BOSTON UNIVERSITY

GRADUATE SCHOOL OF ARTS AND SCIENCES

Dissertation

ADVANCES IN PRIVACY-PRESERVING MACHINE LEARNING

by

OM DIPAKBHAI THAKKAR

B.Tech., Dhirubhai Ambani Institute of Information and Communication Technology, India, 2014

Submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

2019

Approved by

First Reader _____

Adam Smith, Ph.D.
Professor, Computer Science

Second Reader _____

Sofya Raskhodnikova, Ph.D.
Professor, Computer Science

Third Reader _____

Alina Ene, Ph.D.
Assistant Professor, Computer Science

Fourth Reader _____

Lorenzo Orecchia, Ph.D.
Adjunct Professor, Computer Science

ACKNOWLEDGMENTS

The work included in this dissertation was done in collaboration with Raef Bassily, Roger Iyengar, Prateek Jain, Joseph P. Near, Dawn Song, Abhradeep Thakurta, and Lun Wang. In particular, Chapter 4 is based on Iyengar, Near, Song, Thakkar, Thakurta, & Wang (2019b), Chapter 5 contains results from Bassily, Thakurta, & Thakkar (2018), and Chapter 6 comprises work in Jain, Thakkar, & Thakurta (2018). I am grateful to my collaborators for their invaluable contributions to these works. I would also like to thank Vitaly Feldman for his ideas about the possible extensions of the results in Section 5.5, and Ilya Mironov for his helpful feedback on the empirical evaluation in Section 6.4.

This dissertation would not have been possible without the incredible guidance of my advisor, Adam Smith. I always excitedly looked forward to my meetings with Adam because each meeting was a learning experience. Adam's passion for research is contagious, and his patience as well as his skill of thinking from the other person's perspective are virtues that I will always strive to achieve. Adam is truly an inspiration for me, and I will forever be grateful to him for making my Ph.D. journey enjoyable as well as fruitful.

I would like to thank my dissertation committee members Sofya Raskhodnikova, Alina Ene, and Lorenzo Orecchia for agreeing to be readers for this dissertation and providing helpful feedback. I am especially grateful to Sofya for her detailed comments on the first three chapters that improved their presentation dramatically.

I'm really grateful to Abhradeep Thakurta, who has been instrumental in shaping my research to such an extent that he is a coauthor in all of the works included in this dissertation. My interest in machine learning was piqued largely during

my initial interactions with Abhradeep. He has been extremely patient throughout these years in explaining new concepts to me.

I thank my other collaborators, namely Galen Andrew, Steve Chien, Brendan McMahan, Swaroop Ramaswamy, Ryan Rogers, Aaron Roth, Nathan Srebro, and Blake Woodworth, with whom I worked on research projects not included in this dissertation.

I am fortunate to have had three internship opportunities and two semester-long research visits during my Ph.D. I am thankful to many people: the CoreOS team at Apple; Nicholas Carlini, Steve Chien, Úlfar Erlingsson, Brendan McMahan, Nicolas Papernot, Martin Pelikan, and Kunal Talwar at Google; Dawn Song from UC Berkeley; and Shibani Santurkar, and Nirvan Tyagi, for helping me during these experiences and making them enjoyable.

I am thankful to have received support from 1) National Science Foundation grant CAREER CCF-0845701 awarded to Sofya Raskhodnikova, 2) National Science Foundation grant IIS-1832766 awarded to Adam Smith, 3) National Science Foundation grant AF-1763786 awarded to Adam Smith, 4) a grant from the Sloan foundation, and 5) Berkeley Deep Drive. The work in this dissertation was done at Pennsylvania State University, Boston University, University of California, Berkeley, and the Simons Institute for the Theory of Computing.

I extend my gratitude to the departmental staff at both Pennsylvania State University and Boston University for going out of their way to help me efficiently complete all the administrative tasks throughout my Ph.D.

I would like to thank my friends Ramesh Krishnan and Nithin Varma for the innumerous helpful discussions, and being alongside me throughout my Ph.D. journey. I would also like to thank some of the other people who have had a big

influence on my academic career: Vishal Solanki (my high-school teacher who constantly encouraged my curiousity for learning), Hiren Thakkar (my high-school teacher who sparked my love for Computer Science), and Rahul Muthu (my undergraduate advisor, who got me interested in Theoretical Computer Science).

I will forever be indebted to my parents and my sister for their encouragement and support in all of my decisions. I would like to conclude by thanking Nidhi Vyas from the bottom of my heart for her unwavering support and help in all aspects of my life.

ADVANCES IN PRIVACY-PRESERVING MACHINE LEARNING

OM DIPAKBHAI THAKKAR

Boston University, Graduate School of Arts and Sciences, 2019

Major Advisor: Adam Smith, Professor, Computer Science

ABSTRACT

Building useful predictive models often involves learning from personal data. For instance, companies use customer data to target advertisements, online education platforms collect student data to recommend content and improve user engagement, and medical researchers fit diagnostic models to patient data. A recent line of research aims to design learning algorithms that provide rigorous privacy guarantees for user data, in the sense that their outputs—models or predictions—leak as little information as possible about individuals in the training data. The goal of this dissertation is to design private learning algorithms with performance comparable to the best possible non-private ones. We quantify privacy using *differential privacy*, a well-studied privacy notion that limits how much information is leaked about an individual by the output of an algorithm. Training a model using a differentially private algorithm prevents an adversary from confidently determining whether a specific person's data was used for training the model.

We begin by presenting a technique for practical differentially private convex optimization that can leverage any off-the-shelf optimizer as a black box. We also perform an extensive empirical evaluation of the state-of-the-art algorithms on a range of publicly available datasets, as well as in an industry application.

Next, we present a learning algorithm that outputs a private classifier when given black-box access to a non-private learner and a limited amount of unlabeled public data. We prove that the accuracy guarantee of our private algorithm in the PAC model of learning is comparable to that of the underlying non-private learner. Such a guarantee is not possible, in general, without public data.

Lastly, we consider building recommendation systems, which we model using matrix completion. We present the first algorithm for matrix completion with provable user-level privacy and accuracy guarantees. Our algorithm consistently outperforms the state-of-the-art private algorithms on a suite of datasets. Along the way, we give an optimal algorithm for differentially private singular vector computation which leads to significant savings in terms of space and time when operating on sparse matrices. It can also be used for private low-rank approximation.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

AMP	Approximate Minima Perturbation
DP	Differential Privacy
FW	Frank-Wolfe
GD	Gradient Descent
H-F AMP	Hyperparameter-free AMP
NP baseline	Non-private baseline
PGD	Projected Gradient Descent
P-FW	Private Frank-Wolfe
P-SGD	Private Stochastic Gradient Descent
P-PSGD	Private Permutation-based SGD
P-SCPSGD	Private Strongly Convex PSGD
RMSE	Root Mean Squared Error
SGD	Stochastic Gradient Descent

CHAPTER 1

Introduction

Building useful predictive models often involves learning from personal data. For instance, companies use customer data to target advertisements, online education platforms collect student data to recommend content and improve user engagement, and medical researchers fit diagnostic models to patient data. Learning, in the above context, refers to using the data of a small set of individuals sampled from a population to design useful predictive models. The goal is to generate models about characteristics that are not only limited to the sampled individuals, but which generalize to the underlying population as well. For this dissertation, we consider a setting where the sampled individuals contribute their data to a trusted central aggregator. Then, the aggregator runs a learning algorithm on the collected data to generate a predictive model as its output. We provide a schematic of the framework in Figure 1.1.

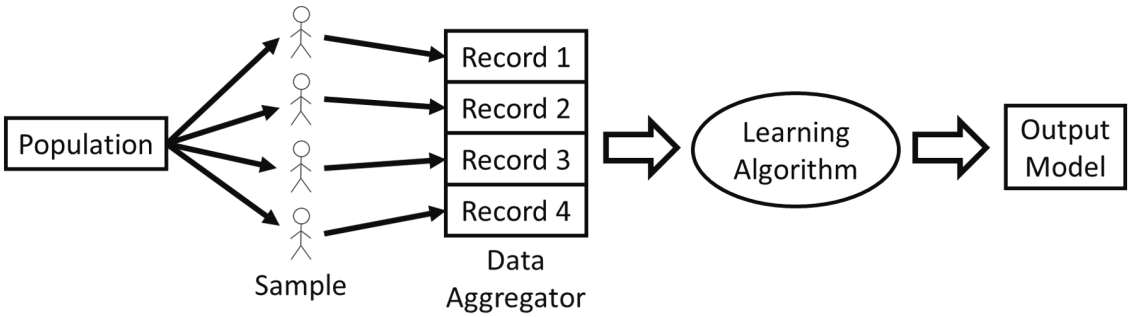

Figure 1.1: A schematic that depicts the considered setting of learning: several individuals sampled from a population contribute their data to a trusted central aggregator, who in turn runs a learning algorithm to generate a predictive model as its output.

As many recent works (Dinur & Nissim (2003); Homer et al. (2008); Sankarara-

man et al. (2009); Bun et al. (2014); Dwork et al. (2015b); Wu et al. (2016); Shokri et al. (2017); Carlini et al. (2018); Melis et al. (2018)) indicate, a model can leak information about the sensitive data it was trained on, even though the data might have never been made public. This motivates the need for providing privacy guarantees to the individuals whose data is a part of the training set.

We quantify privacy using *differential privacy* (Dwork et al. (2006b,a)), a well-studied privacy notion that limits how much information is leaked about an individual by the output of an algorithm. Differential privacy (DP) has been widely adopted by the academic community, as well as big corporations like Google and Apple. The philosophy underlying DP is that the output of an algorithm should not change significantly due to the presence or absence of any individual in the input. In other words, training a model using a differentially private algorithm prevents an adversary from confidently determining whether a specific person's data was used for training the model. This guarantee holds even if the adversary has access to the trained model, and any external side information. We formally define DP in Definition 2.1.3.

In recent years, many works have focused on enabling learning with DP. The aim of this dissertation is to design private learning algorithms that provide generalization guarantees comparable to the best possible non-private ones. One of the highlights of this dissertation is a set of black-box methods for transforming non-private learning methods into private learning algorithms. Such transformations are useful as they tend to be modular, and can take advantage of novel learning techniques which may additionally have been tuned for performance. They can also make use of any customized infrastructure that may have been built for the non-private learning techniques. In contrast, white-box modifications, i.e., trans-

formations dependent on the inner structure of specific techniques, often involve having to make adjustments to the hardware design and software pipelines. This can be time-consuming, expensive, and can result in a loss in efficiency of the technique. For instance, the existing white-box modification of SGD within Tensorflow (Abadi et al. (2015)) results in reduced parallelism for the private technique.

Our first main result is a generic private algorithm for convex optimization that uses non-private algorithms as black-box. Convex optimization is central to machine learning, and the advances therein also have implications to deep learning. Next, we provide black-box transformations for classification tasks in the semi-supervised learning setting.

In addition to black-box transformations, we also provide a private algorithm for recommendation systems, which we model via the problem of matrix completion. Our algorithm builds on the popular Frank-Wolfe method (Frank & Wolfe (1956); Jaggi et al. (2010)), a standard iterative optimization technique having lightweight updates, which enables our algorithm to provide a strong privacy guarantee along with non-trivial utility for this problem.

1.1 THE LANDSCAPE OF "PRIVATE" LEARNING

In this section, we briefly describe some approaches taken by prior works to learn "privately". Our focus is on providing rigorous guarantees, but there is a lot of work on methods with heuristic guarantees. A typical example is k-anonymity (Sweeney (2002)), which has been commonly used to make anonymized releases of sensitive data. Common methods for achieving k-anonymity release a version of the original input dataset in which some attribute values may be hidden, whereas some others may be generalized to broader categories. Even though k-anonymity

protects against a specific class of *linkage attacks*, it does not provide any guarantee on the leakage of specific attributes of individual records. Moreover, anonymized releases of datasets may not exist in isolation, and k-anonymity has been shown (Ganta et al. (2008)) to be vulnerable to *composition attacks* which make use of side information to re-identify overlapping samples from mutiple independently anonymized datasets.

The notion of secure multi-party computation (see Lindell & Pinkas (2008); Evans et al. (2018) for an overview) has been used in settings where rather than contributing private data to a central aggregator (as shown in Figure 1.1), individuals keep their data with themselves but want to collectively run an algorithm. Secure multi-party computation (MPC) has a different, and complementary, goal as compared to differential privacy. It focuses on ensuring that only the output of the joint computation is revealed; raw inputs and all intermediate results are kept secret. However, there can be cases where releasing the output of a computation can reveal the raw inputs. For instance, if the output of a sum of multiple non-negative integers is 0, it reveals that the value of each of the individual inputs is 0. To this end, DP focuses on bounding the leakage of information about any individual input from releasing the output of a computation. DP algorithms can be implemented via an MPC protocol (for example, Dwork et al. (2006a)) to remove the need for a trusted aggregator. In particular, the algorithms proposed in this dissertation can be implemented via MPC.

A related line of work (for example, Konečný et al. (2016); Konecný et al. (2016); McMahan et al. (2017); McMahan & Ramage (2017)) has focused on Federated Learning, a framework which has many interpretations. The core idea of Federated Learning is to collaboratively train a model on a central server without di-

rectly sharing any individual's input data with the server. Although there is no single standard for privacy or confidentiality in the setting of Federated Learning, efficient approaches for MPC (for example, Bonawitz et al. (2017); Reyzin et al. (2018)) can be advantageous in practice for incorporating DP in this setting.

In the absence of a guarantee of privacy, information about training data can be leaked in unexpected ways. There have been many works that demonstrate this by designing attacks to exploit the learning process. Dinur & Nissim (2003) design a general *reconstruction attack*, which reconstructs the input dataset by taking advantage of multiple statistical queries being answered with sufficient accuracy on the input. Other works (for example, Homer et al. (2008); Sankararaman et al. (2009); Bun et al. (2014); Dwork et al. (2015b); Shokri et al. (2017); Melis et al. (2018)) design *membership inference attacks* that infer the presence or absence of a particular record in the training process by exploiting the predictions of the trained model and some side information. Carlini et al. (2018) focus on *memorization attacks* that extract sensitive input samples that were accidentally memorized by high-dimensional models during training.

The initial works demonstrating attacks led to many notions being proposed for bounding the information leakage about inputs from the output of a training algorithm, and DP was one such rigorous notion to come out of those efforts. Although some basic mechanisms for obtaining DP (for example, the Laplace mechanism from Dwork et al. (2006b), and the Gaussian mechanism from Nikolov et al. (2013)) can be composed together to perform various tasks, it is often the case that custom, task-specific techniques provide better utility. Thus, much of the research in this area has focused on building DP mechanisms providing utility guarantees for complex tasks.

There are several major lines of work within differentially private learning. In recent years, designing DP techniques with utility guarantees for convex optimization has been a very active area of research (for example, see Chaudhuri et al. (2011); Kifer et al. (2012); Song et al. (2013); Smith & Thakurta (2013); Bassily et al. (2014a); Jain & Thakurta (2014); Talwar et al. (2014); Wu et al. (2017); Feldman et al. (2018)). There have also been efforts (Abadi et al. (2016); Papernot et al. (2016, 2018)) on effectively training high-dimensional deep neural networks with differential privacy. There is a body of literature, including Kasiviswanathan et al. (2008); Beimel et al. (2010); Chaudhuri & Hsu (2011); Beimel et al. (2013); Bun et al. (2015), that studies the effect of incorporating privacy on the sample complexity for various families of problems in the standard PAC model of learning (Valiant (1984); Kearns & Vazirani (1994)). Lastly, many works (for example, Blum et al. (2005); McSherry & Mironov (2009); Chan et al. (2011); Hardt & Roth (2012); Dwork et al. (2014c); Hardt & Roth (2013); Liu et al. (2015)) have focused on designing DP recommendation systems. To that end, they provide algorithms for matrix completion and low-rank approximation.

In this dissertation, we provide a detailed discussion of prior works on private recommendation systems in Section 3.4, DP convex optimization in Chapter 4, and private PAC learning in Chapter 5.

CHAPTER 2

Preliminaries

In this chapter, we formally define the notation, important definitions, and existing results that have been used in this dissertation.

We denote the data universe by U, and define $\mathcal{D}$ to be a distribution over U. We denote an m-element dataset by $D = \{d_1, d_2, \ldots, d_m\}$. Typically, for each $i \in [m]$ we have d_i drawn independently from $\mathcal{D}$, denoted by $d_i \sim \mathcal{D}$.

2.1 DEFINING "PRIVACY"

Here, we provide formal definitions of the notions we use to quantify privacy, mainly *differential privacy* (DP), and a recent variant called *concentrated differential privacy* (CDP). We also describe some common privacy mechanisms that have been used in subsequent chapters.

DP bounds the effect of a record on the output of an algorithm, and this is captured by a "neighboring" relationship between datasets. There are two notions of neighboring used in this dissertation, which we define next.

Definition 2.1.1 (Neighboring under modification)**.** *A pair of datasets $D, D' \in U^m$ are* neighboring under modification *if D' can be obtained from D by modifying one sample $d_i \in D$ for some $i \in [m]$.*

For defining neighboring under insertion/deletion, we denote the symmetric difference between any two datasets $D, D' \in U^*$ by $D \triangle D'$.

