RECONCILING DATA PRIVACY AND UTILITY IN THE ERA OF BIG DATA

A Dissertation
Presented to
The Academic Faculty

By

Lei Yu

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Computer Science

Georgia Institute of Technology

May 2019

RECONCILING DATA PRIVACY AND UTILITY IN THE ERA OF BIG DATA

Approved by:

Dr. Ling Liu, Advisor
School of Computer Science
Georgia Institute of Technology

Dr. Calton Pu, Advisor
School of Computer Science
Georgia Institute of Technology

Dr. Mustaque Ahamad
School of Computer Science
Georgia Institute of Technology

Dr. Douglas M Blough
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Rachel Cummings
School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Shamkant Navathe
School of Computer Science
Georgia Institute of Technology

Dr. Jamie Morgenstern
School of Computer Science
Georgia Institute of Technology

Date Approved: Mar 28, 2019

To my family.

ACKNOWLEDGEMENTS

First and foremost, I would like to extend my sincere gratitude to my advisor Prof. Ling Liu and co-advisor Prof. Calton Pu for their continuous support of my Ph.D study and research, for their patience, motivation, and enlightened guidance. I feel extremely fortunate to have had the opportunity to work with them in the past four years. Prof. Liu Ling has always offered me insightful comments on my research projects and given me the freedom and encouragement to pursue my research interests. Ling also has always been available to provide sincere and valuable suggestions, helping me make the right decisions in my important life related matters. I wish that I could be as lively, colorful, energetic, and enthusiastic as Ling. Prof. Calton Pu has been my another great mentor. His immense knowledge and wisdom have always enlightened me in our every meeting. He have provided me not only insightful discussions on my research but also invaluable career advices in general. His suggestions will undoubtedly light my way in my future career.

It has been my great honor to be a member of DiSL research lab. I would like to thank every member in the lab, Emre Yigitoglu, Emre Gursoy, Stacey Truex, Qi Zhang, Yang Zhou, Wenqi Cao, Wenqi Wei, Yanzhao Wu, for their support, friendship, and in-depth discussions on my research.

I would also like to thank my committee members Prof. Mustaque Ahamad, Prof. Douglas M Blough, Prof. Shamkant Navathe, Prof. Jamie Morgenstern and Prof. Rachel Cummings who have been very supportive of my research and constructive critiques on my dissertation.

Finally, I would like to dedicate this work to my parents, whose love, patience and support have been my constant source of inspiration. My success is also theirs. The last word of acknowledgment I have saved is for my dear wife Hanchun Zhang, who has been with me all these years, and has made them the best years of my life. Without her love and support, this accomplishment would have been impossible.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

The widespread use of internet-connected mobile devices, internet of things(IoT) and cloud computing has enabled a large scale collection of personal data, including user profiles, daily activities, locations, photos and health states, etc, of millions and billions of users from a wide range of scenarios such as the usage of mobile apps, smart home, and cloud storage services. The availability of these huge amounts of datasets has been driving the breakthrough in deep learning and explosion of data-driven applications for enriching human with life-enhancing experiences. At the same time, however, these datasets often encode privacy-sensitive information related to individuals, which raises serious privacy concerns to the society. Therefore, it is imperative to develop principled privacy preserving approaches to harvesting the power of those big data. This dissertation research contributes original ideas and innovative techniques in applying differential privacy, a rigorous mathematical framework that offers provable privacy guarantee, to protect data privacy with improving the trade-off between privacy and utility in the era of big data from three perspectives respectively: data collection, data usage, and data publication.

The first contribution of this dissertation research is the development of PIVE [1], a two-phase Bayesian differential location privacy framework that aims to protect users location privacy in location based services while ensuring the service quality. With the popularity of location based services for navigation, point-of-interest recommendation and social network etc, the companies that offer such services can continuously collect users locations. The collected location information may open doors to potential misuse and abuse of private location information, exposing users travel patterns and uncovering their health state and political views. PIVE provides a Bayesian differentially private location perturbation mechanism which transforms the users exact location to a perturbed location in a geo-indistinguishable way while being resilient against Bayesian attacks before reporting it to the servers. This approach essentially augments differential location privacy by bounding the inference error of the adversaries with specific prior knowledge, while enabling adaptive privacy control to improve the utility and user experience.

The second contribution of this dissertation research is the development of differentially private deep learning for protecting the privacy of the training data [2]. Because of the breakthrough of deep learning, more companies are interested in training deep neural networks on the collected data to empower their business with new competitive edges. However, a deep neural network usually has millions of model parameters, leading to large effective capacity that could be sufficient for encoding the details of individual data into model parameters. Our research addresses a collection of related topics within the context of deep learning with differential privacy. We provide more refined analysis of the privacy losses for differentially private stochastic gradient descent algorithms(SGD) for different data batching strategies including random reshuffling and random sampling. Also, we propose a family of methods for non-uniformly allocating privacy budget across SGD iterations to improve model accuracy while retaining privacy guarantees.

Last, we propose a differentially private data synthesis approach for data publication. Because the collection of individual data by governments and corporations can create tremendous opportunities for knowledge-based decision making, there is a demand for the exchange and publication of data among various parties. However, publishing data in its original form will violate individual privacy. Instead, releasing synthetic data that mimic original data provides a promising way for privacy preserving data publication while allowing rich data analytics. In particular, we propose to use deep generative models with differentially private training for location data synthesis, compared our approach with conventional methods that rely on sophisticated feature engineering, and examine the utility of synthesized data.

CHAPTER 1
INTRODUCTION

1.1 Motivation

The widespread use of internet-connected mobile devices, internet of things(IoT) and cloud computing has enabled a large scale collection of personal data, including user profiles, daily activities, locations, photos and health states, etc, of millions and billions of users from a wide range of scenarios such as the usage of mobile apps, smart home, and cloud storage services. The availability of these huge amounts of datasets has been driving the breakthrough in deep learning and explosion of data-driven applications for enriching human with life-enhancing experiences. However, because these datasets often encode privacy-sensitive information related to individuals, serious privacy concerns has been rto the society. Therefore, it is imperative to develop principled privacy preserving approaches to harvesting the power of those big data.

One type of widely collected data from users is the location data, due to the widespread use of mobile devices with GPS and popularity of location based services (LBSs) and applications, such as Uber, Yelp and Foursquare. On one hand, the emergence of location aware computing and location-based services creates great opportunities for empowering business with new competitive edges and enriching citizen with life-enhancing experiences. But on the other hand, such continuous collection of mobile users' location information may open doors to potential misuse and abuse of private location information and serious location privacy risks, such as exposing places that a user has visited, the travel patterns of a user, and using the location information to infer users' activities and uncover many unauthorized personal information such as their political views, religious affiliation, or state of health. Therefore, location privacy continues to attract significant attentions in recent years.

The wide availability of personal data, not only limited to location data, collected from individuals and the success of deep learning in a wide range of AI tasks spurs significant interest of applying machine learning techniques to extract useful knowledge from large collection of individual data and provide personalized services. However, recent studies on membership attacks and model inversion attacks have shown potential privacy risks from a number of dimensions. Deep neural networks have a large number of hidden layers, leading to large effective capacity that could be sufficient for encoding the details of some individual data into model parameters or even memorizing the entire data set [3]. It has been shown that individual information can be effectively extracted from neural networks [4, 5]. Therefore, there are severe privacy concerns accompanied with the wide deployment of deep learning applications and deep learning as a service platform. On the other hand, the publishing and sharing of trained models have always been of great interest in deep learning applications. The model owners can also publish their trained models to the cloud and allow other users to get predictions through APIs. In mobile applications, entire models are stored on-device to enable power-efficient and low-latency inference. Transfer learning [6], a key technique of deep learning, can leverage and adapt the already existing models to new classes of data, saving the effort of training the entire neural network from scratch. People who only have small datasets can use the model trained on a large dataset as fixed feature extractor in their neural networks or adapt the model to their own domain. A large amount of pre-trained models have been publicly available in model zoo repositories [7]. In these cases, the model parameters are completely exposed, making it easier for the adversaries to launch inference attacks, such as membership attacks [5] or model inversion attacks [4], to infer sensitive data records about individuals in the training datasets. Therefore, it is imperative to develop principled privacy preserving deep learning techniques to protect private training data against the adversaries with full knowledge of model parameters.

Not only in applying machine learning techniques on the collected dataset, there is also a large interest and demand in releasing and sharing of data among collaborators for research and development. For example, the location trace data collected from individuals can be shared with different parties for different purpose like city/traffic planning, location-driven advertising and human behavior study. But because of privacy concerns, data holders usually are wary or not willing to share their data. To solve such dilemma, privacy preserving data publishing has received considerable attention, which studies how to transform original data into the version immunized against privacy attacks but still allow effective data analytics. Traditional data anonymization techniques [8, 9, 10] transform the data by removing key identifiers or generalizing quasi-identifiers to ensure data privacy. However, it has been shown that they are susceptible to privacy attacks that ”de-anonymize” datasets via linkage to external or public datasets, with some notable examples on Netflix movie rating dataset [11], and the AOL search log [12] and the Washington State health record identification [13].

Data synthesis with differential privacy is a promising approach for privacy preserving data publication [14, 15, 16, 17, 18, 19], which generates synthetic data that mimic original data in terms of important characteristics and that can be released without compromising the privacy of individuals. The synthesized data can act as surrogate for the original dataset to allow data users to run arbitrary statistical analysis of their own as if they had the original data. However, existing methods rely on sophisticated feature engineering and probabilistcally modeling, which makes them difficult to apply on high dimensional data and implement in practice. They usually differ with each other on the selected features, it is not clear how they can adapt to new datasets. On the other hand, deep learning is powerful in learning from high dimensional data with its automatic feature learning and end-to-end modeling. Deep generative models have been proposed to model and generate text and image data. Therefore, deep learning with differential privacy can be an attractive approach for differntially private data synthesis.

1.2 Dissertation Scope and Contributions

The first contribution of this dissertation research is the development of PIVE, a two-phase Bayesian differential location privacy framework that aims to protect users location privacy in location based services while ensuring the service quality. In Phase I, we take into account the user-defined inference error threshold and the prior knowledge about the user's location to determine a subset of locations as the protection location set for protecting the actual location by increasing adversary's expected location inference error. In Phase II, we generate pseudo-locations (i.e., perturbed locations) in the way that satisfies a Bayesian differential privacy that is defined over the protection location set and achieves resiliency against Bayesian attacks. This two-phase location obfuscation is constructed dynamically by leveraging the relationship between two privacy notions based on adversary's current prior information and user-specific privacy requirements on different locations and at different times. Experiments with real-world datasets demonstrate that our PIVE approach effectively guarantees the two privacy notions simultaneously and outperforms the existing mechanisms in terms of adaptive privacy protection in presence of skewed locations and computation efficiency.

The second contribution of this dissertation research is the development of differentially private deep learning for protecting the privacy of the training data. we propose a differentially private approach to train the neural networks with several new techniques to optimize both the privacy loss and model accuracy. We employ a generalization of differential privacy, named as concentrated differential privacy(CDP), with refined privacy loss analysis on different data batching strategies. We implement a dynamic privacy budget allocator over the course of training to improve both the privacy loss and the model accuracy. Extensive experiments demonstrate that our approach effectively improves the privacy loss accounting, the training efficiency and the model quality under a given privacy budget.

Last, we propose Deepsynthesizer, a differentially private data synthesis approach for data publication. Because the collection of individual data by governments and corporations can create tremendous opportunities for knowledge-based decision making, there is a demand for the exchange and publication of data among various parties. However, publishing data in its original form will violate individual privacy. Instead, releasing synthetic data that mimic original data provides a promising way for privacy preserving data publication while allowing rich data analytics. We propose to use deep generative models with differentially private training for location data synthesis and examine the utility of synthesized data in a set of data analytics tasks.

1.3 Dissertation Organization

The rest of this dissertation is organized into three technical chapters and conclude in Chapter 5. Each chapter addresses one or more of the problems described above. In each chapter, we introduce the background of the problem being addressed, present our solution techniques followed by experimental evaluation, and describe related work. In Chapter 2, we present PIVE, our two-phase dynamic differential location privacy framework for protecting users location privacy in location based services. In Chapter 3, we introduce differentially private model publishing approach for deep learning for protecting the privacy of the training data. In Chapter 4, we present Deepsynthesizer, a differentially private data synthesis approach for data publication. We summarize the main contributions of this dissertation in Chapter 5.

CHAPTER 2
BAYESIAN DIFFERENTIAL LOCATION PRIVACY FOR LOCATION BASED SERVICES

2.1 Introduction

Location privacy research has drawn significant interests in recent years. Considering the high utility of location information and personalized privacy risk variations, instead of cryptographic solutions, a large body of location privacy research have been centered on the location obfuscation mechanisms that allow mobile travelers to use LBSs with perturbed location instead of exact location, referred to as *pseudo-location*, such that the release of the pseudo-location can prevent the disclosure of user-specific and request-specific sensitive location information [20, 21, 22, 23, 24, 25], while maintaining desired utility of location information.

Recently, geo-indistinguishability [20] and expected inference error [26, 22] are proposed in the literature as the two statistical notions of location privacy. Geo-indistinguishability is derived from differential privacy [27] and ensures that for any two location points that are geographically close, the location obfuscation mechanism will produce a pseudo-location with similar probabilities. The expected inference error, as a statistical metric instead, takes into account the prior information of an adversary about user's location, and measures location privacy by the expected distance between the estimated location by the adversary and the true location. A number of location obfuscation mechanisms [22, 28] have been developed solely based on the privacy notion of expected inference error.

In this paper, we argue that geo-indistinguishability and expected inference error are two complementary notions for location privacy. Existing geo-indistinguishable mechanisms [20, 21] guarantee location privacy with respect to the information leakage through a differential privacy based location obfuscation mechanism, but they do not consider the inference attacks using prior knowledge [25]. We performed the bound analysis to formally examine the relationship between geo-indistinguishability and expected inference error. We show that geo-indistinguishability may not adequately protect the absolute privacy of user's location against inference attacks with using prior information. On the other hand, the mechanisms with expected inference error as privacy metric are constructed based on the assumption of certain types of prior information that the adversary may have, but without consideration of constraint on the posterior information gain from the release of pseudo-locations. These mechanisms may be vulnerable to inference attacks with arbitrary prior knowledge. Thus, we argue that a strategic combination of the two privacy notions can double shield location privacy by simultaneously limiting information leakage of the location perturbation mechanism and ensuring the inference error to be constrained for inference attacks with prior information the adversary may have.

In addition to combining the two privacy notions for effective defense against inference attacks, we also argue that an effective location obfuscation mechanism should maintain desired location utility and service quality for respective mobile users and their LBSs. In practice, mobile users may have very different privacy requirements for different types of LBSs. Even for the same LBS, users may have different privacy demands for different locations or for the same location at different times. For example, a user may want the expected inference error of adversary to be larger than 1km when he is in a hospital or a religious event, but may reduce this requirement to 200 meters when he is in a restaurant with a lot of other restaurants nearby; or the user may not care about privacy at some places (e.g., her home or office) during certain periods of a day, but needs the privacy at other places, such as her travel routes and stops along some trajectories.

In this paper, we propose to design a Bayesian differential location privacy mechanism that can protect location against Bayesian inference attacks with personalized error bounds. First, we formally study the relationship between geo-indistinguishability and expected inference error and examine their limitations through experimental study. The relationship between two privacy notions helps to determine the noise level of location obfuscation required for protecting a location against inference attacks. Second, we allow users to define personalized error bound for each of their locations and introduce the concept of *protection location set* for each location, which identifies the neighborhood locations based on both the personalized error bound constraint and the prior distribution that the adversary may have based on historical locations of a user, her mobility model or the population density. Based on the above development, we design a two-phase dynamic differential location privacy framework, called PIVE, which integrates geo-indistinguishability and expected inference error to effectively protect location privacy against two popular types of inference attacks: optimal inference attack and Bayesian inference attack. This framework constructs pseudo-locations dynamically and adaptively, based on multiple pieces of information that may change frequently in the spatial-temporal context of a mobile user, such as the user's current location at the time of her service request, her current location privacy requirements, her location utility and LBS quality preferences, and the prior information that the adversary may have at this time. In Phase I, we utilize the user-defined inference error threshold and the prior knowledge about the user's location to determine the protection location set for protecting the actual location of a user and ensuring the lower bound of adversary's expected location inference error over this protection location set. In Phase II, we generate pseudo-locations that achieve geo-indistinguishability on this protection location set. The former aims to bound the expected inference error in the worst case and the latter aims to scope the possible posterior information leakage. The PIVE approach provides dynamic differential location privacy with personalized error bound and can work adaptively in presence of skewed prior distribution of locations and efficiently for the scenarios in which users may have personalized and non-uniform privacy needs at different locations and for different LBSs.

Previous work [25] by Shokri is the first to identify the need for integrating the two privacy notions and to propose a joint optimization approach. This approach combines the two privacy notions together in parallel in a linear program and produce the distribution of perturbed locations statically once for all locations in an area, and we refer to it as the global optimization approach. Compared to the joint optimization [25], PIVE takes a sequential and local approach to combine two privacy notions. It separately applies the expected inference error metric first, which produces a neighborhood protection location set for the user's location by leveraging user defined error bound and the prior information, and then produces the perturbed location by ensuring geo-indistinguishability and at the same time increasing the resilience of perturbed location against inference attacks. Another feature of PIVE that is different from the joint optimization approach is to leverage the user defined personalized error bound (threshold) for different locations or for the same location at different times and for computing the protection location sets dynamically and adaptively. This allows PIVE to balance privacy and utility for different locations while meeting the personalized inference error bound constraint for perturbed locations.

PIVE algorithms are highly efficient in terms of computation complexity, compared to existing mechanisms that need to solve a linear program with $|\mathcal{X}|^2$ decision variables and up to $O(|\mathcal{X}|^3)$ constraints for previous joint optimization approach [25], where $\mathcal{X}$ is the number of all possible locations of a user. First, PIVE only requires the search of a protection location set locally within the neighborhood of a user's current location by leveraging user-defined error bound, and simple probability computation for the exponential mechanism. This locality based design enables PIVE to adapt to the dynamic changes of both prior information and privacy preferences per location more efficiently. Second, PIVE adaptively adjusts the noise level of location obfuscation to prior information through searching a protection location set under the minimum inference error bound constraint, which provides Bayesian differentially private mechanism to generate perturbed locations. We implement the PIVE dynamic location obfuscation mechanism and evaluate PIVE with real-world datasets. Our experimental results show that the PIVE approach effectively guarantees the two privacy notions simultaneously and outperforms the existing mechanisms that secure geo-indistinguishability or that quantify location privacy by expected inference errors.

2.2 Overview

In this section we first introduce the notation of differential privacy, describe the model of location obfuscation and the adversary model used in this paper. Then, we state the problem to be addressed in this paper.

2.2.1 Differential Privacy

Differential privacy is a rigorous mathematical framework that offers provable privacy guarantees for protecting individual data in statistical databases and has recently become a de-facto standard for privacy. It ensures that arbitrary changes to a single individual's row result in only statistically insignificant changes in the outcome of a data analysis. Formally,

Definition 1 (Differential Privacy [27]). *A randomized mechanism $\mathcal{A}$ provides ϵ-differential privacy if for any two neighboring database D_1 and D_2 that differ in only a single entry, $\forall S \subseteq Range(\mathcal{A})$,*

$$\frac{\Pr(\mathcal{A}(D_1) \in S)}{\Pr(\mathcal{A})(D_2) \in S)} \leq e^{\epsilon} \tag{2.1}$$

The standard approach to achieve differential privacy is the sensitivity method [29, 27] (e.g., Laplacian mechanism) that adds to the query output the noise proportional to the sensitivity of the query function. The sensitivity measures the maximum change in the query answers due to the change of a single database entry.

Definition 2 (Sensitivity [29]). *The sensitivity of a query function $q : \mathcal{D} \to \mathbb{R}^d$ is*

$$\Delta q = \max_{D_1, D_2} ||q(D_1) - q(D_2)||_1 \tag{2.2}$$

where D_1, $D_2 \in \mathcal{D}$ are any two neighboring datasets that differ at most one element, $|| \cdot ||_1$ denotes L_1 norm.

To achieve ϵ-differential privacy, the Laplacian mechanism perturbs the output by $q(D)+Lap(\Delta q/\epsilon)$, where $Lap(*) = (Z_1, \ldots, Z_d)$ in which Z_i are drawn i.i.d from Laplace distribution. Such differentially private mechanism ensures that two neighboring datasets are indistinguishable on the distribution of query answers.

The exponential mechanism [30] is another mechanism that preserves ϵ-differential privacy. Given the output range R, a utility function $u : D \times R \to \mathbb{R}$ is defined, which maps the dataset/output pairs to utility scores. The sensitivity of utility function u is

$$\Delta u = \max_{r \in R} \max_{D,D'} |u(D, r) - u(D', r)| \tag{2.3}$$

over any two neighboring datasets D and D'.

Definition 3 (The exponential mechanism [30])**.** *The exponential mechanism selects and outputs an element $r \in R$ with probability proportional to exp$(\frac{\epsilon u(D,r)}{2\Delta u})$.*

2.2.2 Location Obfuscation Mechanism

In this paper we are interested in the location based services in which the users sporadically reveal their locations for issuing spatial queries, e.g., finding the nearby points-of-interests or friends. We do not consider the protection of the users' identities that prevents the adversary to discover which user issues the query. In this case, the typical way to preserve the users' location privacy is to randomly obfuscate the user's actual location to a pseudo-location and report this pseudo-location to the location based service providers. In this paper we assume discretized locations as in [21, 22] and use $\mathcal{X}$ to denote the set of the user's possible locations. An obfuscation mechanism determines the random mapping between the user's actual locations A and pseudo-locations O, with following the probability distribution

$$f(x'|x) = \Pr(O = x'|A = x) \qquad x, x' \in \mathcal{X} \tag{2.4}$$

That is, it takes the actual location x as input and chooses a pseudo-location x' by sampling from the distribution $f(x'|x)$. An obfuscation mechanism is indeed a specification of probability distributions $f(\cdot|\cdot)$ over $\mathcal{X}$. Different obfuscation mechanisms determine such probability distributions in different ways.

2.2.3 Adversary Model

This paper assumes the adversary that has prior knowledge about user's location. We argue that the prior information about users' locations inherently exists because of the publicly available transportation information, geographical information of points of interest, road networks, residential area, population distribution, and human movement pattern, etc. Following previous works [26, 22], the prior knowledge is captured by a prior (probability) distribution π over the set of possible locations of the user, $\mathcal{X}$. The adversary can build π for the target user in multiple ways:

- Using the population density or popularity [31, 32] of every place as π that can be obtained from public traces, check-in datasets or demographic information;
- Using the user's historical access information to a location based service that records his locations from which he sent location based queries [22].
- Using the mobility pattern modeled by Markov chain to infer the possible locations of a user at current time and their probabilities given his previous disclosed locations [28].

In this paper we assume the adversary with prior knowledge of π regardless of in which way it is derived. We also assume that the adversary also knows the location obfuscation mechanism, i.e, how it works and the distribution f. Such adversary is called an informed adversary [29].

The adversary's goal is to infer the user's actual location x. Once the adversary observes the pseudo-location x' reported by the user, he computes the posterior probability distribution, $\Pr(x|x')$ for $x \in \mathcal{X}$, i.e., the probability that x is the actual location that generated

x':

$$\Pr(x|x') = \frac{\pi(x)f(x'|x)}{\sum_{x\in\mathcal{X}}\pi(x)f(x'|x)} \tag{2.5}$$

Based on the posterior distribution, a *Bayesian adversary* can perform **optimal inference attack** [22] which aims to minimize his expected inference error, i.e., the expected distortion between the estimated location $\hat{x}$ and user's actual location x, given an observed pseudo-location x'. That is,

$$\hat{x} = \arg\min_{\hat{x}\in\mathcal{X}} \sum_{x\in\mathcal{X}} \Pr(x|x')d_p(\hat{x}, x) \tag{2.6}$$

where d_p can be Hamming distance or Euclidean distance between locations, or their semantic dissimilarity, which captures the privacy loss from inference attack. We assume d_p to be Euclidean distance d for optimal inference attack.

If d_p is Hamming distance, for which $d_p(\hat{x}, x) = 0$ if $\hat{x} = x$, and $d_p(\hat{x}, x) = 1$ otherwise, it is easy to see that the optimal inference attack actually guesses the actual location as the one having the maximum posterior probability. We call this attack as **Bayesian inference attack**, represented by

$$\hat{x} = \arg\max_{x\in\mathcal{X}} \Pr(x|x') \tag{2.7}$$

2.2.4 Problem Statement

We can categorize the existing research on quantifying location privacy into two broad categories based on two notions of location privacy: geo-indistinguishability and expected inference error. The location privacy solutions that promote geo-indistinguishability are primarily based on the theory of differential privacy [27]. The solutions that quantify location privacy by the amount of expected inference error are typically based on Bayesian theory and thus are referred to as Bayesian optimal mechanisms. The class of solutions based on geo-indistinguish-ability protect location privacy without any assumption of adversary's prior information but consequently do not consider absolute location privacy against inference attacks in terms of expected inference error when the adversary has some prior knowledge about the user's exact location or past released locations. In contrast, the Bayesian optimal mechanisms advocate the background inference resilient location privacy but are not as robust as geo-indistinguishability against adversary with arbitrary prior information.

The problem statement can be summarized from three dimensions. First, geo-indistinguishability and expected inference error are two complementary privacy notions for protecting location privacy against inference attacks. It is critical to understand the relationship between the two privacy notions, and the limitations of existing location obfuscation mechanisms that support only one of the two privacy notions. Second, it is not only beneficial but also feasible to develop a location obfuscation mechanism that can effectively integrate the two privacy notions. Third, incorporating user-defined constraint, such as minimum inference error bound, not only improves the usability perspective, which is critical for the wide deployment of privacy protection models, but also enables adaptive noise adjustment for geo-indistinguishability and supports customizable privacy/utility requirement of mobile users that allows personalized error bounds at different locations, different times, and for different LBSs. This motivates the design and implementation of PIVE, a two-phase dynamic differential location privacy framework for ensuring both notions of location privacy with personalized error bounds.

2.3 Location Privacy Notions

In this section we provide a detailed analysis and illustration of the two location privacy notions: expected inference error and geo-indistinguishability. We first briefly describe each notion, its respective location perturbation model, compare the mechanisms based on these two privacy notions and identify and illustrate their inherent problems through both formal and experimental analysis.

2.3.1 Expected Inference Error

Under the inference attack of Bayesian adversary, the location privacy offered by a mechanism is measured by the expected inference error of the adversary averaged over all possible locations in $\mathcal{X}$, referred to as unconditional expected inference error [26, 22], computed as

$$\sum_{x' \in \mathcal{X}} \Pr(x') \min_{\hat{x} \in \mathcal{X}} \sum_{x \in \mathcal{X}} \Pr(x|x') d_p(\hat{x}, x) \tag{2.8}$$

$$= \sum_{x' \in \mathcal{X}} \min_{\hat{x} \in \mathcal{X}} \sum_{x \in \mathcal{X}} \pi(x) f(x'|x) d_p(\hat{x}, x) \tag{2.9}$$

Similarly, the service quality loss is measured by the unconditional expected distance between actual location and reported pseudo-location over the quality metric $d_q(\cdot)$, i.e.,

$$\sum_{x \in \mathcal{X}} \sum_{x' \in \mathcal{X}} \pi(x) f(x'|x) d_q(x', x) \tag{2.10}$$

where d_q determines the quality loss by reporting x' instead of actual location x. Since the accuracy of location based queries like nearest neighbor and range queries usually depends on the Euclidean distance between the actual location and reported location, we use the Euclidean distance d as d_q, as in previous works [21, 25].

An optimal mechanism [22] has been proposed to maximize the expected inference error (resp. service quality) given the constraint on the service quality loss (resp. expected inference error). In such approach, privacy and quality are controlled in terms of these global performance metrics that are averaged over all locations, which does not provide users a straightforward way to explicitly specify different privacy/quality requirements at different locations and times. Also, for the prior information that is dynamically built by the adversary with mobility model [28], a linear program has to be recomputed under every change. More importantly, the construction relies on the assumption about adversary's prior information, different prior information with higher accuracy level may cause privacy degradation of the mechanism, as shown in [25].

We note that the upper limit of expected inference error is achieved when the maximum tolerable service quality loss becomes sufficiently large or not bounded. In this case, the pseudo-locations are generated independently of user's locations, and the adversary's best strategy is to make guess based on prior distribution. Therefore, the upper limit of expected inference error is

$$ExpErr_{max} = \min_{\hat{x}} \sum_{x \in \mathcal{X}} \pi(x) d_p(\hat{x}, x) \tag{2.11}$$

2.3.2 Geo-indistinguishability

A mechanism satisfies ϵ_g-geo-indistinguishability [20] iff for all x, y,

$$\frac{f(x'|x)}{f(x'|y)} \leq e^{\epsilon_g d(x,y)} \tag{2.12}$$

where $d(x, y)$ is the Euclidean distance between x and y. It ensures that for two locations that are geographically close, the probability distributions of pseudo-locations generated at them are similar. Note, as shown in [20], ϵ_g is decided by a privacy parameter ϵ (≥ 0) and the range of circular region centered at the user's location x. Essentially it means that geo-indistinguishability aims to protect this circular region with guaranteeing ϵ-differential privacy over it. Because the actual location is protected by being hidden among all the locations in the region due to their similar probability distributions for generating pseudo-locations, we call such region as the *protection region* and the set of locations within the region as the *protection location set*. Let D be the diameter of protection region and $\epsilon_g = \epsilon/D$, we say the mechanism is ϵ-differentially private over the region if for any two locations x and y in the protection region

$$e^{-\epsilon} \leq \frac{f(x'|x)}{f(x'|y)} \leq e^{\epsilon}. \tag{2.13}$$

Upper bound of posterior probability: Let Φ be the protection region. An upper bound of the posterior distribution of location $x \in \Phi$, given any observed pseudo-location x', can be obtained as follows:

$$\Pr(x|x') = \frac{\pi(x)f(x'|x)}{\sum_{y\in\mathcal{X}}\pi(y)f(x'|y)} \tag{2.14}$$

$$= \frac{\pi(x)f(x'|x)}{\sum_{y\in\Phi}\pi(y)f(x'|y) + \sum_{y\in\mathcal{X}\setminus\Phi}\pi(y)f(x'|y)} \tag{2.15}$$

$$\leq \frac{\pi(x)f(x'|x)}{\sum_{y\in\Phi}\pi(y)f(x'|y)} \tag{2.16}$$

$$= \frac{\pi(x)}{\sum_{y\in\Phi}\pi(y)f(x'|y)/f(x'|x)} \tag{2.17}$$

Applying (2.13),we have

$$\leq \frac{\pi(x)}{\pi(x) + e^{-\epsilon}\sum_{y\in\Phi,y\neq x}\pi(y)} \tag{2.18}$$

Since $0 < e^{-\epsilon} < 1$,we have

$$\leq e^{\epsilon}\frac{\pi(x)}{\sum_{y\in\Phi}\pi(y)} \tag{2.19}$$

The upper bound of posterior probability (2.19) implies that no matter what prior information the adversary has, geo-indistinguishability constrains the multiplicative distance between posterior distribution $\Pr(x|x')$ and prior distribution $\frac{\pi(x)}{\sum_{y\in\Phi}\pi(y)}$ within e^{ϵ}, and thus limits the posterior information gain of the adversary. This makes location obfuscation more robust against Bayesian adversary compared with the Bayesian mechanism [22] that could be constructed with incomplete knowledge about the adversary's prior information.

