR*, 2019.

[122] C Zhang, S Bengio, M Hardt, B Recht, and O Vinyals. Understanding deep learning requires rethinking generalization. In *ICLR*, 2017.

[123] Jun Zhang, Graham Cormode, Cecilia M Procopiuc, Divesh Srivastava, and Xiaokui Xiao. Private release of graph statistics using ladder functions. In *Proceedings of the 2015 ACM SIGMOD international conference on management of data*, pages 731–745, 2015.

[124] Jun Zhang, Zhenjie Zhang, Xiaokui Xiao, Yin Yang, and Marianne Winslett. Functional mechanism: Regression analysis under differential privacy. *Proceedings of the VLDB Endowment*, 5(11), 2012.

[125] Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. *Advances in Neural Information Processing Systems*, 31:5165–5175, 2018.

[126] Sen Zhang and Weiwei Ni. Graph embedding matrix sharing with differential privacy. *IEEE Access*, 7:89390–89399, 2019.

[127] Shijie Zhang, Hongzhi Yin, Tong Chen, Zi Huang, Lizhen Cui, and Xiangliang Zhang. Graph embedding for recommendation against attribute inference attacks. *arXiv preprint arXiv:2101.12549*, 2021.

[128] Zhengli Zhao, Nicolas Papernot, Sameer Singh, Neoklis Polyzotis, and Augustus Odena. Improving differentially private models with active learning. *arXiv preprint arXiv:1910.01177*, 2019.

[129] Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *arXiv preprint arXiv:1812.08434*, 2018.

[130] Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. In *NIPS*, pages 14774–14784, 2019.