Definition 2.1.2 (Neighboring under insertion/deletion)**.** *A pair of datasets $D, D' \in U^*$ are* neighboring under insertion/deletion *if $|D \triangle D'| = 1$. In other words, D can be obtained from D' by either the addition or deletion of a sample.*

We use the definition of neighboring under modification (Definition 2.1.1) in Chapter 4 and Chapter 6, whereas we use neighboring under insertion/deletion (Definition 2.1.2) in Chapter 5.

2.1.1 Differential Privacy

Here, we define differential privacy, state some of its useful properties, and describe a few common mechanisms used for achieving DP.

Definition 2.1.3 ((ϵ, δ)-Differential Privacy (Dwork et al. (2006b,a))). *A (randomized) algorithm $\mathcal{A}$ with input domain U^* and output range $\mathcal{R}$ is* (ϵ, δ)-differentially private *if for all pairs of neighboring inputs $D, D' \in U^*$, and every measurable $S \subseteq \mathcal{R}$, we have*

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

where probabilities are taken over the coin flips of $\mathcal{A}$.

When $\delta > 0$, the notion is also known as *approximate* DP. On the other hand, when $\delta = 0$, it is known as *pure* DP, and is parameterized only by ϵ.

An important advantage of differential privacy is that it is closed under *post-processing*, and can be adaptively composed, which we describe next.

Lemma 2.1.4 (Post-processing (Dwork et al. (2006b))). *If a mechanism $\mathcal{A} : U^* \to \mathcal{R}$ is (ϵ, δ)-differentially private, then for any function $f : \mathcal{R} \to \mathcal{R}'$, we have that $f \circ \mathcal{A}$ is also (ϵ, δ)-differentially private.*

Lemma 2.1.5 (Basic Composition (Dwork et al. (2006b))). *For each $i \in [k]$, let algorithm $\mathcal{A}_i : U^* \times \prod_{j=1}^{i-1} \mathcal{R}_j \to \mathcal{R}_i$ be (ϵ_i, δ_i)-DP in its first argument. If algorithm*

$\mathcal{A}_{[k]} : U^* \to \prod_{j=1}^{k} \mathcal{R}_j$ is defined such that

$$\mathcal{A}_{[k]}(D) = \left(\mathcal{A}_1(D), \mathcal{A}_2\left(D, \mathcal{A}_1(D)\right), \ldots, \mathcal{A}_k(D, \mathcal{A}_1(D), \ldots, \mathcal{A}_{k-1}(D))\right)$$

then $\mathcal{A}_{[k]}$ is $\left(\sum_{i=1}^{k} \epsilon_i, \sum_{i=1}^{k} \delta_i\right)$-DP.

Lemma 2.1.6 (Advanced Composition (Dwork et al. (2010); Kairouz et al. (2017)))**.** *For each $i \in [k]$, let algorithm $\mathcal{A}_i : U^* \times \prod_{j=1}^{i-1} \mathcal{R}_j \to \mathcal{R}_i$ be (ϵ_i, δ_i)-DP in its first argument. If algorithm $\mathcal{A}_{[k]} : U^* \to \prod_{j=1}^{k} \mathcal{R}_j$ is defined such that*

$$\mathcal{A}_{[k]}(D) = \left(\mathcal{A}_1(D), \mathcal{A}_2\left(D, \mathcal{A}_1(D)\right), \ldots, \mathcal{A}_k(D, \mathcal{A}_1(D), \ldots, \mathcal{A}_{k-1}(D))\right)$$

then for every $\delta' > 0$, algorithm $\mathcal{A}_{[k]}$ is $\left(\epsilon', 1 - (1-\delta')\prod_{i=1}^{k}(1-\delta_i)\right)$-DP, where

$$\epsilon' = \min\left(\sum_{i=1}^{k}\epsilon_i, \sum_{i=1}^{k}\frac{\epsilon_i(e^{\epsilon_i}-1)}{e^{\epsilon_i}+1} + \sqrt{\min\left(\sum_{i=1}^{k} 2\epsilon_i^2 \log\left(e + \frac{\sqrt{\sum_{i=1}^{k}\epsilon_i^2}}{\delta'}\right), \sum_{i=1}^{k} 2\epsilon_i^2 \log\frac{1}{\delta'}\right)}\right).$$

To describe two of the common techniques for achieving differential privacy, we first define the global sensitivity of a function.

Definition 2.1.7 (L_p-sensitivity)**.** *A function $f : U^* \to \mathbb{R}^n$ has L_p-*sensitivity Δ *if*

$$\max_{\substack{D, D' \in U^* \text{ s.t.} \\ (D, D') \text{ neighbors}}} \|f(D) - f(D')\|_p = \Delta.$$

One of the most common techniques for achieving pure differential privacy is the Laplace mechanism, which we define next.

Lemma 2.1.8 (Laplace mechanism (Dwork et al. (2006b))). *If a function $f : U^* \to \mathbb{R}^n$ has L_1-sensitivity Δ, and a mechanism $\mathcal{A}$ on input $D \in U^*$ outputs $f(D) + b$, where $b \sim Lap(\lambda I_{n\times n})$ and $\lambda = \frac{\Delta}{\epsilon}$, then $\mathcal{A}$ satisfies ϵ-differential privacy. Here, $Lap(\lambda I_{n\times n})$ denotes a vector of n i.i.d. samples from the Laplace distribution $Lap(\lambda)$.*

Next, we define one of the most common techniques for achieving approximate differential privacy, which is the Gaussian mechanism.

Lemma 2.1.9 (Gaussian mechanism (Nikolov et al. (2013))). *If a function $f : U^* \to \mathbb{R}^n$ has L_2-sensitivity Δ, and a mechanism $\mathcal{A}$ on input $D \in U^*$ outputs $f(D) + b$, where $b \sim \mathcal{N}\left(0, \frac{\Delta^2}{\epsilon^2}\left(1 + \sqrt{2\log\frac{1}{\delta}}\right)^2 I_{n\times n}\right)$, then $\mathcal{A}$ satisfies (ϵ, δ)-differential privacy. Here, $N(0, \sigma^2 I_{n\times n})$ denotes a vector of n i.i.d. samples from the zero-mean Gaussian distribution having variance σ^2.*

2.1.2 Concentrated Differential Privacy

In Chapter 6, we also use a recently proposed notion of privacy called *zero concentrated differential privacy* (zCDP) (Bun & Steinke (2016)). The definition of concentrated differential privacy (CDP) was given by Dwork & Rothblum (2016), which was then modified by Bun & Steinke (2016).

Definition 2.1.10 (ρ-zCDP (Bun & Steinke (2016))). *A (randomized) algorithm $\mathcal{A}$ with input domain U^* and output range $\mathcal{R}$ is ρ-zCDP if for all pairs of neighboring inputs $D, D' \in U^*$, and for all $\alpha \in (1, \infty)$, we have:*

$$\mathfrak{D}_\alpha(\mathcal{A}(D)||\mathcal{A}(D')) \leq \rho\alpha$$

where $\mathfrak{D}_\alpha(P||Q) = \frac{1}{\alpha-1}\log\left(\mathbb{E}_{x\sim P}\left[\left(\frac{P(x)}{Q(x)}\right)^{\alpha-1}\right]\right)$ denotes the α-Rényi divergence between distributions P and Q.

Next, we provide the post-processing and composition properties of zCDP.

Lemma 2.1.11 (Post-processing (Bun & Steinke (2016))). *If a mechanism $\mathcal{A} : U^* \to \mathcal{R}$ is ρ-zCDP, then for any function $f : \mathcal{R} \to \mathcal{R}'$, we have that $f \circ \mathcal{A}$ is also ρ-zCDP.*

Lemma 2.1.12 (Composition (Bun & Steinke (2016))). *Let $\mathcal{A}_1 : U^* \to \mathcal{R}_1$, and $\mathcal{A}_2 : U^* \times \mathcal{R}_1 \to \mathcal{R}_2$ be two randomized algorithms, where $\mathcal{A}_1$ is ρ_1-zCDP and $\mathcal{A}_2(\cdot, r_1)$ is ρ_2-zCDP for each $r_i \in \mathcal{R}_1$. If algorithm $\mathcal{A} : U^* \to \mathcal{R}_2$ is composed such that $\mathcal{A}(D) = \mathcal{A}_2(D, \mathcal{A}_1(D))$, then $\mathcal{A}$ is $(\rho_1 + \rho_2)$-zCDP.*

Now, we show that zCDP is an intermediate notion between pure DP and approximate DP.

Lemma 2.1.13 (Bun & Steinke (2016)). *If a mechanism $\mathcal{A} : U^* \to \mathcal{R}$ is:*

- *ϵ-DP, then $\mathcal{A}$ is $\left(\frac{1}{2}\epsilon^2\right)$-zCDP.*
- *ρ-zCDP, then $\mathcal{A}$ is $\left(\rho + 2\sqrt{\rho \log(1/\delta)}, \delta\right)$-DP for every $\delta \in (0, 1)$.*

Lastly, we provide the zCDP guarantees of the Gaussian mechanism.

Lemma 2.1.14 (Gaussian mechanism (Bun & Steinke (2016))). *If a function $f : U^* \to \mathbb{R}^n$ has L_2-sensitivity Δ, and a mechanism $\mathcal{A}$ on input $D \in U^*$ outputs $f(D) + b$, where $b \sim \mathcal{N}\left(0, \frac{\Delta^2}{2\rho} I_{n \times n}\right)$, then $\mathcal{A}$ satisfies ρ-zCDP. Here, $N(0, \sigma^2 I_{n \times n})$ denotes a vector of n i.i.d. samples from the zero-mean Gaussian distribution having variance σ^2.*

2.2 THE FRAMEWORK OF LEARNING

Next, we provide some definitions from learning theory that will be useful in subsequent chapters.

Define a function $\ell : \Theta \times U \to \mathbb{R}$ for some set Θ, and let the average over a dataset be defined as $L(\theta; D) = \frac{1}{m} \sum_{i=1}^{m} \ell(\theta; d_i)$ for $\theta \in \Theta, D \in U^m$. Also, define

the expectation of L over a distribution to be $L(\theta; \mathcal{D}) = \mathbb{E}_{D \sim \mathcal{D}^m}(L(\theta; D))$. We first define empirical risk and population risk.

Definition 2.2.1 (Empirical Risk)**.** *Given an element $\hat{\theta} \in \Theta$ and a dataset $D \in U^m$, the* empirical risk *of $\hat{\theta}$ on D is*

$$\mathcal{R}(\hat{\theta}; D) = L(\hat{\theta}; D) - \min_{\theta \in \Theta} L(\theta; D).$$

Given an algorithm $\mathcal{A} : \Theta \times U^m \to \mathbb{R}$, we define the empirical risk of $\mathcal{A}$ on a dataset $D \in U^m$ as $\mathcal{R}(\mathcal{A}; D) = \mathbb{E}(\mathcal{R}(\mathcal{A}(D); D))$, where the expectation is over the internal randomness of $\mathcal{A}$.

Definition 2.2.2 (Population Risk)**.** *Given a distribution $\mathcal{D}$ over U and an element $\hat{\theta} \in \Theta$, the* population risk *of $\hat{\theta}$ on $\mathcal{D}$ is*

$$\mathcal{R}(\hat{\theta}; \mathcal{D}) = L(\hat{\theta}; \mathcal{D}) - \min_{\theta \in \Theta} L(\theta; \mathcal{D}).$$

Given an input dataset $D \sim \mathcal{D}^m$, our central objective in the subsequent chapters of this dissertation will be to find an element $\hat{\theta} \in \Theta$ such that its population risk $\mathcal{R}(\hat{\theta}; \mathcal{D})$ is *low*, and the *privacy* of the input dataset D is guaranteed. In other words, if $\theta_{opt} = \arg\min_{\theta \in \Theta} L(\theta; \mathcal{D})$, i.e., θ_{opt} is an element in the class Θ with a population risk equal to 0, we want $\hat{\theta}$ to be competitive with θ_{opt} while guaranteeing the privacy of dataset D. The class Θ and the optimization function ℓ will vary by the problem we are trying to solve, and each subsequent chapter addresses a special case:

- In Chapter 4, we have $\Theta = \mathbb{R}^n$ as the n-dimensional model space, and the loss function $\ell(\theta; d)$ for $\theta \in \Theta, d \in U$ is convex in the first parameter θ. This formulation falls under unconstrained convex optimization, and finds applications

in a variety of settings, including machine learning via linear regression, logistic regression, or support vector machines (SVMs).

- In Chapter 5, we operate in the standard PAC model of learning, where the universe is the space of feature vectors and labels $U = \mathcal{X} \times \mathcal{Y}$, and the search space $\Theta = \mathcal{H}$ is a hypothesis class where each hypothesis $h \in \mathcal{H}$ is a mapping $h : \mathcal{X} \to \mathcal{Y}$. For $d = (x, y) \in U$, the loss function $\ell(h; d) = \mathbb{I}[h(x) \neq y]$ denotes the classification error.

- In Chapter 6, we solve the problem of matrix completion, where the data universe U consists of the entries of an $(m \times n)$ matrix Y^* s.t. $\|Y^*\|_{\mathsf{nuc}} \leq k$ for some $k > 0$, where $\|Y^*\|_{\mathsf{nuc}}$ denotes the nuclear norm of matrix Y^*. The input is the incomplete matrix $D = \mathsf{P}_\Omega(Y^*)$, where $\Omega = \{(i, j) \subseteq [m] \times [n]\}$ comes from a distribution $\mathcal{D}$ that is uniform over $[m] \times [n]$, the operator $\mathsf{P}_\Omega(Y)_{ij} = Y_{ij}$ if $(i, j) \in \Omega$, and 0 otherwise. The search space $\Theta = \{Y \in \mathbb{R}^{m \times n} : \|Y\|_{\mathsf{nuc}} \leq k\}$, and the loss function $\ell(Y; \mathsf{P}_\Omega(Y^*)_i) = \frac{m}{2|\Omega|}\|\mathsf{P}_\Omega(Y - Y^*)_i\|_F^2$ denotes a scaled version of the Mean Squared Error (MSE) over the observed entries. We also present an optimal differentially private algorithm for singular vector computation, based on Oja's method, that provides significant savings in terms of space and time when operating on sparse matrices. We conduct an empirical evaluation of our algorithm on a suite of datasets, and show that it consistently outperforms the state-of-the-art private algorithms.

CHAPTER 3

Main Results

In this chapter, we first provide an overview of the main results in this dissertation. Next, we describe each of the results in detail.

3.1 OVERVIEW

We start by presenting our two major contributions towards practical differentially private convex optimization. They were published in Iyengar, Near, Song, Thakkar, Thakurta, & Wang (2019b). First, we present Approximate Minima Perturbation, a novel algorithm that can leverage any off-the-shelf optimizer. All the state-of-the-art algorithms have some hyperparameters that need to be tuned for obtaining good utility, thus incurring an additional cost to privacy. In contrast, we show that our algorithm can be employed without any hyperparameter tuning, thus making it attractive for practical deployment. Our second contribution is to perform an extensive empirical evaluation of the state-of-the-art algorithms for differentially private convex optimization, and compare it with our new approach. Our evaluation is on a range of publicly available benchmark datasets, as well as an industry application obtained through a collaboration with Uber.

Next, we present a learning algorithm (inspired by Papernot et al. (2016)) that outputs a private classifier when given black-box access to a non-private learner and a limited amount of unlabelled public data. It was published in Bassily, Thakurta, & Thakkar (2018). We start by providing a new differentially private algorithm for answering a sequence of m online classification queries (given by a sequence of m unlabeled public feature vectors) based on a training set with a fixed privacy budget. Our algorithm follows the paradigm of subsample-and-aggregate

(Nissim et al. (2007)), in which any generic non-private learner is trained on disjoint subsets of the private training set, and then for each classification query, the votes of the resulting classifiers ensemble are aggregated in a differentially private fashion. Our private aggregation is based on a novel combination of the distance-to-instability framework (Smith & Thakurta (2013)), and the sparse-vector technique (Dwork et al. (2009); Hardt & Rothblum (2010)). We show that our algorithm makes a conservative use of the privacy budget. In particular, if the underlying non-private learner yields a classification error of at most $\alpha \in (0, 1)$, then our construction answers more queries, by at least a factor of $1/\alpha$ in some cases, than what is implied by a direct application of the advanced composition theorem for differential privacy. Next, we apply the knowledge transfer technique (which is inspired by the early work of Breiman (1996)) to construct a private learner that outputs a classifier, which can be used to answer an unlimited number of queries.

We analyze our construction in the (agnostic) PAC model, and prove upper bounds on the sample complexity for both the realizable and the non-realizable cases. Similar to non-private sample complexity, our bounds are completely characterized by the VC dimension of the concept class. If only private data is used, there are lower bounds showing that the private sample complexity will have a necessary dependence on the dimensionality of the model (Bassily et al. (2014a)), or the size of the model class (Bun et al. (2015)). Thus, such a guarantee is not possible, in general, without public data.

Lastly, we present the first provably DP algorithm with formal utility guarantees for the problem of user-level privacy-preserving matrix completion. It was published in Jain, Thakkar, & Thakurta (2018). Our algorithm is based on the Frank-Wolfe method for optimization, and it consistently estimates the underlying

preference matrix as long as the number of users m is $\omega(n^{5/4})$, where n is the number of items, and each user provides her preference for at least $\sqrt{n}$ randomly selected items. We also empirically evaluate our algorithm on a suite of datasets. We show that it provides nearly the same accuracy as the state-of-the-art non-private algorithm and outperforms the state-of-the-art private algorithm by 30% in some cases.