Lower bound of inference error: We further consider location privacy in terms of expected inference error. Let z be the estimated location by the adversary, i.e., $z = arg\min_{\hat{x}}\sum_{x\in\mathcal{X}}\Pr(x|x')d(\hat{x},x)$. The conditional expected inference error is

$$\sum_{x\in\mathcal{X}}\Pr(x|x')d_p(z,x) \tag{2.20}$$

Here we consider the lower bound for it, which is indeed achieved in the worst case that the adversary narrows possible guesses to the location set within the protection region that contains the user's actual location. Therefore, the lower bound is

$$\min_{\hat{x}\in\mathcal{X}} \sum_{x\in\Phi} \frac{\Pr(x|x')}{\sum_{y\in\Phi}\Pr(y|x')} d_p(\hat{x}, x) \tag{2.21}$$

Let $z' = arg\min_{\hat{x}\in\mathcal{X}} \sum_{x\in\Phi} \frac{\Pr(x|x')}{\sum_{y\in\Phi}\Pr(y|x')} d_p(\hat{x}, x)$, the above becomes

$$= \sum_{x\in\Phi} \frac{\Pr(x|x')}{\sum_{y\in\Phi}\Pr(y|x')} d_p(z', x) \tag{2.22}$$

$$= \sum_{x\in\Phi} \frac{\pi(x)f(x'|x)}{\sum_{y\in\Phi}\pi(y)f(x'|y)} d_p(z', x) \tag{2.23}$$

Using (2.13), we have

$$\geq e^{-\epsilon} \sum_{x\in\Phi} \frac{\pi(x)}{\sum_{y\in\Phi}\pi(y)} d_p(z', x) \tag{2.24}$$

$$\geq e^{-\epsilon} \min_{\hat{x}\in\Phi} \sum_{x\in\Phi} \frac{\pi(x)}{\sum_{y\in\Phi}\pi(y)} d_p(\hat{x}, x) \tag{2.25}$$

where we have the derivation from (2.24) to (2.25) given that Φ is convex and thus the minimum is obtained when $\hat{x}$ is the weighted geometric median of Φ that lies in the region.

The bounds of posterior probability (2.19) and inference error (2.25) indicate the capability of geo-indistinguishability for defending against Bayesian inference attack (2.7) and optimal inference attack (2.6) respectively. Both of them depend on the prior distribution over protection region Φ, which suggests that geo-indistinguishability may not provide enough location protection against Bayesian adversary with sufficient prior information. The protection of geo-indistinguishability only measures the impact of user's location on the output, but not the inference capability of Bayesian adversary with his prior information. We have argued that certain prior knowledge to identify the user's location inherently exists, but geo-indistinguishable mechanisms produce pseudo-locations as if the adversary does not have any prior knowledge.

Also, we can see that it has limitations for geo-indistinguishable mechanisms in existing works [20, 21, 23, 24] to use uniform differential privacy parameter and protection region radius, independently of the user's locations. Because the prior distribution over protection regions around different locations are mostly different, geo-indistinguishablity may not achieve the same level privacy against Bayesian adversary, indicated by bounds (2.19) and (2.25) that change with priors. For example, in an urban area with many possible locations densely distributed, the user can use a small radius r for his protection region in which ϵ-differential privacy is achieved; but in a rural area, when the user's location is only possible location within it, using a small radius to generate a pseudo-location does not provide sufficient protection. This is indicated by that the upper bound (2.19) achieves maximum e^{ϵ} (≥ 1)and the lower bound (2.25) becomes zero, which actually means no bound for the posterior probability and inference error. Indeed, the adversary can easily associate the pseudo-location with the actual location given the prior knowledge that there is only one possible location in this area.

2.3.3 Experimental Illustration

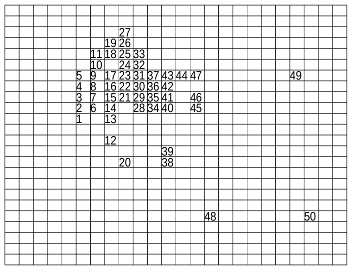


Figure 2.1: The 50 regions in the location dataset.

In this section we evaluate the privacy of geo-indistinguishability against optimal and Bayesian inference attack and validate our analysis result in previous section. In order to see the lack of protection against inference attacks with geo-indistinguishability, we compare a geo-indistinguishable mechanism with a mechanism constructed with expected inference error as privacy metric that is optimal against inference attacks. Two mechanisms are given below:

- The optimal ϵ_g-geo-indistinguishable mechanism [21], denoted by M_{ϵ_g}, that minimizes the service quality loss (2.10) subject to geo-indistinguishability (2.12);

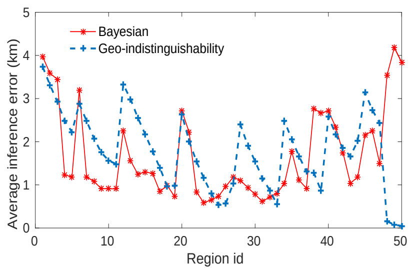


Figure 2.2: The average inference error of optimal inference attack.

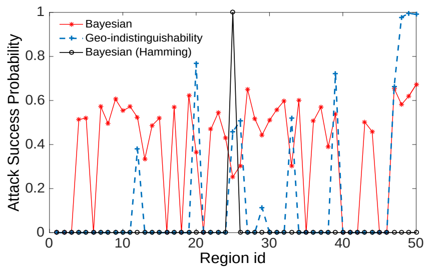


Figure 2.3: The success probability of Bayesian inference attack.

- The optimal Bayesian mechanism [22], denoted by M_B, that maximizes the expected inference error (2.8) under the constraint of the maximum tolerable service quality loss $Q_{loss}^{\max}$ for (2.10).

We choose them because it has been shown in [21] that M_{ϵ_g} and M_B can achieve the same level of location privacy in terms of expected inference error (2.8) defined with Euclidean distance, which enables us to make a fair comparison of them under optimal inference attack. To achieve that, given M_{ϵ_g} and the minimum quality loss q it obtains, M_B is derived with letting $Q_{loss}^{\max} = q$. Besides, we are particularly interested in the local performance of the mechanisms for protecting each location, rather than only considering the global average metrics as previous works [21, 22].

In our experiment we use location data extracted from Geo-life dataset [33, 34, 35]. The details of data processing is described in Section 2.5. Specifically, here we use all-day location data of a single user with id 0, and consider 50 regions shown in Figure 2.1 as $\mathcal{X}$. The prior for the user is computed by counting and normalizing the number of his/her location points falling into each of 50 regions.

Optimal inference attack

We use $\epsilon_g = 0.9$ for M_{ϵ_g} that incurs minimum quality loss 0.89km. Letting $Q_{loss}^{\max} = 0.89$km, we follow the approach in [22] to obtain M_B with maximum expected inference error 0.89km. We simulate a user using two mechanisms at every region in $\mathcal{X}$, repeat the simulation 1000 times, and measure the inference error, i.e., the distance between the actual location and the location inferred by optimal inference attack (2.6), averaged over 1000 times for every region. The result is shown in Figure 2.2.

Though both M_{ϵ_g} and M_B can guarantee the expected inference error 0.89km at most locations, we can see that M_{ϵ_g} has almost zero inference error at regions with id 48, 49 and 50, but the optimal solution against the inference attack M_B incurs much larger inference error at them. This indicates that geo-indistinguishability does not provide sufficient privacy protection against optimal inference attack at these locations. The essential reason is that geo-indistinguishability does not consider any prior distribution the adversary may have. As we can see from Figure 2.1, region 48, 49 and 50 are isolated locations on the prior distribution over $\mathcal{X}$, which means zero probabilities for any other locations in their neighborhood. Such skewed probability distribution lets the upper bound of posterior probability (2.19) to be e^{ϵ} (larger than 1) given any pseudo-locations reported from these isolated locations, which means no bound for the posterior probability and thus it can get close to one on the true locations. Similarly, the lower bound of expected inference error in (2.25) becomes zero at these regions, meaning no guarantee for location privacy in terms of expected inference error. Consequently, with minimization on the quality loss, M_{ϵ_g} has probability larger than 0.9 to report truthfully at these regions, and the posterior probability $\Pr(x|x')$ conditioned on $x' = 48, 49, 50$ get close to one on $x = 48, 49, 50$ respectively.

In Figure 2.4, we vary ϵ_g from 0.7 to 0.1 and $Q_{loss}^{\max}$ from 1 to 2 and measure the expected inference error (2.8). We can see that both M_{ϵ_g} and M_B have the expected inference error to increase to 1.178 and remain the same after that. This value is exactly the value calculated by (2.11), which validates the upper limit (2.11). We note M_{ϵ_g} and M_B achieve this limit in different ways: M_{ϵ_g} always chooses the same region 25 as pseudo-location for any user's location, but M_B turns out to uniformly sample a location from $\mathcal{X}$ as the pseudo-location. In essence, both break the dependency between pseudo-locations and actual locations.

Bayesian inference attack

For Bayesian inference attack, we are interested in the probability that Bayesian inference attack makes correct guesses about the actual location for a user. We replace optimal inference attack in the above simulation with Bayesian inference attack, repeat the simulation 1000 times, and calculate the percentage of successful guesses at every location. Note that M_B is constructed with using Euclidean distance based privacy metric. With using Hamming distance, an optimal Bayesian mechanism against Bayesian inference attack M'_B can be derived in the same way as M_B. Since we focus on Euclidean distance based privacy notions and Hamming distance does not guarantee it, M'_B is only used as a reference for examining the resilience of the two other mechanisms constructed with Euclidean distance based privacy notion against Bayesian inference attack. With the same $Q_{loss}^{\max} = 0.89$km, Figure 2.3 shows the attack success probabilities of three mechanisms. As we can see, M'_B has zero attack success probabilities at all regions except region 25, which demonstrates M'_B's optimality with using Hamming distance against Bayesian inference attack compared with M_B and M_{ϵ_g}. M_B has at least 50% success probabilities at 50% regions but M_{ϵ_g} has zero attack success probabilities at more than 70% regions. The result demonstrates the improvement introduced by geo-indistinguishability compared with M_B, as a result of limiting the relative ratio between posterior distribution and prior distribution.

For M_{ϵ_g}, there are multiple locations, i.e., region 48, 49 and 50, with high success probabilities close to 1. This is because that the posterior probability $\Pr(x|x')$ conditioned on most pseudo-locations reported from these regions is much higher on the actual locations (close to 1) than on others, which also causes lowest inference error at these regions in Figure 2.2. Intuitively, a region has weak protection against inference attacks if the pseudo-locations generated from it are highly associable with the true location, represented by high posterior probability on the true location. Our result demonstrates that no guarantee on the bounds (2.19) and (2.25) for inference attacks allows such strong association to happen to the skewed locations with M_{ϵ_g}, leading to weak protection for these regions. Note that, because for M'_B region 25 is the maximum point of posterior distributions $\Pr(x|x')$ for any x', the adversary always guess 25 no matter where the user is, and thus region 25 has success probability 1. It is not because of the weak protection as for regions 48-50 with M_{ϵ_g}. The posterior probability achieves maximum on region 25 but only up to 0.37, which is much smaller than that on regions 48-50 with M_{ϵ_g}. High posterior probability incurs both low inference error and high success probability, and skewed locations actually show the worst case vulnerabilities of that, thus in our experiment we have particular discussions for such regions.

Because the user has different probabilities to visit every region, we evaluate the expected success probability of Bayesian inference attack as follows:

$$P_s = \sum_{x \in \mathcal{X}} \sum_{x' \in \mathcal{X}} \pi(x) f(x'|x) c(x, x') \tag{2.26}$$

where $c(x, x') = 1$ if the actual location x is correctly inferred given pseudo-location x', i.e., $x = \arg\max_{y \in \mathcal{X}} \Pr(y|x')$; otherwise $c(x, x') = 0$. We compute P_s for two mechanisms shown in Figure 2.3, and obtain $P_s = 0.19$ for M_B and $P_s = 0.27$ for M_{ϵ_g}. M_B that is constructed with prior information against Bayesian inference attack achieves better privacy than M_{ϵ_g}. We also measure the expected attack success probability for M_{ϵ_g} with different privacy parameter ϵ_g, shown in Figure 2.5. Smaller ϵ_g indicates higher privacy. When ϵ_g gets close to zero, the multiplicative distance between posterior and prior distribution approaches to one, which means that the adversary cannot do better than just guessing the actual location by prior knowledge. That's why the curve becomes flat when ϵ_g approaches to zero in Figure 2.5. In contrast, when ϵ_g (as also ϵ) increases, the upper bound of posterior probability (2.19) increases, letting the mechanism truthfully report the

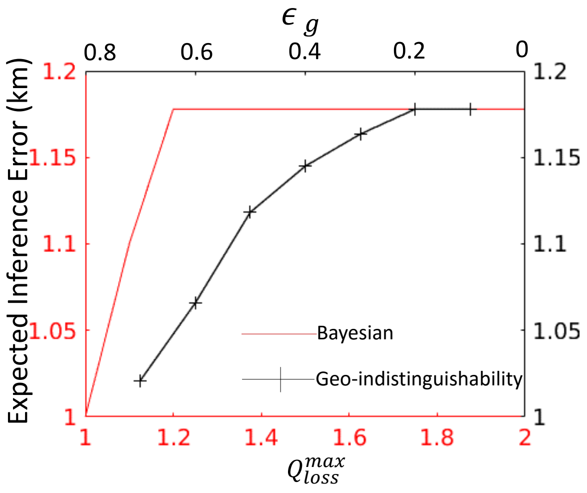


Figure 2.4: Max Expected Error.

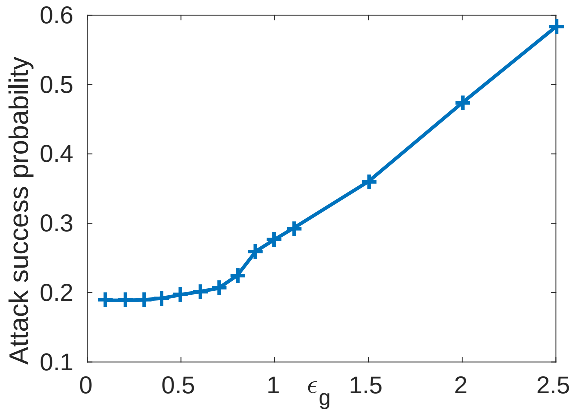


Figure 2.5: ϵ_g v.s. P_s.

locations with higher probabilities under quality loss minimization.

2.3.4 Our Design Objective

From our formal and experimental analysis, we can see that geo-indistinguishability limits the privacy leakage by bounding the relative information gain of the adversary given observed pseudo-locations, regardless of what kind of prior information the adversary may have. But it does not ensure absolute location privacy guarantee in terms of expected inference error against inference attacks (2.6) and (2.7). Bayesian optimal mechanisms protect location privacy by maximizing expected inference error against inference attacks but require assuming a prior location distribution that the adversary has, which is not robust against adversaries with arbitrary knowledge. Thus, it is desirable to have both privacy notions in a location obfuscation mechanism. On the other hand, existing geo-indistinguishable mechanisms suppose uniform differential privacy parameters over every location, which may either cause unnecessarily large noise level at some locations or insufficient noise level at others leading to privacy disclosure, and their formulations do not provide the user a straightforward way to customize his privacy preference for his current location. Considering these issues, we aim to design a location obfuscation mechanism that can effectively combine geo-indistinguishability and expected inference error, while operating adaptively with supporting customizable privacy preferences for the users.

2.4 Our Solution Approach

In this section we describe PIVE, a two-phase dynamic approach to protect location privacy in terms of both geo-indistinguishability and expected inference error. We first present the PIVE, two phase location obfuscation framework, and then describe each phase in detail. In the first phase, we determine a set of locations (i.e., protection location set) to protect user's actual location, with guaranteeing the expected location inference errors with the user-defined threshold and the adversary's prior knowledge with respect to the user's location. we develop a Hilbert curve based method and its optimization for efficiently and accurately determining the protection location set. In the second phase, we devise a differentially private mechanism to generate pseudo-locations with strong utility guarantee with respect to the service quality.

2.4.1 PIVE Two-Phase Framework

Our goal is to design a mechanism that achieves geo-indistinguishability while providing lower bound on expected inference error against optimal inference attacks. A challenging problem is how to integrate both privacy notions to a mechanism designed to obfuscate locations instantly and adaptively. Basically, our solution is to dynamically choose a protection location set to guarantee expected inference error and produce pseudo-locations in a differentially private way for every location in the set.

To introduce our approach, we first define ϵ-differential location privacy over an arbitrary region containing the actual location, as opposed to geo-indistinguishability that is defined over the circular neighborhood centered at the actual location.

Differential privacy requires that a query function has unsubstantial difference for the outputs over any two neighboring datasets that differ only in a single element. The location obfuscation mechanism for a user only involves a single data record, i.e., his current location. Differentially private location obfuscation requires the definition of "neighboring" location points to the user's location, such that they have the similar probabilities to produce a pseudo-location. The neighborhood consisting of all "neighboring" locations indeed functions as "a minimum crowd" for the actual location to "be hidden in a crowd". As mentioned in Section 2.3.2, previous geo-indistinguishabile mechanisms [20, 21] actually regard the circular region centered at the user's location with a uniform radius as such neighborhood for protection. In this paper, we define "neighboring" relationship over a set of locations in an arbitrary region that contains the user's actual location, referred to as *protection location set*, and accordingly differentially private location obfuscation is defined as follows:

Definition 4. *A randomized location obfuscation mechanism $f(\cdot|\cdot)$ satisfies ϵ-differential location privacy on protection location set Φ, if for any locations $x, y \in \Phi$, and any output x',*

$$\frac{f(x'|x)}{f(x'|y)} \leq e^{\epsilon} \tag{2.27}$$

Based on the upper bound of posterior probability (2.19), here ϵ is chosen to achieve a desired bound of the multiplicative distance between posterior distribution and prior distribution, to limit the adversary's posterior information gain.

Then, we consider how to guarantee the expected inference error via protection location set. Let

$$ExpEr(x') = \min_{\hat{x}\in\mathcal{X}} \sum_{x\in\mathcal{X}} \Pr(x|x')d(\hat{x}, x) \tag{2.28}$$

$$E(\Phi) = \min_{\hat{x}\in\Phi} \sum_{x\in\Phi} \frac{\pi(x)}{\sum_{y\in\Phi}\pi(y)} d(\hat{x}, x) \tag{2.29}$$

where we choose Euclidean distance as privacy metric as previous works [21], $ExpEr(x')$ is the conditional expected inference error given any observed pseudo-location x'. For optimal inference attack with using x', according to the lower bound result for expected inference error in (2.25), we have

$$ExpEr(x') \geq e^{-\epsilon}E(\Phi) \tag{2.30}$$

To ensure a lower bound for conditional expected inference error $ExpEr(x')$, we introduce privacy parameter E_m that is specified by the user according to his current location's sensitivity, such that $\forall x'$, $ExpEr(x') \geq E_m$. To ensure that, it is sufficient to satisfy that

$$E(\Phi) \geq e^{\epsilon}E_m \tag{2.31}$$

Then, we have the following theorem (with the above as a proof):

Theorem 1. *For a location obfuscation mechanism that achieves ϵ-differential location privacy on protection location set Φ, if $E(\Phi) \geq e^{\epsilon}E_m$, the optimal inference attack using any observed pseudo-location x', $ExpEr(x') \geq E_m$.*

Based on (2.31), we regard Φ as a variable and propose to dynamically search a region of Φ where the user is located to satisfy $E(\Phi) \geq e^{\epsilon}E_m$. Then, with the protection location set Φ, we propose an exponential mechanism that generates a pseudo-location in the way that achieves ϵ-differential location privacy on Φ, defined in Definition 4. Because the maximum change of the user's location is within the range of Φ, the sensitivity method to achieve differential privacy introduces the noise perturbation proportional to Φ's diameter $D(\Phi)$, as shown in Section 2.4.3. To maximize the utility, the noise magnitude should be

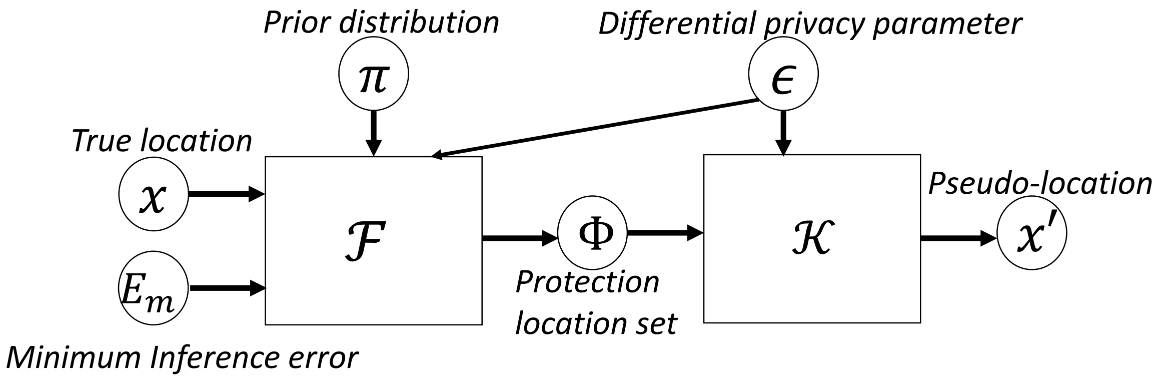


Figure 2.6: The framework of PIVE. $\mathcal{F}$ is the algorithm to determine the protection location set Φ and $\mathcal{K}$ is the differential mechanism to produce a pseudo-location.

minimized, and thus it is desired to find Φ that satisfies (2.31) with a minimum diameter.

Figure 2.6 illustrates the workflow of PIVE. It shows two components with their inputs: the algorithm $\mathcal{F}$ for generating the protection location set and the differentially private mechanism $\mathcal{K}$ for producing a pseudo-location. In essence, via protection location set that is determined with prior distribution π and $e^{\epsilon}E_m$, PIVE achieves differential privacy while guaranteeing a lower bound for the adversary's expected inference error. This framework offers adaptive location protection for users according to their current locations and requirements on two privacy notions (expressed by E_m and ϵ), and the latest prior distribution the adversary could have known (e.g., by inference with the mobility model of users). Previous geo-indistinguishable mechanisms [20, 21] can be regarded as the special cases of our framework with $\mathcal{F}$ using the circular neighborhood with a fixed radius as the protection location set without considering any prior distribution and inference error bound.

PIVE provides two privacy control knobs: 1) the minimum inference error E_m and 2) the differential privacy parameter ϵ. Through these parameters, we allow users to define their desired privacy preferences at different locations. The min error parameter E_m aims to bound the expected inference error in the worst case. The differential privacy parameter ϵ allows users to constrain the posterior information leakage via the provisioning of differential privacy. Given that ϵ-differential privacy is the property of the random mechanism $\mathcal{K}$ producing pseudo-locations, one possible way for a user to set these two parameters is

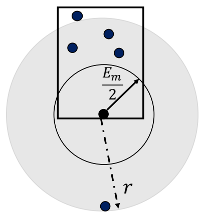


Figure 2.7: The non-optimal case for the simple approach

to use a fixed ϵ for $\mathcal{K}$ and set E_m according to this user's tolerance estimation on the lowest bound of expected inference error of the adversary against the protection region, for example, E_m=0.1km.

2.4.2 Determining Protection Location Set

Given the user's location x, the problem is how to efficiently determine its protection location set Φ ($x \in \Phi$) that satisfies $E(\Phi) \geq e^{\epsilon}E_m$ with a diameter as small as possible. Meanwhile, we note the diameter of the protection location set cannot be less than $e^{\epsilon}E_m$, given by the following theorem.

Theorem 2. *Let $D(\Phi)$ be the diameter of protection location set Φ that is the largest distance between any two locations in Φ. If $E(\Phi) \geq e^{\epsilon}E_m$, we have $D(\Phi) \geq e^{\epsilon}E_m$.*

Proof. $D(\Phi) \geq d(\hat{x}, x)$ for $\forall \hat{x}$, x in Φ, so

$$e^{\epsilon}E_m \leq E(\Phi) \leq \min_{\hat{x}\in\Phi} \sum_{x\in\Phi} \frac{\pi(x)}{\sum_{y\in\Phi}\pi(y)} D(\Phi) = D(\Phi)$$

$\square$

A simple way to determine the protection location set is to gradually increase the radius of circular region centered at the user's location from $e^{\epsilon}E_m/2$ (given by Theorem 2) until it satisfies $E(\Phi) \geq e^{\epsilon}E_m$. However, this approach may produce unnecessarily large diameter, leading to significant service quality loss. Figure 2.7 shows an example in which the desired protection location set is obtained by increasing radius to r to include four location points, resulting in $2r$ diameter. However, in this way we cannot find another qualified set that is the rectangle area in the figure with a much smaller diameter.

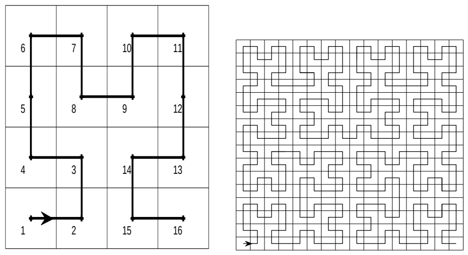


Figure 2.8: Hilbert Curve for 4×4 and 16×16 grid

Hilbert Curve based Search: To efficiently search over the plane for the protection location set, we propose a Hilbert curve based search algorithm. Hilbert curve [36] is a popular member in the family of space-filling curves. It provides a mapping from a data point in a 2-D space to a point in one dimensional space that preserves the proximity of data. That is, points which are close to one another in the 2-D space will also remain close to each other in the transformed 1-D space. It has been shown that Hilbert curves have the superior distance preserving properties [37]. Figure 2.8 shows the Hilbert curves for 4×4 and 16×16 grids in 2-D space. The Hilbert curve maps a location point x to a 1-D value denoted by $H(x)$. We call $H(x)$ as the Hilbert value of x. The locations in $\mathcal{X}$ is sorted by their Hilbert values, and the rank of a location x in the sorted $\mathcal{X}$ is denoted by $R(x)$.

Given user's location x, our algorithm searches the neighborhood of x along the Hilbert curve to find a protection location set Φ that satisfies $x \in \Phi$ and $E(\Phi) \geq e^{\epsilon} E_m$. The basic search strategy in the algorithm can be generally described as follows. Let $x_{-l}, x_{-l+1}, \ldots$, $x_0 (= x)$, $x_1, x_2, \ldots, x_r$ be the sequence of locations in the searching neighborhood of x along the Hilbert curve, sorted by their Hilbert values. For each x_i ($-l \leq i \leq 0$), the algorithm checks every interval from x_i to x_j for $0 \leq j \leq r$ in the sequence, denoted by $[x_i, x_j]$, and evaluate $E([x_i, x_j])$ by (2.29). Once an interval that has $E([x_i, x_j]) \geq e^{\epsilon} E_m$ is found for x_i, the algorithm stops interval check for x_i, adds location set in $[x_i, x_j]$ to a candidate list, and repeats with the next x_i. Finally, the set having the smallest diameter in the list is returned, with breaking ties by a random choice.

In the case that all locations in Φ have zero prior probabilities, i.e., $\sum_{y \in \Phi} \pi(y) =$

Algorithm 1: Protection Location Set Search Algorithm

```
Input: $x$: user's location $E_m$: error bound, $\epsilon$: privacy parameter
1  if $\pi(x) = 0$ then
2      $S \leftarrow \{l \mid H(l) \in [H(x) - range, H(x) + range]$ on $H$ $\}$;
3  else
4      $S \leftarrow \{l \mid H(l) \in [R(x) - range, R(x) + range]$ on sorted $\mathcal{X}\}$;
5  Let $S$ be $x_{-l}, x_{-l+1}, \ldots, x_0 = x, x_1, x_2, \ldots, x_r$;
6  $L \leftarrow \emptyset$;
7  for $i$ from $-l$ to $0$ do
8      for $j$ from $0$ to $r$ do
9          $\Phi = \{x_k \mid i \leq k \leq j\}$ ;
10         Calculate $E(\Phi)$ by(2.29) ;
11         if $E(\Phi) \geq e^{\epsilon} E_m$ then
12             Add $\Phi$ to $L$;
13             break ;
14 return a set having the smallest diameter in $L$;
```

0 in (2.29), we define $\frac{\pi(x)}{\sum_{y\in\Phi}\pi(y)} = \frac{1}{|\Phi|}$, because a uniform distribution is assumed in an area when the adversary does not have any prior information about it. Accordingly, the algorithm searches Φ for locations with zero and non-zero prior probability in $\mathcal{X}$ in different ways. Because over the plane any locations outside $\mathcal{X}$ (e.g., the un-numbered regions in Figure 2.1) indeed have zero prior probabilities, the protection location set Φ for the user's location x with $\pi(x) = 0$ can involve them with $E(\Phi)$ being computed in the defined way. Thus, the searching range is determined as all the locations on the plane with Hilbert values in a range $[H(x) - range, H(x) + range]$. For the protection location set of x with $\pi(x) > 0$, the locations with zero prior probabilities contribute zero to $E(\Phi)$ in (2.29) and thus the locations outside $\mathcal{X}$ are not considered. The searching range is defined as the locations with ranks in $[R(x) - range, R(x) + range]$ over the sorted sequence of $\mathcal{X}$. The algorithm applies the search strategy mentioned above to the searching range to obtain the protection location set. The pseudo-code is given in Algorithm 1.