Our matrix completion algorithm involves computing a rank one approximation of a matrix. For this purpose, we give an optimal differentially private algorithm for singular vector computation, based on Oja's method, that provides significant savings in terms of space and time when operating on sparse matrices. Further, we show that it can be used for differentially private low-rank approximation and thus might be of independent interest.

3.2 PRIVATE CONVEX OPTIMIZATION

With the recent advancements in machine learning and big data, private convex optimization has proven to be useful for large-scale learning over sensitive user data that has been collected by organizations. Our objective is to provide insight into practical differentially private convex optimization, with a specific focus on the classical technique of objective perturbation (Chaudhuri et al. (2011); Kifer et al. (2012)). Our main technical contribution is to design a new algorithm for private convex optimization that is amenable to real-world scenarios, and provide privacy and utility guarantees for it. In addition, we conduct a broad empirical evaluation of approaches for private convex optimization, including our new approach. Our evaluation is more extensive than prior works (Chaudhuri et al. (2011); Song et al. (2013); Jain & Thakurta (2014); Wu et al. (2017)). Apart from these, we also consider

four industry use cases, obtained in collaboration with Uber Technologies, Inc. We provide advice and resources for practitioners, including open-source implementations (Iyengar et al. (2019a)) of the algorithms evaluated, and benchmarks on all the public datasets considered.

3.2.1 Objective Perturbation and its Practical Feasibility

We focus our attention on the technique of objective perturbation, because prior works (Chaudhuri et al. (2011); Kifer et al. (2012)) as well as our own preliminary empirical results have hinted at its superior performance. The standard technique of objective perturbation (Chaudhuri et al. (2011)) consists of a two-stage process: "perturbing" the objective function by adding a random linear term, and releasing the minima of the perturbed objective. It has been shown (Chaudhuri et al. (2011); Kifer et al. (2012)) that releasing such a minima can be done while achieving DP guarantees.

However, objective perturbation provides privacy guarantees *only if* the output of the mechanism is the *exact minima* of the perturbed objective. Practical algorithms for convex optimization often involve the use of first-order iterative methods, such as gradient descent or stochastic gradient descent (SGD) due to their scalability. However, such methods typically offer convergence rates that depend on the number of iterations carried out by the method, so they are *not* guaranteed to reach the exact minima in finite time. As a result, it is not clear if objective perturbation in its current form can be applied in a practical setting, where one is usually constrained by resources such as computing power, and reaching the minima might not be feasible.

We answer the following question in the affirmative: Does there exist an alter-

native to standard objective perturbation which provides comparable privacy and utility guarantees when the released model is *not necessarily* the minima of the perturbed objective? A major implication of this result is that one can use first-order iterative methods in combination with such an approach. This can be highly beneficial in terms of the time taken to obtain a private solution, as first-order methods often find a "good" solution quickly.

3.2.2 Our Approach: Approximate Minima Perturbation

We propose Approximate Minima Perturbation (AMP), a strengthened alternative to objective perturbation. Our method provides privacy and utility guarantees when the released model is a noisy "approximate" minima for the perturbed objective. We measure the convergence of a model in terms of the Euclidean norm of its gradient of the perturbed objective. The scale of the noise added to the approximate minima contains a parameter representing the maximum tolerable gradient norm. This results in a trade-off between the gradient norm bound (consequently, the amount of noise to be added to the approximate minima) and the difficulty, in practice, of being able to obtain an approximate minima within the norm bound. This can be useful in settings where a limited computing power is available, which can in turn act as a guide for setting an appropriate norm bound. We note that if the norm bound is set to zero, then this approach reduces to the setting of standard objective perturbation. Approximate Minima Perturbation also brings with it certain distinct advantages, which we will describe next.

Approximate Minima Perturbation works for *all* convex objective functions: Previous works (Chaudhuri et al. (2011); Kifer et al. (2012)) provide privacy guarantees for objective perturbation *only* when the objective is a loss function of a

generalized linear model, i.e., when the loss function is parameterized by an inner product of the feature vector of the data, and the model. On the other hand, the guarantees provided for AMP hold for *any* objective function, and our analysis extends to objective perturbation as well. In both cases, the objective functions are assumed to possess standard properties like *Lipschitz continuity* and *smoothness*.

Approximate Minima Perturbation is the first feasible approach that can leverage any off-the-shelf optimizer: AMP can accommodate any off-the-shelf optimizer as a black-box for carrying out its optimization step. This enables a simple implementation that inherits the scalability properties of the optimizer used, which can be particularly important in situations where high-performance optimizers are available. AMP is the *only* known feasible algorithm for DP convex optimization that allows the use of any off-the-shelf optimizer.

Approximate Minima Perturbation has a competitive hyperparameter-free variant: To ensure privacy for an algorithm, its hyperparameters must be chosen either independently of the data, or by using a differentially private hyperparameter tuning algorithm. Previous work (Chaudhuri et al. (2011); Chaudhuri & Vinterbo (2013); Abadi et al. (2016)) has shown this to be a challenging task. AMP has only four hyperparameters: one related to the Lipschitz continuity of the objective, and the other three related to splitting the privacy budget within the algorithm. In Section 4.4.1, we present a data-independent method for setting all of them. Our empirical evaluation demonstrates that the resulting hyperparameter-free variant of AMP yields comparable accuracy to the standard variant with its hyperparameters tuned in a data-dependent manner (which may be non-private).

3.2.3 Empirical Evaluation, & Resources for Practitioners

We also report on an extensive and broad empirical study of the state-of-the-art differentially private convex optimization techniques. In addition to AMP and its hyperparameter-free variant, we evaluate four existing algorithms: private gradient descent with minibatching (Bassily et al. (2014a); Abadi et al. (2016)), both the variants (convex, and *strongly convex*) of the private Permutation-based SGD algorithm (Wu et al. (2017)), and the private Frank-Wolfe algorithm (Talwar et al. (2014)).

Our evaluation is the largest to date, including a total of 13 datasets. We include datasets with a variety of different properties, including four high-dimensional datasets and four use cases represented by datasets obtained in collaboration with Uber.

The results of our empirical evaluation demonstrate three key findings. First, we confirm the expectation that the cost of privacy decreases as dataset size increases. For *all* the use cases in our evaluation, we obtain differentially private models that achieve an accuracy within 4% of the non-private baseline even for conservative settings of the privacy parameters. For reasonable values of the privacy parameters, the accuracy of the best private model is within 2% of the baseline for two of these datasets, essentially identical to the baseline for one of them, and even slightly higher than the baseline for one of the datasets! This provides empirical evidence to further the claims of previous works (Bassily et al. (2014b); Dwork et al. (2015a)) that DP can also act as a type of regularization, reducing the generalization error.

Second, our results demonstrate a general ordering of algorithms in terms of empirical accuracy. Our results show that AMP generally outperforms all the other

algorithms across all the considered datasets. Under specific conditions like high-dimensionality of the dataset and sparsity of the optimal predictive model for it, we see that private Frank-Wolfe provides the best performance.

Third, our results show that a hyperparameter-free variant of AMP achieves nearly the same accuracy as the standard variant with its hyperparameters tuned in a data-dependent manner. Approximate Minima Perturbation is therefore simple to deploy in practice as it can leverage any off-the-shelf optimizer, and it has a competitive variant that does not require any hyperparameter tuning.

We provide an open-source implementation (Iyengar et al. (2019a)) of the algorithms evaluated, including our Approximate Minima Perturbation, and a complete set of benchmarks used in producing our empirical results. In addition to enabling the reproduction of our results, this set of benchmarks will provide a standard point of reference for evaluating private algorithms proposed in the future. Our open-source release represents the first benchmark for differentially private convex optimization.

3.2.4 Main Contributions

Our main contributions are as follows:

- We propose Approximate Minima Perturbation, a strengthened alternative to objective perturbation that provides privacy guarantees even for an approximate minima of the perturbed objective, and therefore allows the use of *any* off-the-shelf optimizer. No previous approach provides this capability. Compared to previous approaches, AMP also provides improved utility in practice, and works with any convex loss function under standard assumptions.

- We conduct the largest empirical study to date of state-of-the-art DP convex optimization approaches, including as many as nine public datasets, four of which are high-dimensional. Our results demonstrate that AMP generally provides the best accuracy.

- We evaluate DP convex optimization on four use cases obtained in collaboration with Uber. Our results suggest that for the large-scale datasets used in practice, privacy-preserving models can obtain essentially the same accuracy as non-private models for reasonable values of the privacy parameters. In one case, we show that a DP model achieves a *higher* accuracy than the non-private baseline.

- We present a competitive hyperparameter-free variant of AMP, allowing the approach to be deployed without the need for tuning on publicly available datasets, or by a DP hyperparameter tuning algorithm.

- We release open-source implementations (Iyengar et al. (2019a)) of all the algorithms we evaluate, and the first benchmarks for differentially private convex optimization algorithms on as many as nine public datasets.

3.3 MODEL-AGNOSTIC PRIVATE LEARNING

The main goal in the standard setting of differentially private learning is to design a differentially private learner that, given a private training set as input, outputs a model (or, a classifier) that is safe to publish. Despite being a natural way to define the private learning problem, there are several limitations with this standard approach. First, there are pessimistic lower bounds in various learning problems implying that the error associated with the final private model will generally have

necessary dependence on the dimensionality of the model (Bassily et al. (2014a)), or the size of the model class (Bun et al. (2015)). Second, this approach often requires non-trivial, white-box modification of the existing non-private learners (Kasiviswanathan et al. (2008); Chaudhuri et al. (2011); Kifer et al. (2012); Smith & Thakurta (2013); Bassily et al. (2014a); Talwar et al. (2015); Abadi et al. (2016)), which can make some of these constructions less practical since they require making changes in the infrastructure of the existing systems. Third, designing algorithms for this setting often requires knowledge about the underlying structure of the learning problem, e.g., specific properties of the model class (Beimel et al. (2010, 2013); Bun et al. (2015)); or convexity, compactness, and other geometric properties of the model space (Bassily et al. (2014a); Talwar et al. (2015)).

We study the problem of differentially private learning when the learner has access to a limited amount of *public unlabeled* data. Our central goal is to characterize in a basic model, such as the standard PAC model, the improvements one can achieve for private learning in such a relaxed setting compared to the aforementioned standard setting. Towards this goal, we first consider a simpler problem, namely, privately answering classification queries given by a sequence of *public unlabeled* data $\mathcal{Q} = \{\tilde{x}_1, \cdots, \tilde{x}_{\widetilde{m}}\}$. In this problem, one is given a *private labeled* dataset denoted by D, and the goal is to design an (ϵ, δ)-differentially private algorithm that labels all $\widetilde{m}$ public feature vectors in $\mathcal{Q}$. In designing such an algorithm, there are four main goals we aim to achieve: (i) We wish to provide an algorithm that enables answering as many classification queries as possible while ensuring (ϵ, δ)-differential privacy. This is a crucial property for the utility of such an algorithm since the utility in this problem is limited by the number of queries we can answer while satisfying the target privacy guarantee. (ii) We want to have a

modular design paradigm in which the private algorithm can use any generic non-private algorithm (learner) in a *black-box* fashion, i.e., it only has oracle access to the non-private algorithm. This property is very attractive from a practical standpoint as the implementation of such an algorithm does not require changing the internal design of the existing non-private algorithms. (iii) We want to have a design paradigm that enables us to easily and formally transfer the accuracy guarantees of the underlying non-private algorithm into meaningful accuracy guarantees for the private algorithm. The most natural measure of accuracy in that setting would be the misclassification rate. (iv) We want to be able to use such an algorithm together with the public unlabeled data to construct a *differentially private learner* that outputs a classifier, which can then be used to answer as many classification queries as we wish. In particular, given the second goal above, the final private learner would be completely agnostic to the intricacies of the underlying non-private learner and its model. Namely, it would be oblivious to whether the model is simple logistic regression, or a multi-layer deep neural network.

Given the above goals, a natural framework to consider is *knowledge aggregation and transfer*, which is inspired by the early work of Breiman (1996). The general idea is to train a non-private learner on different subsamples from the private dataset to generate an ensemble of classifiers. The ensemble is collectively used in a differentially private manner to generate privatized labels for the given unlabeled public data. Finally, the public data together with the private labels are used to train a non-private learner, which produces a final classifier that is safe to publish.

3.3.1 Our Techniques

We give a new construction for privately answering classification queries that is based on a novel framework combining two special techniques in the literature of differential privacy, namely, the subsampling stability framework (Nissim et al. (2007); Smith & Thakurta (2013)) and the sparse vector technique (Dwork et al. (2009); Hardt & Rothblum (2010); Dwork et al. (2014a)). Our construction also follows the knowledge aggregation and transfer paradigm, but it exploits the stability properties of good non-private learners in a quantifiable and formal manner. Our construction is based on the following idea: if a good learner is independently trained b times on equally sized, independent training sets, then one would expect the corresponding output classifiers $h_1, \cdots, h_b$ to predict "similarly" on a new example from the same distribution. Using this idea, we show that among $\widetilde{m}$ classification queries, one only needs to "pay the price of privacy" for the queries for which there is significant disagreement among the b classifiers. Using our construction and the unlabeled public data, we also provide a final private learner. We show via formal and quantifiable guarantees that our construction achieves our four main goals stated earlier.

We note that our framework is not restricted to classification queries; it can be used for privately answering any sequence of online queries that satisfy certain stability properties in the sense of Smith & Thakurta (2013).

3.3.2 Main Contributions

Answering online classification queries using privacy conservatively: In Section 5.4, we give our (ϵ, δ)-DP construction for answering a sequence of online classification queries. Our construction uses *any generic* non-private learner in a

black-box fashion. The privacy guarantee is completely independent of the non-private learner and its accuracy. Moreover, the accuracy guarantee can be obtained directly from the accuracy of the non-private learner, i.e., the construction allows us to directly and formally "transform" the accuracy guarantee for the non-private learner into an accuracy guarantee for the final private algorithm.

We provide a new privacy analysis for the novel framework combining subsampling stability and sparse vector techniques. We analyze the accuracy of our algorithm in terms of its misclassification rate, defined as the ratio of misclassified queries to the total number of queries, in the standard (agnostic) PAC model. Our accuracy analysis is new, and is based on a simple counting argument. We consider both the realizable and non-realizable (agnostic) cases. In the realizable case, the underlying non-private learner is assumed to be a PAC learner for a hypothesis class $\mathcal{H}$ of VC-dimension V. The private training set consists of m labeled examples, where the labels are generated by some unknown hypothesis $h^* \in \mathcal{H}$. The queries are given by a sequence of $\widetilde{m}$ i.i.d. unlabeled domain points drawn from the same distribution as the domain points in the training set. We show that, with high probability, our private algorithm can answer up to $\approx m/V$ queries with a misclassification rate of $\approx V/m$, which is essentially the optimal misclassification rate attainable *without privacy*. Thus, answering those queries essentially comes with no cost for privacy. When answering $\widetilde{m} > m/V$ queries, the misclassification rate is $\approx \widetilde{m}V^2/m^2$. A straightforward application of the advanced composition theorem of differential privacy would have led to a misclassification rate $\approx \sqrt{\widetilde{m}}V/m$, which can be significantly larger than our rate. This is because our construction pays a privacy cost only for "hard" queries for which the PAC learner tends to be incorrect. Our result for the realizable case is summarized below. We also provide

an analogous statement for the non-realizable case (Theorem 13).

Informal Theorem 3.3.1 (Corresponding to Theorem 12)**.** *Given a PAC learner for a class $\mathcal{H}$ of VC-dimension V, a private training set of size m, and assuming realizability, our private construction (Algorithm 10) answers a sequence of up to $\widetilde{\Omega}(m/V)$ binary classification queries such that, with high probability, the misclassification rate is $\widetilde{O}(V/m)$. When the number of queries $\widetilde{m}$ is beyond $\widetilde{\Omega}(m/V)$, then with high probability, the misclassification rate is $\widetilde{O}\left(\widetilde{m}V^2/m^2\right)$.*

A model-agnostic private learner with formal guarantees: In Section 5.5, we use the knowledge transfer technique to bootstrap a private *learner* from our construction above. The idea is to use our private construction to label a sufficient number of public feature vectors. Then, we use these newly labeled public data for training a non-private learner to finally output a classifier. Since there is no privacy constraint associated with the public data, the overall construction remains private as differential privacy is closed under post-processing. Note that this construction also uses the non-private learner as a black box, and hence it is agnostic to the structure of such learner and the associated model. This general technique has also been adopted in Papernot et al. (2016). Our main contribution here is that we provide formal and explicit utility guarantees for the final private learner in the standard (agnostic) PAC model. Our guarantees are in terms of upper bounds on the sample complexity in both realizable and non-realizable cases. Let $L(h; \mathcal{D}) \triangleq \mathop{\mathbb{P}}_{(x,y)\sim\mathcal{D}}[h(x) \neq y]$ denote the true classification error of a hypothesis h. Given black-box access to an agnostic PAC learner for a class $\mathcal{H}$ of VC-dimension V, we obtain the following results:

Informal Theorem 3.3.2 (Corresponding to Theorems 14, 17)**.** *Let $0 < \alpha < 1$. Let m be the size of the private training set. There exists an (ϵ, δ)-differentially private algorithm*

(Algorithm 11) that, given access to $\widetilde{m} = \widetilde{O}\left(\frac{V}{\alpha^2}\right)$ *unlabeled public data points, w.h.p. outputs a classifier* $\hat{h} \in \mathcal{H}$ *such that the following guarantees hold: (i) Realizable case:* $L(\hat{h}; \mathcal{D}) \leq \alpha$ *for* $n = \widetilde{O}\left(V^{3/2}/\alpha^{3/2}\right)$*, and (ii) Agnostic case:* $L(\hat{h}; \mathcal{D}) \leq \alpha + O(\gamma)$ *for* $n = \widetilde{O}\left(V^{3/2}/\alpha^{5/2}\right)$*, where* $\gamma = \min_{h \in \mathcal{H}} L(h; \mathcal{D})$.

Our bounds are only a factor of $\widetilde{O}\left(\sqrt{V/\alpha}\right)$ worse than the corresponding optimal non-private bounds. In the agnostic case, however, we note that the accuracy of the output hypothesis in our case has a suboptimal dependency (by a small constant factor) on $\gamma \triangleq \min_{h \in \mathcal{H}} L(h; \mathcal{D})$.

We note that the same construction can serve as a private learner in a less restrictive setting where only the labels of the training set are considered private information. This setting is known as *label-private* learning, and it has been explored before in Chaudhuri & Hsu (2011) and Beimel et al. (2016). Both works have only considered pure, i.e., $(\epsilon, 0)$, differentially private learners, and their constructions are white-box, i.e., they do not allow for using a black-box non-private learner. The bounds in Chaudhuri & Hsu (2011) involve smoothness assumptions on the underlying distribution. In Beimel et al. (2016), an upper bound on the sample complexity is derived for the realizable case. Their bound is a factor of $O(1/\alpha)$ worse than the optimal non-private bound for the realizable case.

3.4 PRIVATE MATRIX COMPLETION

Collaborative filtering (or matrix completion) is a popular approach for modeling the recommendation system problem, where the goal is to provide personalized recommendations about certain items to a user (Koren & Bell (2015)). In other words, the objective of a personalized recommendation system is to learn the entire users-items preference matrix $Y^* \in \mathbb{R}^{m \times n}$ using a small number of user-item pref-

erences $Y^*_{ij}, (i,j) \in [m] \times [n]$, where m is the number of users and n is the number of items. Naturally, in absence of any structure in Y^*, the problem is ill-defined as the unknown entries of Y^* can be arbitrary. Hence, a popular modeling hypothesis is that the underlying preference matrix Y^* is low-rank, and thus, the collaborative filtering problem reduces to that of low-rank matrix completion (Recht (2011); Candes & Recht (2012)). One can also enhance this formulation using side-information like user-features or item-features (Yu et al. (2014)).

Naturally, personalization problems require collecting and analyzing sensitive customer data like their preferences for various items, which can lead to serious privacy breaches (Korolova (2010); Narayanan & Shmatikov (2010); Calandrino et al. (2011)). We attempt to address this problem of privacy-preserving recommendations using collaborative filtering (McSherry & Mironov (2009); Liu et al. (2015)). We answer the following question in the **affirmative**: *Can we design a matrix completion algorithm which keeps* **all** *the ratings of a user private, i.e., guarantees* **user-level privacy** *while still providing accurate recommendations?* In particular, we provide the *first* differentially private matrix completion algorithms with *provable* accuracy guarantees.

Most of the prior works on DP matrix completion and low-rank approximation (Blum et al. (2005); Chan et al. (2011); Hardt & Roth (2012, 2013); Kapralov & Talwar (2013); Dwork et al. (2014b)) have provided guarantees which are non-trivial only in the *entry-level* privacy setting, i.e., they preserve privacy of only a single rating of a user. Hence, they are not suitable for preserving a user's privacy in practical recommendation systems. In fact, their trivial extension to user-level privacy leads to vacuous bounds (see Table 3.1). Some works (McSherry & Mironov (2009); Liu et al. (2015)) do serve as an exception, and directly address the user-level privacy

problem. However, they only show empirical evidences of their effectiveness; they do not provide formal error bounds. In case of Liu et al. (2015), the DP guarantee itself might require an exponential amount of computation. In contrast, we provide an efficient algorithm based on the classic Frank-Wolfe (FW) procedure (Frank & Wolfe (1956)), and show that it gives strong utility guarantees while preserving user-level privacy. Furthermore, we empirically demonstrate its effectiveness on various benchmark datasets.

Our private FW procedure needs to compute the top right singular vector of a sparse user preference matrix, while preserving DP. For practical recommendation systems with a large number of items, this step turns out to be a significant bottleneck both in terms of space as well as time complexity. To alleviate this issue, we provide a method, based on the celebrated Oja's algorithm (Jain et al. (2016)), which is nearly optimal in terms of the accuracy of the computed singular vector while still providing significant improvement in terms of space and computation. In fact, our method can be used to speed-up even the vanilla differentially private PCA computation (Dwork et al. (2014c)). To the best of our knowledge, this is the first algorithm for DP singular value computation with optimal utility guarantee, that also exploits the sparsity of the underlying matrix.

Notion of privacy: In the context of matrix completion, where the goal is to release the *entire preference matrix* while preserving privacy, the standard notion of differential privacy (Definition 2.1.3) implies that the computed ratings/preferences for any particular user cannot depend strongly on *her own personal preferences*. Naturally, the resulting preference computation is going to be trivial and inaccurate (which also follows from the reconstruction attacks of Dinur & Nissim (2003) and Hardt & Roth (2012)).

To alleviate this concern, we consider a relaxed but natural DP notion (for recommendation systems) called *joint differential privacy* (Kearns et al. (2014)). Consider an algorithm $\mathcal{A}$ that produces individual outputs Y_i for each user i, i.e., the i-th row of preference matrix Y. Joint DP ensures that for each user i, the output of $\mathcal{A}$ for all other users (denoted by Y_{-i}) does not reveal "much" about the preferences of user i. That is, the recommendations made to all the users except the i-th user do not depend significantly upon the i-th user's preferences. Although not mentioned explicitly, previous works on DP matrix completion (McSherry & Mironov (2009); Liu et al. (2015)) strive to ensure Joint DP.

Granularity of privacy: DP protects the information about a user in the context of presence or absence of her data record. Prior works on DP matrix completion (McSherry & Mironov (2009); Liu et al. (2015)), and its close analogue, low-rank approximation (Blum et al. (2005); Chan et al. (2011); Hardt & Roth (2012); Dwork et al. (2014c); Hardt & Roth (2013)), have considered different variants of the notion of a data record. Some have considered a single entry in the matrix Y^* as a data record (resulting in *entry-level privacy*), whereas others have considered a more practical setting where the complete row is a data record (resulting in *user-level privacy*). Here, we present all our results in the strictly harder user-level privacy setting. To ensure a fair comparison, we present the results of prior works in the same setting.

3.4.1 Problem Definition: Matrix Completion

The goal of a low-rank matrix completion problem is to estimate a low-rank (or a convex relaxation of bounded nuclear norm) matrix $Y^* \in \mathbb{R}^{m \times n}$, having seen only a small number of entries from it. Here, m is the number of users, and n is the

number of items. Let $\Omega = \{(i,j) \subseteq [m] \times [n]\}$ be the index set of the observed entries from Y^*, and let $\mathsf{P}_\Omega : \mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n}$ be a matrix operator s.t. $\mathsf{P}_\Omega(Y)_{ij} = Y_{ij}$ if $(i,j) \in \Omega$, and 0 otherwise. Given, $\mathsf{P}_\Omega(Y^*)$, the objective is to output a matrix Y such that the following generalization error, i.e., the error in approximating a uniformly random entry from the matrix Y^*, is minimized:

$$L(Y) = \mathbb{E}_{(i,j)\sim_{\mathsf{unif}}[m]\times[n]}\left[\left(Y_{ij} - Y^*_{ij}\right)^2\right]. \tag{3.1}$$

Generalization error captures the ability of an algorithm to predict unseen samples from Y^*. We would want the generalization error to be $o(1)$ in terms of the problem parameters when $\Omega = o(mn)$. Throughout this section and Chapter 6, we will assume that $m > n$.

3.4.2 Main Contributions

We provide the first joint DP algorithm for low-rank matrix completion with formal non-trivial error bounds, which are summarized in Tables 3.1 and 3.2. At a high level, our key result can be summarized as follows:

Informal Theorem 3.4.1 (Corresponds to Corollary 6.2.1)**.** *Assume that each entry of a hidden matrix $Y^* \in \mathbb{R}^{m\times n}$ is in $[-1,1]$, and there are $\sqrt{n}$ observed entries per user. Also, assume that the nuclear norm of Y^* is bounded by $O(\sqrt{mn})$, i.e., Y^* has nearly constant rank. Then, there exist (ϵ,δ)-joint differentially private algorithms that have $o(1)$ generalization error as long as $m = \omega(n^{5/4})$.*

In other words, even with $\sqrt{n}$ observed ratings per user, we obtain asymptotically the correct estimation of each entry of Y^* on average, as long as m is large enough. The sample complexity bound dependence on m can be strengthened by

making additional assumptions, such as *incoherence*, on Y^*.

Our algorithm is based on two important ideas: a) using local and global computation, b) using the Frank-Wolfe method as a base optimization technique.

Local and global computation: The key idea that defines our algorithm, and allows us to get strong error bounds under joint DP is splitting the algorithm into two components: *global* and *local*. Recall each row of the hidden matrix Y^* belongs to an individual user. The global component of our algorithm computes statistics that are aggregate in nature (e.g., computing the correlation across columns of the revealed matrix $\mathsf{P}_\Omega(Y^*)$). On the other hand, the local component independently fine-tunes the statistics computed by the global component to generate accurate predictions for each user. Since the global component depends on the data of all users, adding noise to it (for privacy) does not significantly affect the accuracy of the predictions. McSherry & Mironov (2009); Liu et al. (2015) also exploit a similar idea of segregating the computation, but they do not utilize it formally to provide non-trivial error bounds.

Frank-Wolfe based method: We use the standard nuclear norm formulation (Recht (2011); Shalev-Shwartz et al. (2011); Tewari et al. (2011); Candes & Recht (2012)) for the matrix completion problem:

$$\min_{\|Y\|_{\mathsf{nuc}} \leq k} L(Y; \Omega), \tag{3.2}$$

where $L(Y;\Omega) = \frac{1}{2|\Omega|}\|\mathsf{P}_\Omega(Y - Y^*)\|_F^2$, the sum of singular values of Y is denoted by $\|Y\|_{\mathsf{nuc}}$, and the underlying hidden matrix Y^* is assumed to have nuclear norm of at most k. Note that we denote the empirical risk of a solution Y given the observed indices Ω by $L(Y;\Omega)$. We use the popular Frank-Wolfe algorithm (Frank & Wolfe (1956); Jaggi et al. (2010)) as our algorithmic building block. At a high-level, FW

computes the solution to (3.2) as a convex combination of rank-one matrices, each with nuclear norm at most k. These matrices are added iteratively to the solution.

Our main contribution is to design a version of the FW method that preserves Joint DP. That is, if the standard FW algorithm decides to add matrix $u \cdot v^T$ during an iteration, our private FW computes a noisy version of $v \in \mathbb{R}^n$ via its global component. Then, each user computes the respective element of $u \in \mathbb{R}^m$ to obtain her update. The noisy version of v suffices for the Joint DP guarantee, and allows us to provide the strong error bound in Theorem 3.4.1 above.

We want to emphasize that the choice of FW as the underlying matrix completion algorithm is critical for our system. FW updates via rank-one matrices in each step. Hence, the error due to noise addition in each step is small (i.e., proportional to the rank), and allows for an easy decomposition into the local-global computation model. Other standard techniques like proximal gradient descent based techniques (Cai et al. (2010); Lin et al. (2010)) can involve nearly *full-rank* updates in an iteration, and hence might incur large error, leading to arbitrary inaccurate solutions. Note that though a prior work (Talwar et al. (2015)) has proposed a DP Frank-Wolfe algorithm for high-dimensional regression, it was for a completely different problem in a different setting where the segregation of computation into global and local components was not necessary.

Private singular vector of sparse matrices using Oja's method: Our private FW requires computing a noisy covariance matrix which implies $\Omega(n^2)$ space/time complexity for n items. Naturally, such an algorithm does not scale to practical recommendation systems. In fact, this drawback exists even for standard private PCA techniques (Dwork et al. (2014c)). Using insights from the popular Oja's method, we provide a technique (see Algorithm 14) that has a linear dependency on n as

long as the number of ratings per user is small. Moreover, the performance of our private FW method isn't affected by using this technique.

SVD-based method: We also extend our technique to a singular value decomposition (SVD) based method for matrix completion/factorization. Our utility analysis shows that there are settings where this method outperforms our FW-based method, but in general it can provide a significantly worse solution. The main goal is to study the power of the simple SVD-based method, which is still a popular method for collaborative filtering.

Empirical results: Finally, we show that along with providing strong analytical guarantees, our private FW also performs well empirically. We show its efficacy on benchmark collaborative filtering datasets like Jester (Goldberg et al. (2001)), MovieLens (Harper & Konstan (2015)), the Netflix prize dataset (Bennett & Lanning (2007)), and the Yahoo! Music recommender dataset (Yahoo (2011)). Our algorithm consistently beats (in terms of accuracy) the existing state-of-the-art in DP matrix completion (SVD-based method by McSherry & Mironov (2009), and a variant of projected gradient descent (Cai et al. (2010); Bassily et al. (2014b); Abadi et al. (2016)).

3.4.3 Comparison to Prior Work

As discussed earlier, our results are the first to provide non-trivial error bounds for DP matrix completion. For comparing different results, we consider the following setting of the hidden matrix $Y^* \in \mathbb{R}^{m \times n}$ and the set of released entries Ω: i) $|\Omega| \approx m\sqrt{n}$, ii) each row of Y^* has an L_2 norm of $\sqrt{n}$, and iii) each row of $\mathsf{P}_\Omega(Y^*)$ has L_2-norm at most $n^{1/4}$, *i.e.*, $\approx \sqrt{n}$ random entries are revealed for each row. Furthermore, we assume the spectral norm of Y^* is at most $O(\sqrt{mn})$, and Y^* is

rank-one. These conditions are satisfied by a matrix $Y^* = u \cdot v^T$, where $\sqrt{n}$ random entries are observed *per user*, and $u_i, v_j \in [-1, 1]\ \forall i, j$.

Algorithm	**Bound on m**	**Bound on $\lvert\Omega\rvert$**
Nuclear norm min. (non-private) (Shalev-Shwartz et al. (2011))	$\omega(n)$	$\omega(m\sqrt{n})$
Noisy SVD + kNN (McSherry & Mironov (2009))	–	–
Noisy SGLD (Liu et al. (2015))	–	–
Private FW (Jain et al. (2018))	$\omega(n^{5/4})$	$\omega(m\sqrt{n})$

Table 3.1: Sample complexity bounds for matrix completion. $m =$ no. of users, $n =$ no. of items. The bounds hide privacy parameters ϵ and $\log(1/\delta)$, and polylog factors in m, n.

Algorithm	**Error**
Randomized response (Blum et al. (2005)) (Chan et al. (2011); Dwork et al. (2014b))	$O(\sqrt{m+n})$
Gaussian measurement (Hardt & Roth (2012))	$O\left(\sqrt{m} + \sqrt{\frac{\mu n}{m}}\right)$
Noisy power method (Hardt & Roth (2013))	$O(\sqrt{\mu})$
Exponential mechanism (Kapralov & Talwar (2013))	$O(m+n)$
Private FW (Jain et al. (2018))	$O\left(m^{3/10}n^{1/10}\right)$
Private SVD (Jain et al. (2018))	$O\left(\sqrt{\mu\left(\frac{n^2}{m} + \frac{m}{n}\right)}\right)$

Table 3.2: Error bounds ($\|Y - Y^*\|_F$) for low-rank approximation. $\mu \in [0, m]$ is the incoherence parameter (Definition 29). The bounds hide privacy parameters ϵ and $\log(1/\delta)$, and polylog factors in m an n. Rank of the output matrix Y_{priv} is $O\left(m^{2/5}/n^{1/5}\right)$ for Private FW, whereas it is $O(1)$ for the others.

In Table 3.1, we provide a comparison based on the sample complexity, i.e., the number of users m and the number observed samples $|\Omega|$ needed to attain a generalization error of $o(1)$. We compare our results with the best non-private algorithm for matrix completion based on nuclear norm minimization (Shalev-Shwartz et al. (2011)), and the prior work on DP matrix completion (McSherry & Mironov

(2009); Liu et al. (2015)). We see that for the same $|\Omega|$, the sample complexity on m increases from $\omega(n)$ to $\omega(n^{5/4})$ for our FW-based algorithm. While McSherry & Mironov (2009); Liu et al. (2015) work under the notion of Joint DP as well, they do not provide any formal accuracy guarantees.

Interlude: Low-rank approximation. We also compare our results with the prior work on a related problem of DP low-rank approximation. Given a matrix $Y^* \in \mathbb{R}^{m \times n}$, the goal is to compute a DP low-rank approximation Y_{priv}, s.t. Y_{priv} is close to Y^* either in the spectral or Frobenius norm. Notice that this is similar to matrix completion if the set of revealed entries Ω is the complete matrix. Hence, our methods can be applied directly. To be consistent with the existing literature, we assume that Y^* is rank-one matrix, and each row of Y^* has L_2-norm at most one. Table 3.2 compares the various results. While all the prior works provide trivial error bounds (in both Frobenius and spectral norm, as $\|Y^*\|_2 = \|Y^*\|_F \leq \sqrt{m}$), our methods provide non-trivial bounds. The key difference is that we ensure Joint DP (Definition 18), while existing methods ensure the stricter standard DP (Definition 2.1.3), with the exponential mechanism (Kapralov & Talwar (2013)) ensuring $(\epsilon, 0)$-standard DP.