The $range$ must be large enough to have better chance to find a qualified protection location set. Given $T = e^{\epsilon}E_m$ and Theorem 2, $range$ can be decided heuristically. We can traverse along the Hilbert curve in both directions from user's location x. Once reaching the locations a and b in each direction with their distances to x being some multiple of T, we set $range = \max(|H(a) - H(x)|, |H(b) - H(x)|)$. In our implementation we simply choose sufficiently large range that incurs low failure rate for finding protection location set. We can also specify an upper bound for $range$ to limit the searching cost and avoid large region that causes unacceptable quality loss. Note the algorithm may not find any qualified protection location set, for example, in the case that the user's current location is only possible location for him and all other locations has zero prior probabilities on the plane. Thus, if an empty set is returned, which indicates the location privacy cannot be protected, the user can choose to suppress location report.

Improvement with Multiple Rotated Hilbert Curves: Although using Hilbert curve enables efficient search over 2-D plane, a drawback is that the search is conducted along a single direction and the searched regions can only be ones that consist of neighboring locations on the curve. A cell actually can have four neighbors on the plane while two neighboring cells may be far apart on the curve (e.g., location 2 and 15 in Figure 2.8). Since there are regions where locations are not adjacent on the curve, we propose to use multiple different Hilbert curves to connect locations in different ways such that more possible regions can be involved, which can improve the chance to find the protection location set with a smaller diameter. Previous works have utilized multiple Hilbert curves to improve the quality of k-Nearest Neighbor queries [38] and reduce cloaking area [39]. In PIVE, given a Hilbert curve H over $2^n \times 2^n$ grid, other three Hilbert curves are generated by rotating it 90, 180, 270 degrees clockwise about the center point. We use Algorithm 1 to find the protection set of the user's location x for each Hilbert curve, and choose the one with smallest diameter among four results.

2.4.3 Differentially Private Mechanism

Given the protection location set Φ, PIVE achieves differential location privacy on it through the exponential mechanism [30]. Considering the set $\mathcal{X}$ as the output range of location obfuscation, the utility of output x' is measured by the distance between x' and user's location

x in Φ. Smaller distance has higher utility. As the protection location set decides "neighboring" locations to the user's location, the sensitivity of the utility function u is

$$\Delta u = \max_{x' \in \mathcal{X}} \max_{x,y \in \Phi} |d(x,x') - d(y,x')| \tag{2.32}$$

It is easy to see, according to triangle inequality, for any $x, y \in \Phi$, $|d(x,x') - d(y,x')| \leq d(x,y) \leq D(\Phi)$, so $\Delta u = D(\Phi)$ where $D(\Phi)$ is the diameter of Φ.

Exponential mechanism $\mathcal{K}$: Given the user's location x and location protection set Φ, the exponential mechanism $\mathcal{K}$ selects and outputs a location $x' \in \mathcal{X}$ with probability proportional to $\exp(\frac{-\epsilon d(x,x')}{2D(\Phi)})$.

The mechanism $\mathcal{K}$ samples each location x' from $\mathcal{X}$ with the probability $w_x \exp(\frac{-\epsilon d(x,x')}{2D(\Phi)})$ where w_x is the normalization factor for the probability distribution over $\mathcal{X}$,

$$w_x = 1 \Big/ \Big(\sum_{x' \in \mathcal{X}} exp\big(\frac{-\epsilon d(x,x')}{2D(\Phi)}\big)\Big) \tag{2.33}$$

Following the proof of Theorem of McSherry and Talwar [30], we can easily obtain the theorem below,

Theorem 3. *The exponential mechanism $\mathcal{K}$ preserves ϵ-differential privacy on the protection location set Φ.*

Proof.

$$\begin{aligned}
\frac{f(x'|x)}{f(x'|y)} &= \frac{w_x \exp\Big(-\epsilon d(x,x')/(2D(\Phi))\Big)}{w_y \exp\Big(-\epsilon d(y,x')/(2D(\Phi))\Big)} \\
&\leq \frac{w_x}{w_y} e^{\epsilon |d(x,x') - d(y,x')|/(2D(\Phi))} \leq \frac{w_x}{w_y} e^{\epsilon d(x,y)/2D(\Phi)} \\
&\leq \frac{w_x}{w_y} e^{\epsilon/2} \leq \frac{\Big(\sum_{x' \in \mathcal{X}} exp\big(\frac{-\epsilon d(y,x')}{2D(\Phi)}\big)\Big)}{\Big(\sum_{x' \in \mathcal{X}} exp\big(\frac{-\epsilon d(x,x')}{2D(\Phi)}\big)\Big)} e^{\epsilon/2} \\
&\leq \frac{\Big(\sum_{x' \in \mathcal{X}} exp\big(\frac{-\epsilon (d(x,x') - D(\Phi))}{2D(\Phi)}\big)\Big)}{\Big(\sum_{x' \in \mathcal{X}} exp\big(\frac{-\epsilon d(x,x')}{2D(\Phi)}\big)\Big)} e^{\epsilon/2} \\
&\leq \frac{\Big(\sum_{x' \in \mathcal{X}} exp\big(\frac{-\epsilon d(x,x')}{2D(\Phi)}\big)\Big)}{\Big(\sum_{x' \in \mathcal{X}} exp\big(\frac{-\epsilon d(x,x')}{2D(\Phi)}\big)\Big)} e^{\epsilon/2} e^{\epsilon/2} \leq e^{\epsilon}
\end{aligned} \tag{2.34}$$

□

The exponential mechanism provides strong utility guarantees since it discounts the pseudo-locations exponentially quickly as their distances to the actual location increase. To see that, we have the following theorem

Theorem 4. *Given the actual location x, let x' be the pseudo-location randomly sampled from $\mathcal{X}$ by the exponential mechanism $\mathcal{K}$, with probability at least $1-\delta$ we will have*

$$d(x,x') \leq \frac{2D(\Phi)}{\epsilon}\Big(\ln|\mathcal{X}| + \frac{\epsilon}{2} - \ln|\Phi| - \ln\delta\Big) \tag{2.35}$$

Proof. For any x' that has $d(x,x') \geq c$, the probability it is sampled with is at most $w_x\mathrm{exp}(\frac{-\epsilon c}{2D(\Phi)})$. Thus, the total probability of $d(x,x') \geq c$ for all x' is at most $w_x|\mathcal{X}|\mathrm{exp}(\frac{-\epsilon c}{2D(\Phi)})$. On the other hand,

$$\begin{aligned}
w_x &= 1\Big/\Big(\sum_{x'\in\Phi} exp(\frac{-\epsilon d(x,x')}{2D(\Phi)}) + \sum_{x'\in\mathcal{X}\backslash\Phi} exp(\frac{-\epsilon d(x,x')}{2D(\Phi)})\Big)\\
&\leq 1\Big/\Big(\sum_{x'\in\Phi} exp(\frac{-\epsilon d(x,x')}{2D(\Phi)})\Big) \leq 1\Big/\Big(\sum_{x'\in\Phi} e^{-\epsilon/2}\Big)\\
&= \frac{e^{\epsilon/2}}{|\Phi|}
\end{aligned}$$

Thus, we have $\mathrm{Pr}(d(x,x') \geq c) \leq \frac{|\mathcal{X}|e^{\epsilon/2}}{|\Phi|}\mathrm{exp}(\frac{-\epsilon c}{2D(\Phi)})$. Let δ be the right-hand side and we can derive (2.35). □

Because the searching range is limited in Algorithm 1, $\ln|\Phi|$ is bounded by $\ln(2range)$ that is a small constant (e.g, at most 4 in our experiment), while $D(\Phi)$ can vary a lot given possible sparse location distribution. Therefore, the value of right hand side of (2.35) mainly depends on $\frac{D(\Phi)}{\epsilon}$. Given fixed E_m, increasing ϵ can incur a protection location set with a larger diameter since $D(\Phi) \geq e^{\epsilon}E_m$ given by Theorem 2. Because $D(\Phi)$ increases exponentially with ϵ, with increasing ϵ from the value close to zero, $\frac{D(\Phi)}{\epsilon}$ will decrease first and then increase. Therefore, we expect that the service quality and also location privacy will exhibit the similar changing pattern, which is demonstrated in our evaluation.

2.5 Evaluation

In this section we first evaluate the performance of our PIVE mechanism, and compare PIVE approach with other mechanisms on location privacy and service quality. Our evaluation shows that PIVE effectively combines two privacy notions, and efficiently addresses the issues of existing location obfuscation mechanisms.

We use the dataset provided by authors of [21]. The dataset was extracted from the GeoLife GPS Trajectories dataset [33, 34, 35], which contains 17621 traces collected from 182 users in Beijing, China, during a period of over five years. The traces record users outdoor movements with locations being logged every 1-5 seconds or every 5-10 meters. The details of data processing can be found in [21] and here we provide a brief description. The map of Beijing is divided into a grid of regions 0.658km wide and 0.712km high, the 50 "most popular" regions of the grid is used as the set of all locations $\mathcal{X}$, as shown in Figure 2.1, and the users who have few recorded points for each time period at these regions are filtered out. The final dataset contains 84 users. The prior for each user is computed by counting and normalizing the number of points falling in each of 50 regions with in different time periods (all day, morning, afternoon and night). In this paper we use all-day prior to construct mechanisms. In order to demonstrate the performance in a single user setting, at default we always choose the user with id 0, as in Section 2.3.3.

2.5.1 Performance of Protection Location Set Search

Given ϵ and E_m, a threshold $T = e^{\epsilon} E_m$ is determined and Algorithm 1 searches a location protection set Φ for user's location that has $E(\Phi)$ in (2.29) no less than T while with the smallest diameter. In this section we study the performance of our search algorithm in terms of the diameter $D(\Phi)$ and value $E(\Phi)$. In the algorithm, we choose sufficient large $range$=50 at default. The searching range $range$ decides the chance to find a qualified location protection set for a location. To see its impact, we test $range$ with values 20, 40, 50 and 60 for $T = 2$ that is corresponding to the largest average diameter in Figure 2.9. The number of regions for which the algorithm fails to find qualified protection location set is 19, 8, 6, 6 for each $range$ value respectively. We can see that from $range$=50 that is the size of $\mathcal{X}$, the number of such regions remains to be 6. Smaller $range$ 40 has approximate number of failures as 50. Since the size of 50 regions is small, in our experiment we choose 50 that incurs smallest number of failures. Within a large size of $\mathcal{X}$, $range$ can be a relatively smaller value.

We vary T from 0.1 and 2.0 and measure the diameters of protection (location) sets obtained by our algorithm for user's location at each of 50 regions. The results are shown in Figure 2.9 where the whiskers represents minimum and maximum diameters in each group. It is clear that the average diameter of all regions increases with the threshold T. The diameters for isolated region 48, 49, and 50 remain between 4km and 5km under different T. They are maximum ones in the results from T=0.1 to 0.8. For some regions like 24, 25, 32, 33 and 34, the diameters become higher than 12km from T=1.2. From T=1.4 to 2.0, the algorithm cannot find qualified protection sets for regions like 24, 25, 26, 32 and 33. Figure 2.10 shows the corresponding $E(\Phi)$ values of obtained protection sets for every region under different T. As we can see, the average $E(\Phi)$ value increases linearly with T and is approximate to T. This indicates that our algorithm effectively finds the qualified protection location set with $E(\Phi) \geq T$ as desired. We also observe that the maximum $E(\Phi)$ for each T is about 2km from T=0.1 to 1.5, which is because that the protection set for region 49 always has maximum $E(\Phi)$ 2km. By further looking into the results, we find that for both region 49 and 50, the protection location set remains the same from T=0.1 to 1.5, resulting the same diameter and $E(\Phi)$. Region 49 always has protection set $\{47, 49\}$, and 50 has $\{48, 50\}$. From Figure 2.1, we can see the reason is that they are isolated regions and their nearest neighbors are 47 and 48 respectively that provide qualified protection sets. For $T > 1.5$, region 50 has to involve another region 45 to satisfy $E(\Phi) \geq T$.

To see how the diameter of protection location set varies among different regions, we

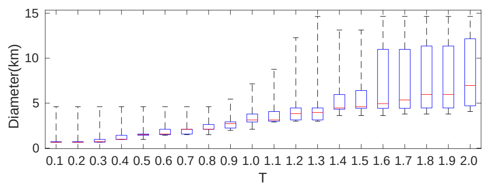


Figure 2.9: Boxplot of Diameter with different T

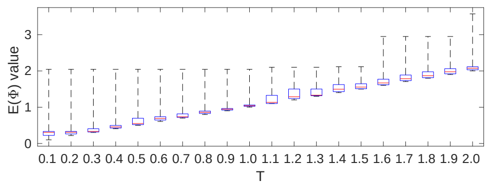
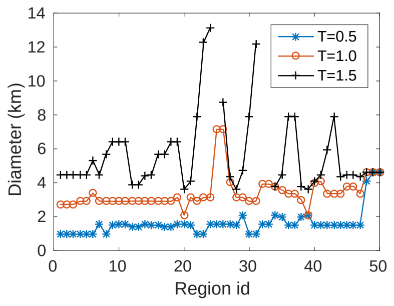


Figure 2.10: Boxplot of $E(\Phi)$ with different T

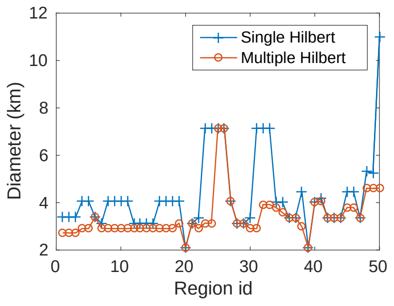
show the results of T=0.5, 1.0 and 1.5 in Figure 2.11a. It is clear that the diameter of protection location set for each region increases with T. The curve is discontinuous at some points for T=1.5 because the algorithm cannot find qualified set at those locations. The diameters for regions 49 and 50 remains the same with three different T due to the reasons mentioned above.

Improvement with Multiple Hilbert curves: Our algorithm utilizes multiple Hilbert curves that are generated by the rotation of the original Hilbert curve to find protection location set with the smallest diameter. To see the effectiveness of such improvement, we compare the diameter of every region with the search algorithm using one single Hilbert curve and multiple ones respectively, under a given T. Figure 2.11b shows the result with T=1.0. We can see that using multiple Hilbert curves effectively reduces the diameter of protection location set. At some regions the improvement is significant. For example, the diameter is reduced by more than half at region 31 and 50. Such improvement holds for different T values.

We further investigate the diameters for all 84 users in the dataset and show the result with T=1 in Figure 2.12. All users have approximate average diameters between 2km and 4km, but the maximum diameter for some users can be as large as 14km. Large diameter will incur significant noise and extremely low utility. To avoid that, a maximum tolerable diameter D_m can be specified in the mechanism, such that the mechanism can use the



(a) Diameter of Φ for regions



(b) Single V.S. multiple Hilbert

Figure 2.11: Diameter of protection location set for every region.

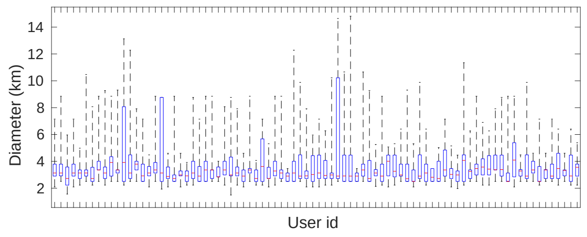


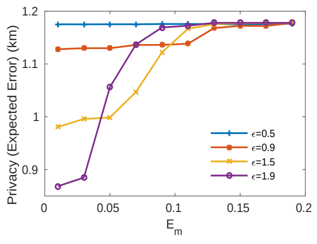
Figure 2.12: The diameters under $T = 1$ for every user.

location set with maximum $E(\Phi)$ among those with diameters no larger than D_m if the diameter of the produced protection location set exceeds D_m.

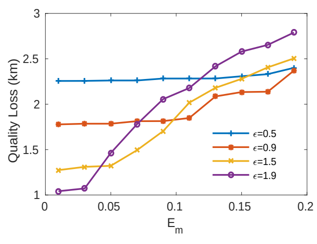
2.5.2 Location Privacy and Service Quality

In this section, we evaluate the impact of differential privacy parameter ϵ and inference error threshold E_m on location privacy and service quality under a single user setting. Although PIVE allows different privacy parameters at different locations, we use uniform parameters over all locations and unconditional expected inference error (2.8) and quality loss (2.10) as privacy and quality metric, in order to examine the effects of different ϵ and E_m on the performance.

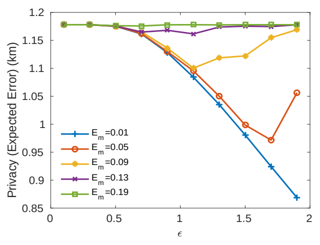
Figure 2.13a and 2.13b show that under different ϵ, both location privacy and quality loss monotonically increase with E_m. This is because that higher E_m leads to larger diameter of the protection location set, and the pseudo-location is more likely to be further from the actually location and thus incurs lower utility, which is indicated by Theorem 4. The monotonic relationship between E_m and the corresponding location privacy (i.e., expected



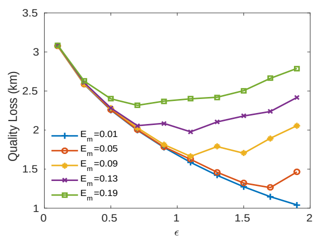
(a) Privacy V.S. E_m



(b) Quality Loss V.S. E_m



(c) Privacy V.S. ϵ



(d) Quality Loss V.S. ϵ

Figure 2.13: Impact of privacy parameters ϵ and E_m

inference error) indicates that E_m is an effective control knob to guarantee the expected inference error. The difference in order of magnitude between E_m and corresponding expected inference error is because E_m is the lower bound of the expected inference error given any pseudo-locations in the worst case that the adversary have identified the protection set. Therefore, E_m should be determined with consideration of such worst case to protect location privacy in terms of unconditional expected inference error.

For smallest ϵ=0.5 indicating the strongest privacy guarantee, $e^{\epsilon}E_m$ increases linearly with E_m with a small factor e^{ϵ}, that is to say, the impact of diameter changes on the privacy and quality is much smaller compared with that of ϵ. In contrast, under larger ϵ like 1.9 that incur weak requirement for differential privacy, $E(\Phi)$ increases with E_m with a much larger factor e^{ϵ}, which incurs larger diameter variance. Therefore, E_m has more significant impact on location privacy and quality loss for ϵ=1.9, indicating by its highest curve steepness in Figure 2.13a and 2.13b. Also, in Figure 2.13a, location privacy for different ϵ increases to the same upper limit 1.178 as in Section 2.3.3. ϵ=0.5 achieves this limit regardless of E_m. Other cases have location privacy approximate to the upper limit starting from E_m=0.1. Therefore, we can choose E_m no larger than 0.1 for improving utility. Accordingly, in our comparison experiment we focus on E_m=0.05 and 0.09.

We further examine the impact of ϵ on location privacy and quality loss with given E_m. The results are shown in Figure 2.13c and 2.13d. We can see that the relationship between ϵ and location privacy as well as quality loss is not monotonic. Location privacy and quality loss first decrease with ϵ and then increase. This result confirms our discussion following Theorem 4. The reason is that, at first ϵ takes control of location privacy and quality loss, and thus increasing ϵ incurs lower location privacy and quality loss. As the diameter increases exponentially with ϵ, the diameter takes effects, thus increasing ϵ causes higher privacy and quality loss. Comparing Figure 2.13c and 2.13d, we can see that the turning points of both metrics under the same E_m occurs at the same ϵ values.

2.5.3 Comparison with other mechanisms

In this section we compare PIVE with typical geo-indistinguishable mechanisms to verify the advantage of introducing inference error bound. Because PIVE focuses on local performance of privacy protection for every region rather than the global average performance examined in previous works [21, 25, 22], we compare PIVE with other mechanisms mostly in a single user setting, in order to check the privacy protection performance at each individual region. We examine the performance of PIVE for every user, and only show results with regard to user with id 0 due to the similar behaviors of these mechanisms for other users. It is worth to note that PIVE provides the users a way to specify different privacy requirements for different locations through two privacy parameters E_m and ϵ. Given that the existing mechanisms like optimal geo-indistinguishable mechanism do not support different privacy specifications for different locations, we set the same privacy parameters everywhere for PIVE in order to make meaningful comparisons.

We first consider an exponential mechanism EM, that is like the one proposed in PIVE except using uniform constant diameter for every location's protection location set. It represents geo-indistinguishable mechanisms like discrete Planar Laplace Mechanism that en-

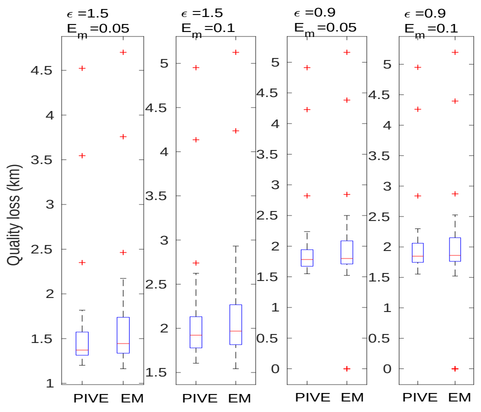
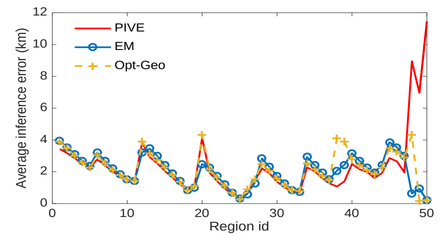


Figure 2.14: Boxplot of quality loss of 84 users

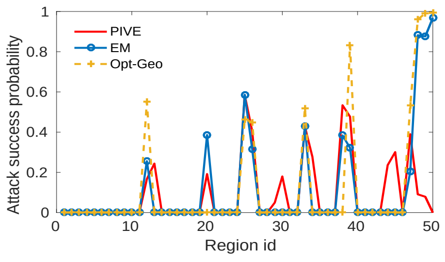
sure ϵ-differential privacy in the circular neighborhood centered at the user's location. To make a fair comparison, for each user, we run PIVE with different ϵ and E_m, and obtain its location privacy (i.e., expected inference error (2.8)). Then, given the same ϵ, we derive EM by choosing the diameter to achieve the same location privacy. To deal with floating point comparisons, two values with less than 0.005 difference are regarded to be equal for location privacy. Figure 2.14 shows boxplots of the quality losses of all users for PIVE and EM respectively under different pairs of ϵ and E_m. In each subfigure, we can see that overall PIVE achieves smaller quality loss than EM, though they have the same location privacy. This is because that PIVE adaptively determines protection location sets to implement geo-indistinguishability but EM uses protection regions of uniform radius everywhere. At some locations with sufficient number of possible locations in their neighborhood, PIVE can use smaller diameters than at locations in sparse areas for providing the same level of location privacy. Comparing these subfigures, we can see that lower ϵ or higher E_m, both indicating higher privacy requirements, incur larger quality loss.

We further look into the level of privacy protection for a single user at every region, with using the same simulation approach described in Section 2.3.3. Suppose $E_m = 0.05$ and $\epsilon = 1.5$ for PIVE. We derive EM with $\epsilon = 1.5$ and also the optimal geo-indistinguishable mechanism Opt-geo (M_{ϵ_g} in Section 2.3.3) with $\epsilon_g = 0.7$ such that they achieve the same expected inference error as PIVE. Figure 2.15a and 2.15b show the average error of optimal inference attack and success probability of Bayesian inference attack against each region for three mechanisms. We can see that for both E_M and Opt-geo, the skewed regions 48, 49 and 50 suffer strong association between pseudo-locations and true locations, manifested by approximate zero inference error for the optimal inference attack and high success probability for Bayesian inference attack simultaneously. PIVE is resilient to such vulnerable cases due to the skewed probability distribution on these isolated regions by finding a sufficient protection location set and ensuring the lower bound of inference error in the worst cases. By using a protection region to include other possible locations, PIVE avoids the strong association between pseudo-location and true location for skewed cases that happen to E_M and Opt-geo. That is the reason of why PIVE has much larger inference error and approximate zero attack success probabilities at these isolated locations. Furthermore, with PIVE, the Bayesian inference attack success probability is capped to be no more than 60% as shown in Figure 2.15b. To see the PIVE's difference compared with others against Bayesian inference attack, in Table 2.1 we show the percentage of regions that have the Bayesian inference attack success probability higher than $X\%$ with X range 50%$\sim$90% for all mechanisms. As we can see, given different threshold X, PIVE obtains the least percentage of regions that have success probability larger than X. Comparing the service quality losses of three mechanisms, we have EM=1.49 $>$ PIVE=1.32 $>$ Opt-geo=1.02. PIVE has smaller quality loss than EM, which has been explained above, and Opt-geo achieves smallest quality losses due to its global optimization on service quality. Here we also note that region 25 is not skewed location and does not have as strong association issue (high posterior probability) as the skewed locations, thus with just satisfying minimum lower bound 0.05, PIVE does not have much effect on the inference error and success probability on region 25, compared with others mechanisms.

Next, we compare PIVE with the joint optimization mechanism [25] in terms of effectivess for privacy protection by combining geo-indistinguishability and expected inference error. We choose the same parameters for PIVE as in the previous expeirments, and then

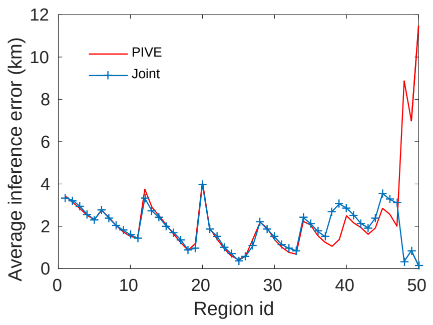


(a) Optimal inference attack

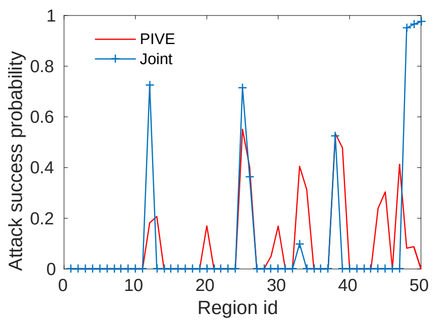


(b) Bayesian inference attack

Figure 2.15: Comparison of local privacy protection at every region



(a) Optimal inference attack



(b) Bayesian inference attack

Figure 2.16: Comparison of PIVE and joint mechanism

use its expected inference error as the minimum desired distortion privacy level d_m for constructing the optimal joint mechanism. ϵ_g in the joint mechanism is chosen to achieve the same location privacy as PIVE in terms of unconditional expected inference error. Figure 2.16 shows the average inference error and success probability for two inference attacks respectively at each region, with d_m=0.9986km and ϵ_g=0.8. It can be seen that (1) the joint mechanism and PIVE exhibit similar performance at most locations with small variation; and (2) the joint mechanism incurs the weak regions, e.g., region id 48, 49 and 50, against inference attacks, despite having bound on global expected distortion metric. These weak regions represent some skewedness as they are far away from the rest of the regions. Con-

X.	PIVE	EM	Opt-geo	Joint
$> 50\%$	6%	8%	14%	12%
$> 70\%$	0%	6%	8%	10%
$> 90\%$	0%	6%	6%	4%

Table 2.1: The percentage of locations exceeding given success probability

cretely, PIVE has the average inference error bounded to be no lower than 0.22 and at the same time the Bayesian inference attack success probability capped to be no higher than 60% (Table 2.1 shows PIVE has smaller percentage of regions compared to the joint mechanism with different X). In comparison, the joint optimization achieves good privacy at most of the locations but fail to avoid the worst case scenarios when the location dataset contains some skewed locations.

Note for the joint mechanism, when ϵ_g increases, its performance gets close to the optimal Bayesian mechanism, since its linear model is equal to the optimal Bayesian mechanism with geo-indistinguishability constraint and larger ϵ_g will relax the constraint of geo-indistinguishability. But it will cause less robustness against the adversary with arbitrary prior information. PIVE shows the benefits of both privacy notions against optimal inference attack simultaneously: similar (or sometimes slightly lower) expected inference error as the joint mechanism under strong geo-indistinguishability (with ϵ_g=0.8) except regions 48-50 where PIVE provides similar privacy protection as the optimal Bayesian mechanism given in Figure 2.2. We would like to make two remarks: (1) The level of privacy protection offered by both PIVE and joint optimization are exceeding the user-defined lower error bound at most locations, thus are acceptable for users as good privacy protection, even though the inference error of PIVE can be slightly lower at some locations. (2) For the weak locations, PIVE shows high resilience and adaptivity to the skewed distribution against inference attacks, compared to all three existing approaches (see Figure 15 and Figure 16).

2.6 Related Work

Location privacy research started about ten years ago with the notion of location k-anonymity with two landmark results: (i) uniform location k-anonymity [40] and (ii) user-defined, personalized location k-anonymity [41]. The location k-anonymity based solutions hide a user's exact location point using a spatial region that meets the two constraints: (a) it contains the exact location point of the user; and (b) there are at least $k - 1$ other users who will use the same location region as their released location to meet the k anonymity requirement. Alternatively, some location obfuscation mechanisms achieve privacy by using landmark objects or random perturbation instead of k-anonymity. [42] proposes to use the location of a closest landmark object as the perturbed location such that the LBS severs process the location query based on the landmark. [31] proposes to search the region that has sufficient user footprints such that the user can feel safe for his location privacy. However, neither user-defined privacy notion nor any formal privacy notion is provided and guaranteed by the proposed region-based location cloaking mechanism.