CHAPTER 4

Private Convex Optimization

In this chapter, we will look at a technique for practical differentially private convex optimization. We will also look at an extensive empirical evaluation, which includes many high-dimensional publicly available benchmark datasets, corroborating that this technique performs well in practice.

4.1 ADDITIONAL PRELIMINARIES

Given an m-element dataset $D = \{d_1, d_2, \ldots, d_m\}$, s.t. $d_i \sim \mathcal{D}$ for $i \in [m]$, the objective is to get a model $\hat{\theta}$ from the following unconstrained optimization problem:

$$\hat{\theta} \in \arg\min_{\theta \in \mathbb{R}^n} L(\theta; D),$$

where $L(\theta; D) = \frac{1}{m} \sum_{i=1}^{m} \ell(\theta; d_i)$ is the empirical risk, $n > 0$, and $\ell(\theta; d_i)$ is defined as a loss function for d_i that is convex in the first parameter $\theta \in \mathbb{R}^n$. This formulation falls under the framework of ERM, which is useful in various settings, including the widely applicable problem of classification in machine learning via linear regression, logistic regression, or support vector machines. The notation $\|x\|$ is used to represent the L_2-norm of a vector x. Next, we define certain basic properties of functions that will be helpful in further sections.

Definition 4.1.1. *A function* $f : \mathbb{R}^n \to \mathbb{R}$:

- *is a* convex function *if for all* $\theta_1, \theta_2 \in \mathbb{R}^n$*, we have*

$$f(\theta_1) - f(\theta_2) \geq \langle \nabla f(\theta_2), \theta_1 - \theta_2 \rangle$$

- *is a* ξ-strongly convex function *if for all* $\theta_1, \theta_2 \in \mathbb{R}^n$*, we have*

$$f(\theta_1) \geq f(\theta_2) + \langle \nabla f(\theta_2), \theta_1 - \theta_2 \rangle + \frac{\xi}{2}\|\theta_1 - \theta_2\|^2$$

  or equivalently, $\langle \nabla f(\theta_1) - \nabla f(\theta_2), (\theta_1 - \theta_2) \rangle \geq \xi \|\theta_1 - \theta_2\|^2$

- *has* L_q-Lipschitz constant Δ *if for all* $\theta_1, \theta_2 \in \mathbb{R}^n$*, we have*

$$|f(\theta_1) - f(\theta_2)| \leq \Delta \cdot \|\theta_1 - \theta_2\|_q$$

- *is* β-smooth *if for all* $\theta_1, \theta_2 \in \mathbb{R}^n$*, we have*

$$\|\nabla f(\theta_1) - \nabla f(\theta_2)\| \leq \beta \cdot \|\theta_1 - \theta_2\|$$

Lastly, we define Generalized Linear Models (GLMs).

Definition 4.1.2 (Generalized Linear Model)**.** *For a model space* $\theta \in \mathbb{R}^n$*, where* $n > 0$*, the sample space* U *in a* Generalized Linear Model (GLM) *is defined as the cartesian product of a* n*-dimensional feature space* $\mathcal{X} \subseteq \mathbb{R}^n$ *and a label space* $\mathcal{Y}$*, i.e.,* $U = \mathcal{X} \times \mathcal{Y}$*. Thus, each data sample* $d_i \in U$ *can be decomposed into a feature vector* $x_i \in \mathcal{X}$*, and a label* $y_i \in \mathcal{Y}$*. Moreover, the loss function* $\ell(\theta; d_i)$ *for a GLM is a function of* $x_i^T\theta$ *and* y_i*.*

In this chapter, we will use the notion of neighboring under modification (Definition 2.1.1) for the guarantee of DP (Definition 2.1.3).

4.2 RELATED WORK

Convex optimization in the non-private setting has a long history; several excellent resources provide a good overview (Boyd & Vandenberghe (2004); Bubeck et al.

(2015)). A lot of recent advances have been made in the field of convex Empirical Risk Minimization (ERM) as well. A comprehensive list of works on *stochastic* convex ERM has been provided in Zhang et al. (2017), whereas Feldman (2016) provides dimension-dependent lower bounds for the sample complexity required for stochastic convex ERM and uniform convergence.

A large body of existing work examines the problem of differentially private convex ERM. The techniques of output perturbation and objective perturbation were first proposed in Chaudhuri et al. (2011). Near dimension-independent risk bounds for both the techniques were provided in Jain & Thakurta (2014); however, the bounds are achieved for the standard settings of the techniques, which provide privacy guarantees only for the minima of their respective objective functions. A private SGD algorithm was first given in Song et al. (2013), and optimal risk bounds were provided for a later version of private SGD in Bassily et al. (2014a). A variant of output perturbation was proposed in Wu et al. (2017) that requires the use of permutation-based SGD, and reduces sensitivity using properties of that algorithm. Several works (Kifer et al. (2012); Smith & Thakurta (2013)) deal with DP convex ERM in the setting of high-dimensional sparse regression, but the algorithms in these works also require obtaining the minima. The Frank-Wolfe algorithm (Frank & Wolfe (1956)) has also seen a resurgence lately (Jaggi (2013); Lacoste-Julien & Jaggi (2013); Lacoste-Julien & Jaggi (2015); Lacoste-Julien (2016)). We study the performance of a DP version of Frank-Wolfe (Talwar et al. (2014)) in our empirical analysis.

There are also works in DP convex optimization apart from the ERM model. Many recent works (Jain et al. (2012); Duchi et al. (2013); Thakurta & Smith (2013)) examine the setting of online learning, whereas high-dimensional kernel learning

is considered in Jain & Thakurta (2013); these settings are quite different from ours, and the results are incomparable. There have also been works (Zhang et al. (2012); Wu et al. (2015)) on DP regression analysis, a subset of DP convex optimization. However, the privacy guarantees in these hold only if the algorithms are able to find some minima. There have also been advances in DP non-convex optimization, including deep learning (Shokri & Shmatikov (2015); Abadi et al. (2016)). A broad survey of works in DP machine learning has been provided in Ji et al. (2014).

Previous empirical evaluations have provided limited insight into the practical performance of the various algorithms for DP convex optimization. Output perturbation and objective perturbation are evaluated on two datasets in Chaudhuri et al. (2011) and Jain & Thakurta (2014), and private SGD is evaluated in Song et al. (2013). Wu et al. (2017) perform the broadest comparison, including their own approach, and two variants of private SGD (Song et al. (2013); Bassily et al. (2014a)) on six datasets, but they do not include objective perturbation. No prior evaluation considers the state-of-the-art algorithms from all three major lines of work in the area (output perturbation, objective perturbation, and private SGD). Moreover, none of the prior evaluations considers high-dimensional data—a maximum of 75 dimensions is considered in Wu et al. (2017).

Our empirical evaluation is the most complete to date. We evaluate state-of-the-art algorithms from all 3 lines of work on 9 public datasets and 4 use cases. We consider low-dimensional and high-dimensional (as many as 47,236 dimensions) datasets. In addition, we release open-source implementations for all algorithms, and benchmarking scripts to reproduce our results (Iyengar et al. (2019a)).

4.3 APPROXIMATE MINIMA PERTURBATION

In this section, we will describe Approximate Minima Perturbation, a strengthened alternative to objective perturbation that provides DP guarantees in the case even when the output of the algorithm is not the actual minima of the perturbed objective function. The perturbed objective takes the form $L(\theta; D) + \frac{\Lambda}{2}\|\theta\|^2 + \langle b, \theta \rangle$, where b is a random variable drawn from an appropriate distribution, and Λ is an appropriately chosen regularization constant. We make two crucial improvements over the original objective perturbation algorithm (Chaudhuri et al. (2011); Kifer et al. (2012)):

- The privacy guarantee of objective perturbation holds only at the *exact* minima of the underlying optimization problem, which is *never guaranteed in practice given finite time*. We show that AMP provides a privacy guarantee even for an *approximate* solution.

- Earlier privacy analyses for objective perturbation (Chaudhuri et al. (2011); Kifer et al. (2012)) hold only when the loss function $\ell(\theta; d)$ is a loss for a GLM (see Definition 4.1.2), as they implicitly make a rank-one assumption on the Hessian of the loss $\bigtriangledown^2 \ell(\theta; d)$. Via a careful perturbation analysis of the Hessian, we extend the analysis to any convex loss function under standard assumptions. It is important to note that AMP reduces to objective perturbation if the "approximate" minima condition is tightened to getting the actual minima of the perturbed objective.

Algorithmic description: Given a dataset $D = \{d_1, d_2, \ldots, d_m\}$, where each $d_i \sim \mathcal{D}$, we consider (objective) functions of the form $L(\theta; D) = \frac{1}{m} \sum_{i=1}^{m} \ell(\theta; d_i)$, where $\theta \in \mathbb{R}^n$ is a model, loss $\ell(\theta; d_i)$ has L_2-Lipschitz constant Δ for all d_i, is convex in θ, has a

continuous Hessian, and is β-smooth in both the parameters.

At a high level, Approximate Minima Perturbation provides a convergence-based solution for objective perturbation. In other words, once the algorithm finds a model θ_{approx} for which the norm of the gradient of the perturbed objective $\nabla L_{priv}(\theta_{approx}; D)$ is within a pre-determined threshold γ, it outputs a noisy version of θ_{approx}, denoted by θ_{out}. Since the perturbed objective is strongly convex, it is sufficient to add Gaussian noise, with standard deviation σ_2 having a linear dependence on the norm bound γ, to θ_{approx} to ensure DP.

Details of AMP are provided in Algorithm 1. Note that although we get a relaxed constraint on the regularization parameter Λ (in Algorithm 1) if the loss function ℓ is a loss for a GLM, the privacy guarantees hold for general convex loss functions as well. The parameters (ϵ_1, δ_1) within the algorithm represent the amount of the privacy budget dedicated to perturbing the objective, with the rest of the budget (ϵ_2, δ_2) being used for adding noise to the approximate minima θ_{approx}. On the other hand, the parameter ϵ_3 intuitively represents the part of the privacy budget ϵ_1 allocated to scaling the noise added to the objective function. The remaining budget $(\epsilon_1 - \epsilon_3)$ is used to set the amount of regularization used.

Privacy and utility guarantees: Here, we provide the privacy and utility guarantees for Algorithm 1. While we provide a complete privacy analysis (Theorem 1), we only state the utility guarantee (Theorem 2) as it is a slight modification from previous work (Kifer et al. (2012)).

Theorem 1 (Privacy guarantee)**.** *Algorithm 1 is (ϵ, δ)-differentially private.*

Proof Idea. For obtaining an (ϵ, δ)-DP guarantee for Algorithm 1, we first split the output of the algorithm into two parts: one being the exact minima of the perturbed objective, whereas the other containing the exact minima, the approximate

Algorithm 1 Approximate Minima Perturbation

Input: Dataset: $D = \{d_1, \cdots, d_m\}$; loss function: $\ell(\theta; d_i)$ that has L_2-Lipschitz constant Δ, is convex in θ, has a continuous Hessian, and is β-smooth for all $\theta \in \mathbb{R}^n$ and all d_i; Hessian rank bound parameter: r which is the minimum of n and twice the upper bound on the rank of ℓ's Hessian; privacy parameters: (ϵ, δ); gradient norm bound: γ.

1: Set $\epsilon_1, \epsilon_2, \epsilon_3, \delta_1, \delta_2 > 0$ such that $\epsilon = \epsilon_1 + \epsilon_2$, $\delta = \delta_1 + \delta_2$, and $0 < \epsilon_1 - \epsilon_3 < 1$

2: Set $\Lambda \geq \frac{r\beta}{\epsilon_1 - \epsilon_3}$

3: $b_1 \sim \mathcal{N}(0, \sigma_1^2 I_{n\times n})$, where $\sigma_1 = \frac{\left(\frac{2\Delta}{m}\right)\left(1+\sqrt{2\log\frac{1}{\delta_1}}\right)}{\epsilon_3}$

4: Let $L_{priv}(\theta; D) = \frac{1}{m}\sum_{i=1}^{m} \ell(\theta; D_i) + \frac{\Lambda}{2m}\|\theta\|^2 + b_1^T\theta$

5: $\theta_{approx} \leftarrow \theta$ such that $\|\nabla L_{priv}(\theta; D)\| \leq \gamma$

6: $b_2 \sim \mathcal{N}(0, \sigma_2^2 I_{n\times n})$, where $\sigma_2 = \frac{\left(\frac{m\gamma}{\Lambda}\right)\left(1+\sqrt{2\log\frac{1}{\delta_2}}\right)}{\epsilon_2}$

7: Output $\theta_{out} = \theta_{approx} + b_2$

minima obtained in Step 1 of the algorithm, as well as the Gaussian noise added to it. For the first part, we bound the ratio of the density of the exact minima taking any particular value, under any two neighboring datasets, by e^{ϵ_1} with probability at least $1 - \delta_1$. We first simplify such a ratio, as done in Chaudhuri et al. (2011) via the function inverse theorem, by transforming it to two ratios: one involving only the density of a function of the minima value and the input dataset, and the other involving the determinant of this function's Jacobian. For the former ratio, we start by bounding the sensitivity of the function using the L_2-Lipschitz constant Δ of the loss function. Then, we use the guarantees of the Gaussian mechanism to obtain a high-probability bound (shown in Lemma 4.3.1). We bound the latter ratio (in Lemma 4.3.2) via a novel approach that uses the β-smoothness property of the loss. Next, we use the the gradient norm bound γ, and the strong convexity of the perturbed objective to obtain an (ϵ_2, δ_2)-DP guarantee for the second part of the split output. Lastly, we use the basic composition property of DP (Lemma 2.1.5) to get the statement of the theorem. □

Proof. Define $\theta_{min} = \arg\min_{\theta \in \mathbb{R}^n} L_{priv}(\theta; D)$. Fix a pair of neighboring datasets $D^*, D' \in \mathcal{D}^m$, and some $\alpha \in \mathbb{R}^n$. First, we will show that:

$$\frac{\text{pdf}_{D^*}(\theta_{min} = \alpha)}{\text{pdf}_{D'}(\theta_{min} = \alpha)} \leq e^{\epsilon_1} \text{ w.p. } \geq 1 - \delta_1. \tag{4.1}$$

We define $b(\theta; D) = -\nabla L(\theta; D) - \frac{\Lambda\theta}{m}$ for $D \in \mathcal{D}^m$ and $\theta \in \mathbb{R}^n$. Changing variables according to the function inverse theorem (Theorem 17.2 in Billingsley (1995)), we get

$$\frac{\text{pdf}_{D^*}(\theta_{min} = \alpha)}{\text{pdf}_{D'}(\theta_{min} = \alpha)} = \frac{\text{pdf}(b(\alpha; D^*); \epsilon_1, \delta_1, \Delta)}{\text{pdf}(b(\alpha; D'); \epsilon_1, \delta_1, \Delta)} \cdot \frac{|\det(\nabla b(\alpha; D'))|}{|\det(\nabla b(\alpha; D^*))|} \tag{4.2}$$

We will bound the ratios of the densities and the determinants separately. First, we will show that for $\epsilon_3 < \epsilon_1$, $\frac{\text{pdf}(b(\alpha;D^*);\epsilon,\delta,\Delta)}{\text{pdf}(b(\alpha;D');\epsilon,\delta,\Delta)} \leq e^{\epsilon_3}$ w.p. at least $1 - \delta_1$, and then we will show that $\frac{|\det(\nabla b(\alpha;D'))|}{|\det(\nabla b(\alpha;D^*))|} \leq e^{\epsilon_1 - \epsilon_3}$ if $\epsilon_1 - \epsilon_3 < 1$.

Lemma 4.3.1. *We define $b(\theta; D) = -\nabla L(\theta; D) - \frac{\Lambda\theta}{m}$ for $D \in \mathcal{D}^m$, and $\theta \in \mathbb{R}^n$. Then, for any pair of neighboring datasets $D^*, D' \in \mathcal{D}^m$, and $\epsilon_3 < \epsilon_1$, we have*

$$\frac{pdf(b(\alpha; D^*); \epsilon_1, \delta_1, \Delta)}{pdf(b(\alpha; D'); \epsilon_1, \delta_1, \Delta)} \leq e^{\epsilon_3} \text{ w.p. at least } 1 - \delta_1.$$

Here, $L_{priv}(\alpha;)$ is defined as in Algorithm 1.

Proof. Assume w.l.o.g. that $d_i \in D^*$ has been replaced by d_i' in D'. We first bound the L_2-sensitivity of $b(\alpha;)$:

$$\|b(\alpha; D^*) - b(\alpha; D')\| \leq \frac{\|\nabla \ell(\alpha; d_i') - \nabla \ell(\alpha; d_i)\|}{m} \leq \frac{2\Delta}{m},$$

where the last inequality follows as $\|\nabla \ell(\alpha;)\| \leq \Delta$.