Recently, two stronger privacy notions are proposed based on statistical quantification of attack resilience: expected inference error [22, 26] and geo-indistinguishability [20]. The former advocates the privacy notion based on its attack resilience to the prior information of adversary by measuring the expected inference error and the latter promotes the differential privacy notion to constrain the posterior information gain of an adversary based on the release of pseudo-locations of mobile user. A number of location obfuscation mechanisms [22, 20, 32, 21, 25] have been developed based on them. For example, based on the prior distribution of user's location, Shokri et al. [22] proposed an optimal construction mechanism for location perturbation against inference attacks through linear programming. The mechanism aims to maximize the expected inference error (resp. service quality) given the constraint on the service quality loss(resp. expected inference error). The service quality loss is characterized by the expected distance between real and reported locations. Based on Shokri et al.'s optimization framework, Theodorakopoulos et al [28] advocated to follow a user over his trajectory and maximizes privacy for each location with considering privacy leakage due to location correlation between past, current and future locations in a trajectory. Andrés et al. [20] proposed the notion of geo-indistinguishability. A Planar Laplace (LP) mechanism is developed to achieve the ϵ geo-indistinguishability by adding noise to actual location drawn from a polar Laplacian distribution. Several recent location privacy development projects [23, 24, 43] have adopted or extended ϵ geo-indistinguishability for location privacy protection. Bordenabe et al. [21] proposed an optimal geo-indistinguishable mechanism to minimize the service quality loss. Similar to [22], it uses linear programing to minimize global expected service quality loss, with a uniform privacy parameter for geo-indistinguishability. Chatzikokolakis et al [32] defines privacy mass over the point of interests on the plane and adaptively decide the privacy parameter of geo-indistinguishability for a location with considering local characteristics of each area.

The mechanisms in [21, 22, 25] follow a global optimization framework: given the privacy or service quality constraints, a linear programming model is formulated to maximize service quality or privacy respectively. Such formulation uses uniform differential privacy parameter and global privacy/quality metrics averaged over all locations, which offers uniform privacy/utility with respect to all locations and all LBSs. It could be a difficult task to pre-determine the constraint for every location where a user will ask for any LBS service request with his personalized and spatial-temporal dependent as well as LBS dependent privacy requirement. Besides, these techniques are computationally costly due to solving a linear program with $|\mathcal{X}|^2$ decision variables, and the perturbation solution is statically constructed once for all locations, which can be prohibitively expensive for frequently changing prior information and frequently changing privacy/utility preference by users at different locations and times.

Our work is primarily related to two recent research efforts in [25] and [32]. Concretely, Shokri [25] is the first to propose a joint mechanism to integrate the two privacy notions using a linear programming framework, demonstrating the potential for improvement on privacy protection. However, the joint optimization mechanism uses uniform differential privacy parameter and global privacy/utility metrics by averaging over all locations. We argue that an overall metric for all locations and a per-location based metric may result in different allocations of privacy and utility. Thus PIVE is more suitable to situations where mobile users may have different privacy/utility preferences for different locations, at different time and working with different LBSs. Next, unlike most existing geo-indistinguishable mechanisms that consider uniform differential privacy parameters for all users and all locations, Chatzikolakis and his co-authors [32] propose to adaptively decide the noise level of geo-indistinguishability according to the privacy characteristics of local area. They compute the density of a local area for each location and adds less noise for perturbed location if the density of the actual location area is high and more noise when the actual location falls into the low density areas. However, the density of a local area is defined in terms of the public locations such as restaurants, churches and hospitals. Thus this approach assumes that these different types of public locations are of the same privacy sensitivity for all mobile users at all time, and thus fails to model the personalized geo-indistinguishability with respect to different locations, different times and different LBSs. In comparison, PIVE adaptively adjusts the noise level of location obfuscation according to a personalized error bound and the prior distribution in the local area.

2.7 Concluding Remarks

Privacy Claim: The propose PIVE approach aims for Bayesian differential location privacy that enhances differential privacy with the resiliency against Bayesian inference attacks. It leverages the prior information of Bayesian inference attacks to improve the privacy parameter ϵ of differential location privacy for the resiliency against Bayesian inference attacks for the locations that presents skewed prior distribution. On the other hand, the differential location privacy is not equivalent to the standard differential privacy. The differential location privacy is always defined over a protection region to make any two locations inside the region differentially private, i.e., the distribution of outcome fake locations from these two locations are ratio bounded by e^{ϵ}.

Utility Claim: With achieving the same resiliency against Bayesian inference attacks, PIVE improves the location utility compared with existing geo-indistinguishable mechanisms, because it adapts the noise introduced by differential location privacy according to the prior information around a location.

CHAPTER 3
DIFFERENTIALLY PRIVATE MODEL PUBLICATION FOR DEEP LEARNING

3.1 Introduction

In recent years, deep learning techniques based on artificial neural networks have dramatically advanced the state of art in a wide range of AI tasks such as speech recognition, image classification, natural language processing and game playing. Its success relies on three sources of development: high performance computing, large-scale datasets, and and the increasing number of open source deep learning frameworks, such as TensorFlow, Caffe, Torch.

Privacy Concerns in Deep Learning. However, recent studies on membership attacks and model inversion attacks have shown potential privacy risks from a number of dimensions. First, when large scale datasets collected via crowdsourcing platform are collected via crowdsourcing platforms from individuals, such as location, images, medical and financial data of the users, and users usually do not have any control over how their data is being used or shared. Second, deep neural networks have a large number of hidden layers, leading to large effective capacity that could be sufficient for encoding the details of some individual data into model parameters or even memorizing the entire data set [3]. It has been shown that individual information can be effectively extracted from neural networks [4, 5]. Therefore, there are severe privacy concerns accompanied with the wide deployment of deep learning applications and deep learning as a service platform.

On the other hand, the publishing and sharing of trained models have always been of great interest in deep learning applications. Google's cloud machine learning services provide several pre-trained models usable out-of-the-box through a set of APIs. The model owners can also publish their trained models to the cloud and allow other users to get predictions through APIs. In mobile applications, entire models are stored on-device to enable power-efficient and low-latency inference. Transfer learning [6], a key technique of deep learning, can leverage and adapt the already existing models to new classes of data, saving the effort of training the entire neural network from scratch. People who only have small datasets can use the model trained on a large dataset as fixed feature extractor in their neural networks or adapt the model to their own domain. It is believed to be the next driver of machine learning success in industry and significantly stimulate the sharing of pre-trained models. A large amount of pre-trained models have been publicly available in model zoo repositories [7]. In these cases, the model parameters are completely exposed, making it easier for the adversaries to launch inference attacks, such as membership attacks [5] or model inversion attacks [4], to infer sensitive data records about individuals in the training datasets. Even only providing the query API to access remote trained models, the model parameters still can be extracted from prediction queries and in turn used to infer the sensitive training data [44]. Therefore, it is imperative to develop principled privacy preserving deep learning techniques to protect private training data against the adversaries with full knowledge of model parameters.

Deep learning with Differential Privacy. Although privacy preserving machine learning has attracted much attention over the last decade, the privacy preserving deep learning proposal was first appeared in 2015 [45]. It argues for privacy preserving model training in a collaborative federated learning system which involves multiple participants jointly train a model by only sharing sanitized parameters and their updates with each other while keeping their training data private and local. Following this work, deep learning with differential privacy was proposed in 2016 [46]. Differential privacy, a defacto standard for privacy that offers provable privacy guarantees, has been applied for privacy preserving machine learning [47, 48, 49, 50, 51]. Differential privacy requires characterization of output differences over any two input datasets differing in at most one element, but it is not known to characterize that for deep learning because the internal representations of deep neural networks is already notoriously difficult to understand. The prior works [46, 52, 53] suggest to use norm gradient clipping in the stochastic gradient descent (SGD) algorithm to bound the influence of any single example on the gradients and apply differentially private mechanisms to perturb the gradients accordingly. By ensuring each gradient descent step differentially private, the final output model satisfies a certain level of differential privacy given its composition property. It is known that the SGD training process of a deep neural network tends to involve a large number of iterations. Given the targeted differential privacy guarantee, its differentially private version needs a tight estimation on the privacy loss for the composition of differentially private iterations, such that the algorithm can track accumulative privacy loss during the training process and terminate it before the loss exceeds the privacy budget. Unfortunately, Abadi and his co-authors [46] have shown existing strong composition theorem [54] for differential privacy does not yield a tight analysis. To address this problem, the moments accountant method is proposed, which tracks the log moments of privacy loss variable and provide much tighter estimate of the privacy loss for composing Gaussian mechanisms with random sampling.

In this paper, we address several issues with using the above differentially private SGD algorithm and privacy accounting method for deep learning. The first problem is related to underestimation of privacy loss caused by data batching strategies. For the computation efficiency, the SGD algorithm usually takes small batches from the training dataset each time to compute gradients and update model parameters. The previous work [46] exploits privacy amplification of random sampling to produce tighter estimation on privacy loss. It is based on assumption that the data batches for mini-batch SGD input are generated through random sampling with replacement on the training dataset. In practice, for better efficiency, the data batching strategy is implemented through random reshuffling that randomly shuffles the training dataset and then partition them into batches of the similar size. We note that the random sampling and random reshuffling are two different implementation strategies for data batching. Our analysis and experiments show that they actually cause distinct privacy loss. Thus, the analysis of composition of differentially private mechanisms depends on how data is accessed by every mechanism, and simply treating random reshuffling and random sampling as the same data access will lead to underestimation of privacy loss.

The second issue is the need of a tight analysis on accumulative privacy loss for differentially private SGD that tends to have a large number of iterations. To address this problem, we propose to use concentrated differential privacy (CDP), a generalization of differential privacy recently introduced by Dwork and Rothblum [55]. It is developed to focus on the cumulative privacy loss for a large number of computations and provide a sharper analysis tool. Based on CDP, we analyze the privacy loss under different data batch strategies and develop privacy accounting methods for each respectively. In particular, for the random reshuffling, we show that our analysis provides a tighter estimation on privacy loss than the approach that applies strong composition theorem even with taking advantage of the privacy amplification effect of random sampling by assuming random reshuffling as a random sampling process. For the random sampling batching strategy, we show that CDP is not able to capture its privacy amplification effect. We address this problem by a relaxation and conversion to traditional (ϵ, δ)-differential privacy. Compared with the moments account method that require numerical computation of log moments for a range of moment orders, our method for privacy accounting under random sampling produces very close estimation but with much simpler and faster computation.

The third novelty of our approach to differentially private model publishing for deep learning is our development of dynamic privacy budget allocation to improve the model accuracy under differentially private training. The perturbed gradients during the training process inevitably degrade the model accuracy but for the deep learning accuracy is the most important goal. Because we aim to achieve differential privacy guarantee on the final output model, we have much less concern on the privacy loss of single iteration, which naturally provide opportunities to optimize the model accuracy via adjusting privacy budget allocation for every training iteration. In this paper we develop several different ways for dynamic privacy budget allocation and our extensive experiments demonstrate its benefit on improving the model accuracy. It is worth to note that, the techniques proposed in this paper, including the CDP based privacy accounting, its refinement under different data batching strategies, and dynamic privacy budget allocation over iterations, not only apply to neural networks, but also apply to any other iterative model training algorithms.

3.2 Background

3.2.1 Deep Learning

Deep learning uses neural networks that are defined as a hierarchical composition of parameterized functions to model the input data. For supervised learning, the training data are labeled with correct classes, and a multi-layer neural network is deployed to model the correlation between data instances and their labels. A typical neural network consists of $n(n > 1)$ layers of neurons. Each layer of neurons is parameterized by a weight matrix $W^{(l)}$ and a bias vector $b^{(l)}$. Layers apply an affine transformation to the previous layer's output and then computes an activation function σ over that. Typical examples of the activation function σ are sigmoid, rectified linear unit(ReLU) and tanh.

The training of a neural network aims to learn the parameters $\theta = \{W^{(l)}, b^{(l)} | 1 \leq l \leq n\}$ that minimize a loss function L defined to represent the penalty for misclassifying the training data. It is usually a non-convex optimization problem and solved by gradient descent. The gradient descent method iteratively computes the gradient of the loss function L and updates the parameters every step until the loss converges to a local optimum. In practice, the training of neural networks uses the mini-batch stochastic gradient descent (SGD) algorithm, which is much more efficient for large datasets. At each step a batch B of examples is sampled from the training dataset and the gradient of the average loss is computed, i.e., $\frac{1}{|B|}\sum_{x \in B} \nabla_\theta L(\theta, x)$ as $\nabla_\theta L(\theta)$. The SGD algorithm then applies the following update rule for parameters θ

$$\theta = \theta - \alpha \nabla_{\theta} L(\theta) \tag{3.1}$$

where α is the learning rate. The running time of the mini-batch SGD algorithm is usually expressed as the number of *epochs*. Each epoch consists of all of the batches of the training dataset, i.e., in an epoch every example has been seen once. Within an epoch, the pass of one batch of examples for updating the model parameters is called one *iteration*.

3.2.2 Differential Privacy

Differential privacy is a rigorous mathematical framework that formally defines the privacy properties of data analysis algorithms. Informally it requires that any changes to a single data point in the training dataset can only cause statistically insignificant changes to the algorithm's output.

Definition 5 (Differential Privacy [27]). *A randomized mechanism $\mathcal{A}$ provides (ϵ, δ)-differential privacy if for any two neighboring database D and D' that differ in only a single entry, $\forall S \subseteq Range(\mathcal{A})$,*

$$\Pr(\mathcal{A}(D) \in S) \leq e^{\epsilon} \Pr(\mathcal{A})(D') \in S) + \delta \tag{3.2}$$

If $\delta = 0$, $\mathcal{A}$ is said to be ϵ-differential privacy. In the rest of this paper, we write (ϵ, δ)-DP for short.

The standard approach to achieving differential privacy is the sensitivity method [29, 27] that adds to the output some noise that is proportional to the sensitivity of the query function. The sensitivity measures the maximum change of the output due to the change of a single database entry.

Definition 6 (Sensitivity [29]). *The sensitivity of a query function $q : \mathcal{D} \rightarrow \mathbb{R}^d$ is*

$$\Delta = \max_{D,D'} ||q(D) - q(D')|| \tag{3.3}$$

where D, $D' \in \mathcal{D}$ are any two neighboring datasets that differ at most one element, $||\cdot||$ denotes L_1 or L_2 norm.

In this paper, we choose the Gaussian mechanism that uses L_2 norm sensitivity. It adds zero-mean Gaussian noise with variance $\Delta^2\sigma^2$ in each coordinate of the output $q(D)$, as

$$q(D) + \mathcal{N}(0, \Delta^2\sigma^2\mathbf{I}) \tag{3.4}$$

It satisfies (ϵ,δ)-DP if $\sigma^2 > 2\log(\frac{1.25}{\delta})/\epsilon^2$ and $\epsilon \in (0, 1)$ [56].

3.2.3 Concentrated Differential Privacy

Concentrated differential privacy (CDP) is a generalization of differential privacy recently introduced by Dwork and Rothblum [55]. It aims to make privacy-preserving algorithms more practical for large numbers of computations than traditional DP while still providing strong privacy guarantees. It allows the computation to have much less concern about single-query loss but high probability bounds for the cumulative loss, and provides sharper and more accurate analysis on the cumulative loss for multiple computations compared to the popular (ϵ, δ)-DP.

CDP considers privacy loss on an outcome o as a random variable when the randomized mechanism $\mathcal{A}$ operates on two adjacent database D and D':

$$L^{(o)}_{(\mathcal{A}(D)||\mathcal{A}(D'))} \triangleq \log\frac{\Pr(\mathcal{A}(D) = o)}{\Pr(\mathcal{A}(D') = o)} \tag{3.5}$$

The (ϵ, δ)-DP guarantee ensures that the privacy loss variable is bounded by ϵ but exceeds that with probability no more than δ. As a relaxation to that, (μ, τ)-concentrated differential privacy, (μ, τ)-CDP for short [55], ensures that the mean (i.e., expectation) of the privacy loss is no more than μ and the probability of the loss exceeding its mean by an amount of $t\cdot\tau$ is bounded by $e^{-t^2/2}$. An alternative formulation of CDP to Dwork and Rothblum's (μ, τ)-CDP is proposed by Bun and Steinke [57], called "zero-concentrated differential privacy" (zCDP for short). Instead of mean concentrated as (μ, τ)-CDP, zCDP makes privacy loss concentrated around zero (hence the name), still following sub-Gaussian such that larger deviations from zero become increasingly unlikely.

Definition 7 (Zero-Concentrated Differential Privacy (zCDP)[57])**.** *A randomized mechanism $\mathcal{A}$ is ρ-zero concentrated differentially private (i.e., ρ-zCDP) if for any two neighboring databases D and D' that differ in only a single entry and all $\alpha \in (1, \infty)$,*

$$D_\alpha(\mathcal{A}(D)||\mathcal{A}(D')) \triangleq \frac{1}{\alpha - 1} \log\left(\mathbb{E}\left[e^{(\alpha-1)L^{(o)}}\right]\right) \leq \rho\alpha \tag{3.6}$$

Where $D_\alpha(\mathcal{A}(D)||\mathcal{A}(D'))$ called α-Rényi divergence between the distributions of $\mathcal{A}(D)$ and $\mathcal{A}(D')$.

The (ϵ, δ)-DP bounds the privacy loss by ensuring $\Pr(L^{(o)} > \epsilon) \leq \delta$. In contrast, zCDP entails a bound on the moment generating function of privacy loss $L^{(o)}$, indicated by an equivalent form of (3.6)

$$\mathbb{E}\left[e^{(\alpha-1)L^{(o)}}\right] \leq e^{(\alpha-1)\alpha\rho} \tag{3.7}$$

This implies that for zCDP, privacy loss $L^{(o)}$ is assumed to be a sub-Gaussian random variable such that it has a strong tail decay property, namely, $\Pr(L^{(o)} > t + \rho) \leq e^{-t^2/(4\rho)}$ for all $t > 0$ [57]. The following propositions are restatements of some zCDP results given in [57] that will be used in our paper.

Proposition 1. *If $\mathcal{A}$ provides ρ-zCDP, then $\mathcal{A}$ provides $(\rho + 2\sqrt{\rho \log(1/\delta)}, \delta)$-DP for any $\delta > 0$.*

Proposition 2. *The Gaussian mechanism with noise $\mathcal{N}(0, \Delta^2\sigma^2\mathbf{I})$ satisfies ($\frac{1}{2\sigma^2}$)-zCDP.*

We use zCDP instead of original (μ, τ)-CDP because zCDP is comparable to (ϵ, δ)-DP, as indicated by Proposition 1 and is immune to post-processing while (μ, τ)-CDP is not closed under post-processing [57].

3.2.4 Composition

Differential privacy offers elegant composition properties that enable more complex algorithms and data analysis task via the composition of multiple differentially private building blocks. The composition should have privacy guarantees degraded gracefully with multiple outputs that may be subjected to the joint analysis from building blocks.

For a sequential composition of k mechanisms $\mathcal{A}_1, \ldots, \mathcal{A}_k$ satisfying (ϵ_i, δ_i)-DP for i=1,. . ., k respectively, the basic composition result [56] shows that the privacy composes linearly, i.e., the sequential composition satisfies $(\sum_i^k \epsilon_i, \sum_i^k \delta_i)$-DP. When $\epsilon_i = \epsilon$ and $\delta_i = \delta$, the strong composition bound from [54] states that the composition satisfies $(\epsilon\sqrt{2klog(1/\delta')} + k\epsilon(e^\epsilon - 1), k\delta + \delta')$-DP. For zCDP, it has a simple linear composition property [57]:

Theorem 5. *Two randomized mechanisms $\mathcal{A}_1$ and $\mathcal{A}_2$ satisfy ρ_1-zCDP and ρ_2-zCDP respectively, their sequential composition $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ satisfies $(\rho_1 + \rho_2)$-zCDP.*

Compared with (ϵ, δ)-DP, CDP provides a tighter bound on the cumulative privacy loss under composition, which makes it more suitable for algorithms running a large number of iterations. In other words, while providing the same privacy guarantee, CDP allows lower noise scale and thus better accuracy. Consider k iterative composition of a Gaussian mechanism with noise $\mathcal{N}(0, \sigma^2\mathbf{I})$. To guarantee the final (ϵ, δ)-DP, in terms of (ϵ_i, δ_i)-DP for every iteration, the permitted loss ϵ_i of each iteration is $\epsilon_i = \epsilon/(2\sqrt{2k\log(1/(\delta - k\delta_i))})$, which will be very low when k is large. The noise scale is $\sigma = \frac{\sqrt{k}}{\epsilon}(4\sqrt{\log(1.25/\delta_i)\log(1/(\delta - k\delta_i))})$. Suppose $\delta_i = \delta/(k+1)$, we then have $\sigma > \frac{\sqrt{k}}{\epsilon}(4\log((k+1)/\delta))$. In contrast, with using ρ_i-zCDP for every iteration, because $\rho \approx \epsilon^2/(4\log(1/\delta))$ zCDP satisfies (ϵ, δ)-DP [57] and $\rho_i = \frac{1}{k}\rho$. It is easy to show that the noise scale $\sigma = \frac{\sqrt{k}}{\epsilon}\sqrt{2log(1/\delta)}$ which is multiple times smaller than the noise scale derived under (ϵ, δ)-DP. On the other hand, a single parameter ρ of zCDP and its linear composition naturally fit the concept of a privacy budget. Thus, zCDP is an appropriate choice for privacy accounting.

3.3 Overview

Because it is difficult to characterize the maximum difference of the model parameters over any two neighboring datasets for neural networks, differentially private deep learning [46, 52, 53] relies on differentially private stochastic gradient descent (DP-SGD) to control the influence of training data on the model. This approach explicitly bounds per-example gradients $\nabla_\theta L(\theta, x)$ in every iteration by clipping the L_2 norm of gradient vectors. Given a clipping threshold C, this is done by replacing the gradient vector g with $\mathbf{g}/\max(1, \frac{||\mathbf{g}||_2}{C})$ which scales g down to norm C if $||\mathbf{g}||_2 > C$. A Gaussian mechanism with L_2 norm sensitivity of C is then applied to perturb the gradients before the gradient descent step in Eq. (3.1) updates the model parameters. Because each SGD step is differentially private, by the composition property of differential privacy, the final model parameters are also differentially private. The problem with DP-SGD is that the training of a deep neural network (DNN) tends to have a large number of iterations, which causes large cumulative privacy loss at the end. Therefore, a tight estimation of privacy loss under composition is critical for allowing lower noise scale or more training iterations (for desired accuracy) when we have a fixed privacy budget.

To analyze the cumulative privacy loss of DP-SGD, we employ concentrated differential privacy(CDP) which was developed to accommodate a larger number of computations and provides sharper and tighter analysis of privacy loss than the strong composition theorem of (ϵ, δ)-DP. One way to track the privacy loss of DP-SGD is the Moments Accountant (MA) method proposed by Abadi et al. [46]. It assumes that the data batches for mini-batch SGD are generated by randomly sampling examples from the training dataset with replacement, MA takes advantage of the privacy amplification effect of random sampling to achieve a much tighter estimate on privacy loss than the strong composition theorem. It has been shown in [58] that running an (ϵ, δ) differentially private mechanism over a set of examples each of which is independently sampled with probability q $(0 < q < 1)$

achieves $(\log(1 + q(e^{\epsilon} - 1)), q\delta)$-DP. However, in practice, random batches are generated by randomly shuffling examples and then partitioning them into batches for computation efficiency, which is distinct from random sampling with replacement. By analyzing the privacy loss under these two data batching methods, random sampling with replacement and random reshuffling respectively, we show that 1) random sampling with replacement and random reshuffling result in different privacy loss; and 2) privacy accounting using the MA method underestimates the actual privacy loss of their neural network training, because it simply regards random reshuffling as random sampling with replacement. To address these problems, we develop different privacy accounting methods for each of the batching methods, and our algorithm makes proper choices depending on which method is used for data batching. For privacy accounting under random sampling based batching, we show that CDP is unable to capture the privacy amplification effect of random sampling. To address that, we propose a relaxation of zCDP and convert it to (ϵ, δ)-DP. Compared with MA, which needs to compute log moments of privacy loss over a range of moment orders, our method uses explicit expressions to compute privacy loss and thus is more efficient, particularly when the noise scale of the Gaussian mechanism dynamically changes along the training.

In our approach, dynamic privacy budget allocation is applied to DP-SGD to improve the model accuracy. In model publishing the privacy loss of each learning step is not our primary concern. This allows us to allocate different privacy budgets to different training epochs as long as we maintain the same overall privacy guarantee. Our dynamic budget allocation approach is in contrast to the previous work [46], which employs a uniform privacy budget allocation, and uses the same noise scale in each step of the whole training process. Our dynamic privacy budget allocation approach leverages several different ways to adjust the noise scale. Our experimental results demonstrate that this approach achieves better model accuracy while retaining the same privacy guarantee.

Algorithm 2 presents our DP-SGD algorithm. In each iteration, a batch of examples

Algorithm 2: Differentially Private SGD Algorithm

```
Input: Training examples $\{x_1, \ldots, x_N\}$, learning rate $\eta_t$, group size $L$, gradient norm bound
       $C$, total privacy budget $\rho_{total}$
1  Initialize $w_0$ ;
2  Initialize cumulative privacy loss $c_t^{priv} = 0$;
3  for $t = 1 : T$ do
4      Dynamic privacy budget allocation:
5      $\sigma_t \leftarrow AdpBudgetAlloc(\rho_{total}, t, T, schedule)$;
6      update $c_t^{priv}$ according to data batching method, $t$ and $\sigma_t$;
7      If $c_t^{priv} > \rho_{total}$, break ;
8      data batching:
9      Take a batch of data samples $\mathbb{B}_t$ from the training dataset;
10     $B = |\mathbb{B}_t|$;
11     Compute gradient:
12     For each $i \in \mathbb{B}_t$, $\mathbf{g}_t(x_i) \leftarrow \nabla_{w_t}\mathcal{L}(w_t, x_i)$;
13     Clip gradient:
14     $\hat{\mathbf{g}}_t(x_i) \leftarrow \mathbf{g}_t(x_i)/max\left(1, \frac{||\mathbf{g}_t(x_i)||_2}{C}\right)$;
15     Add noise:
16     $\widetilde{\mathbf{g}}_t \leftarrow \frac{1}{B}\big(\sum_i \hat{\mathbf{g}}_t(x_i) + \mathcal{N}(0, \sigma_t^2 C^2 \mathbb{I})\big)$;
17     Descent:
18     $w_{t+1} \leftarrow w_t - \eta_t \widetilde{\mathbf{g}}_t$ ;
19 Output $w_T$ ;
```

is sampled from the training dataset and the algorithm computes the gradient of the loss on the examples in the batch and uses the average in the gradient descent step. The gradient clipping bounds per-example gradients by l_2 norm clipping with a threshold C. The Gaussian mechanism adds random noise $\mathcal{N}(0, \sigma_t^2 C^2 \mathbb{I})$ to $\sum_i \hat{\mathbf{g}}_t(x_i)$ to perturb the gradients in every iteration. We have a total privacy budget ρ_{total} and cumulative privacy cost c_t^{priv}. The way to update c_t^{priv} depends on which batching method is used. If the privacy cost exceeds the total budget, then the training is terminated. In the pseudo code, the function $AdpBudgetAlloc(\rho_{total}, t, T, schedule)$ is used to obtain the noise scale σ_t for the current training step t, according to $schedule$ that decides how the noise scale is adjusted during the training time.

3.4 Details of Our Approach

In this section, we present the details of our approach for differentially private deep learning. We first propose our dynamic privacy budget allocation techniques and then develop privacy accounting methods based on zCDP for different data batching methods.

3.4.1 Dynamic Privacy Budget Allocation

In Algorithm 2, the privacy budget allocated to an epoch decides the noise scale of Gaussian mechanism used by each iteration within that epoch. For a given privacy budget ρ_{total}, the final model accuracy depends on how the privacy budget is distributed over the training epochs. Our approach aims to optimize the budget allocation over the training process to obtain a differentially private DNN model with better accuracy.

Concretely, our dynamic privacy budget allocation follows the idea that, as the model accuracy converges, it is expected to have less noise on the gradients, which allows the learning process to get closer to the local optimal spot and achieve better accuracy. A similar strategy has been applied to the learning rate of DNNs in common practice. It is often recommended to reduce the learning rate as the training progresses, instead of using a constant learning rate throughout all epochs, to achieve better accuracy [59, 60]. Therefore, we propose a set of methods for privacy budget allocation, which effectively improve the model accuracy by dynamically reducing the noise scale over the training time (as demonstrated by our experiments).

Adaptive schedule based on public validation dataset

One approach for adjusting the noise scale is to monitor the validation error during training and reduce the noise scale whenever the validation error stops improving. We propose an adaptive privacy budget allocation that dynamically reduces the noise scale according to the validation accuracy. Every time when the validation accuracy improves by less than a threshold δ, the noise scale is reduced by a factor of k until the total privacy budget runs out. However, when the validation dataset is sampled from the private training dataset, the schedule has some dependency on the private dataset, which adds to the privacy cost. In this case, if we can leverage a small publicly available dataset from the same distribution and use it as our validation dataset, then it will not incur any additional privacy loss.

In our approach, with a public validation dataset, the validation accuracy is checked periodically during the training process to determine if the noise scale needs to be reduced for subsequent epochs. The epochs over which the validation is performed are referred to as validation epochs. Let σ_e be the noise scale for the DP-SGD training in the validation epoch e, and S_e be the corresponding validation accuracy . The noise scale for the subsequent epochs is adjusted based on the accuracy difference between current epoch e and the previous validation epoch $e - 1$. Initially, $S_0 = 0$,

$$\sigma'_e = \begin{cases} k\sigma_e, & \text{if } S_e - S_{e-1} \leq \delta \quad (3.8) \\ \sigma_e & \quad (3.9) \end{cases}$$

The updated noise scale σ'_e is then applied to the training until the next validation epoch $e+1$. The above equations amount to say that if the improvement on the validation accuracy $S_e - S_{e-1}$ is less than the threshold δ, it triggers the decay of noise scale $\sigma_e = k\sigma_e$ where k $(0 < k < 1)$ is decay rate, a hyperparameter for the schedule. We note that the validation accuracy may not increase monotonically as the training progresses, and its fluctuations may cause unnecessary reduction of noise scale and thus waste on the privacy budget. This motivates us to use the moving average of validation accuracy to improve the effectiveness of validation-based noise scale adjustment: at validation epoch e, we define an averaged validation accuracy $\bar{S}_e$ over the previous m validation epochs from e, including itself, as follows:

$$\bar{S}_e = \frac{1}{m} \sum_{i=e-m+1}^{e} S_i \tag{3.10}$$

The schedule checks the averaged validation accuracy every $period$ ($period \geq m$) number of validation epochs and compares the current result with that of the last checking time to decide if the noise scale needs to be reduced according to Eq. (3.8).

Pre-defined schedules

When the public validation dataset is not available, we propose to use an alternative approach that pre-defines how the noise scale decreases over time without accessing any datasets or checking the model accuracy. Concretely, in our approach, the noise scale is reduced over time according to some decay functions. The decay functions update the noise scale by epoch, while the noise scale keeps the same for every iteration within an epoch. By leveraging the Gaussian mechanism that adds noise from $\mathcal{N}(0, \sigma_t^2 C^2 \mathbb{I})$, we provide four instances of decay functions, all of which take the epoch number t as an argument.
a) Time-Based Decay: It is defined with the mathematical form

$$\sigma_t = \sigma_0/(1 + kt)$$

where σ_0 is the initial noise parameter for σ, t is the epoch number and k ($k > 0$) is decay rate. when $k < 1$, it is known as "search-then-converge" [61], which decreases the noise scale linearly during the SGD search phase when t is less than the "search time" $1/k$, and decreases the noise by $1/t$ when t is greater than $1/k$.
b) Exponential Decay: It has the mathematical form

$$\sigma_t = \sigma_0 e^{-kt}$$

where k $(k > 0)$ is decay rate.
c) Step Decay: Step decay reduces the learning rate by some factor every few epochs. The mathematical form is

$$\sigma_t = \sigma_0 * k^{\lfloor t/period \rfloor}$$

where k $(1 > k > 0)$ is the decay factor and $period$ decides how often to reduce noise in terms of the number of epochs.

d) Polynomial Decay: It has the mathematical form

$$\sigma_t = (\sigma_0 - \sigma_{end}) * (1 - t/period)^k + \sigma_{end}$$

where $k (k > 0)$ is the decay power and $t < period$. A polynomial decay function is applied to the initial σ_0 within the given number of epochs defined by $period$ to reach σ_{end} ($\sigma_{end} < \sigma_0$). When $k = 1$, this is a linear decay function.

The schedules for noise decay are not limited to the above four instances. In this paper we choose to use these four because they are simple, representative, and also used by learning rate decay in the performance tuning of DNNs. Users can apply them in various ways. For example, a user can use them in the middle of training phase, but keep constant noise scale before and after. Note that in the time-based decay and exponential decay, t can be replaced by $\lfloor t/period \rfloor$ as done in the step decay such that the decaying is applied every $period$ number of epochs. The polynomial decay requires a specific end noise scale after $period$ epochs, and we make the noise scale constant after the training time exceed $period$ epochs.

Privacy Preserving Parameter Selection

The proposed schedules require a set of pre-defined hyperparameters, such as decay rate and period. Their values decide the training time and affect the final model accuracy. It is expected to find the optimal hyperparameters for the schedules to produce the most accurate model. A straightforward approach is to test a list of k candidates by training k neural networks respectively and trivially choose the one that achieves the highest accuracy, though this adds the privacy cost up to $k\rho_{total}$. A better approach is to apply differentially private parameter tuning, such as the mechanism proposed by Gupta et al. [50]. The idea is to partition the dataset to $k + 1$ equal portions, train k models with using k schedules on k different data portions respectively, and evaluate the number of incorrect predictions for each model, denoted by z_i ($1 \leq i \leq k$), on the remaining data portion. Then, the

Exponential Mechanism [30] is applied, which selects and outputs a candidate with the probability proportional to $exp(\frac{-\epsilon z_i}{2})$. This parameter tuning procedure satisfies ϵ-DP, and accordingly satisfies $\frac{1}{2}\epsilon^2$-zCDP [57].

3.4.2 Refined Privacy Accountant

The composition property of zCDP allows us to easily compute cumulative privacy loss for the iterative SGD training algorithm. Suppose that each iteration satisfies ρ-zCDP and the training runs T iterations, then the whole training process satisfies $(T\rho)$-zCDP. In this section we show that 1) the composition can be further refined by considering the property of the mini-batch SGD algorithm, and 2) more importantly, different batching methods lead to different privacy loss. In particular, we analyze the privacy loss composition under two common batching methods: random sampling with replacement and random reshuffling. With random reshuffling, the training dataset is randomly shuffled and then partitioned into batches of similar size and SGD sequentially processes one batch at a time. It is a random sampling process without replacement. For random sampling with replacement, each example in a batch is independently sampled from the training dataset with replacement. Because these two data batching methods have different privacy guarantees, for tracking privacy loss correctly, it is important for the users to choose the right accounting method based on the batching method they use.

Under random reshuffling

SGD takes disjoint data batches as input within an epoch with random reshuffling. We note that the existing results [55, 57, 29] on the composition of a sequence of differential private mechanisms $\mathcal{A}_1, \ldots, \mathcal{A}_k$ assume that each mechanism $\mathcal{A}_i$ runs with the same dataset X as input. It is expected that their composition has less cumulative privacy loss if each of differentially private mechanisms runs on disjoint datasets. The formal composition result in this scenario is detailed in Theorem 6.

Theorem 6. *Suppose that a mechanism $\mathcal{A}$ consists of a sequence of k adaptive mechanisms, $\mathcal{A}_1, \ldots, \mathcal{A}_k$, where each $\mathcal{A}_i : \prod_{j=1}^{i-1} \mathcal{R}_j \times \mathcal{D} \to \mathcal{R}_i$ and $\mathcal{A}_i$ satisfies ρ_i-zCDP ($1 \le i \le k$). Let $\mathbb{D}_1, \mathbb{D}_2, \ldots, \mathbb{D}_k$ be the result of a randomized partitioning of the input domain $\mathbb{D}$. The mechanism $\mathcal{A}(D) = (\mathcal{A}_1(D \cap \mathbb{D}_1), \ldots, \mathcal{A}_k(D \cap \mathbb{D}_k))$ satisfies*

$$\begin{cases} \rho\text{-zCDP}, & \text{if } \rho_i = \rho, \forall i \\ \max_i \rho_i\text{-zCDP}, & \text{if } \rho_i \neq \rho_j \text{ for some } i, j \end{cases} \tag{3.11}$$

Proof. Suppose two neighboring datasets D and D'. Without loss of generality, assume that D contains one more element d_e than D'. Let $D_i = D \cap \mathbb{D}_i$ and $D'_i = D' \cap \mathbb{D}_i$. Accordingly, there exists j such that D_j contains one more element than D'_j, and for any $i \neq j$, $D_i = D'_i$. Consider any sequence of outcomes $o = (o_1, \ldots, o_k)$ of $\mathcal{A}_1(D_1), \ldots, \mathcal{A}_k(D_k)$.

Because only D_j is different from D'_j, for any $i \neq j$, we have

$$\Pr[\mathcal{A}_i(D_i) = o_i | \mathcal{A}_{i-1}(D_{i-1}) = o_{i-1}, \ldots, \mathcal{A}_1(D_1) = o_1]$$

equal to

$$\Pr[\mathcal{A}_i(D'_i) = o_i | \mathcal{A}_{i-1}(D'_{i-1}) = o_{i-1}, \ldots, \mathcal{A}_1(D'_1) = o_1]$$

Then, we have

$$\begin{aligned} L_j^{(o)} &\triangleq \log \left(\frac{\Pr(\mathcal{A}(D) = o}{\Pr(\mathcal{A}(D') = o)} \right) \\ &= \log \left(\frac{\prod_{i \in [n]} \Pr[\mathcal{A}_i^{D_i} = o_i | \mathcal{A}_{i-1}^{D_{i-1}} = o_{i-1}, .., \mathcal{A}_1^{D_1} = o_1]}{\prod_{i \in [n]} \Pr[\mathcal{A}_i^{D'_i} = o_i | \mathcal{A}_{i-1}^{D'_{i-1}} = o_{i-1}, .., \mathcal{A}_1^{D'_1} = o_1]} \right) \\ &= \log \left(\frac{\Pr[\mathcal{A}_j^{D_j} = o_j | \mathcal{A}_{j-1}^{D_{j-1}} = o_{j-1}, .., \mathcal{A}_1^{D_1} = o_1]}{\Pr[\mathcal{A}_j^{D'_j} = o_j | \mathcal{A}_{j-1}^{D'_{j-1}} = o_{j-1}, .., \mathcal{A}_1^{D'_1} = o_1]} \right) \\ &\triangleq c_j(o_j; o_1, \ldots, o_{j-1}) \end{aligned}$$

where $\mathcal{A}_i^{D_i}$ denotes $\mathcal{A}_i(D_i)$ for short.

Once the prefix $(o_1, \ldots, o_{j-1})$ is fixed,

$$C_j \triangleq c_j(o_j; o_1, \ldots, o_{j-1}) = \log\left(\frac{\Pr(\mathcal{A}_j(D_j) = o_j)}{\Pr(\mathcal{A}_j(D'_j) = o_j)}\right)$$

By the ρ_j-zCDP property of $\mathcal{A}_j$, $\mathbb{E}\left[e^{(\alpha-1)C_j}\right] \le e^{(\alpha-1)\alpha\rho_j}$, thus

$$\begin{aligned}\mathbb{E}\left[e^{(\alpha-1)L_j^{(o)}}\right] &= \mathbb{E}\left[\exp\Big((\alpha-1)C_j\Big)\right] \\ &\le e^{(\alpha-1)\alpha\rho_j}\end{aligned}$$

Because of randomized partition of the input domain $\mathbb{D}$, the extra element d_e of D is randomly mapped to k partitions. Therefore, j is uniformly distributed over $\{1, \ldots, k\}$, and thus the privacy loss $L^{(o)}$ under random data partition is the mixture of independent random variables $L_1^{(o)}, \cdots, L_k^{(o)}$,

$$f(L^{(o)}) = \frac{1}{k} f(L_1^{(o)}) + \ldots + \frac{1}{k} f(L_k^{(o)})$$

where $f(X)$ is the probability distribution function of X.

We have

$$\mathbb{E}\left[e^{(\alpha-1)L^{(o)}}\right] = \frac{1}{k}\sum_{j=1}^{k}\mathbb{E}\left[\exp\Big((\alpha-1)L_j^{(o)}\Big)\right]$$

Because $L_j^{(o)}$ satisfies zCDP, by (3.7) then we have

$$\le \frac{1}{k}\sum_{j=1}^{k}\exp((\alpha-1)\alpha\rho_j)$$

If $\rho_j = \rho\ \forall j$, we have $\mathbb{E}\left[e^{(\alpha-1)L^{(o)}}\right] \le \exp((\alpha-1)\alpha\rho)$, and thus the mechanism $\mathcal{A}(D)$ satisfies ρ-zCDP.

If not all ρ_j are the same, we replace each ρ_j with $\max_j \rho_j$, we have $\mathbb{E}\left[e^{(\alpha-1)L^{(o)}}\right] \le \exp((\alpha-1)\alpha \max_j \rho_j)$, and the mechanism $\mathcal{A}(D)$ satisfies $\max_i \rho_i$-zCDP. $\square$

Theorem 6 provides a tighter characterization of privacy loss for the composition of mechanisms having disjoint input data. Assume for all $i \in (1, \ldots, k)$, $\rho_i = \rho$. Given Theorem 6, it is trivial to demonstrate that mechanism $\mathcal{A}$ satisfies ρ-zCDP. Compared with a guarantee of $(k\rho)$-zCDP using the composition property in Proposition 5, we demonstrate that the total privacy loss of sequential computations on disjoint datasets is just ρ, equivalent with one computation step in the sequence. For ϵ-DP, the similar result in [62] of the parallel composition theorem says that when each ϵ_i-differentially private mechanism queries a disjoint subset of data in parallel and work independently, their composition provides $(\max_i \epsilon_i)$-DP instead of the $\sum_i \epsilon_i$ derived from a naive composition.

In the differentially private mini-batch SGD algorithm shown in Algorithm 2, each iteration step t satisfies $(\frac{1}{2\sigma_t^2})$-zCDP according to Proposition 2. Suppose that each iteration from the same epoch uses the same noise scale σ and each uses a disjoint data batch. Then, by Theorem 6, we know that the computation of this epoch still satisfies $(\frac{1}{2\sigma^2})$-zCDP. Because the training dataset is repeatedly used every epoch, the composition of the epoch level computations follows normal composition of Proposition 5. Thus, when the training runs a total of E epochs and each epoch e satisfies ρ_e-zCDP, the whole training procedure satisfies $(\sum_{e=1}^{E} \rho_e)$-zCDP.

Under random sampling with replacement

We have shown a privacy amplification effect resulting from the disjoint data access of every iteration within one epoch under random reshuffling. In contrast, the MA method [46] exploits the privacy amplification effect of random sampling with replacement. In this section, we examine how random sampling with replacement affects the privacy loss in terms of zCDP. Intuitively, the random sampling with replacement introduces more uncertainty than the random reshuffling process which samples data batches without replacement. However, our analysis shows that CDP cannot characterize the privacy amplification effect of random sampling. It is because of the restrictive notion of sub-Gaussianity in CDP which requires moment constraints on all orders, i.e., $\alpha \in (1, \infty)$ in Eq. (3.7). To address this problem, we propose a relaxation of zCDP and convert it to (ϵ, δ)-DP. This then allows us to capture the privacy amplification of random sampling.

Suppose a new mechanism $\mathcal{A}'$ that runs ρ-zCDP mechanism $\mathcal{A}$ on a random subsample of dataset $\mathbb{D}$ where each example is independently sampled with probability q. Without loss of generality, we fix $\mathbb{D}$ and consider a neighboring dataset $\mathbb{D}'$= $\mathbb{D} \cup d_e$. we use $\Lambda(*)$ to denote the sampling process over dataset $*$, let T be any subsample that does not include d_e and $T' = T \cup d_e$. Because d_e is randomly sampled with probability q, $\mathcal{A}'(\mathbb{D}')$ is distributed identically as $u_0 \triangleq \Pr(\Lambda(\mathbb{D}) = T)\mathcal{A}(T)$ with probability $(1-q)$, and as $u_1 \triangleq \Pr(\Lambda(\mathbb{D}') = T' | d_e \in T')\mathcal{A}(T')$ with probability q. Therefore, the following holds:

$$\mathcal{A}'(\mathbb{D}) \sim u_0, \qquad \mathcal{A}'(\mathbb{D}') \sim qu_1 + (1-q)u_0 \tag{3.12}$$

By $\Pr\big(\Lambda(\mathbb{D}) = T\big)$=$\Pr\big(\Lambda(\mathbb{D}') = T' | d_e \in T'\big)$ due to $T' = T \cup d_e$, it is easy to prove that $\mathcal{A}'$ still satisfies ρ-zCDP. Suppose $\mathcal{A}'$ that runs ρ-zCDP mechanism $\mathcal{A}$ on a random subsample of dataset $\mathbb{D}$. Following the result (3.12) in the paper, the proof is as follows: Because $\Pr(\Lambda(\mathbb{D}') = T' | d_e \in T') = \Pr(\Lambda(\mathbb{D}) = T)$ and $\mathcal{A}$ satisfies ρ-zCDP, Rényi divergence

$$D_\alpha(u_0 || u_1) \leq D_\alpha(\mathcal{A}(T) || \mathcal{A}(T')) \leq \alpha\rho \tag{3.13}$$

$$D_\alpha(u_1 || u_0) \leq D_\alpha(\mathcal{A}(T') || \mathcal{A}(T)) \leq \alpha\rho \tag{3.14}$$

Using jointly quasi-convexity of Rényi divergence [63], we have

$$D_\alpha(u_0 || qu_1 + (1-q)u_0) \leq D_\alpha(u_0 || u_1) \leq \alpha\rho \tag{3.15}$$

$$D_\alpha(qu_1 + (1-q)u_0 \;||\; u_0) \leq D_\alpha(u_1 || u_0) \leq \alpha\rho \tag{3.16}$$

and thus $\mathcal{A}'$ still satisfies ρ-zCDP.

No privacy amplification for $\mathcal{A}'$ in terms of CDP. Consider

$$(\alpha - 1)D_\alpha(\mathcal{A}'(\mathbb{D}') || \mathcal{A}'(\mathbb{D}))$$

$$
\begin{aligned}
&= \log\left(\mathbb{E}_{u_0}\left[\left(\frac{qu_1+(1-q)u_0}{u_0}\right)^{\alpha}\right]\right) > \log\left(\mathbb{E}_{u_0}\left[\left(q\frac{u_1}{u_0}\right)^{\alpha}\right]\right)\\
&= \alpha \log q + (\alpha-1)D_\alpha(u_1||u_0) \qquad (3.17)
\end{aligned}
$$

When $\alpha \to \infty$, we have

$$
D_\alpha(\mathcal{A}'(\mathbb{D}')||\mathcal{A}'(\mathbb{D})) > \alpha\rho + \frac{\alpha}{\alpha-1}\log q \to \alpha\rho + \log q \qquad (3.18)
$$

This shows that the sampling does not produce any reduction with regard to q on ρ and still $D_\alpha(\mathcal{A}'(\mathbb{D}')||\mathcal{A}'(\mathbb{D})) = \Theta(\alpha\rho)$. Therefore, by definition, zCDP is not able to capture the privacy amplification effect of random sampling.

The reason there is no privacy amplification with random sampling on zCDP is that concentrated DP requires a sub-Gaussian distribution for privacy loss, and thus moments must be bounded by $\exp(O(\alpha^2))$ at all orders $\alpha \in (1, +\infty)$. This is a fairly strong condition. Alternatively, it requires that a mechanism $\mathcal{A}$ has α-Rényi divergence bounded by $\alpha\rho$ for all $\alpha \in (1, +\infty)$. To demonstrate that, we assume a Gaussian mechanism $\mathcal{A}$ that $\mathcal{A}(T) \sim N(0, \sigma^2)$ and $\mathcal{A}(T') \sim N(1, \sigma^2)$. Denote $N(0, \sigma^2)$ by p_0 and $N(1, \sigma^2)$ by p_1. It is trivial to show that $D_\alpha(u_1||u_0) = D_\alpha(p_1||p_0)$ and $D_\alpha(qu_1+(1-q)u_0||u_0) = D_\alpha(qp_1+(1-q)p_0||p_0)$. We can then numerically compute $D_\alpha(u_1||u_0)$ and $D_\alpha(qu_1+(1-q)u_0||u_0)$ with $q = 0.01$ and $\sigma = 4$. Results are shown in Figure 3.1. $D_\alpha(u_1||u_0)$ is linear in α because $D_\alpha(u_1||u_0) = \alpha\rho = \alpha/(2\sigma^2)$. However, we can see that $D_\alpha(qu_1 + (1-q)u_0||u_0)$ has a changing point at $\alpha = 147$. Before this changing point it is close to zero, because $q = 0.01$ is very small and the two distributions $qu_1 + (1-q)u_0$ and u_0 are close to each other. At higher orders, the sampling effect vanishes and the divergence increases at the rate of $D_\alpha(u_1||u_0)$. This indicates that the privacy amplification effect from random sampling does not hold for all orders $\alpha \in (1, +\infty)$, and we cannot improve the privacy metric in terms of zCDP under the random sampling. On the other hand, Figure 3.1 suggests that we need to analyze the α-Rényi divergence within a limited range of α to capture the privacy amplification effect. We will show that having a bound on $D_\alpha(\cdot||\cdot)$ within a limited range of α

makes it possible to capture the privacy amplification effect of random sampling. However, such constraint does not fit into the definition of CDP since it indicates a sub-exponential privacy loss variable, a relaxation to sub-Gaussianity in the definition of CDP. Therefore, in this paper we address it by converting the α-Rényi divergence under such relaxation to traditional (ϵ, δ)-DP.

In the following we show the conversion to (ϵ, δ)-DP in a general form for a Gaussian mechanism with bounded α-Rényi divergence in some limited range of α parameterized by q and σ, and then consider specific cases with concrete values for the range of α and bounds on α-Rényi divergence.

Suppose that $f : \mathbb{D} \to \mathbb{R}^p$ with $||f(\cdot)||_2 \leq 1$. Consider a mechanism $\mathcal{A}'$ that runs a Gaussian mechanism adding noise $\mathcal{N}(0, \sigma^2\mathbb{I})$ over a random subsample $J \subseteq \mathbb{D}$ where each example is independently sampled with probability q, i.e., $\mathcal{A}'(\mathbb{D}) = \sum_{i\in J} f(x_i) + \mathcal{N}(0, \sigma^2\mathbb{I})$. Let $D_\alpha(\cdot||\cdot)$ be the α-Rényi divergence between $\mathcal{A}'(\mathbb{D})$ and $\mathcal{A}'(\mathbb{D}')$ for two neighboring datasets $\mathbb{D}$ and $\mathbb{D}'$. Let $P(*)$ and $U_\alpha(*)$ be bounded functions (e.g., polynomial function) of q and σ. We assume $D_\alpha(\cdot||\cdot)$ is bounded by $\alpha \cdot P(q, \sigma)$ within a limited range of $1 < \alpha \leq U_\alpha(q, \sigma)$. Then, we have the following theorem:

Theorem 7. *Let $\widehat{\rho} = P(q, \sigma)$ and $u_\alpha = U_\alpha(q, \sigma)$. If the mechanism $\mathcal{A}'$ has*

$$D_\alpha(\mathcal{A}'(\mathbb{D})||\mathcal{A}'(\mathbb{D}')) \leq \alpha\widehat{\rho} \tag{3.19}$$

for $1 < \alpha \leq u_\alpha$, it satisfies

$$\begin{cases} \Big(\widehat{\rho} + 2\sqrt{\widehat{\rho}\log(1/\delta)}, \delta\Big) - DP, \text{ if } \delta \geq 1/\exp(\widehat{\rho}(u_\alpha - 1)^2) & (3.20) \\ \Big(\widehat{\rho}u_\alpha - \dfrac{\log\delta}{u_\alpha - 1}, \delta\Big) - DP, \text{otherwise} & (3.21) \end{cases}$$

Proof. Let $Z = L^{(o)}_{(\mathcal{M}(D)||\mathcal{M}(D'))}$ be privacy loss random variable, then for $1 < \alpha \leq \sigma^2 \log\frac{1}{q\sigma} + 1$,

$$\mathbb{E}\left[e^{(\alpha-1)Z}\right] = e^{(\alpha-1)D_\alpha(M(\mathbb{D})||M(\mathbb{D}'))} \leq e^{(\alpha-1)\alpha\widehat{\rho}} \tag{3.22}$$

By Markov' inequality,

$$P(Z \geq \epsilon) = P(e^{(\alpha-1)Z} > e^{(\alpha-1)\epsilon}) \leq \frac{\mathbb{E}\left[e^{(\alpha-1)Z}\right]}{e^{(\alpha-1)\epsilon}} \tag{3.23}$$

$$\leq \exp((\alpha-1)(\widehat{\rho}\alpha - \epsilon)) \tag{3.24}$$

The unconstrained minimum of function $g(\alpha) = (\alpha - 1)(\widehat{\rho}\alpha - \epsilon)$ occurs at $\alpha^* = (\epsilon + \widehat{\rho})/(2\widehat{\rho})$, and the minimum value is $-(\epsilon - \widehat{\rho})^2/(4\widehat{\rho})$. If $\alpha^* \leq u_\alpha$, this unconstrained minimum corresponds to the constrained minimum as well that is subject to $\alpha \leq u_\alpha$. Let $\delta = \exp\big(-(\epsilon - \widehat{\rho})^2/(4\widehat{\rho})\big)$, and then we have $\epsilon = \widehat{\rho} + 2\sqrt{\widehat{\rho}log(1/\delta)}$, which has the same form as in Proposition 1. In this case,

$$\alpha^* = (2\widehat{\rho} + 2\sqrt{\widehat{\rho}log(1/\delta)})/(2\widehat{\rho})$$

, and it requires

$$(2\widehat{\rho} + 2\sqrt{\widehat{\rho}log(1/\delta)})/(2\widehat{\rho}) \leq u_\alpha \tag{3.25}$$

From (3.25) we have

$$\delta \geq 1/\exp(\widehat{\rho}(u_\alpha - 1)^2) \tag{3.26}$$

Otherwise, if

$$\delta < 1/\exp(\widehat{\rho}(u_\alpha - 1)^2) \tag{3.27}$$

which means

$$\alpha^* > u_\alpha$$

Then, because the function $g(\alpha)$ is monotonically decreasing in the interval $(0, \alpha^*]$, the constrained minimum is achieved at the boundary point

$$\alpha^+ = u_\alpha$$

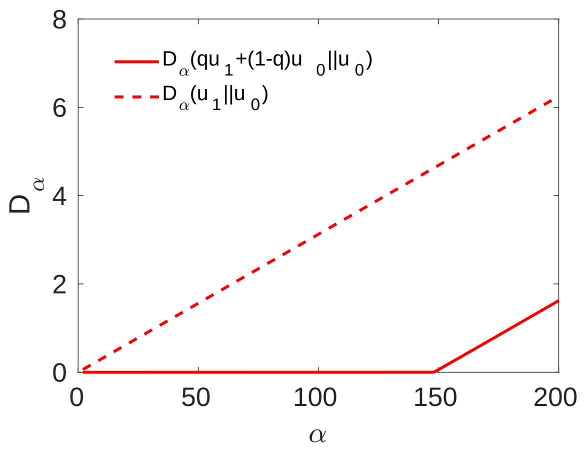


Figure 3.1: α-Rényi divergence under sampling(q=0.01, σ=4)

and accordingly we let $\delta = \exp\big((\alpha^+ - 1)(\widehat{\rho}\alpha^+ - \epsilon)\big)$ and have

$$\epsilon = \widehat{\rho}\alpha^+ - \frac{log\delta}{\alpha^+ - 1}$$

. $\square$

Theorem 7 shows how to convert bounded α-Rényi divergence within a limited range of α to (ϵ, δ)-DP, once the bound functions $P(q,\sigma)$ and $U_\alpha(q,\sigma)$ are determined. Proper choice of $U_\alpha(q,\sigma)$ for the range of α (for example, let $U_\alpha(q,\sigma) < 147$ in Figure 1) with a corresponding $P(q,\sigma)$ can capture the privacy amplification effect of random sampling with replacement, which we will discuss soon. Combined with the composition rules given below, Theorem 7 provides an easy way to estimate privacy loss for DP-SGD under random sampling based batching.

Composition. Now we consider the composition of a sequence of Gaussian mechanisms with random sampling. Suppose k mechanisms, denoted by $\mathcal{M}$=($\mathcal{A}'_1$,..., $\mathcal{A}'_k$) where each $\mathcal{A}'_i$ uses sampling ratio q_i and noise scale σ_i. Because the constraint of α in Eq. (3.19) depends on the sampling ratio and noise scale, we examine their composition in two cases:

1.) Each mechanism uses the same q and σ. For $1 < \alpha \leq u_\alpha$, by the composition property of α-Rényi divergence [57], we have $D_\alpha(M(\mathbb{D})||M(\mathbb{D}')) \leq k\alpha P(q,\sigma)$. The conversion to (ϵ, δ)-DP can be done by letting $\widehat{\rho} = kP(q,\sigma)$ in (3.20) and (3.21) in Theorem 7.

2.) The sampling ratio and noise scale are different for each mechanism. Then, for each $\mathcal{A}_i$, we have $D_\alpha(\mathcal{A}_i(\mathbb{D})||\mathcal{A}_i(\mathbb{D}')) \leq \alpha P(q_i, \sigma_i)$ for $1 < \alpha \leq U_\alpha(q_i, \sigma_i)$. To allow the composition of α-Rényi divergence of mechanisms with different q_i and σ_i, we constrain α to the range $1 < \alpha \leq \min_i\{U_\alpha(q_i, \sigma_i)\}$. It is then clear that $D_\alpha(M(\mathbb{D})||M(\mathbb{D}')) \leq \alpha(\sum_i P(q_i, \sigma_i))$ holds within this α range. Letting $\widehat{\rho} = \sum_i P(q_i, \sigma_i)$ and replacing q and σ by q_j and σ_j where $j = \arg\min_i\{U_\alpha(q_i, \sigma_i)|1 \leq i \leq k\}$ in Eq. (3.20) and (3.21), we can still obtain the corresponding (ϵ,δ)-DP.

When random sampling is used for batching, Algorithm 2 follows the above method to estimate privacy loss in terms of (ϵ, δ)-DP. In particular, the algorithm specifies a fixed $\delta = \delta_0$ and a total privacy budget ϵ_{total}, and at every iteration step t, it updates $\widehat{\rho} = \sum_{i=0}^{t} P(q_i, \sigma_i)$ and computes the corresponding cumulative privacy loss ϵ_t. If $\epsilon_t > \epsilon_{total}$, the training is terminated and the final model satisfies $(\epsilon_{total}, \delta_0)$-DP.

The bound on α-Rényi divergence. To apply Theorem 7 while capturing the privacy amplification effect of random sampling, proper range constraint $U_\alpha(q, \sigma)$ and the bound $P(q, \sigma)$ have to be determined. We noted that, an asymptotic bound on the log moment was given in Lemma 3 in previous work [46] when $q \leq \frac{1}{16\sigma}$. By definition, α-Rényi divergence is equal to the log moment multiplied by a factor of $\frac{1}{\alpha-1}$. Then, it is easy to know that, under the same condition, $D_\alpha(\cdot||\cdot)$ has an asymptotic bound of $\frac{q^2}{1-q}\alpha/\sigma^2 + O(q^3(\alpha-1)^2/\sigma^3)$ for $1 < \alpha \leq \sigma^2 \log \frac{1}{q\sigma} + 1$. This bound exhibits the privacy amplification with having a factor of q^2 on α. Therefore, a possible solution here is to determine an appropriate $P(q, \sigma)$ within $1 < \alpha \leq U_\alpha(q, \sigma) = \sigma^2 \log \frac{1}{q\sigma} + 1$ given $q \leq \frac{1}{16\sigma}$.

In the conference version of our paper [2], we noted that in previous work [46] the proof of Theorem 1 uses an approximation $q^2\lambda^2/\sigma^2$ as the log moment bound. Accordingly, we set $P(q, \sigma) = q^2/\sigma^2$ for Theorem 7. The problem with it is that the bound $q^2\lambda^2/\sigma^2$ does not strictly bound the log moment at order λ due to the approximation, although our experiment in [2] shows that using $P(q, \sigma) = q^2/\sigma^2$ produces valid upper bound and very close result to that produced by MA [46] which computes the exact log moment value numerically.

We examined this problem and experimentally confirmed it. In this paper, instead, we use $q^2\lambda(\lambda+1)/\sigma^2$ as the upper bound of the log moment because our numerical computation shows that it is valid in a wide range of parameter settings, and at the same time upper bound $q^2\alpha/\sigma^2$ of α-Rényi divergence still holds in this case. In our validation, we compare the bound $q^2\alpha/\sigma^2$ of α-Rényi divergence with the result numerically computed with the MA implementation [46] considering the factor of $\frac{1}{\alpha-1}$ difference between log moment and α-Rényi divergence. We test σ and q with a precision of 0.001, by varying σ from 2 to 30 and q from 0.001 to $\frac{1}{16\sigma}$ with a step size of 0.001. To avoid the overflow in numerical computation of MA, we limit $U_\alpha = min(\sigma^2 \log \frac{1}{q\sigma} + 1, 200)$. In our validation, we compute the result on integer values of $\alpha \in (1, U_\alpha)$. It is interesting that our result shows that this upper bound holds in all the parameter settings we checked for q and σ.