Setting $\sigma_1 \geq \frac{\left(\frac{2\Delta}{m}\right)\left(1+\sqrt{2\log\frac{1}{\delta_1}}\right)}{\epsilon_3}$ for $\epsilon_3 < \epsilon_1$, we get the statement of the lemma from

the guarantees of the Gaussian mechanism (Lemma 2.1.9). □

Lemma 4.3.2. *Let $b(\theta; D)$ be defined as in Lemma 4.3.1. Then for any pair of neighboring datasets $D^*, D' \in \mathcal{D}^m$, if $\epsilon_1 - \epsilon_3 < 1$, we have*

$$\frac{|\det(\nabla b(\alpha; D'))|}{|\det(\nabla b(\alpha; D^*))|} \leq e^{\epsilon_1 - \epsilon_3}.$$

Proof. Assume w.l.o.g. that $d_i \in D^*$ is replaced by d_i' in D'. Let $A = m\nabla^2 L(\alpha; D^*)$, and $E = \nabla^2 \ell(\alpha; d_i') - \nabla^2 \ell(\alpha; d_i)$. As the $(n \times n)$ matrix E is the difference of the Hessians of the loss of two individual samples, we can define a bound r on the rank of E as follows:

$$r = \min\left\{n, 2 \cdot \left(\text{upper bound on rank of } \nabla^2 \ell(\alpha;)\right)\right\}$$

Let $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ be the eigenvalues of A, and $\lambda_1' \leq \lambda_2' \leq \cdots \leq \lambda_n'$ be the eigenvalues of $A + E$. Thus,

$$\begin{aligned}
\frac{|\det(\nabla b(\alpha; D'))|}{|\det(\nabla b(\alpha; D^*))|} &= \frac{\det\left(\frac{A+E+\Lambda\mathbb{I}_n}{m}\right)}{\det\left(\frac{A+\Lambda\mathbb{I}_n}{m}\right)} = \frac{\prod_{i=1}^{n}(\lambda_i' + \Lambda)}{\prod_{i=1}^{n}(\lambda_i + \Lambda)} \\
&= \prod_{i=1}^{n}\left(1 + \frac{\lambda_i' - \lambda_i}{\lambda_i + \Lambda}\right) \leq \prod_{i=1}^{n}\left(1 + \frac{|\lambda_i' - \lambda_i|}{\Lambda}\right) \\
&= 1 + \sum_{i=1}^{n}\frac{|\lambda_i' - \lambda_i|}{\Lambda} + \sum_{\substack{i,j\in[n],\\ i\neq j}}\frac{\prod_{k\in\{i,j\}}|\lambda_k' - \lambda_k|}{\Lambda^2} + \cdots \\
&\leq 1 + \frac{r\beta}{\Lambda} + \frac{(r\beta)^2}{\Lambda^2} + \cdots \leq \frac{\Lambda}{\Lambda - r\beta}
\end{aligned}$$

The first inequality follows since A is a positive semi-definite matrix (as ℓ is con-

vex) and thus, $\lambda_j \geq 0$ for all $j \in [n]$. The second inequality follows as i) the rank of E is at most r, ii) both A and $A+E$ are positive semi-definite (so $\lambda_j, \lambda'_j \geq 0$ for all $j \in [n]$), and iii) we have an upper bound β on the eigenvalues of A and $A+E$ due to $\ell(\theta; d_j)$ being convex in θ, having a continuous Hessian, and being β-smooth. The last inequality follows if $\Lambda > r\beta$. Also, we want $\frac{\Lambda}{\Lambda - r\beta} \leq exp(\epsilon_1 - \epsilon_3)$, which implies $\Lambda \geq \frac{r\beta}{1-exp(\epsilon_3-\epsilon_1)} \geq \frac{r\beta}{\epsilon_1-\epsilon_3}$. Both conditions are satisfied by setting $\Lambda = \frac{r\beta}{\epsilon_1-\epsilon_3}$ as $\epsilon_1 - \epsilon_3 < 1$. □

From Equation 4.2, and Lemmas 4.3.1 and 4.3.2, we get that $\frac{\text{pdf}_{D^*}(\theta_{min}=\alpha)}{\text{pdf}_{D'}(\theta_{min}=\alpha)} \leq e^{\epsilon_1}$ w.p. $\geq 1 - \delta_1$. In other words, θ_{min} is (ϵ_1, δ_1)-differentially private.

Now, since we can write θ_{out} as $\theta_{out} = \theta_{min} + (\theta_{approx} - \theta_{min} + b_2)$, we will prove that releasing $(\theta_{approx} - \theta_{min} + b_2)$ is (ϵ_2, δ_2)-differentially private.

Lemma 4.3.3. *For $D \in \mathcal{D}^m$, let $\gamma \geq 0$ be chosen independent of D, and let $\theta_{min} = \arg\min_{\theta \in \mathbb{R}^n} L_{priv}(\theta; D)$. If $\theta_{approx} \in \mathbb{R}^n$ s.t. $\|\nabla L_{priv}(\theta_{approx}; D)\| \leq \gamma$, then releasing $(\theta_{approx} - \theta_{min} + b_2)$, where $b_2 \sim \mathcal{N}(0, \sigma_2^2 I_{n\times n})$ for $\sigma_2 = \frac{\left(\frac{m\gamma}{\Lambda}\right)\left(1+\sqrt{2\log\frac{1}{\delta_2}}\right)}{\epsilon_2}$, is (ϵ_2, δ_2)-DP.*

Proof. We start by bounding the L_2-norm of $(\theta_{approx} - \theta_{min})$:

$$\|\theta_{approx} - \theta_{min}\| \leq \frac{m\|\nabla L_{priv}(\theta_{approx}; D) - \nabla L_{priv}(\theta_{min}; D)\|}{\Lambda} \leq \frac{m\gamma}{\Lambda} \tag{4.3}$$

The first inequality follows as L_{priv} is $\frac{\Lambda}{m}$-strongly convex, and the second inequality follows as $\nabla L_{priv}(\theta_{min}; D) = 0$ and $\nabla L_{priv}(\theta_{approx}; D) \leq \gamma$. Now, setting $\sigma_2 = \frac{\left(\frac{m\gamma}{\Lambda}\right)\left(1+\sqrt{2\log\frac{1}{\delta_2}}\right)}{\epsilon_2}$, we get the statement of the lemma by the properties of the Gaussian mechanism (Lemma 2.1.9). □

As $\epsilon_1 + \epsilon_2 = \epsilon$, and $\delta_1 + \delta_2 = \delta$, we get the privacy guarantee of Algorithm 1 by Equation 4.1, Lemma 4.3.3, and basic composition of DP (Lemma 2.1.5). □

Next, we provide the utility guarantee (in terms of excess empirical risk) for Algorithm 1 in Theorem 2.

Theorem 2 (Utility guarantee (adapted from Kifer et al. (2012))). *Let $\hat{\theta}$ be the true unconstrained minimizer of $L(\theta; D) = \frac{1}{m}\sum_{i=1}^{m} \ell(\theta; d_i)$, and $r = \min\{n, 2R_{\ell_H}\}$ where R_{ℓ_H} is an upper bound on the rank of the Hessian of ℓ. In Algorithm 1, if $\epsilon_i = \frac{\epsilon}{2}$ for $i \in \{1, 2\}$, $\epsilon_3 = \max\left\{\frac{\epsilon_1}{2}, \epsilon_1 - 0.99\right\}$, $\delta_j = \frac{\delta}{2}$ for $j \in \{1, 2\}$, and we set the regularization parameter $\Lambda = \Theta\left(\frac{1}{\|\hat{\theta}\|}\left(\frac{\Delta\sqrt{rn\log 1/\delta}}{\epsilon} + \sqrt{\frac{m^2\Delta\gamma\sqrt{n\log 1/\delta}}{\epsilon}}\right)\right)$ such that it satisfies the constraint in Step 1, then the following is true:*

$$\mathbb{E}\left(L(\theta_{out}; D) - L(\hat{\theta}; D)\right) = O\left(\frac{\|\hat{\theta}\|\Delta\sqrt{rn\log\frac{1}{\delta}}}{m\epsilon} + \|\hat{\theta}\|\sqrt{\frac{\Delta\gamma\sqrt{n\log\frac{1}{\delta}}}{\epsilon}}\right).$$

Proof. For bounding the expected risk of the algorithm, we first need to bound its empirical risk (Lemma 4.3.4).

Lemma 4.3.4 (Empirical Risk). *Let $\hat{\theta}$ be the minimizer of $L(\theta; D) = \frac{1}{m}\sum_{i=1}^{m} \ell(\theta; d_i)$, and θ_{min} be the minimizer of $L_{priv}(\theta; D) = L(\theta; D) + \frac{\Lambda}{2m}\|\theta\|^2 + b_1^T\theta$, where b_1 is as defined in Algorithm 1. Also, let θ_{out} be the output of Algorithm 1. We have:*

$$L(\theta_{out}; D) - L(\hat{\theta}; D) \leq \Delta\left(\frac{m\gamma}{\Lambda} + \|b_2\|\right) + \frac{\Lambda\|\hat{\theta}\|^2}{2m} + \frac{2n\|b_1\|^2}{\Lambda}.$$

Proof. We have

$$L(\theta_{out}; D) - L(\hat{\theta}; D) = (L(\theta_{out}; D) - L(\theta_{min}; D)) + \left(L(\theta_{min}; D) - L(\hat{\theta}; D)\right) \quad (4.4)$$

First, we will bound $(L(\theta_{out}; D) - L(\theta_{min}; D))$. We have:

$$\begin{aligned} L(\theta_{out}; D) - L(\theta_{min}; D) &\leq |L(\theta_{out}; D) - L(\theta_{min}; D)| \leq \Delta \|\theta_{out} - \theta_{min}\| \\ &= \Delta \|\theta_{approx} - \theta_{min} + b_2\| \leq \Delta \|\theta_{approx} - \theta_{min}\| + \Delta \|b_2\| \\ &\leq \Delta \left(\frac{m\gamma}{\Lambda} + \|b_2\| \right) \end{aligned} \tag{4.5}$$

The second inequality above follows from the Lipschitz property of $L(; D)$. The first equality follows as $\theta_{out} = \theta_{min} + (\theta_{approx} - \theta_{min} + b_2)$, whereas the last inequality follows from inequality 4.3.

Next, we bound $\left(L(\theta_{min}; D) - L(\hat{\theta}; D) \right)$ on the lines of the proof of Lemma 3 in Kifer et al. (2012). Let $\theta^{\#} = \arg\min_{\theta \in \mathbb{R}^p} L^{\#}(\theta; D)$, where $L^{\#}(\theta; D) = L(\theta; D) + \frac{\Lambda}{2n}\|\theta\|^2$. As a result, $L_{priv}(\theta; D) = L^{\#}(\theta; D) + b_1^T \theta$. So, we have:

$$\begin{aligned} L(\theta_{min}; D) - L(\hat{\theta}; D) &= L^{\#}(\theta_{min}; D) - L^{\#}(\theta^{\#}; D) + L^{\#}(\theta^{\#}; D) - L^{\#}(\hat{\theta}; D) \\ &\quad + \frac{\Lambda \|\hat{\theta}\|^2}{2m} - \frac{\Lambda \|\theta_{min}\|^2}{2m} \\ &\leq L^{\#}(\theta_{min}; D) - L^{\#}(\theta^{\#}; D) + \frac{\Lambda \|\hat{\theta}\|^2}{2m} \end{aligned} \tag{4.6}$$

The inequality above follows as $L^{\#}(\theta^{\#}; D) \leq L^{\#}(\hat{\theta}; D)$.

Let us now bound $L^{\#}(\theta_{min}; D) - L^{\#}(\theta^{\#}; D)$. To this end, we first observe that since L_{priv} is $\frac{\Lambda}{m}$-strongly convex in θ, we have that

$$\begin{aligned} L_{priv}(\theta^{\#}; D) &\geq L_{priv}(\theta_{min}; D) - \nabla L_{priv}(\theta_{min}; D)^T (\theta_{min} - \theta^{\#}) + \frac{\Lambda}{2m} \|\theta^{\#} - \theta_{min}\|^2 \\ &= L_{priv}(\theta_{min}; D) + \frac{\Lambda}{2m} \|\theta^{\#} - \theta_{min}\|^2 \end{aligned} \tag{4.7}$$

The equality above follows as $\|\nabla L_{priv}(\theta_{min}; D)\| = 0$.

Substituting the definition of $L_{priv}(;D)$ in equality 4.7, we get that

$$L^{\#}(\theta_{min};D) - L^{\#}(\theta^{\#};D) \leq b_1^T(\theta^{\#} - \theta_{min}) - \frac{\Lambda}{2m}\|\theta^{\#} - \theta_{min}\|^2 \tag{4.8}$$

$$\leq \|b_1\| \cdot \|\theta^{\#} - \theta_{min}\| \tag{4.9}$$

Inequality 4.9 above follows by the Cauchy–Schwarz inequality.

Now, since $L^{\#}(\theta_{min};D) - L^{\#}(\theta^{\#};D) \geq 0$, it follows from inequalities 4.8 and 4.9 that

$$\|b_1\| \cdot \|\theta^{\#} - \theta_{min}\| \geq \frac{\Lambda}{2m}\|\theta^{\#} - \theta_{min}\|^2$$

$$\Rightarrow \|\theta^{\#} - \theta_{min}\| \leq \frac{2m\|b_1\|}{\Lambda} \tag{4.10}$$

We get the statement of the lemma from equation 4.4, and inequalities 4.5, 4.6, 4.9, and 4.10. □

Now, we are ready to prove Theorem 2. The proof is on the lines of the proof of Theorem 4 in Kifer et al. (2012). First, let us get a high probability bound on $L(\theta_{out};D) - L(\hat{\theta};D)$. To this end, we will first bound $\|b_1\|$ and $\|b_2\|$ w.h.p., where $b_s \sim \mathcal{N}(0,\sigma_s^2 I_{n\times n})$ for $s \in \{1,2\}$. Using Lemma 2 from Dasgupta & Schulman (2007), we get that w.p. $\geq 1 - \frac{\alpha}{2}$,

$$\|b_s\| \leq \sigma_s\sqrt{2n\log\frac{2}{\alpha}}.$$

Substituting this into Lemma 4.3.4, we get that w.p. $\geq 1 - \alpha$,

$$L(\theta_{out};D) - L(\hat{\theta};D) \leq \Delta\left(\frac{m\gamma}{\Lambda} + \sigma_2\sqrt{2n\log\frac{2}{\alpha}}\right) + \frac{\Lambda\|\hat{\theta}\|^2}{2m} + \frac{4m\sigma_1^2 n\log\frac{2}{\alpha}}{\Lambda}.$$

It is easy to see that by setting $\epsilon_i = \frac{\epsilon}{2}$ for $i \in \{1, 2\}$, $\epsilon_3 = \max\left\{\frac{\epsilon_1}{2}, \epsilon_1 - 0.99\right\}$, $\delta_j = \frac{\delta}{2}$ for $j \in \{1, 2\}$, and $\Lambda = \Theta\left(\frac{\Delta\sqrt{rn\log 1/\delta}}{\epsilon\|\hat{\theta}\|} + \frac{m}{\|\hat{\theta}\|}\sqrt{\frac{\Delta\gamma\sqrt{n\log 1/\delta}}{\epsilon}}\right)$ such that it satisfies the constraint in Step 1 in Algorithm 1, we get the statement of the theorem. □

Remark: For loss functions of Generalized Linear Models, we have $r = 2$. Here, for small values of γ (for example, $\gamma = O\left(\frac{1}{m^2}\right)$), the excess empirical risk of Approximate Minima Perturbation is asymptotically the same as that of objective perturbation (Kifer et al. (2012)), and has a better dependence on m than that of Private Permutation-based SGD (Wu et al. (2017)). Specifically, the dependence is $\propto \frac{1}{m}$ for Approximate Minima Perturbation, and $\propto \frac{1}{\sqrt{m}}$ for Private PSGD.

Towards Hyperparameter-free Approximate Minima Perturbation: AMP can be considered to have the following hyperparameters: the Lipschitz constant Δ, and the privacy parameters ϵ_2, δ_2, and ϵ_3 which split the privacy budget within the algorithm. A data-independent approach for setting these parameters can eliminate the need for hyperparameter tuning with this approach, making it convenient to deploy in practice.

For practical applications, given a sensitive dataset and a convex loss function, the Δ hyperparameter can be thought of as a trade-off between the sensitivity of the loss and the amount of external interference required to achieve that sensitivity, for instance, *sample clipping* (defined in Section 4.4.1) on the data. In the next section, we provide a hyperparameter-free variant of AMP that has performance comparable to the standard variant in which all the hyperparameters are tuned.

4.4 EXPERIMENTAL RESULTS

Our evaluation seeks to answer two major research questions:

1. **What is the cost (to accuracy) of privacy?** How close can a DP model come to the non-private baseline? For industrial use cases, is the cost of privacy low enough to make DP learning practical?

2. **Which algorithm provides the best accuracy in practice?** Is there a total order on the available algorithms? Does this ordering differ for datasets with different properties?

Additionally, we also attempt to answer the following question which can result in a significant advantage for the deployment of a DP model in practice:

3. **Can Approximate Minima Perturbation be deployed without hyperparameter tuning?** Can its hyperparameters ($\Delta, \epsilon_2, \delta_2$, and ϵ_3) be set in a data-independent manner?

Summary of results: *Question #1:* Our results demonstrate that for datasets of sufficient size, *the cost of privacy is negligible*. Experiments 1 (on low-dimensional datasets), 2 (on high-dimensional datasets), and 3 (on datasets obtained in collaboration with Uber) evaluate the cost of privacy using logistic loss. Our results show that for large datasets, a DP model exists that approaches the accuracy of the non-private baseline at reasonable privacy budgets. Experiment 3 shows that for the larger datasets common in practical settings, a privacy-preserving model can produce even *better* accuracy than the non-private one, which suggests that privacy-preserving learning is indeed practical.