In summary, we have shown that CDP is not able to capture the privacy amplification effect of random sampling. We address this issue by bounding α-Rényi divergence over a constrained range of α instead of $(1, \infty)$ and convert to (ϵ, δ)-DP. Comparing with MA, our approach produces a bit higher privacy loss estimation due to the use of an upper bound instead of exact log moment values. However, it provides an alternative but easy way to estimate privacy loss, especially when we use different sampling ratios and noise scales for each iteration of DP-SGD for dynamic privacy budget allocation.

More importantly, we have provided formal analysis to show that the compositions of differential privacy under two batching methods are distinct. As demonstrated by our experimental results, this causes different privacy loss. Therefore, we argue that the privacy accounting method has to be chosen according to which data batching method is used. In our implementation, we focus on random reshuffling, because it is a common practice in the neural network implementation [64, 65]. In fact, several existing deep learning frameworks such as TensorFlow provide convenient random reshuffling APIs for generating batches. It is also numerically observed that random reshuffling outperforms its random sampling with replacement [66].

3.4.3 DP Composition Under Dynamic Schedules

For pre-defined schedules, once the hyperparameters are specified, they follow the decay functions to update the noise scale without accessing the data and the model, and thus do not incur any additional privacy cost. Since the noise scale is updated by epoch, each iteration step within an epoch uses the same noise scale. Suppose the epoch t uses the noise scale σ_t. Each iteration of epoch t is then $\rho = 1/(2\sigma_t^2)$ zCDP by Proposition 2, and the total privacy cost of epoch t can be calculated by Theorem 6 or 7 depending on which batching method is used. Over the course of training, the cumulative privacy loss is updated at each epoch, and once the cumulative privacy loss exceeds the fixed privacy budget ρ_{total}, the training is terminated. To achieve a target training time under a given total privacy budget, we can determine the exact values of hyperparameters for these schedules in advance before the training time.

For the validation-based schedule, the access to the public validation dataset does not incur additional privacy cost. With this schedule, the composition of differential privacy involves adaptive choices of the privacy parameter ρ at every epoch, which is corresponding to the noise scale σ of the Gaussian mechanism. This means that the choice of privacy parameters itself is a function of the realized outcomes of the previous rounds. It has been shown by Rogers et al. [67] that the strong composition theorem for (ϵ, δ)-DP fails to hold in this adaptive privacy parameter setting since the theorem requires the privacy parameters to be pre-defined ahead of time. To address this problem, they define the privacy loss as a random variable as done in Eq. (3.5) for CDP and develop the composition for (ϵ, δ)-DP using privacy filters. Privacy filters provide a way to halt the computation with probability $1-\delta$ before the realized privacy loss exceeds ϵ. Our approach relies on zCDP which defines privacy loss as in Eq. (3.5) by nature and therefore the composition accumulating privacy cost with regard to Rényi divergence holds for the adaptive parameter settings.

3.5 Experimental Results

In this section, we evaluate the proposed privacy accounting methods, and demonstrate the effectiveness of dynamic privacy budget allocation on different learning tasks. Our implementation is based on the TensorFlow implementation [68] of DP-SGD in the paper[46].

3.5.1 Comparing Privacy Accounting Approaches

In Section 3.4.2 we derive different privacy accounting methods for two data batching methods: random reshuffling (RF) and random sampling with replacement (RS). We refer to them as zCDP(RF) and zCDP(RS) respectively. To numerically compare them with other privacy accounting methods including strong composition [54] and the moments accountant (MA) method [46], we unify them into (ϵ, δ)-DP. Following [46], we assume that the batches are generated with RS for both the strong composition and MA. We use the implementation of [46] in TensorFlow to compute MA. For strong composition, we apply the strong composition theorem in [54] to the composition of $(\log(1 + q(e^{\epsilon} - 1)), q\delta)$-DP mechanisms that are the privacy amplified version of (ϵ, δ)-DP mechanisms running with random sampling ratio q. We compute (ϵ, δ)-DP for zCDP(RF) with Proposition 1 and for zCDP(RS) with the methods in Section 3.4.2.

In our experimental setting, we assume a batch size B for random reshuffling. For random sampling we assume a sampling ratio $q = \frac{B}{N}$ given a total of N samples. In the following, when we vary q, it is equivalent to change the batch size for random reshuffling. The number of iterations in one epoch is $\frac{1}{q}$. For simplicity, we use the same noise scale for Gaussian mechanism $\mathcal{N}(0, C^2\sigma^2\mathbf{I})$ for every iteration and set $q = 0.01$ and $\sigma = 6$ by default. Given $\sigma = 6$, the Gaussian mechanism satisfies both ($\epsilon = 0.808$, $\delta = 1e - 5$)-DP and $\rho = 0.0139$-zCDP. We track the cumulative privacy loss by epoch with different privacy accounting methods and convert the results to ϵ in terms of (ϵ, δ)-DP with fixed $\delta = 1e - 5$.

Figure 3.2 shows the growth of privacy loss metric ϵ during the training process. It shows that zCDP(RF) has lower estimation on privacy loss than that of the strong composition during the training. The final spent ϵ at epoch 400 by zCDP(RF) and strong composition are 21.5 and 34.3 respectively. Although random sampling introduces higher uncertainty and thus less privacy loss than random shuffling, zCDP(RF) still achieves lower and thus tighter privacy loss estimation even than the strong composition with random sampling. This demonstrates the benefit of CDP for composition of a large number of computations. The results for zCDP(RS) and MA are very close to each other because that they both exploit the moment bounds of privacy loss to achieve tighter tail bound and take advantage of the privacy amplification of random sampling. The final spent ϵ is 2.37 and 1.67 for zCDP(RS) and MA respectively. The reason for zCDP(RS) to have a slightly higher estimation is that its conversion to (ϵ, δ)-DP explicitly uses the log moment bound instead of the numerical computation of log moments. The benefit of zCDP(RS) is that it is easy to compute with explicit expressions in Theorem 7, and its composition for dynamic privacy budget allocation is also simple and thus more efficient.

Figure 3.2 shows that zCDP(RF) has higher privacy loss compared to MA and zCDP(RS), because more uncertainty is introduced with RS. However, it is worth noting that the common practice in deep learning is to use RF, including the implementation of [46]. Thus, zCDP(RF) is the proper choice for them and also straightforward due to the composition property of ρ-zCDP which simply adds up on ρ values. The results show that MA underestimates the real privacy loss when treating the random reshuffling as random sampling with replacement.

We further examine how zCDP(RF) and zCDP(RS) change with the sampling ratio q and the noise scale σ. Using the default $\sigma = 6$, Figure 3.3a shows the privacy loss ϵ at the end of 200 training epochs with varying q values. For zCDP(RF), the cumulative privacy loss does not change with q. This is because the composition of ρ-zCDP iterations within one epoch still satisfies ρ-zCDP by Theorem 6 and across epochs the linear composition

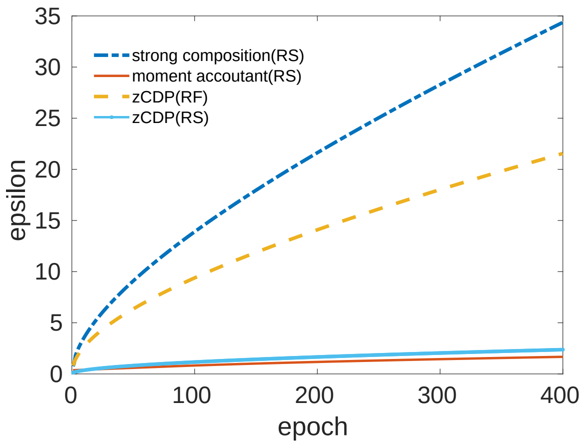
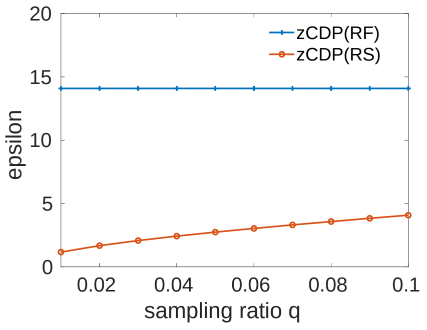
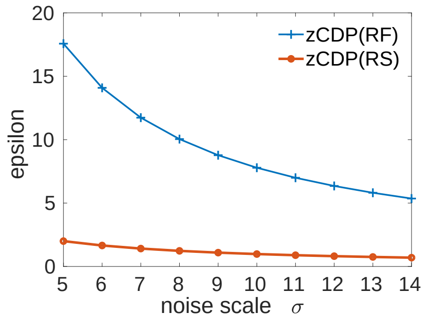


Figure 3.2: Privacy parameter ϵ v.s. epoch



(a) ϵ v.s. q



(b) ϵ v.s. σ

Figure 3.3: privacy loss ϵ v.s. sampling ratio q & noise scale σ

of ρ-zCDP in Theorem 5 is applied, which makes the final privacy loss depend exclusively on the number of training epochs. We have fixed 200 epochs so the final privacy loss does not change. In contrast, the privacy loss given by zCDP(RS) increases with the sampling ratio q, which can be seen in Eq. (3.20) where ϵ increases with $\widehat{\rho}$ which is proportional to q^2. Similarly, Figure 3.3b shows the privacy loss after 200 epochs by varying noise scales with the same q=0.01. We observe that increasing σ from 5 to 14 significantly reduces ϵ for zCDP(RF) but has noticeably less impact on zCDP(RS). It suggests that under random sampling, a small sampling ratio contributes much more on privacy than the noise scale σ. This indicates that we may reduce the noise scale to improve the model accuracy without degrading much privacy. However, for random reshuffling, the privacy loss does not depend on the sampling ratio (i.e., the batch size) but is decided by σ, so it is more critical to achieve a good trade-off between privacy and model accuracy in this case. Our privacy budget allocation techniques optimize this trade-off by dynamically adjusting σ during the training to improve model accuracy while retaining the same privacy guarantee.

3.5.2 Evaluating Dynamic Privacy Budget Allocation

In this section we evaluate the effectiveness of dynamic privacy budget allocation compared to uniform privacy budget allocation adopted by Abadi et al. [46]. Since the TensorFlow implementation uses random reshuffling to generate batches, privacy accounting in Section 3.4.2 should be used to avoid the underestimation of privacy loss. We therefore use ρ as the metric to represent the privacy budget and loss. Because the techniques for adjusting noise scales are independent of the batching method, the benefit of dynamic privacy budget allocation on model accuracy demonstrated under random reshuffling also applies to random sampling.

Datasets and Models

Our experiments use three datasets and different default neural networks for each dataset.
MNIST. This is a dataset of handwritten digits consists of 60,000 training examples and 10,000 testing examples [69] formatted as 28X28 size gray-level images. In our experiment, the neural network model for MNIST follows the settings in previous work [46] for comparison: a 60-dimensional PCA projection layer followed by a simple feed-forward neural network comprising a single hidden layer of 1000 ReLU units. The output layer is softmax of 10 classes corresponding to the 10 digits. The loss function computes cross-entropy loss. A batch size 600 is used. The non-private training of this model can achieve 0.98 accuracy with 100 epochs.
Cancer Dataset. This dataset [70] consists of 699 patient examples. Each example has 11 attributes including an id number, a class label that corresponds to the type of breast cancer (benign or malignant), and the 9 features describing breast fine-needle aspirates. After excluding 16 examples with missing values, we use 560 examples for training and

Table 3.1: Budget Allocation Schedules under Fixed Budget $\rho_{total} = 0.78125$, initial noise scale $\sigma_0 = 10$ for dynamic schedules. The parameters for validation based schedule is k=0.7,m=5, δ=0.01,$period$=10.

	Uniform $\sigma_c = 8$	Time (k=0.05)	Step(k=0.6, $period$=10)	Exp (k=0.01)	Poly(k=3,σ_{end}=2, $period$=100)	Validation
epochs	100	38	31	71	44	64
training accuracy	0.918	0.934	0.928	0.934	0.930	0.930
testing accuracy	0.919	0.931	0.929	0.929	0.932	0.930
non-private SGD	0.978 /0.970	0.959 /0.957	0.955 /0.954	0.971 /0.965	0.963 /0.959	0.97 /0.964
uniform #epochs		0.922 /0.925	0.921 /0.925	0.922 /0.925	0.925 /0.929	0.924 /0.926

123 examples for testing. A neural network classifier with 3 hidden layers, containing 10, 20, and 10 ReLU units, is trained to predict whether a breast tumor is malignant or benign. Each iteration takes the whole training data set as a batch and thus each iteration is one epoch. The non-private training of this model achieves testing and training accuracy 0.96 with 800 epochs.

CIFAR-10. The CIFA-10 dataset consists of 32×32 color images with three channels (RGB) in 10 classes including ships, planes, dogs and cats. Each class has 6000 images. There are 40,000 examples for training, 10,000 for testing and 10,000 for validation. For experiments on CIFAR, we use a pre-trained VGG16 neural network model [71]. Following the previous work [46], we assume the non-private convolutions layers that are trained over a public dataset (ImageNet [72] for VGG16) and only retrain a hidden layer with 1000 units and a softmax layer with differential privacy. We use 200 training epochs and batch size of 200. The corresponding non-private training achieves 0.64 training accuracy and 0.58 testing accuracy.

Results on MNIST

In differentially private model training, we keep the batch size at 600, clip the gradient norm of every layer at 4, and use fixed noise scale $\sigma_{pca} = 16$ for differentially private PCA. Note that, since the PCA part has constant privacy cost $1/(2\sigma_{pca}^2)$ in terms of ρ-zCDP, we exclude it from the total privacy budget ρ_{total} in our experiment, i.e., ρ_{total} is only for the DP-SGD in Algorithm 2. A constant learning rate 0.05 is used by default. We evaluate the model accuracy during training under different privacy budget allocation schedules. For the validation-based schedule, we divide the training dataset into 55,000 examples for training and 5000 examples for validation, and perform validation every epoch.

The results in Table 3.1 demonstrate the benefit of dynamic privacy budget allocations and the effect of earlier training termination on the model accuracy. For comparison, we consider the uniform privacy budget allocation in [46] with a constant noise scale σ_c=8 for every epoch as our baseline. We choose a fixed total privacy budget ρ_{total}=0.78125. This results in 100 training epochs in the baseline case. We test all dynamic schedules with the hyperparameters given in the table and present their testing and training accuracy in numbers rounded to two decimals. The training is terminated when the privacy budget runs out, and the hyerparameters are chosen from a set of candidates to demonstrate varied training times in epochs which are reported in the table. We also ran a non-private version of SGD with using the same training time as these schedules to see the impact of DP-SGD on accuracy. We can see from Table 3.1 that the baseline with constant σ_c=8 achieves 0.918 training accuracy and 0.919 testing accuracy. By comparison, all non-uniform privacy budget allocation schedules improve the testing/training accuracy by $1\%\sim1.6\%$ while running fewer epochs. Because DP-SGD is a randomized procedure and the numbers in the table vary among trials, we repeat all the experiments 10 times and report in Figure 3.5 the mean accuracy along with the min-max bar for every schedule. These results show that dynamic schedules consistently achieve higher accuracy than the baseline. Therefore, given a fixed privacy budget, the dynamic budget allocation can achieve better accuracy than using the

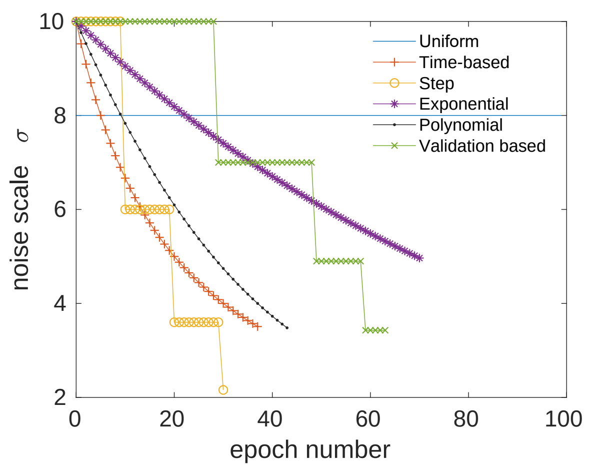


Figure 3.4: The change of noise scale σ during training

uniform budget allocation.

The accuracy improvement shown with dynamic schedules in the above example comes from two sources: less training time and non-uniformity of budget allocation (i.e., decaying of the noise scale). With uniform allocation under a fixed privacy budget, reducing the training time increases the privacy budget allocated to every epoch and thus decreases the noise scale used by the Gaussian mechanism. Therefore, when the model training benefits more from the reduced perturbation rather than longer training time, using less training time can improve the model accuracy. As verification, we apply uniform budget allocation with the noise scale $\sqrt{T/(2\rho_{total})}$ to achieve the same training time T as the corresponding dynamic schedule. The training accuracy and testing accuracy are presented respectively in the final row of Table 3.1. All cases outperform the baseline case with $\sigma_c = 8$ with less than 100 training epochs, indicating the benefit of trading the training time for more privacy budget per epoch. however, it is worth noting that in certain cases, increasing the noise scale to prolong the training time may help improve the accuracy, exemplified by the result of the validation-based schedule on the Cancer dataset. Overall, when compared with the uniform allocation under the same training time, dynamic schedules demonstrate higher accuracy, therefore illustrating the benefit of non-uniformity and dynamic budget allocation.

Table 3.2: decay rate values for different training times

#epochs	30	40	50	60	70	80	90	100
Time	0.076	0.0441	0.0281	0.019	0.0132	0.0093	0.0067	0.0048
Step	0.5459	0.7008	0.7922	0.851	0.891	0.919	0.94	0.956
Exp	0.0442	0.0282	0.0193	0.0138	0.0101	0.0075	0.0056	0.0041
Poly	6.2077	3.5277	2.1948	1.4317	0.9549	0.6382	0.4167	0.1626

Figure 3.4 shows how the noise scale σ changes with the epochs under different schedules in Table 3.1. The curves terminate at the end of the training due to the depleted privacy budget. For the validation-based decay, the duration of the noise scale keeping unchanged decreases over the training time. The noise scale keeps 10 for 29 epochs, 7 for 20 epochs and 4.9 for 10 epochs. It is because that, as the model converges, the increment rate of the validation accuracy declines and it is more often to find that the accuracy increment does not exceed the given threshold.

We additionally manipulate different hyperparamters individually while keeping the rest constant to demonstrate their effects on training/testing accuracy and training time.. By default all accuracy numbers are the average of five trials.

The effect of decay functions. In our previous experiments, we evaluate four types of decay functions for the pre-defined schedules. Here we compare their effects on the model accuracy with constant training time. Given the total privacy budget and initial noise scale, we can use the composition theorem of ρ-zCDP to search for proper values of the parameter k in these functions for the schedule to achieve a target training time. Table 3.2 provides the values of k for different decay functions to achieve training times of 60, 70, 80, 90, and 100 epochs respectively. These values are derived via search in step size $1e-4$, with the same initial noise scale and other parameters as stated in Table 3.1.

Figure 3.6 shows the training and testing accuracy of pre-defined schedules under different training times along with the accuracy achieved by uniform budget allocation [46] using the same privacy budget. We observe that all training instances using pre-defined schedules achieve higher accuracy compared to the uniform budget allocation given a fixed

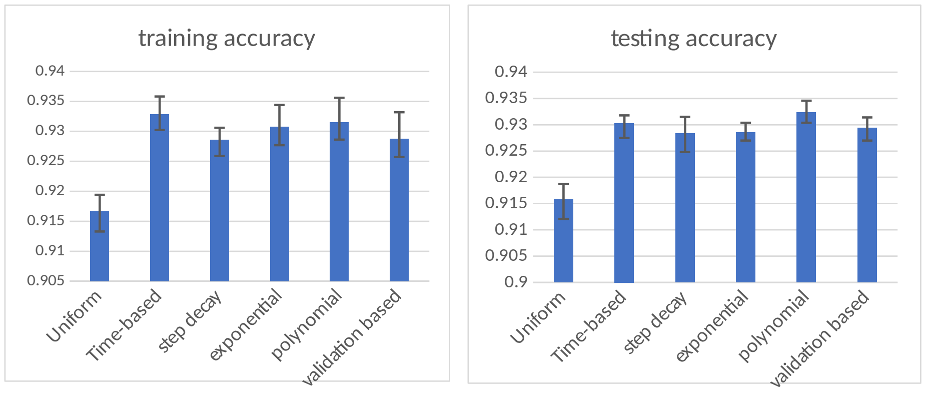


Figure 3.5: The accuracy comparison of different schedules

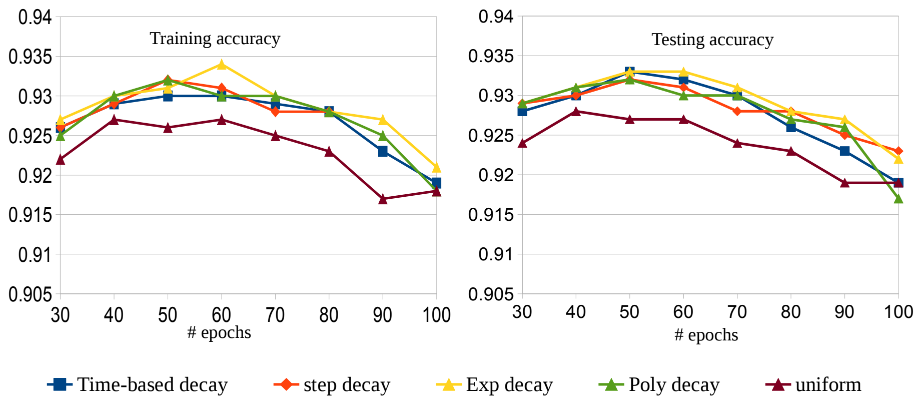


Figure 3.6: The accuracy under fixed training time

training time. However, there is no clear winner among the different decay functions. Their accuracy increases from 30 to 50 or 60 epochs and then decreases as the training time increases from 50 or 60 to 100 epochs. At 100 epochs, all pre-defined schedules have an accuracy closer to that of the uniform budget allocation schedule running 100 epochs. Due to the similar behaviors of the different decay functions, we choose to simply set the decay function to the exponential decay in subsequent experiments unless otherwise stated.

Decay rate. The decay rate k decides how fast the noise scale decays. Keeping other parameters as the same as those reported in Table 3.1, we vary k from 0.005 to 0.5 for exponential decay and from 0.3 to 0.9 for validation-based decay. Figure 3.7 and 3.8 show the accuracy and training time under different values of k. We make three observations. First, in both cases, there exists an optimal decay rate to achieve maximum accuracy. For exponential decay, the best accuracy occurs at k=0.2; for validation-based decay, it occurs at k=0.7. Second, for exponential decay, with the increase of k, the noise scale decreases at a higher rate. The privacy budget is therefore spent faster and the training time strictly decreases. For the validation-based decay, it reduces the noise scale to a k fraction of the original, so the decay rate is actually 1-k and the training time should increase with k. Figure 3.8 shows that the training time overall increases with k but with some random fluctuations. This is because that the validation-based decay adjusts the noise scale according to the validation accuracy which may change during the training in a non-deterministic way.

Third, the results at the ends of the x-axis in both figures indicate two interesting facts. At one end, although the lowest decay rate leads to the longest training time, it also produces the worst accuracy because the noise scale decays more slowly in this case and therefore more epochs will suffer relatively higher noise scales. This degrades the efficiency of the learning process and lowers the accuracy. At the other end of the axis, the highest decay rate causes the training to stop much earlier, resulting in an insufficient training time which also degrades the accuracy.

Learning rate. Next we fix the decay rate k=0.0138 for exponential decay. Given a privacy budget 0.78125, the training lasts 60 epochs. With an initial learning rate 0.1, we linearly decrease the learning rate to end_{lr} over 10 epochs and then fix it at end_{lr} thereafter. We vary end_{lr} from 0.01 to 0.07. Figure 3.9 shows that the accuracy decreases significantly when the learning rate is too small or too large.

Number of hidden units/layers. Next, we vary the number of hidden units in the model from 200 to 1600. The results are shown in Figure 3.10. Although more hidden units increase the sensitivity of the gradient, leading to more noise added at each iteration, we observe that increasing the number of hidden units does not decease the model accuracy under the exponential decay schedule. This is consistent with the observation in [46] using uniform budget allocation. This shows that the effectiveness of dynamic budget allocation schedules scales to neural networks of different sizes. We also vary the number of hidden

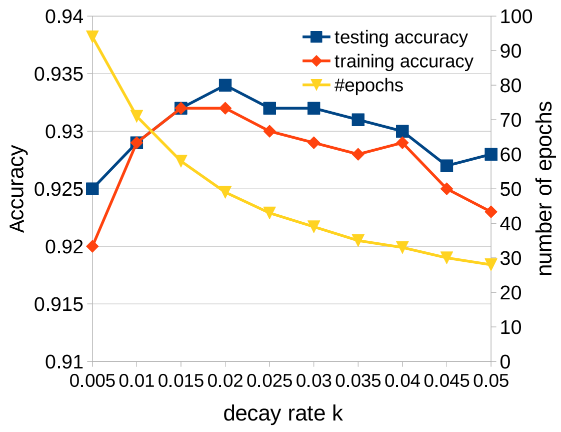


Figure 3.7: exponential decay

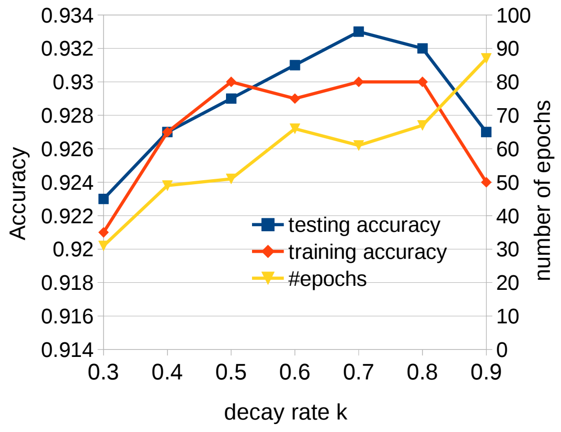


Figure 3.8: validation-based

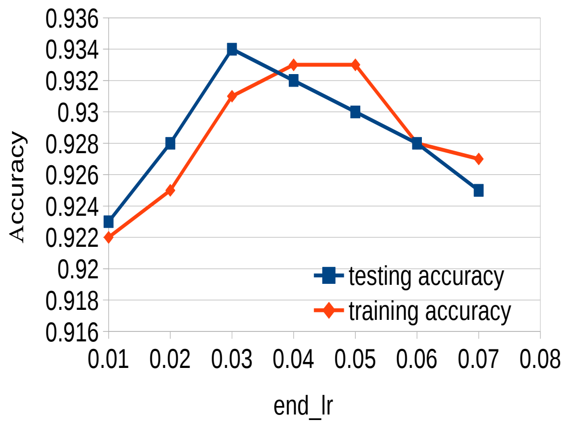


Figure 3.9: learning rate

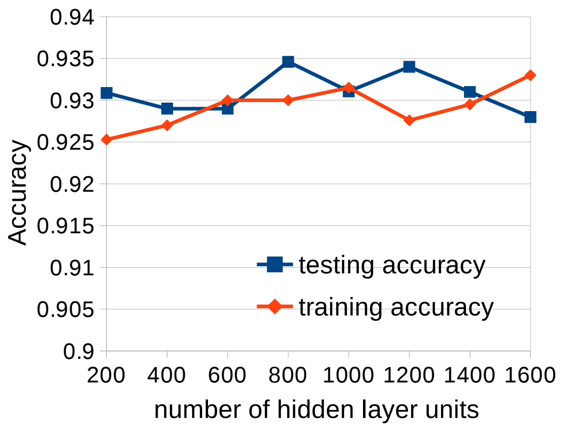


Figure 3.10: hidden units

layers from 1 to 3, each with 1000 hidden units. The accuracy results are given in Table 3.3 and are consistent with [46] under the uniform allocation wherein the authors claim that, for MNIST, one hidden layer combined with PCA works better than networks with more layers.

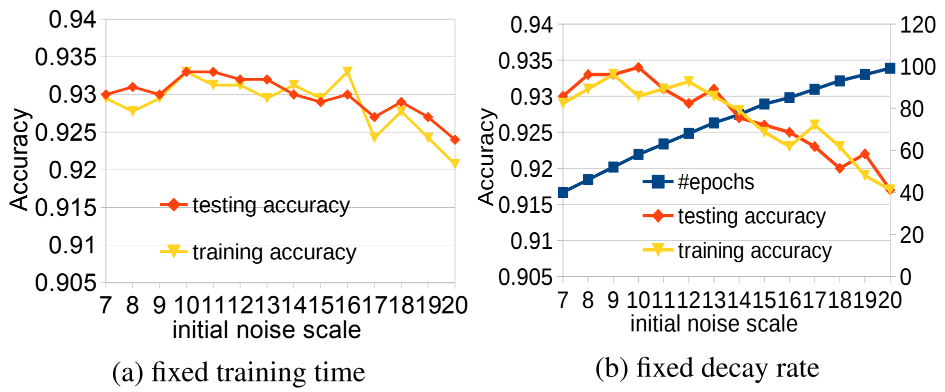
Table 3.3: Accuracy results under different number of layers

Layer numbers	1	2	3
training accuracy	0.9315	0.9228	0.902
testing accuracy	0.931	0.921	0.898

Initial noise scale. In the previous experiments, the initial noise scale σ_0=10 is set as the default. To examine the effect of σ_0, we vary its value from 7 to 20 and measure the model accuracy under the fixed privacy budget 0.78125 in two cases: 1) for each σ_0, we choose the exponential decay rate to achieve a fixed training time of 60 epochs; 2) the exponential decay rate is fixed to 0.015, leading to variation in training time. Figure 3.11a and 3.11b show that, overall, increasing the initial noise scale reduces accuracy. In comparing the two figures, we observe that when the training time is fixed, the choice of σ_0 has less impact on accuracy. This is because, when the training time is fixed, a larger σ_0 results in a higher decay rate of the noise scale which benefits accuracy. However, for fixed decay rate, although higher σ_0 leads to more training epochs, there is no accuracy improvement. This indicates that the model accuracy is more sensitive to the noise scale than the training time.

Accuracy and privacy in training. Figure 3.12 and 3.13 illustrate the change of model accuracy and privacy loss during training time for schedules with the same parameters as in Table 3.1 except the parameters explicitly noted in the figures. Figure 3.13 shows that the uniform privacy budget allocation in [46] incurs linear growth of privacy loss in terms of ρ-zCDP while our dynamic budget allocation schedules have faster growth rate due to the reduction of noise scale with time. All instances stop when the given total privacy budget of 0.78125 is reached. Combined with Figure 3.12, we can see that the exponential decay schedule consistently achieves better accuracy before the training ends compared to the uniform allocation, thanks to its faster noise scale reduction, while the validation-based schedule performs more conservatively and has a relatively longer training time.

An important implication of Figure 3.12 is that the gap between uniform budget allocation and non-private SGD indicates the maximum potential for the accuracy improvement through dynamic budget allocation over the uniform allocation. The proposed dynamic budget allocation schedules provide users a way to improve the accuracy of DP-SGD to approach that of non-private SGD. It is not possible for dynamic budget allocation to completely close this gap because gradient perturbation inevitably hurts model accuracy. Therefore, we argue that the effectiveness of dynamic privacy budget allocation should be evaluated on how much it can reduce the gap between non-private SGD and DP-SGD with uniform allocation. In our experiment, the accuracy difference between non-private SGD and DP-SGD of uniform case is 0.05 at the end of training. The exemplified schedules reduce this difference by 20%$\sim$30%. One of our ongoing research directions is to investigate the ways to effectively find the best hyperparameters to apply these schedules.



(a) fixed training time (b) fixed decay rate

Figure 3.11: Initial noise scale

Results on other datasets

We repeat the experiments on the Cancer Dataset and CIFAR-10 datasets. By applying exponential decay and validation-based decay to each learning task, we compare corresponding model accuracy with the uniform allocation method [46]. In this set of experiments, we first consider a uniform schedule that uses a constant noise scale to achieve a desired training time under the given privacy budget. Then we choose a value around this noise scale as the initial noise scale for decay schedules. A set of candidates for the decay rate is evaluated, and we use each candidate to train a model and compare achieved model accuracy. The parameters for the schedules we used are given in Table 3.4.

Table 3.4: Schedule parameters

Dataset	Uniform	Exp Decay	Validation
Cancer	σ=25, #epochs=500	σ_0=30, k=0.001	σ_0=35,k=0.99, $period$=50,δ =0.01, m=1
CIFAR-10	σ=8, #epochs=200	σ_0=6, k=0.001	σ_0=6,k=0.99, $period$=10,δ =0.05, m=5

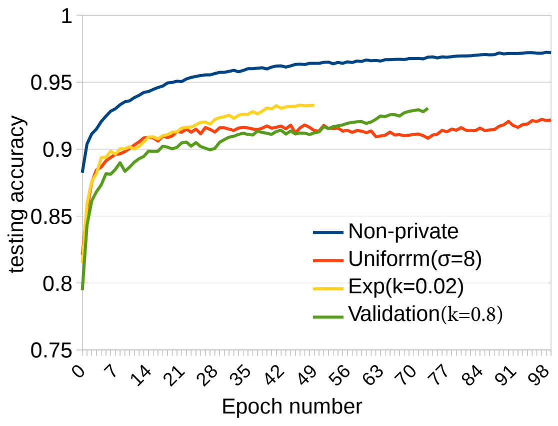


Figure 3.12: Accuracy in training

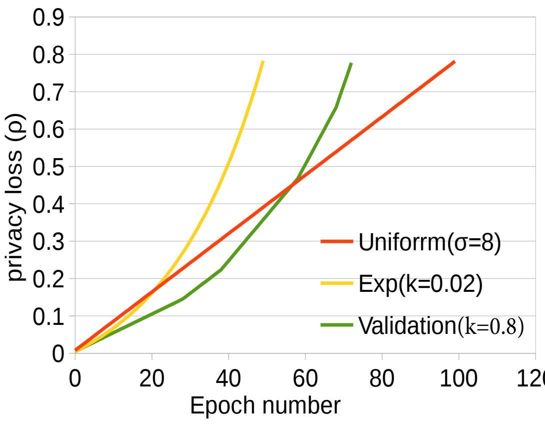


Figure 3.13: Privacy in training

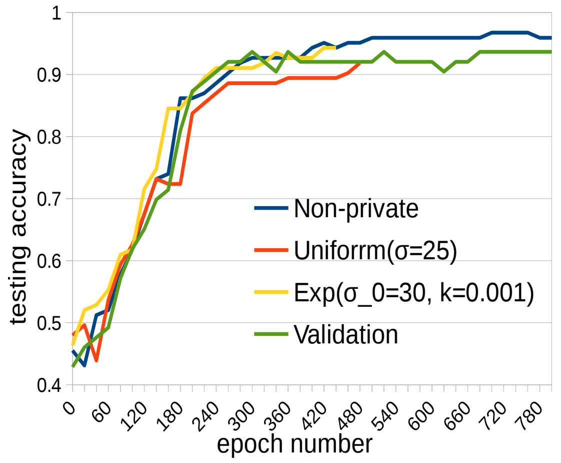


Figure 3.14: Accuracy (Cancer)

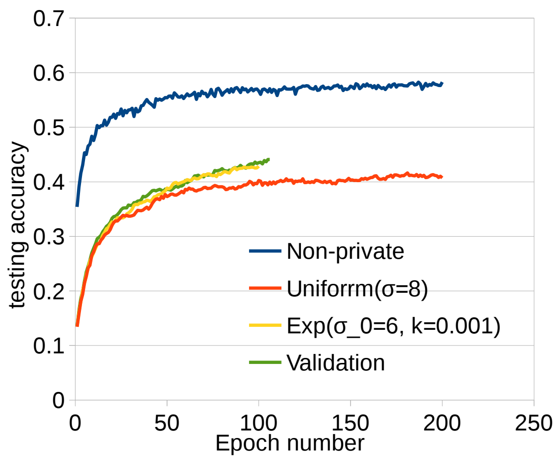


Figure 3.15: Accuracy (Cifar-10)

Results for testing and training accuracy are show in Figures 3.14 and 3.15. For the Cancer dataset, the exponential decay produces the model accuracy closer to the non-private SGD, about 3% higher accuracy than the uniform allocation case, and reduces the gap between non-private SGD and DP-SGD with uniform allocation by 70%. The validation-based schedule produces about 1.8% higher accuracy than the uniform case, with taking advantage of a longer training time as shown in Figure 3.14. For CIFAR-10, the exponential decay achieves 2% higher accuracy than the uniform case, and reduces the gap by about 12%. The validation-based schedule improves model accuracy by 4% over the uniform case, and reduces the gap by about 19%.

3.6 Discussion

We discuss a number of nuances/caveats as take-away remarks for deploying differentially private deep learning in practice for model publishing.

Understanding privacy parameter. Although differential privacy (DP) as a theory has evolved through different forms, today it is still not clear how a realistic privacy benefit can be realized as a function of the privacy parameters in the DP definitions such as the ϵ and δ parameters in traditional DP and the ρ in zCDP. These privacy parameters lack understandable interpretations to the end-users. For ρ-zCDP, results like Proposition 1 would help if ϵ and δ had straightforward privacy-related interpretations. Advancement in interpretability and usability of DP parameters by end-users and domain-scientists can have profound impact on the practical deployment of differential privacy.

Data Dependency. The characteristics of input data, for example, dependency among training instances or dependency in the presence of training instances can render a differentially private mechanism ineffective for protecting the privacy of individuals [73, 74, 52]. The baseline definition of differential privacy is focused on the privacy of a single instance and therefore when multiple instances of the same user are present, a DP mechanism needs to be extended to group-level differential privacy to provide sufficient protection. One direction of our future work is to investigate and explore the ways of extending our DP-SGD techniques to provide a group-level privacy guarantee.

Resilience to Privacy Risks and Attacks. Differentially private deep learning aims to compute model parameters in a differentially private manner to limit the privacy risk associated with output model parameters. There are a number of known attacks in deep learning such as model inversion attacks and membership inference attacks. Model inversion attacks exploit the prediction output along with model access to infer an input instance. Membership inference attacks exploit the black box access to the prediction API to infer the membership of individual training instances. However, there is no formal study on whether or not a differentially private deep learning model is resilient to such attacks and what types of privacy risks known in practice can be protected with high certainty by a differentially private DNN model. In fact, DP only absolves the differentially-private release as a (quantifiably) strong cause of an inference. The work [75] provides an upper bound on the inferential privacy guarantee for differentially private mechanisms. DP, however, does not prevent the inference. This is another grand challenge in differential privacy and data privacy in general.

3.7 Related Work

Privacy threats in machine learning Existing works [76, 4, 5, 77] have shown that machine learning models and their usage may leak information about individuals in the training dataset and input data. Fredrikson, et al. [76] proposed a model inversion attack, which uses the output/prediction produced by a model to infer the unknown features of the input data and apply this attack against decision trees and neural networks in a pharmacogenetics scenario [4]. Reza et al. [5] developed a membership inference attack that aims to determine if an individual record was used as part of the training dataset for the model using only the black-box access to the target model. Song et al. [77] proposed training phase attacks which perform minor modifications to training algorithms to make them output models which encode a significant amount of information about the training dataset while achieving high quality metrics like accuracy and generalizability. In addition, model extraction attacks proposed in [44] aim to duplicate the functionality of the model with black-box access. Such attacks can be leveraged to infer information about the model's training dataset.

Privacy-preserving deep learning To enable deep learning over the data from multiple parties while preserving the privacy of each party's training dataset, Reza et al [45] proposed a distributed deep learning framework that lets the participants train their model independently on their own dataset and only selectively share a subsets of their models' parameters during training. Abadi et al. [46] proposed a differentially private SGD algorithm for deep learning to offer provable privacy guarantees on the output model. DP [27] as a defacto standard for privacy has been applied to various machine learning algorithms, such as logistic regression [47, 48], support vector machines [49] and risk minimization [50, 51], aiming to limit the privacy risk associated with the output model parameters on the training dataset. Our work in this paper is primarily related to [46]. We improve their approach in a number of ways. For example, instead of using traditional (ϵ, δ)-differential privacy, we apply concentrated differential privacy [55, 57] to provide tight cumulative privacy loss estimation over a large number of computations. Furthermore, we characterize the effect of data batching methods on the composition of differential privacy and propose a dynamic privacy budget allocation framework for improving the model accuracy.

CHAPTER 4
DEEPSYNTHESIZER: DIFFERENTIALLY PRIVATE DATA SYNTHESIS WITH DEEP LEARNING

4.1 Introduction

With the advancement of computing and storage devices and networking infrastructure, the digital information about individuals are collected by governments and corporations at a tremendous scale. Because the collection of individual data can create wide opportunities for knowledge-based decision making, there is a large interest and demand for the sharing of data among collaborators or releasing data publicly. However, the collected data related to individuals often encode their privacy sensitive information, which impose serious privacy risks. Therefore, privacy preserving data publishing that studies how to transform original data into the version immunized against privacy attacks but still allow effective data analytics has received considerable attention. Traditional data anonymization techniques [8, 9, 10] transform the data by removing key identifiers or generalizing quasi-identifiers to ensure data privacy. However, it has been shown that they are susceptible to privacy attacks that ”de-anonymize” datasets via linkage to external or public datasets, with some notable examples on Netflix movie rating dataset [11], and the AOL search log [12] and the Washington State health record identification [13].

Data synthesis with differential privacy provides a promising approach for privacy preserving data publication and has been receiving much attention recently [14, 15, 16, 17, 18, 19]. It aims to generate synthetic data that mimic original data in terms of important characteristics and that can be released without compromising the privacy of individuals. Its advantages lie on that: on one hand, differential privacy [27] provides a rigorous privacy framework offering provable privacy guarantee and has recently emerged as a de-facto standard for data privacy; on the other hand, the important motivation for producing synthetic data is that the synthesized data can act as surrogate for the original dataset to allow data users to run arbitrary statistical analysis of their own as if they had the original data. Note that data synthesis is non-interactive setting for differential privacy. In the interactive settings differential private access mechanisms are used to answer the queries to the original database. Due to the composition of differential privacy, only a certain number of queries can be allowed on the same dataset before a pre-specified privacy budget is exhausted. In contrast, data synthesis eliminates the need to design various differentially private mechanisms to sanitize results for different types of queries. It avoids the requirement in interactive settings for trusted data curators to run these mechanisms and consistently monitor queries. Also, it does not have limits on the number of queries to be made on the synthetic data and thus largely facilitate exploratory, open-ended and iterative analyses.

The common framework for all the differentially private data synthesis approaches can be summarized to three steps: 1) carefully select features of interest and importance and extract them from original data sets in a differentially private way; 2) learn a data model or estimate model parameters in a differentially private way; 3) generate samples from the data model. For instance, PrivBayes [16], a differentially private method for releasing tabular dataset, 1) generate a set of conditional distributions of datsets and construct a Bayesian network over the attributes, 2) derive the approximate distribution model of tuples, and 3) sample tuples from the distribution; DPT [17], a system for generating synthetic trajectory data, 1) extract prefix tree for each resolution in a hierarchy of reference systems privately, 2)assume Markov model for mobility trajectories and estimate transition probabilities privately, 3) sample trajectories from the model; Pygmalion [18], a differentially private graph synthesis approach, extracts the graph's degree correlation statistics, and learn a dK-graph model from which the synthetic graphs can be generated. All the existing works conduct sophisticated process to determine and extract features of dataset in order to maximally capture the characteristics of original dataset. More often they involves multiple features and models, and thus require an allocation of privacy budget for each part respectively. However, it is not known how to optimally allocate it and thus most works choose evenly allocation privacy budget among each part. For high dimensional data, the data synthesis becomes more challenging because statistical models with high dimension is computationally impractical to use and excessive noise will be added to satisfy differential privacy.

In this paper we explore the application of differentially private deep learning for data synthesis. The power of deep learning is partially due to its ability to automatically learn appropriate features from raw datasets, which obviates manual feature engineering. Also, deep neural networks has been shown to effectively overcome the curse of dimensional with using large datasets and work well with high dimensional data such as images and text. Therefore, it is promising to use it for data synthesis to effectively avoid sophisticated feature extraction process in previous works and high dimensionality problem. Differentially private data synthesis is simply achieved with training the deep generative models with differentially private SGD algorithm.

An important application of data synthesis is for the publication of location trace data. The collection of people's precise location traces has been made quite easy and common due to the widespread use of mobile devices with GPS and location based services. The release of location trace data has always been of tremendous interest for research and development because it can be used for city/traffic planning, location-driven advertising and human behavior study. However, because of the concerns over the location privacy of individuals, the data holders are wary about publishing these data. Privacy-preserving data synthesis offer a promising solution for this dilemma, by sharing synthesized data with provable privacy guarantee, while ensuring high data utility that is preserving similar aggregate properties as the original location traces. A number of methods [17, 78] have been proposed for location trace synthesis, with differential privacy guarantee. In this paper, we propose DeepSynthesizer that applies deep generative models with differential privacy to synthesize location traces. DeepSynthesizer uses a recurrent neural network model for modeling sequential location trajectory data. Compared with the existing methods [17, 78], DeepSynthesizer provides an end-to-end approach for location data synthesis with differential privacy guarantee, which obviates sophisticated feature selection, data structure, sampling techniques and related privacy budget allocations among different feature extraction steps involved in their design. Therefore, the approach of DeepSynthesizer is easier to implement, deploy and tune in practice. Another important advantage of DeepSynthesizer is a straightforward realization of differential privacy at user level by the mini-batch training of deep learning. The target dataset usually contains a group of records (e.g., trajectories for location data) collected from one user. The conventional methods [17, 78] focus on differential privacy at record level, which may fail to protect the privacy of trajectories at the user level since the presence/absence of one user results in more than one records change in the dataset. The group privacy in differentially privacy tells that any ϵ-differentially private mechanism is $k\epsilon$ differentially private for groups of size k. Because their design is highly coupled with record level differential privacy, it can be difficult to adapt them to protect user level differential privacy, if not impossible. Instead, because of the mini-batch nature of training algorithms for deep neural network models, we show that DeepSynthesizer can have formal privacy guarantees of user-level privacy.

Moreover, we compare DeepSynthesizer with state-of-art differentially private data synthesis approaches in terms of the utility of synthetic location data and privacy cost. The utility of synthetic data is analyzed in two scenarios: extracting statistics for data analysis and performing prediction tasks using machine learning. Our experiment demonstrates that deep learning with differential privacy can significantly facilitate privacy preserving location data synthesis. For the privacy cost, the training of deep neural network models can have unlimited privacy cost because it is an iterative process and each iteration adds to the privacy cost. We investigate the privacy cost under different conditions such as training time, data size and batch size, and the achieved utility. We aim to provide some guidelines for users to choose between deep learning based differentially private data synthesis and

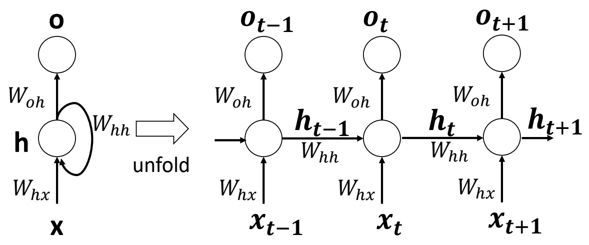


Figure 4.1: An illustration of the RNN model and its unfolding in time steps.

conventional methods.

4.2 Preliminaries

4.2.1 Neural Network Models for Sequential Data

Recurrent neural networks (RNNs) have become the state of the art for sequence modeling and generation, especially in the tasks of machine translation [79], text generation [80], speech recognition [81] and sentiment analysis [82]. It is a deep feedforward neural network whose weights are shared across time. At each time step, the RNN receives an input, updates its hidden state and makes a prediction. The high dimensional hidden state and non-linear activation function enable the RNN to integrate information over many time steps and make accurate predictions.

A standard RNN model can be formalized as follows [83]: Given a sequence of inputs $\{x_1, x_2, \ldots, x_T\}$, the RNN computes a sequence of hidden states $\{h_1, h_2, \ldots, h_T\}$ and a sequence of outpus $\{o_1, o_2, \ldots, o_T\}$ by iterativelly computes the following for $t = 1$ to T:

$$h_t = tanh(W_{hh}h_{t-1} + W_{hx}x_t + b_h) \tag{4.1}$$

$$o_t = W_{oh}h_t + b_o \tag{4.2}$$

where W_{hh} is the hidden-to-hidden weight matrix, W_{hx} is the input-to-hidden weight matrix, W_{oh} is the hidden-to-output weight matrix and the vectors b_h and b_o are the biases. Figure 4.1 illustrates an example of RNN model.

Long short term memory (LSTM) [84] is one of the most popular variations of RNNs

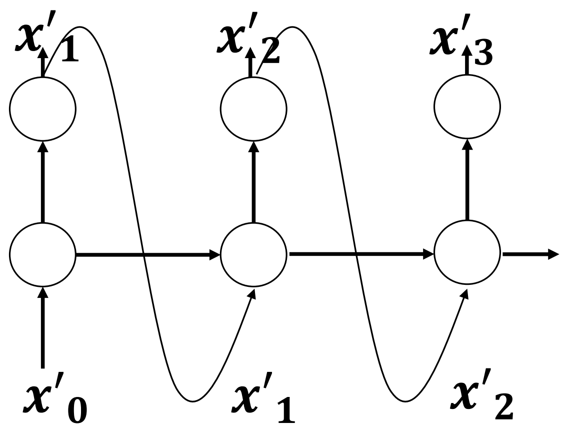


Figure 4.2: An illustration of a trained RNN model for generating sequences.

which is designed to avoid the gradient vanishing/exploding problem [85] of RNNs for learning long-range temporal dependencies. It has been successfully applied in the tasks of speech recognition [86], image captioning [87]. and human mobility modeling [88, 89].

RNN as a generative model. The RNN model can be used as a generative model with predicting the next value in a sequence. Given a training word sequence $\{x_1, x_2, \ldots, x_T\}$ from a fixed vocabulary $\{w_1, \ldots, w_k\}$ of size k, the RNN computes the output vectors $\{o_1, o_2, \ldots, o_T\}$ and accordingly obtains a sequence of predictive distribution $P(x_{t+1}|x_1, \ldots, x_t) = softmax(o_t)$ where the softmax function computes $P(x_{t+1} = w_j) = exp(o_t^{(j)})/\sum_j exp(o_t^{(k)})$ for $j = 1, \ldots, k$. For the language modeling, the learning objective is to maximize the log probability of the training sequence $\sum_t = 0^{T-1} logP(x_{t+1}|x_{\leq t}$. To generate a sentence, we can iterative sample a word from the conditional distribution $P(x_{t+1}|x_{\leq t}$ that can be obtained from the output of the RNN model, and use it as the input for sampling the next word from $P(x_{t+2}|x_{\leq t+1}$.

4.2.2 Differential Privacy

Differential privacy is a rigorous mathematical framework that formally defines the privacy properties of data analysis algorithms. Informally it requires that any changes to a single data point in the training dataset can only cause statistically insignificant changes to the algorithm's output.

Definition 8 (Differential Privacy [27]). *A randomized mechanism $\mathcal{A}$ provides (ϵ, δ)-differential privacy if for any two neighboring database D and D' that differ in only a single entry, $\forall S \subseteq Range(\mathcal{A})$,*

$$\Pr(\mathcal{A}(D) \in S) \leq e^{\epsilon} \Pr(\mathcal{A})(D') \in S) + \delta \tag{4.3}$$

If $\delta = 0$, $\mathcal{A}$ is said to be ϵ-differential privacy. In the rest of this paper, we write (ϵ, δ)-DP for short.

The standard approach to achieving differential privacy is the sensitivity method [29, 27] that adds to the output some noise that is proportional to the sensitivity of the query function. The sensitivity measures the maximum change of the output due to the change of a single database entry.

Definition 9 (Sensitivity [29]). *The sensitivity of a query function $q : \mathcal{D} \to \mathbb{R}^d$ is*

$$\Delta = \max_{D,D'} ||q(D) - q(D')|| \tag{4.4}$$

where D, $D' \in \mathcal{D}$ are any two neighboring datasets that differ at most one element, $|| \cdot ||$ denotes L_1 or L_2 norm.

A standard differentially private mechanism for achieving (ϵ, δ)-differential privacy is Gaussian mechanism that uses L_2 norm sensitivity. It adds zero-mean Gaussian noise with variance $\Delta^2\sigma^2$ in each coordinate of the output $q(D)$, as

$$q(D) + \mathcal{N}(0, \Delta^2\sigma^2\mathbf{I}) \tag{4.5}$$

It satisfies (ϵ,δ)-DP if $\sigma^2 > 2\log(\frac{1.25}{\delta})/\epsilon^2$ and $\epsilon \in (0, 1)$ [56].

4.2.3 Differentially Private Deep Learning

Differentially private deep learning was first formally proposed in [46]. It relies on differentially private stochastic gradient descent (DP-SGD) to preserve the privacy of training data. DP-SGD computes per-example gradients $\nabla_\theta L(\theta, x)$ of the loss L on the input x with regard to the model parameter θ in every iteration and clipping the L_2 norm of gradient vectors. Given a clipping threshold C, this is done by replacing the gradient vector g with $\mathbf{g}/\max(1, \frac{||\mathbf{g}||_2}{C})$ which scales g down to norm C if $||\mathbf{g}||_2 > C$. A Gaussian mechanism with L_2 norm sensitivity of C is then applied to perturb the gradients before applying gradient descent to update the model parameters. The details of DP-SGD algorithm can be found in Chapter 3.

To tightly track the cumulative privacy loss of DP-SGD during training, Abadi et al. [46] proposed Moments Accountant (MA). This method assumes that the data batches for mini-batch SGD are generated by randomly sampling examples from the training dataset with replacement, By taking advantage of the privacy amplification effect of random sampling, it achieves a much tighter estimate on privacy loss than the strong composition theorem.

4.3 Design of DeepSynthesizer

4.3.1 Overview

Consider a set of real location trajectories denoted by $D_{real} = \{\tau_1, \tau_2, \ldots, \tau_i, \ldots\}$ and each $\tau_i =< l_1, l_2, \ldots, l_m >$ is a trajectory that consists of a sequence of locations with the timestamps. Each l_i is a three item tuple (x_i, y_i, t_i, u_i) where x_i and y_i are the geographic coordinates of the location, t_i is the timestamp, and u_i is the user id. Because the location is usually recorded in a continuous geographic domain, a common way to analyze the trajectories is to discretize the geographic domain of D_{real} by imposing a uniform grid structure over the space and choose the centroids as anchor points. A location can then be mapped to the cell it belongs to and a trajectory is encoded as a sequence of cells. Figure 4.3 shows an example.

DeepSynthesizer aims to train a LSTM based neural network model with (encoded) D_{real} as the training dataset and uses it as the generative model for data synthesis. In this paper we want to keep neural network architecture design for DeepSynthesizer as simple as

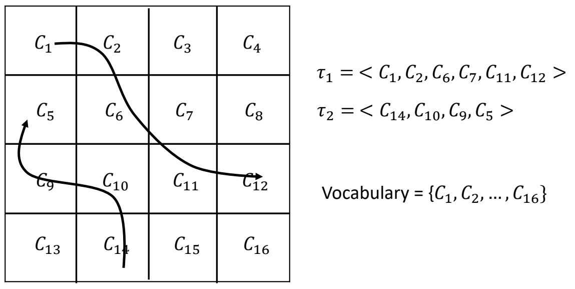

C_1
C_2
C_3
C_4
C_5
C_6
C_7
C_8
C_9
C_{10}
C_{11}
C_{12}
C_{13}
C_{14}
C_{15}
C_{16}
$\tau_1 = < C_1, C_2, C_6, C_7, C_{11}, C_{12} >$
$\tau_2 = < C_{14}, C_{10}, C_9, C_5 >$
Vocabulary = $\{C_1, C_2, \ldots, C_{16}\}$

Figure 4.3: An illustration of 4×4 grid and two encoded trajectories.

possible as a demonstration of the capability of differentially private deep learning for data synthesis. More sophisticated neural network model designs like [89] may further improve the utility of synthesized data, which however is not our focus in this paper.

DeepSynthesizer has options for different privacy guarantee at two different levels, either at record level or at user level. For the differential privacy at the record level, the neighboring trajectory datasets D and D' defined in 4.3 differs in only one trajectory. Per record privacy is preserved in the sense that changing a single record in the dataset only result in statistically insignificant changes in the outcome. However, in the trajectory dataset, record level differential privacy is insufficient, because one individual may contribute many trajectories to the dataset. For example, in Geo-life dataset [35], every user contributes about 5 to 800 trajectories. Since an individual user may repeat a sensitive route several times and thus creates multiple similar trajectories records, they should be protected as a whole in order to protect the sensitive route. To achieve that, differential privacy should be applied at user level to protect all the trajectory data contributed by a single user, rather than a single trajectory. The differential privacy at user-level is formally treated in [52] for differentially private language modeling. The user-level differential privacy applies differential privacy definition in 4.3 over user-adjacent datasets. D and D' are user-adjacent datasets if D' can be formed by adding or removing all of the data records associated with a single user from D.

Suppose a trajectory dataset is collected from N users, each user contributes his trajectories multiple times, and each trajectory is labeled with its source of user id. We group the trajectories by their sources of users, that is, $D = D_{u_1} \bigcup D_{u_2} \bigcup \ldots \bigcup D_{u_N}$. Accordingly $D' = D \bigcup D_{u_{N+1}}$ is regarded as a neighboring trajectory dataset to D. By using each D_{u_i} as a single example for training the neural network, the per-example gradient clipping/perturbation in differentially private SGD can be naturally applied and then the example level differential privacy in DP-SGD achieves the differential privacy for trajectories at user level.

4.3.2 Details of DeepSynthesizer

Input Sequences

The spatial discretization requires a grid to divide the geographic domain of trajectories into cells. Given the cell size of $w \times w$ and a $N \times N$ uniform grid on the space domain, each grid cell is assigned with a unique symbol and each location in the trajectory is mapped to the symbol of the cell that the location belongs to. A trajectory is then encoded as a sequence of cell symbols and all cell symbols compose a vocabulary of $N \times N$. An example is shown in Figure 4.3. A starting symbol $\top$ and a stopping symbol $\perp$ are introduced into the vocabulary, and prepended and appeneded respectively to the encoded trajectories. As a result, $\tau_2 =< \top, C_{14}, C_{10}, C_9, C_5, \perp >$ in Figure 4.3. These two special symbols are used to capture the starting and stopping probabilities for a trajectory.

Multi-Layer LSTM Model

We use LSTM neural network for trajectory modeling. In particular, we consider a multiple-layer LSTM network, which use the hidden state of the previous layer as the input of each corresponding LSTM block in the next layer. Figure 4.4 demonstrate a two layer LSTM neural network. Each LSTM block maintains a hidden vector h_t and a cell state vector c_t, and feed them into itself in the next time step. The cell state is the memory unit for keeping track of long-term dependencies between the elements in the input sequence.

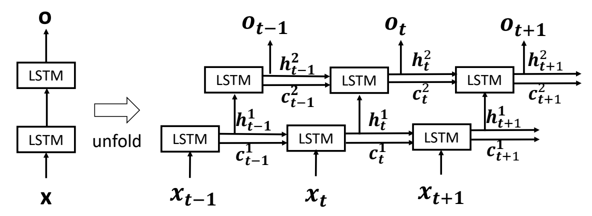


Figure 4.4: An illustration of two layer LSTM neural network.

Differentially Private Neural Network Training

To achieve record level differential privacy, we can directly apply the differentially private SGD (DP-SGD) algorithm in Chapter 3 for training. For simplicity,our implementation use uniform noise scale along the training process for DeepSynthesizer. To achieve lower privacy cost, we use random sampling to construct mini-batch.

However, the previous differentially private SGD algorithm cannot be directly used to achieve user level differential privacy. A direct adaption of differentially private SGD algorithm for user level privacy is presented in Algorithm 3. In this algorithm, a batch of users are randomly sampled, and for each sampled user we obtain the gradient of the model parameters with respect to a batch of the user's data. Per-user gradient here is equivalent to per-example gradient in the original DP-SGD algorithm, and therefore the algorithm has the same per-example gradient clipping and perturbation method. An improvement to this straightforward adaption is differentially private FederatedAveraging (FedAvg) algorithm [52]. The key idea is to perform local training with the user's whole data instead of one batch in one epoch, compute the change of model parameters at each sampled user, and aggregate the model changes from the sampled users. In this paper we focus on Algorithm 3 and evaluate its performance, and leave FedAvg as our future work.