We also present the performance of private algorithms using Huber SVM loss (on all the datasets mentioned above) in Section 4.4.6. The general trends from the experiments using Huber SVM loss are identical to those obtained using logistic loss.

Question #2: Our experiments demonstrate that AMP generally provides the best accuracy among all the evaluated algorithms. Moreover, experiment 2 shows that under specific conditions, private Frank-Wolfe can provide the best accuracy. In all the regimes, the results generally show an improvement over other approaches.

Question #3: Our experiments also demonstrate that a simple data-independent method can be used to set $\Delta, \epsilon_2, \delta_2$, and ϵ_3 for AMP, and that this method provides good accuracy across datasets. For most values of ϵ, our data-independent approach provides nearly the same accuracy as the version tuned using a grid search (which may be non-private).

4.4.1 Experiment Setup

Algorithms evaluated: Our evaluation includes one algorithm drawn from each of the major approaches to private convex optimization: objective perturbation, output perturbation, private gradient descent, and the private Frank-Wolfe algorithm. For each approach, we select the best-known algorithm and configuration.

For objective perturbation, we implement AMP (Algorithm 1) as it is the only practically feasible objective perturbation approach; for all variants pertaining to the standard regime in Chaudhuri et al. (2011), obtaining some exact minima is necessary for achieving a privacy guarantee. For all the experiments, we tune the value of the hyperparameters $\Delta, \epsilon_2, \delta_2$, and ϵ_3 using the grid search described later.

We also evaluate a hyperparameter-free variant of AMP that sets the hyperparameters $\Delta, \epsilon_2, \delta_2$ and ϵ_3 independent of the data. We describe the strategy in detail towards the end of this subsection.

For output perturbation, we implement Private Perturbation-based SGD (Wu

et al. (2017)) (PSGD), as it is the only practically feasible variant of output perturbation; for all variants pertaining to the standard regime in Chaudhuri et al. (2011), obtaining some exact minima is necessary for achieving a privacy guarantee. We evaluate both the variants, with minibatching, proposed in Wu et al. (2017): convex (Algorithm 2), and strongly convex (Algorithm 3). For the convex variant, we evaluate all three proposed learning rate schemes (constant learning rate, decreasing learning rate, and square-root learning rate). We include results only for constant learning rate, as our experiments show that this scheme produces the most accurate models.

Algorithm 2 Differentially Private Permutation-based Stochastic Gradient Descent (Wu et al. (2017))

Input: Data set: $D = \{d_1, \cdots, d_m\}$, loss function: $\ell(\theta; d_i)$ with L_2-Lipschitz constant Δ, privacy parameters: (ϵ, δ), number of passes: T, minibatch size: k, constant learning rate: η.

$\theta \leftarrow 0^n$
Let τ be a random permutation of $[m]$
for $t = 1$ to $T - 1$ **do**
  for $b = 1$ to $\frac{m}{k}$ **do**
    Let $s_1 = d_{\tau(bk)}, \cdots, s_k = d_{\tau(b(k+1)-1)}$
    $\theta \leftarrow \theta - \eta(\frac{1}{k}\sum_{i=1}^{k} \nabla\ell(\theta; s_i))$
$\sigma^2 \leftarrow \frac{8T^2\Delta^2\eta^2\log(\frac{2}{\delta})}{k^2\epsilon^2}$
$b \sim \mathcal{N}(0, \sigma^2 I_{n\times n})$
Output $\theta_{priv} = \theta + b$

For private gradient descent, we implement a variant of the private SGD algorithm originally proposed in Bassily et al. (2014a). Our variant (Algorithm 4) leverages the Moments Accountant (Abadi et al. (2016)), incorporates minibatching, and sets the noise parameter based on the desired number of iterations (as compared to a fixed m^2 iterations in Bassily et al. (2014a)).

For private Frank-Wolfe, we implement the version (Algorithm 5) originally

Algorithm 3 Differentially Private Strongly Convex Permutation-based Stochastic Gradient Descent (Wu et al. (2017))

Input: Data set: $D = \{d_1, \cdots, d_m\}$, loss function: $\ell(\theta; d_i)$ that is ξ-strongly convex and β-smooth with L_2-Lipschitz constant Δ, privacy parameters: (ϵ, δ), number of passes: T, minibatch size: k.

$\theta \leftarrow 0^n$
Let τ be a random permutation of $[m]$
for $t = 1$ to $T - 1$ **do**
  $\eta_t \leftarrow \min \frac{1}{\beta}, \frac{1}{\xi t}$
  for $b = 1$ to $\frac{m}{k}$ **do**
    Let $s_1 = d_{\tau(bk)}, \cdots, s_k = d_{\tau(b(k+1)-1)}$
    $\theta \leftarrow \theta - \eta_t(\frac{1}{k}\sum_{i=1}^{k} \nabla \ell(\theta; s_i))$
$\sigma^2 \leftarrow \frac{8\Delta^2 \log(\frac{2}{\delta})}{\xi^2 m^2 \epsilon^2}$
$b \sim \mathcal{N}(0, \sigma^2 I_{n \times n})$
Output $\theta_{priv} = \theta + b$

proposed in Talwar et al. (2014). This algorithm performs constrained optimization by design. Following the advice of Jaggi (2013), we use a decreasing learning rate for better accuracy guarantees. Unlike the other algorithms, private Frank-Wolfe has nearly dimension-independent error bounds, so it should be expected to perform comparatively better on high-dimensional datasets.

For each algorithm, we evaluate the variant that provides (ϵ, δ)-differential privacy. Most algorithms can also provide ϵ-differential privacy at an additional cost to accuracy.

Datasets: Table 4.1 lists the public datasets used in our experimental evaluation. Each of these datasets is available for download, and our open-source release contains scripts for downloading and pre-processing these datasets. It also contains scripts for generating both the synthetic datasets. As RCV1 has multi-label classification over 103 labels (with most of the labels being used for a very small proportion of the dataset), for this dataset we consider the task of predicting whether a sample is categorized under the most frequently used label or not.

Algorithm 4 Differentially Private Minibatch Stochastic Gradient Descent (Bassily et al. (2014a); Abadi et al. (2016))

Input: Data set: $D = \{d_1, \cdots, d_m\}$, loss function: $\ell(\theta; d_i)$ with L_2-Lipschitz constant Δ, privacy parameters: (ϵ, δ), number of iterations: T, minibatch size: k, learning rate function: $\eta : [T] \to \mathbb{R}$.

1: $\sigma^2 \leftarrow \frac{16\Delta^2 T \log \frac{1}{\delta}}{m^2\epsilon^2}$
2: $\theta_1 = 0^n$
3: **for** $t = 1$ to $T - 1$ **do**
4: $\quad s_1, \cdots, s_k \leftarrow$ Sample k samples uniformly with replacement from D
5: $\quad b_t \sim \mathcal{N}(0, \sigma^2 I_{n\times n})$
6: $\quad \theta_{t+1} = \theta_t - \eta(t)[(\frac{1}{k}\sum_{i=1}^{k} \nabla\ell(\theta; s_i)) + b_t]$
7: Output θ_T

Algorithm 5 Differentially Private Frank-Wolfe (Talwar et al. (2015))

Input: Data set: $D = \{d_1, \cdots, d_m\}$, loss function: $L(\theta; D) = \frac{1}{n}\sum_{i=1}^{n} \ell(\theta; d_i)$ (with L_1-Lipshitz constant Δ for ℓ), privacy parameters: (ϵ, δ), convex set: $C = conv(S)$ with $\|C\|_1$ denoting $max_{s\in S}\|s\|_1$ and S being the set of corners.

1: Choose an arbitrary θ_1 from C
2: $\sigma^2 \leftarrow \frac{32\Delta^2\|C\|_1^2 T log(1/\delta)}{m^2\epsilon^2}$
3: **for** $t = 1$ to $T - 1$ **do**
4: $\quad \forall s \in S, \alpha_s \leftarrow \langle s, \bigtriangledown L(\theta_t; D)\rangle + Lap(\sigma)$, where $Lap(\lambda) \sim \frac{1}{2\lambda}e^{-|x|/\lambda}$
5: $\quad \widetilde{\theta}_t \leftarrow \arg\min_{s\in S} \alpha_s$
6: $\quad \theta_{t+1} \leftarrow (1 - \eta_t)\theta_t + \eta\widetilde{\theta}_t$, where $\eta_t = \frac{1}{t+1}$
7: Output $\theta^{priv} = \theta_T$

The selected datasets include datasets from 2 categories, *low-dimensional*, and *high-dimensional*. We define low-dimensional datasets to be ones where $m \gg n$ (m is the number of samples and n is the number of dimensions). High-dimensional datasets are defined as those for which m and n are on roughly the same scale, i.e. $m \leq n$ (or nearly so). We consider the Synthetic-H, Gisette, Real-sim, and RCV1 datasets to be high-dimensional.

To obtain training and testing sets, we randomly shuffle the dataset, take the first 80% as the training set, and the remaining 20% as the testing set.

Table 4.1: Datasets used in our evaluation

Dataset	# Samples	# Dim.	# Classes
Low-Dimensional Datasets (Public)			
Synthetic-L	10,000	20	2
Adult	45,220	104	2
KDDCup99	70,000	114	2
Covertype	581,012	54	7
MNIST	65,000	784	10
High-Dimensional Datasets (Public)			
Synthetic-H	5,000	5,000	2
Gisette	6,000	5,000	2
Real-sim	72,309	20,958	2
RCV1	50,000	47,236	2
Industrial Datasets (Uber)			
Dataset #1	4m	23	2
Dataset #2	18m	294	2
Dataset #3	18m	20	2
Dataset #4	19m	70	2

Sample clipping: Each of the algorithms we evaluate has the requirement that the loss have a Lipschitz constant. We can enforce this requirement for the loss functions we consider by bounding the norm for each sample. We can accomplish this by pre-processing the dataset, but it must be done carefully to preserve DP.

For all the algorithms except private Frank-Wolfe, to make the loss have an L_2-Lipschitz constant Δ, we bound the influence of each sample (x_i, y_i) by clipping the feature vector x_i to $\left(x_i \cdot \min\left(1, \frac{\Delta}{\|x_i\|}\right)\right)$. This transformation is independent of other samples, and thus preserves DP; it has also been previously used, e.g. in Abadi et al. (2016). As the private Frank-Wolfe algorithm requires the loss to have a relaxed L_1-Lipschitz constant Δ, it suffices (using Theorem 1 from Paulavičius & Žilinskas (2006)) to bound the L_∞-norm of each sample (x_i, y_i) by Δ. We achieve this by clipping each dimension $x_{i,j}$, where $j \in [n]$, to $\min(x_{i,j}, \Delta)$.

Hyperparameters: Each of the evaluated algorithms has at least one hyperparam-

eter. The values for these hyperparameters should be tuned to provide the best accuracy, but the tuning should be done privately in order to guarantee end-to-end differential privacy. Although a number of differentially private hyperparameter tuning algorithms have been proposed (Chaudhuri et al. (2011); Chaudhuri & Vinterbo (2013); Abadi et al. (2016)) to address this problem, they add more variance in the performance of each algorithm, thus making it more difficult to compare the performance across different algorithms.

In order to provide a fair comparison between algorithms, we use a grid search to determine the *best* value for each hyperparameter. Our grid search considers the hyperparameter values listed in Table 4.2. In addition to the standard algorithm hyperparameters (Λ, η, T, k), we tune the clipping parameter Δ used in preprocessing the datasets, and the constraint on the model space used by private Frank-Wolfe, Private SGD when using regularized loss, and Private strongly convex PSGD. The parameter C controls the size of the L_1/L_2-ball from which models are selected by private Frank-Wolfe/the other algorithms respectively. For AMP, we set $\epsilon_2 = f \cdot \epsilon$, $\delta_2 = f \cdot \delta$, and tune for f. Here, f denotes the fraction of the budget (ϵ, δ) that is allocated to (ϵ_2, δ_2). Also, since the valid range of the hyperparameter ϵ_3 depends on the value of ϵ_1, we set $\epsilon_3 = f_1 \cdot \epsilon_1$, and tune for f_1. We also ensure that the constraint on ϵ_3 in Line 1 of Algorithm 1 is satisfied. Note that tuning hyperparameters may be non-private, but it enables a direct comparison of the algorithms themselves.

We consider a range of values for the privacy parameter ϵ. Following Wu et al. (2017), we set the privacy parameter $\delta = \frac{1}{m^2}$, where m is the size of the training data. The complete set of values considered is listed in Table 4.2. For multiclass classification datasets such as MNIST and Covertype, we implement the one-vs-

Table 4.2: Hyperparameter & privacy parameter values

Hyperparameter	Values Considered
Λ (regularization factor)	$10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 0$
η (learning rate)	$0.001, 0.01, 0.1, 1$
T (number of iterations)	5, 10, 100, 1000, 5000
k (minibatch size)	50, 100, 300
Δ (clipping threshold)	0.1, 1, 10, 100
C (model constraint)	1, 10
f (output budget fraction)	0.001, 0.01, 0.1, 0.5
f_1 (privacy budget fraction)	0.9, 0.92, 0.95, 0.98, 0.99

Privacy Parameter	Values Considered
ϵ	$10^{-2}, 10^{-\frac{3}{2}}, 10^{-1}, 10^{-\frac{1}{2}}, 10^{0}, 10^{\frac{1}{2}}, 10^{1}$
δ	$\frac{1}{n^2}$

all strategy by training a binary classifier for each class, and split ϵ and δ equally among the binary classifiers so that we can achieve an overall (ϵ, δ)-DP guarantee by using basic composition (Lemma 2.1.5).

Algorithm Implementations: The implementations used in our evaluation are written in Python, and are available in our open source release (Iyengar et al. (2019a)). For Approximate Minima Perturbation, we define the loss and gradient according to Algorithm 1, and leverage SciPy's `minimize` procedure to find the approximate minima.

For all datasets, our implementation is able to achieve $\gamma = \frac{1}{m^2}$, where m is the size of the training data. For low-dimensional datasets, our implementation of AMP uses SciPy's `BFGS` solver, for which we can specify the desired norm bound γ. The `BFGS` algorithm stores the full Hessian of the objective, which does not fit in memory for the sparse high-dimensional datasets in our study. For these, we define an alternative low-memory implementation using SciPy's `L-BFGS-B` solver, which does not store the full Hessian.

Experiment procedure: Our experiment setup is designed to find the best possible accuracy achievable for a given setting of the privacy parameters. To ensure a fair comparison, we begin every run of each algorithm with the initial model 0^n. Because each of the evaluated algorithms introduces randomness due to noise, we train 10 independent models for each combination of the hyperparameter setting. We report the mean accuracy and standard deviation for the combination of the hyperparameter setting with the highest mean accuracy over the 10 independent runs. The results shown are for hyperparameters tuned via the mean test set accuracy. Since all the considered algorithms aim to minimize the empirical loss, we also conducted experiments by tuning via the mean training set accuracy. Both settings provided visibly identical results.

Differences with the setting in Wu et al. (2017): Although both the studies have 3 datasets in common (Covertype, KDDCup99, and MNIST), our setting is slightly different from Wu et al. (2017) for all 3 of them. For Covertype, our study uses all 7 classes, while Wu et al. (2017) uses a binary version. For KDDCup99, we use a 10% sample of the full dataset (as in Chaudhuri et al. (2011)), while Wu et al. (2017) uses the full dataset. For MNIST, we use all 784 dimensions, while Wu et al. (2017) uses random projection to reduce the dimensionality to 50.

The results we obtain for both the variants of the Private PSGD algorithm (Wu et al. (2017)) are based on faithful implementations of those algorithms. We tune the hyperparameters for both, using the grid search described earlier.

Non-private baseline: Note that one of the main objectives of this study is to determine the cost of privacy in practice for convex optimization. Hence, to provide a point of comparison for our results, we also train a non-private baseline model for each experiment. We use Scikit-learn's `LogisticRegression` class to train

this model on the same training data as the private algorithms, and test its accuracy on the same testing data as the private algorithms. We do not perform sample clipping when training this model.

Strategy for Hyperparameter-free Approximate Minima Perturbation: Now, we describe a data-independent approach for setting Approximate Minima Perturbation's only hyperparameters, $\Delta, \epsilon_2, \delta_2$, and ϵ_3, for *both* the loss functions we consider (see Section 4.4.2). For Δ, we find that setting $\Delta = 1$ achieves a good trade-off between the amount of noise added for perturbing the objective, and the information loss after sample clipping across all datasets. Next, we consider *only* the synthetically generated datasets for setting the hyperparameters specific to AMP. Fixing $\gamma = \frac{1}{m^2}$, we find that setting $\epsilon_2 = 0.01 \cdot \epsilon$ and $\delta_2 = 0.01 \cdot \delta$ achieves a good trade-off between the budget for perturbing the objective, and the amount of noise that its approximate minima can tolerate. For setting ϵ_3, we consider two separate cases:

- For $\epsilon_1 = 0.99 \cdot \epsilon$, and $\epsilon_3 = f_1 \cdot \epsilon_1$, we see that setting $f_1 = 0.99$ for $\epsilon_1 = 0.0099$, $f_1 = 0.95$ for $\epsilon_1 \in \{0.0313, 0.099\}$, and $f_1 = 0.9$ for $\epsilon_1 \in \{0.313, 0.99, 3.13, 9.99\}$ yields a good accuracy for Synthetic-L. Hence, we observe that for very low values of ϵ_1, a good accuracy is yielded by ϵ_3 close to ϵ_1 (i.e., most of the budget is used to reduce the scale of the noise, and the influence of regularization is kept large). As ϵ_1 increases, we see that it is more beneficial to reduce the effects of regularization. We fit a basic polynomial curve of the form $y = a + bx^{-c}$, where $a, b, c > 0$, to the above-stated values to get a dependence of f_1 (the privacy budget fraction) in terms of ϵ_1. We combine it with the lower bound imposed on f_1 by Theorem 1 (for instance, we require $f_1 \geq 0.9$ for $\epsilon_1 = 9.99$) to obtain the following data-independent relationship between ϵ_1 and ϵ_3 for low-dimensional

datasets:

$$\epsilon_3 = \max\left\{\min\left\{0.887 + \frac{0.019}{\epsilon_1^{0.373}}, 0.99\right\}, 1 - \frac{0.99}{\epsilon_1}\right\} \cdot \epsilon_1$$

- For Synthetic-H, we see that setting $f_1 = 0.97$ yields a good accuracy for all the values of ϵ_1 considered. Thus, combining it with the lower bound imposed on f_1 by Theorem 1, we obtain the following relationship for high-dimensional datasets:

$$\epsilon_3 = \max\left\{0.97, 1 - \frac{0.99}{\epsilon_1}\right\} \cdot \epsilon_1$$

Note that the results for this strategy are consistent for both loss functions across all the public and the industrial datasets considered, *none* of which were used in defining the strategy except for setting the Lipschitz constant Δ of the loss. They can be considered to be effectively serving as test-cases for the strategy.