Algorithm 3: User-level Differentially Private SGD Algorithm

```
Input: examples from users $\{u_1, \dots, u_k\}$, learning rate $\eta_t$, gradient norm bound $C$, total
       privacy budget $\rho_{total}$
1  Initialize $w_0$ ;
2  Initialize cumulative privacy loss $c_t^{priv} = 0$;
3  for $t = 1\ :\ T$ do
4      update $c_t^{priv}$ according to data batching method, $t$ and $\sigma_t$;
5      If $c_t^{priv} > \rho_{total}$, break ;
6      $U^t \leftarrow$ sample users with probability $q$ ;
7      for each user $u \in U^t$ do
8          select a batch of size $B$ from $u$'s examples;
9          $\mathbf{g}_u(B) \leftarrow \bigtriangledown_{w_t}\mathcal{L}(w_t, B)$;
10         $\hat{\mathbf{g}}_u \leftarrow \mathbf{g}_u(B)/max\left(1, \frac{||\mathbf{g}_u(B)||_2}{C}\right)$;
11     $\widetilde{\mathbf{g}}_t \leftarrow \frac{1}{|U^t|}\left(\sum_u \hat{\mathbf{g}}_u + \mathcal{N}(0, \sigma_t^2 C^2 \mathbb{I})\right)$;
12     Descent:
13     $w_{t+1} \leftarrow w_t - \eta_t \widetilde{\mathbf{g}}_t$ ;
14 Output $w_T$ ;
```

4.4 Evaluation

4.4.1 Dataset

In this paper we use Brinkhoff dataset for our experiment. It contains 50,000 vehicle trajectories simulated by Brinkhoff network generator for moving object [90]. The dataset was generated on the map of Oldenburg, Germany with simulating the movement of 50,000 vehicles, and the locations were sampled at equal time intervals.

Another dataset we consider is GeoLife GPS Trajectories dataset which contains 17621 traces from 182 users, moving mainly in the north-west of Beijing, China, in a period of over five years (from April 2007 to August 2012). We preprocessed data to limit the spatial domain to be within the 50th ring road of Beijing city.

4.4.2 Utility Metrics

To measure the utility of synthesized data, we use three metrics, including trip distribution, diameter distribution and frequent patterns, introduced in the previous works [17, 78].

- Diameter Distribution: The diameter for a trajectory is defined as the maximum distance between any pair of locations. For a trajectory dataset D, we quantize the diameter into 25 equal width buckets $[0, x), [x, 2x), \ldots, [24x, 25x)$, compute the diameter of each trajectory and obtain its empirical distribution over the buckets. Here x is set to 500m in our experiment. We compute the diameter distribution for both original trajectory dataset D_{real} and synthetic dataset D_{syn}, denoted as $\mathbb{Q}(D_{real})$ and $\mathbb{Q}(D_{syn})$ respectively. We measure the error in diameter distribution by Jensen Shannon divergence $JSD(\mathbb{Q}(D_{real}), \mathbb{Q}(D_{syn}))$.

- Trip Distribution: We use $D_{c_s \rightsquigarrow c_e}$ to denote the set of trajectories in D which starts from the region/cell c_s and ends at the cell c_e. The trip distribution $Pr(c_s \rightsquigarrow c_e)$ for any two cells c_s and c_e of the grid measures the probability of a trajectory starting at c_s and ending at c_e respectively, which can be computed as $Pr(c_s \rightsquigarrow c_e) = \frac{|D_{c_s \rightsquigarrow c_e}|}{|D|}$. We compute the trip distribution for D_{real} and D_{syn}, denoted as $\mathbb{T}(D_{real})$ and $\mathbb{T}(D_{syn})$ respectively, and measure the error in trip distribution by Jensen Shannon divergence $JSD(\mathbb{T}(D_{real}), \mathbb{T}(D_{syn}))$.

- Frequent Patterns: Frequent travel patterns are always of interest in location data mining tasks.We consider trajectories that are encoded to the sequence of cells given a grid discretization. pattern is then an ordered sequence of cells. We find the top k patterns from D_{real} and D_{syn} respectively, denoted by $\mathbb{F}^k(D_{real})$ and $\mathbb{F}^k(D_{syn})$. The error in frequent patterns is measured by the F1-measure, $F_1(\mathbb{F}^k(D_{real}), \mathbb{F}^k(D_{syn}))$ i.e., harmonic mean of precision and recall, between two top-k frequent pattern sets.

4.4.3 Comparison with Existing Generators

In this section we compare DeepSynthesizer with recent proposed methods for location trace synthesis, DPT [17] , SGLT [91], and AdaTrace [78], at the record level differential privacy. This is because that all the existing approaches are designed on record level dif-

Table 4.1: Comparison with existing conventional methods on Brinkhoff Data.

		ngram	DPT	SGLT	AdaTrace	DS(NoDP)	DS(DP)
Trip Error	ϵ=0.5	0.156	0.170	0.298	0.052	0.153	0.182
	ϵ=1.0	0.150	0.120	0.298	0.045	0.153	0.167
	ϵ=2.0	0.139	0.106	0.298	0.043	0.153	0.158
Diameter Error	ϵ=0.5	0.151	0.143	0.126	0.022	0.07	0.12
	ϵ=1.0	0.103	0.084	0.126	0.023	0.07	0.095
	ϵ=2.0	0.092	0.061	0.126	0.022	0.07	0.081
FP F1	ϵ=0.5	0.41	0.56	0.47	0.61	0.61	0.45
	ϵ=1.0	0.39	0.64	0.47	0.62	0.61	0.56
	ϵ=2.0	0.42	0.71	0.47	0.62	0.61	0.60

ferential privacy and cannot be adapted to user-level differential privacy. In particular, we use Brinkhoff dataset for the comparison, because the dataset does not involve any user level differential privacy and each trajectory is generated by a single vehicle. To ensure comparable experiment results with AdaTrace, we follow the similar settings (e.g., similar grid size). The results are provided in Table 4.1, with reusing the metrics reported in [78] . DeepSynthesizer is named as DS in the table. In the table, DS(NoDP) denotes DeepSynthesizer without differential privacy in which case we add no noise to SGD training, so the metrics demonstrate the performance of our LSTM generative model. The model is trained with 5 epochs, and batch size 100. Also, we evaluate DeepSynthesizer with differential privacy guarantee (DS(DP)). The small batch size with respect to large number of trajectories 50000 is equivalent to a very low sampling ratio 0.002. Due to the privacy amplification of random sampling based data batching, the privacy cost is small even with 20000 training steps. As we can see, DeepSynthesizer can obtain comparable data utility with the same privacy cost compared with existing conventional methods. It is expected to

4.4.4 The impact of DP-SGD at user level

In this section we consider Algorithm 3. It is equivalent to the DP-SGD algorithm discussed in Chapter 3, in the sense of that per-example gradient is replaced by per-user gradient. In the Geo-life dataset, all trajectories are grouped by user ids, and thus we consider the user-

Table 4.2: Evaluation on Geo-life dataset.

	DS(NoDP)	DS(DP)
Trip Error	0.45	0.51
Diameter Error	0.10	0.21
FP F1	0.40	0.33

level differential privacy for the training over this dataset. For simplicity, we removed the trajectories that has length more than 1000, resulting in 12167 trajectories of 182 users. Our interest is to examine how much privacy cost when achieving comparable data utility compared with non-private training. The result is show in Table 4.2.

However, in this case, the final privacy cost is much higher than in the Brinkhoff data case. The total number of users is 182, and the smallest sampling ratio we can have is 1/182. The data utility metrics in Table 4.2 has privacy cost $\epsilon = 5.57$, with $\delta = 1e - 5$.

4.5 Related Work

Privacy-preserving synthetic data generation Data synthesizing protects data privacy by sampling data from a pre-trained statistical model and releasing the sampled data in place of the original data. Many works have been proposed for generating privacy-preserving synthetic data. The recent work [91] proposes a trajectory synthesizing approach. It aims to capture both geographic and semantic features of real location traces, and based on that to construct a Markov chain model. The synthetic trajectories sampled from the model are passed through a privacy test to ensure plausible deniability. But it does not provide formal privacy guarantees.

Differentially private data synthesizing Mohammed et al. [14] proposed DiffGen that performs data generalization by partitioning attribute domain in a differentially private way to effectively anonymize raw data. It only generalizes predictor attributes for maximizing the class homogeneity within each partition. It has problems to deal with high-dimensional and large-domain data. Li. et al. [15] developed DPCopula for synthesizing multivariate data by using Copula functions to take into account the dependency structure.

PrivBayes [16] is a differentially private method for releasing high-dimensional tabular data. It uses a Bayesian network and perturbed marginals for differential privacy to derive an approximation of the data distribution in the dataset. The tuples are sampled from the approximate distribution to construct a synthetic dataset. DPT [17] is a system to synthesize mobility data while ensuring ϵ-differential privacy. It uses hierarchical reference systems to capture individual movement speeds, computes prefix tree counts privately and based on that estimates the transition probabilities of Markov model. The synthetic trajectories are sampled from the model. Sala et al. proposed Pygmalion [18] that extracts degree correlation statistics from the original graph data, adds noise to resulting dataset and generates synthetic graph. Jorgensen et al. proposed to adapt Attributed Graph Model(AGM) to add differential pravacy while preserving the utility of the synthesized graphs. Xiao et al. [19] proposed to uses a statistical hierarchical random graph(HRG) model and guarantee differential privacy by sampling possible HRG structures via Markov chain Monte Carlo.

Data synthesizing using deep learning There are a few works that consider generating artificial datasets via deep learning approach. Kulkarni et al. [88] proposes to use recurrent neural networks to learn patterns of mobility traffic data and generate synthetic dataset. Wu et al. [92] also designs RNN based models for trajectory modeling. You et al. [93] propose a deep generative model for graphs. However, they focus on the modeling without providing formal privacy guarantee. Triastcyn et al. [94] proposes using a generative adversarial network to synthesize data and achieve differential privacy with differentially private SGD training. But it focuses on the image data.

CHAPTER 5
CONCLUSION

In this dissertation, we have presented PIVE, a two-phase dynamic differential location privacy framework for providing stronger notion of location privacy in terms of background knowledge based inference attacks. It makes three novel contributions. First, we formally study the relationship between geo-indistinguishability and expected inference error, and demonstrate inherent problems of using geo-indistinguishability alone as the ultimate goal of location privacy protection through formal analysis and experimental illustration. Second, we propose a dynamic differential location privacy protection framework, where we first determine a set of protection locations by guaranteeing the expected inference error bound defined by a mobile user with respect to her service request by taking into account the adversary's prior distribution of the user's locations. Then, we generate the pseudo-locations in a differentially private way. Third, this two-phase framework constructs location obfuscation dynamically by capturing the relationship between two privacy notions based on adversary's current prior information and user-specific privacy requirements for different spatial-temporal contexts. Our experimental evaluation shows that the proposed PIVE approach effectively guarantees the two privacy notions simultaneously and outperforms the existing mechanisms that either offer geo-indistinguishability or quantify location privacy by expected inference errors, in terms of adaptive privacy protection and computation efficiency.

We have presented our approach to differentially private deep learning for model publishing with three original contributions. First, since the training of neural networks involves a large number of iterations, we apply CDP for privacy accounting to achieve tight estimation on privacy loss. Second, we distinguish two different data batching methods and propose privacy accounting methods for each to enable accurate privacy loss estimation. Third, we have implemented several dynamic privacy budget allocation techniques for improving model accuracy over existing uniform budget allocation schemes. Our experiments on multiple datasets demonstrate the effectiveness of dynamic privacy budget allocation.

Finally we have presented DeepSynthesizer that provides an end-to-end privacy preserving location data synthesis approach through differentially private deep learning. DeepSynthesizer uses the recurrent neural network as generative model for data synthesis and applyies differentially private SGD algorithm to train it to protect the privacy of the input dataset. We have compared DeepSynthesizer with existing conventional methods in terms of the utility of synthesized data. Our experiment shows that DeepSynthesizer can achieve comparable performance without sophiscated feature engineering, tuning and careful privacy budget allocation. In addition, DeepSyntheizer considers user level differential privacy that aims to protect a single user's contribution to the input dataset as a whole, which cannot be easily achieved by conventional methods.

REFERENCES

[1] L. Yu, L. Liu, and C. Pu, “Dynamic differential location privacy with personalized error bounds,” in *24th Annual Network and Distributed System Security Symposium, NDSS*, 2017.

[2] L. Yu, L. Liu, C. Pu, M. E. Gursoy, and S. Truex, “Differentially private model publishing for deep learning,” in *IEEE Symposium on Security and Privacy (SP)*, 2019.

[3] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, “Understanding deep learning requires rethinking generalization,” *CoRR*, vol. abs/1611.03530, 2016.

[4] M. Fredrikson, S. Jha, and T. Ristenpart, “Model Inversion Attacks That Exploit Confidence Information and Basic Countermeasures,” in *Proc. of ACM CCS*, New York, NY, USA: ACM, 2015, pp. 1322–1333, ISBN: 978-1-4503-3832-5.

[5] R. Shokri, M. Stronati, and V. Shmatikov, “Membership Inference Attacks against Machine Learning Models,” in *IEEE Symposium on Security and Privacy (S&P)*, 2017.

[6] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks?” *CoRR*, vol. abs/1411.1792, 2014.

[7] *Model zoo*, http://caffe.berkeleyvision.org/model_zoo.html, accessed 2017.

[8] L. Sweeney, “K-anonymity: A model for protecting privacy,” *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, vol. 10, no. 5, pp. 557–570, Oct. 2002.

[9] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, “L-diversity: Privacy beyond k-anonymity,” *ACM Trans. Knowl. Discov. Data*, vol. 1, no. 1, Mar. 2007.

[10] N. Li, T. Li, and S. Venkatasubramanian, “T-closeness: Privacy beyond k-anonymity and l-diversity,” in *2007 IEEE 23rd International Conference on Data Engineering*, 2007, pp. 106–115.

[11] A. Narayanan and V. Shmatikov, “Robust de-anonymization of large sparse datasets,” in *Proceedings of the 2008 IEEE Symposium on Security and Privacy*, ser. SP ’08, Washington, DC, USA: IEEE Computer Society, 2008, pp. 111–125, ISBN: 978-0-7695-3168-7.

[12] M. Barbaro and T. Z.Jr, "A face is exposed for aol searcher no. 4417749," *New York Times*, 2006.

[13] L. Sweeney, "Matching known patients to health records in washington state data," *CoRR*, vol. abs/1307.1370, 2013. arXiv: `1307.1370`.

[14] N. Mohammed, R. Chen, B. C. Fung, and P. S. Yu, "Differentially private data release for data mining," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '11, San Diego, California, USA: ACM, 2011, pp. 493–501, ISBN: 978-1-4503-0813-7.

[15] H. Li, L. Xiong, and X. Jiang, "Differentially private synthesization of multi-dimensional data using copula functions," in *In EDBT*, 2014.

[16] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao, "Privbayes: Private data release via bayesian networks," in *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '14, Snowbird, Utah, USA: ACM, 2014, pp. 1423–1434, ISBN: 978-1-4503-2376-5.

[17] X. He, G. Cormode, A. Machanavajjhala, C. M. Procopiuc, and D. Srivastava, "Dpt: Differentially private trajectory synthesis using hierarchical reference systems," *Proc. VLDB Endow.*, vol. 8, no. 11, pp. 1154–1165, Jul. 2015.

[18] A. Sala, X. Zhao, C. Wilson, H. Zheng, and B. Y. Zhao, "Sharing graphs using differentially private graph models," in *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, ser. IMC '11, Berlin, Germany: ACM, 2011, pp. 81–98, ISBN: 978-1-4503-1013-0.

[19] Q. Xiao, R. Chen, and K.-L. Tan, "Differentially private network data release via structural inference," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '14, New York, New York, USA: ACM, 2014, pp. 911–920, ISBN: 978-1-4503-2956-9.

[20] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Geo-indistinguishability: Differential privacy for location-based systems," in *Proceedings of ACM CCS*, Berlin, Germany: ACM, 2013, pp. 901–914, ISBN: 978-1-4503-2477-9.

[21] N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Optimal geo-indistinguishable mechanisms for location privacy," in *Proceedings of ACM CCS*, Scottsdale, Arizona, USA: ACM, 2014, pp. 251–262, ISBN: 978-1-4503-2957-6.

[22] R. Shokri, G. Theodorakopoulos, C. Troncoso, J.-P. Hubaux, and J.-Y. Le Boudec, "Protecting location privacy: Optimal strategy against localization attacks," in *Pro-*

ceedings of ACM CCS, Raleigh, North Carolina, USA: ACM, 2012, pp. 617–627, ISBN: 978-1-4503-1651-4.

[23] K. Fawaz and K. G. Shin, "Location privacy protection for smartphone users," in *Proceedings of ACM CCS*, Scottsdale, Arizona, USA: ACM, 2014, pp. 239–250, ISBN: 978-1-4503-2957-6.

[24] K. Fawaz, H. Feng, and K. G. Shin, "Anatomization and protection of mobile apps' location privacy threats," in *Proceedings of the 24th USENIX Conference on Security Symposium*, ser. SEC'15, Washington, D.C.: USENIX Association, 2015, pp. 753–768, ISBN: 978-1-931971-232.

[25] R. Shokri, "Privacy games: Optimal user-centric data obfuscation," *PoPETs*, vol. 2015, no. 2, pp. 299–315, 2015.

[26] R. Shokri, G. Theodorakopoulos, J.-Y. Le Boudec, and J.-P. Hubaux, "Quantifying location privacy," in *Security and Privacy (SP), 2011 IEEE Symposium on*, 2011, pp. 247–262.

[27] C. Dwork, "Differential privacy," in *Automata, Languages and Programming*, ser. Lecture Notes in Computer Science, vol. 4052, 2006, pp. 1–12, ISBN: 978-3-540-35907-4.

[28] G. Theodorakopoulos, R. Shokri, C. Troncoso, J.-P. Hubaux, and J.-Y. Le Boudec, "Prolonging the hide-and-seek game: Optimal trajectory privacy for location-based services," in *Proceedings of the 13th Workshop on Privacy in the Electronic Society*, ser. WPES '14, Scottsdale, Arizona, USA: ACM, 2014, pp. 73–82, ISBN: 978-1-4503-3148-7.

[29] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Proceedings of the Third Conference on Theory of Cryptography*, ser. TCC'06, New York, NY: Springer-Verlag, 2006, pp. 265–284, ISBN: 3-540-32731-2, 978-3-540-32731-8.

[30] F. McSherry and K. Talwar, "Mechanism design via differential privacy," in *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, ser. FOCS '07, 2007, pp. 94–103, ISBN: 0-7695-3010-9.

[31] T. Xu and Y. Cai, "Feeling-based location privacy protection for location-based services," in *Proceedings of ACM CCS*, Chicago, Illinois, USA: ACM, 2009, pp. 348–357, ISBN: 978-1-60558-894-0.

[32] K. Chatzikokolakis, C. Palamidessi, and M. Stronati, "Constructing elastic distinguishability metrics for location privacy," *PoPETs*, vol. 2015, no. 2, pp. 156–170, 2015.

[33] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, "Mining interesting locations and travel sequences from gps trajectories," in *Proceedings of the 18th International Conference on World Wide Web*, ser. WWW '09, Madrid, Spain: ACM, 2009, pp. 791–800, ISBN: 978-1-60558-487-4.

[34] Y. Zheng, Q. Li, Y. Chen, X. Xie, and W.-Y. Ma, "Understanding mobility based on gps data," in *Proceedings of the 10th International Conference on Ubiquitous Computing*, ser. UbiComp '08, Seoul, Korea: ACM, 2008, pp. 312–321, ISBN: 978-1-60558-136-1.

[35] Y. Zheng, X. Xie, and W.-Y. Ma, "Geolife: A collaborative social networking service among user, location and trajectory," *IEEE Data(base) Engineering Bulletin*, 2010.

[36] J. K. Lawder and P. J. H. King, "Using space-filling curves for multi-dimensional indexing," in *Proceedings of the 17th British National Conferenc on Databases: Advances in Databases*, ser. BNCOD 17, London, UK, UK: Springer-Verlag, 2000, pp. 20–35, ISBN: 3-540-67743-7.

[37] B. Moon, H. v. Jagadish, C. Faloutsos, and J. H. Saltz, "Analysis of the clustering properties of the hilbert space-filling curve," *IEEE Trans. on Knowl. and Data Eng.*, vol. 13, no. 1, pp. 124–141, Jan. 2001.

[38] A. Khoshgozaran and C. Shahabi, "Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy," in *Proceedings of the 10th International Conference on Advances in Spatial and Temporal Databases*, ser. SSTD'07, Boston, MA, USA: Springer-Verlag, 2007, pp. 239–257, ISBN: 978-3-540-73539-7.

[39] H. Ngo and J. Kim, "Location privacy via differential private perturbation of cloaking area," in *2015 IEEE 28th Computer Security Foundations Symposium*, 2015, pp. 63–74.

[40] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *Proceedings of MobiSys*, San Francisco, California: ACM, 2003, pp. 31–42.

[41] B. Gedik and L. Liu, "Location privacy in mobile systems: A personalized anonymization model," in *Distributed Computing Systems, 2005. ICDCS 2005. Proceedings. 25th IEEE International Conference on*, 2005, pp. 620–629.

[42] J. I. Hong and J. A. Landay, "An architecture for privacy-sensitive ubiquitous computing," in *Proceedings of MobiSys*, Boston, MA, USA: ACM, 2004, pp. 177–189, ISBN: 1-58113-793-1.

[43] *Location guard*, https://addons.mozilla.org/en-US/firefox/addon/location-guard/.

[44] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, "Stealing Machine Learning Models via Prediction APIs," in *USENIX Security Symposium*, Austin, TX: USENIX Association, 2016, pp. 601–618, ISBN: 978-1-931971-32-4.

[45] R. Shokri and V. Shmatikov, "Privacy-Preserving Deep Learning," in *Proc. of ACM CCS*, New York, NY, USA: ACM, 2015, pp. 1310–1321, ISBN: 978-1-4503-3832-5.

[46] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep Learning with Differential Privacy," in *Proc. of ACM CCS*, New York, NY, USA: ACM, 2016, pp. 308–318, ISBN: 978-1-4503-4139-4.

[47] K. Chaudhuri and C. Monteleoni, "Privacy-preserving logistic regression," in *Proc of NIPS*, Vancouver, British Columbia, Canada: Curran Associates Inc., 2008, pp. 289–296, ISBN: 978-1-6056-0-949-2.

[48] J. Zhang, Z. Zhang, X. Xiao, Y. Yang, and M. Winslett, "Functional mechanism: Regression analysis under differential privacy," *Proc. VLDB Endow.*, vol. 5, no. 11, pp. 1364–1375, Jul. 2012.

[49] B. I. P. Rubinstein, P. L. Bartlett, L. Huang, and N. Taft, "Learning in a large function space: Privacy-preserving mechanisms for svm learning," *CoRR*, vol. abs/0911.5708, 2009.

[50] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, "Differentially private empirical risk minimization," *J. Mach. Learn. Res.*, vol. 12, pp. 1069–1109, Jul. 2011.

[51] R. Bassily, A. D. Smith, and A. Thakurta, "Private empirical risk minimization, revisited," *CoRR*, vol. abs/1405.7085, 2014. arXiv: 1405.7085.

[52] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, "Learning differentially private recurrent language models," in *ICLR*, vol. abs/1710.06963, 2018. arXiv: 1710.06963.

[53] S. Song, K. Chaudhuri, and A. D. Sarwate, "Stochastic gradient descent with differentially private updates," in *2013 IEEE Global Conference on Signal and Information Processing*, 2013, pp. 245–248.

[54] C. Dwork, G. N. Rothblum, and S. Vadhan, "Boosting and differential privacy," in *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, 2010, pp. 51–60.

[55] C. Dwork and G. N. Rothblum, "Concentrated Differential Privacy," *CoRR*, Mar. 2016.

[56] C. Dwork and A. Roth, “The algorithmic foundations of differential privacy,” *Found. Trends Theor. Comput. Sci.*, vol. 9, no. 3&#8211;4, pp. 211–407, Aug. 2014.

[57] M. Bun and T. Steinke, “Concentrated Differential Privacy: Simplifications, Extensions, and Lower Bounds,” *CoRR*, May 2016.

[58] N. Li, W. Qardaji, and D. Su, “On sampling, anonymization, and differential privacy or, k-anonymization meets differential privacy,” in *Proc. of ASIACCS*, Seoul, Korea: ACM, 2012, pp. 32–33, ISBN: 978-1-4503-1648-4.

[59] *Decaying the learning rate*,
https://www.tensorflow.org/api_guides/python/train.

[60] F. Chollet, “Xception: Deep learning with depthwise separable convolutions,” *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1800–1807, 2017.

[61] C. Darken and J. Moody, “Note on learning rate schedules for stochastic optimization,” in *Proc of Advances in Neural Information Processing Systems (NIPS)*, Denver, Colorado, USA, 1990, pp. 832–838, ISBN: 1-55860-184-8.

[62] F. D. McSherry, “Privacy integrated queries: An extensible platform for privacy-preserving data analysis,” in *Proc. of ACM SIGMOD*, Providence, Rhode Island, USA: ACM, 2009, pp. 19–30, ISBN: 978-1-60558-551-2.

[63] T. van Erven and P. Harremos, “Rényi divergence and kullback-leibler divergence,” *IEEE Transactions on Information Theory*, vol. 60, no. 7, pp. 3797–3820, 2014.

[64] *Input pipeline performance guide*,
https://www.tensorflow.org/performance/datasets_performance.

[65] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[66] M. Gürbüzbalaban, A. Ozdaglar, and P. Parrilo, “Why Random Reshuffling Beats Stochastic Gradient Descent,” *ArXiv e-prints*, Oct. 2015. arXiv: `1510.08560 [math.OC]`.

[67] R. M. Rogers, S. P. Vadhan, A. Roth, and J. Ullman, “Privacy odometers and filters: Pay-as-you-go composition.,” in *NIPS*, D. D. Lee, M. Sugiyama, U. von Luxburg, I. Guyon, and R. Garnett, Eds., 2016, pp. 1921–1929.

[68] https://github.com/tensorflow/models/tree/master/research/differential_privacy.

[69] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[70] *Breast cancer wisconsin (original) data set*, http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+W

[71] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014. arXiv: `1409.1556`.

[72] *Imagenet*, http://www.image-net.org/.

[73] D. Kifer and A. Machanavajjhala, "No free lunch in data privacy," in *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '11, Athens, Greece, 2011, pp. 193–204, ISBN: 978-1-4503-0661-4.

[74] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the gan: Information leakage from collaborative deep learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '17, Dallas, Texas, USA, 2017, pp. 603–618, ISBN: 978-1-4503-4946-8.

[75] A. Ghosh and R. Kleinberg, "Inferential privacy guarantees for differentially private mechanisms," *CoRR*, vol. abs/1603.01508, 2016. arXiv: `1603.01508`.

[76] M. Fredrikson, E. Lantz, S. Jha, S. Lin, D. Page, and T. Ristenpart, "Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing," in *USENIX Security Symposium*, San Diego, CA, 2014, pp. 17–32, ISBN: 978-1-931971-15-7.

[77] C. Song, T. Ristenpart, and V. Shmatikov, "Machine learning models that remember too much," in *Proc. of ACM CCS*, Dallas, Texas, USA: ACM, 2017, pp. 587–601, ISBN: 978-1-4503-4946-8.

[78] M. E. Gursoy, L. Liu, S. Truex, L. Yu, and W. Wei, "Utility-aware synthesis of differentially private and attack-resilient location traces," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '18, Toronto, Canada: ACM, 2018, pp. 196–211, ISBN: 978-1-4503-5693-0.

[79] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. NIPS*, Montreal, CA, 2014.

[80] I. Sutskever, J. Martens, and G. Hinton, "Generating text with recurrent neural networks," in *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ser. ICML'11, Bellevue, Washington, USA: Omnipress, 2011, pp. 1017–1024, ISBN: 978-1-4503-0619-5.

[81] A. Graves, A. Mohamed, and G. E. Hinton, "Speech recognition with deep recurrent neural networks," *CoRR*, vol. abs/1303.5778, 2013. arXiv: `1303.5778`.

[82] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *In Proceedings of EMNLP*, 2013, pp. 1631–1642.

[83] G. Chen, "A gentle tutorial of recurrent neural network with error backpropagation," *CoRR*, vol. abs/1610.02583, 2016.

[84] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, Dec. 1997.

[85] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.

[86] H. Sak, A. W. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *INTERSPEECH*, 2014.

[87] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang, "Bottom-up and top-down attention for image captioning and visual question answering," in *CVPR*, Jun. 2018, pp. 6077–6086.

[88] V. Kulkarni and B. Garbinato, "Generating synthetic mobility traffic using rnns," in *Proceedings of the 1st Workshop on Artificial Intelligence and Deep Learning for Geographic Knowledge Discovery*, ser. GeoAI '17, Los Angeles, California: ACM, 2017, pp. 1–4, ISBN: 978-1-4503-5498-1.

[89] Z. Lin, M. Yin, S. Feygin, M. S. Transportation, J.-F. Paiement, and A. P. CEE, "Deep generative models of urban mobility," in *KDD*, 2017.

[90] T. Brinkhoff, "A framework for generating network-based moving objects," *GeoInformatica*, vol. 6, no. 2, pp. 153–180, 2002.

[91] V. Bindschaedler and R. Shokri, "Synthesizing plausible privacy-preserving location traces," in *2016 IEEE Symposium on Security and Privacy (SP)*, 2016, pp. 546–563.

[92] W. S.B.Z.W. W. Hao Wu Ziyang Chen, "Modeling trajectories with recurrent neural networks," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 2017, pp. 3083–3090.

[93] J. You, R. Ying, X. Ren, W. L. Hamilton, and J. Leskovec, "Graphrnn: A deep generative model for graphs," *CoRR*, vol. abs/1802.08773, 2018. arXiv: `1802.08773`.

[94] A. Triastcyn and B. Faltings, "Generating differentially private datasets using gans," *CoRR*, vol. abs/1803.03148, 2018. arXiv: `1803.03148`.