4.4.2 Loss Functions

Our evaluation considers the loss functions for two commonly used models: logistic regression and Huber SVM. First, we present the results for logistic regression in detail. The results for Huber SVM are available in Section 4.4.6.

Logistic regression: The L_2-regularized logistic regression loss function on a sample (x, y) with $y \in \{1, -1\}$ is $\ell(\theta, (x, y)) = \ln(1 + \exp(-y\langle\theta, x\rangle)) + \frac{\Lambda}{2}\|\theta\|^2$.

Our experiments consider both the regularized and un-regularized (i.e., $\Lambda = 0$) settings. The un-regularized version has L_2-Lipschitz constant Δ when for each sample x, $\|x\| \leq \Delta$. It is also Δ^2-smooth. The regularized version has L_2-Lipschitz constant $\Delta + \Lambda C$ when for each sample x, $\|x\| \leq \Delta$, and for each model θ, $\|\theta\| \leq C$. It is also $(\Delta^2 + \Lambda)$-smooth, and Λ-strongly convex.

Table 4.3: List of abbreviations used for algorithms

Abbreviation	Full-form
NP baseline	Non-private baseline
AMP	Approximate Minima Perturbation
H-F AMP	Hyperparameter-free AMP
P-SGD	Private SGD
P-PSGD	Private PSGD
P-SCPSGD	Private Strongly Convex PSGD
P-FW	Private Frank-Wolfe

4.4.3 Experiment 1: Low-Dimensional Datasets

In Figure 4.1, we show the results of the experiments with logistic regression on low-dimensional data. All four algorithms perform better in comparison with the non-private baseline for binary classification tasks (Synthetic-L, Adult, and KDDCup99) than for multi-class problems (Covertype and MNIST), because ϵ and δ must be split among the binary classifiers built for each class.

Figure 4.4 contains precise accuracy numbers for each dataset for reasonably low values of ϵ. These results provide a more precise comparison between the four algorithms, and quantify the accuracy loss versus the non-private baseline for each one. Across all datasets, Approximate Minima Perturbation generally provides the most accurate models across ϵ values.

4.4.4 Experiment 2: High-Dimensional Datasets

For this experiment, we repeat the procedure in Experiment 1 on high-dimensional data, and present the results in Figure 4.2. The results are somewhat different in the high-dimensional regime. We observe that although Approximate Minima Perturbation generally outperforms all the other algorithms, the private Frank-Wolfe algorithm performs the best on Synthetic-H. From prior works (Jain & Thakurta

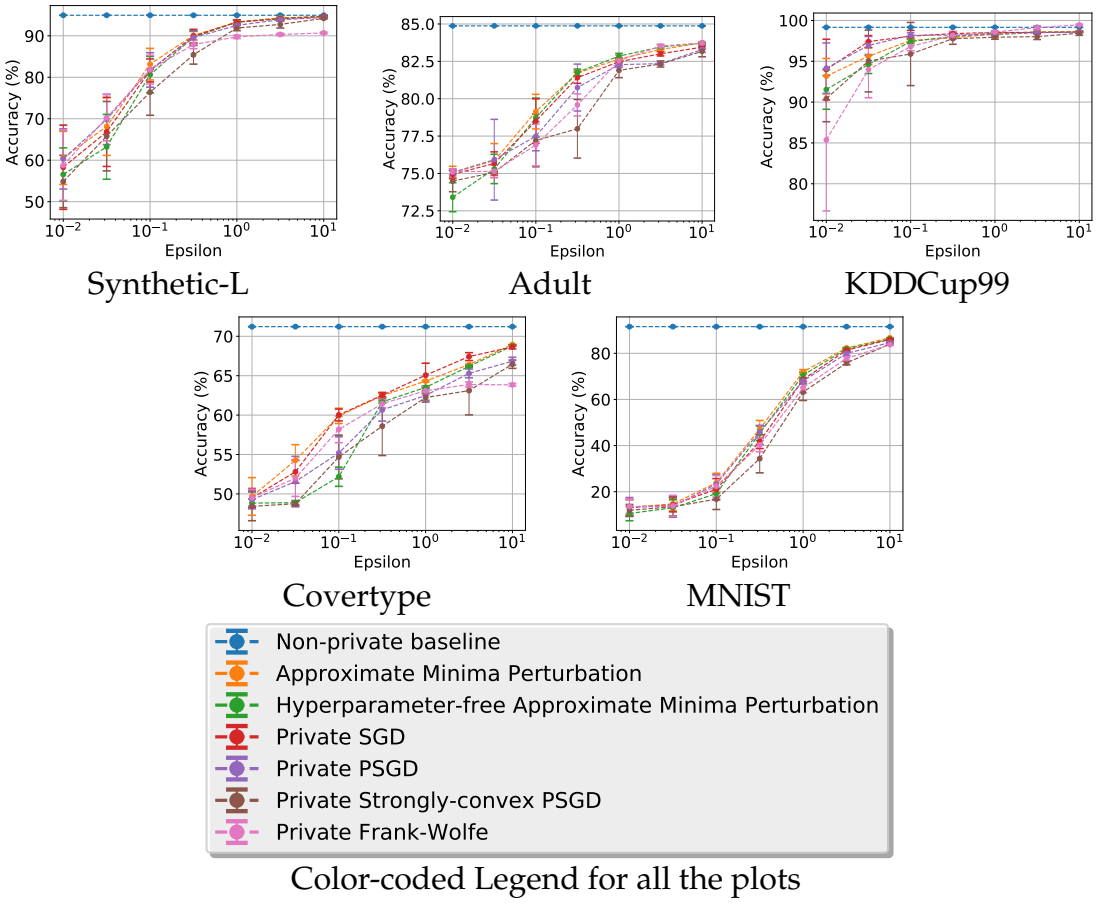

Figure 4.1: Accuracy for logistic regression on low-dimensional datasets. Horizontal axis depicts varying values of ϵ; vertical axis shows accuracy on the testing set.

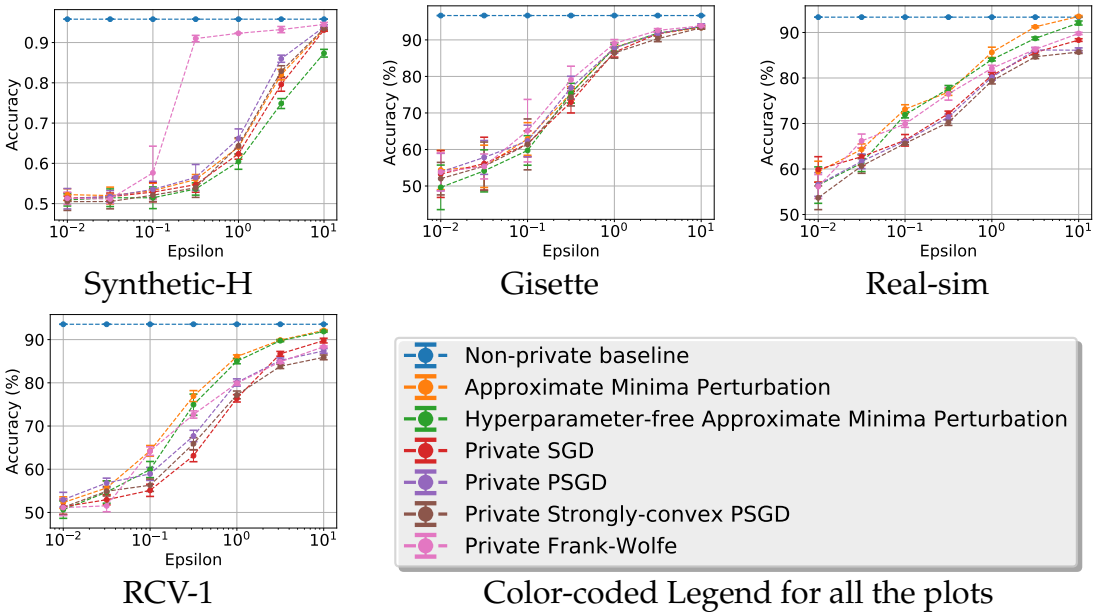

Figure 4.2: Accuracy for logistic regression on high-dimensional datasets. Horizontal axis depicts varying values of ϵ; vertical axis shows accuracy on the testing set.

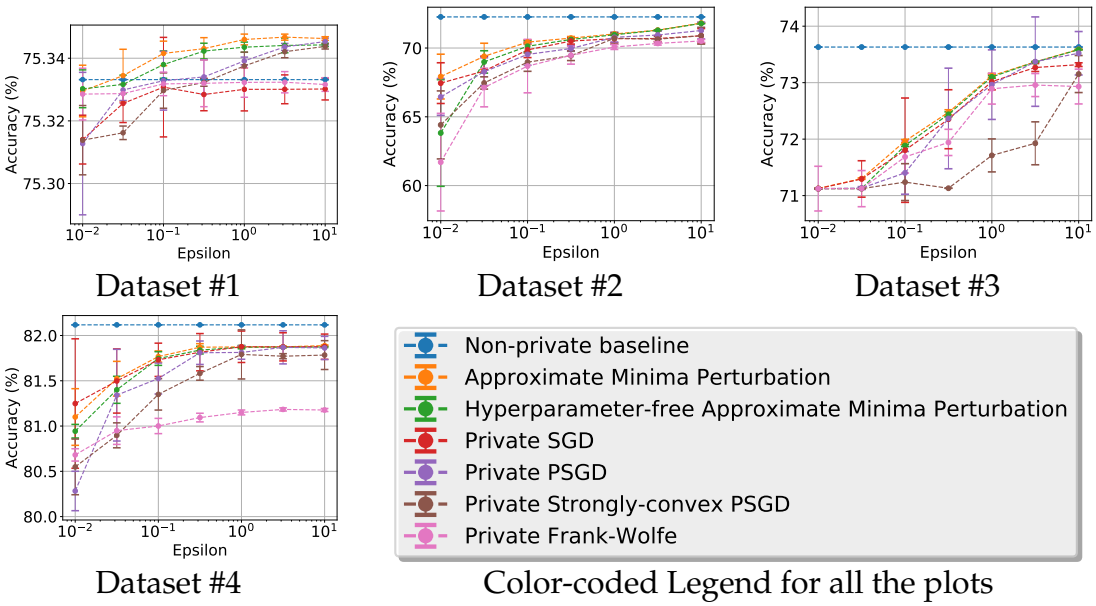

Figure 4.3: Accuracy results for logistic regression on industrial datasets. Horizontal axis depicts varying values of ϵ; vertical axis shows accuracy on the testing set.

Dataset	NP baseline	AMP	H-F AMP	P-SGD	P-PSGD	P-SCPSGD	P-FW
Low-Dimensional Binary Datasets ($\epsilon = 0.1$)							
Synthetic-L	94.9	**83.1**	80.6	81.6	81.7	76.4	81.8
Adult	84.8	**79.1**	78.7	78.5	77.4	77.2	76.9
KDDCup99	99.1	97.5	97.4	98.0	**98.1**	95.8	96.8
Low-Dimensional Multi-class Datasets ($\epsilon = 1$)							
Covertype	71.2	64.3	63.5	**65.0**	62.4	62.2	63.0
MNIST	91.5	**71.9**	70.5	68.6	68.0	63.2	65.0

Figure 4.4: Accuracy results (in %) for logistic regression on low-dimensional datasets. For each dataset, the result in bold represents the DP algorithm with the best accuracy for that dataset. We report the accuracy for $\epsilon = 1$ for multi-class datasets, as compared to $\epsilon = 0.1$ for datasets with binary classification, because multi-class classification is a more difficult task than binary classification. A key for the abbreviations used for the algorithms is provided in Table 4.3.

(2014); Talwar et al. (2014)), we know that both objective perturbation and the private Frank-Wolfe have near dimension-independent utility guarantees when the loss is of a GLM, and we indeed observe this expected behavior from our experiments. As in experiment 1, we present precise accuracy numbers for $\epsilon = 0.1$ in Figure 4.5.

Private Frank-Wolfe works best when the optimal *model* is sparse (i.e., a few important features characterize the classification task well), as in the Synthetic-H dataset, which is well-characterized by just ten important features. This is because private Frank-Wolfe adds at most a single feature to the model at each iteration, and noise increases with the number of iterations. However, noise does *not* increase with the *total* number of features, since it scales with the bound on the L_∞-norm of the samples. This behavior is in contrast to Approximate Minima Perturbation (and the other algorithms considered in our evaluation), for which noise scales with the bound on the L_2-norm of the samples. Private Frank-Wolfe

Dataset	NP baseline	AMP	H-F AMP	P-SGD	P-PSGD	P-SCPSGD	P-FW
High-Dimensional Datasets ($\epsilon = 0.1$)							
Synthetic-H	95.8	53.2	51.4	52.8	53.5	52.0	**57.6**
Gisette	96.6	**62.8**	59.7	61.5	62.3	61.3	58.3
Real-sim	93.3	**73.1**	71.9	66.3	66.1	65.6	69.8
RCV1	93.5	**64.2**	59.9	55.1	58.9	56.2	64.1

Figure 4.5: Accuracy results (in %) for logistic regression on high-dimensional datasets. For each dataset, the result in bold represents the DP algorithm with the best accuracy for that dataset. A key for the abbreviations used for the algorithms is provided in Table 4.3.

therefore approaches the non-private baseline better than the other algorithms for high-dimensional datasets with sparse models, even at low values of ϵ.

4.4.5 Experiment 3: Industrial Use Cases

For this experiment, we repeat the procedure in Experiment 1 on industrial use cases, obtained in collaboration with Uber. These use cases are represented by four datasets, each of which has separately been used to train a production model deployed at Uber. The details of these datasets are listed in Table 4.1. The results of this experiment are depicted in Figure 4.3, with more precise results for $\epsilon = 0.1$ in Figure 4.6.

The industrial datasets are much larger than the datasets considered in Experiment 1. The difference in scale is reflected in the results: all of the algorithms converge to the non-private baseline for very low values of ϵ. These results suggest that in many practical settings, the *cost of privacy is negligible*. In fact, for Dataset #1, some differentially private models exhibit a slightly *higher* accuracy than the non-private baseline for a wide range of ϵ. For instance, even Hyperparameter-

[1]For Dataset #1, AMP slightly outperforms even the NP baseline, as can been seen from Figure 4.3.

Dataset	NP baseline	AMP	H-F AMP	P-SGD	P-PSGD	P-SCPSGD	P-FW
Industrial Datasets ($\epsilon = 0.1$)							
Dataset #1	75.3	**75.3**[1]	75.3	75.3	75.3	75.3	75.3
Dataset #2	72.2	**70.4**	70.1	69.8	69.5	68.9	68.6
Dataset #3	73.6	**71.9**	71.8	71.8	71.4	71.2	71.6
Dataset #4	82.1	**81.7**	81.7	81.7	81.5	81.3	81.0

Figure 4.6: Accuracy results (in %) for logistic regression on industrial datasets. For each dataset, the result in bold represents the DP algorithm with the best accuracy for that dataset. A key for the abbreviations used for the algorithms is provided in Table 4.3.

free AMP, which is end-to-end differentially private as there is no tuning involved, yields an accuracy of 75.34% for $\epsilon = 0.1$ versus the non-private baseline of 75.33%. Some prior works (Bassily et al. (2014b); Dwork et al. (2015a)) have theorized that differential privacy could act as a type of regularization for the system, and improve the generalization error; this empirical result of ours aligns with this claim.

4.4.6 Results for Huber SVM

Here, we report the results of experiments with the Huber SVM loss function. The Huber SVM loss function is a differentiable and smooth approximation of the standard SVM's hinge loss. We define the loss function as in Bassily et al. (2014b). Defining $z = y\langle x, \theta \rangle$, the Huber SVM loss function is:

$$\ell(\theta, (x, y)) = \begin{cases} 1 - z & 1 - z > h \\ 0 & 1 - z < -h \\ \frac{(1-z)^2}{4h} + \frac{1-z}{2} + \frac{h}{4} & \text{otherwise} \end{cases}$$

As with logistic regression, the Huber SVM loss function has L_2-Lipschitz con-

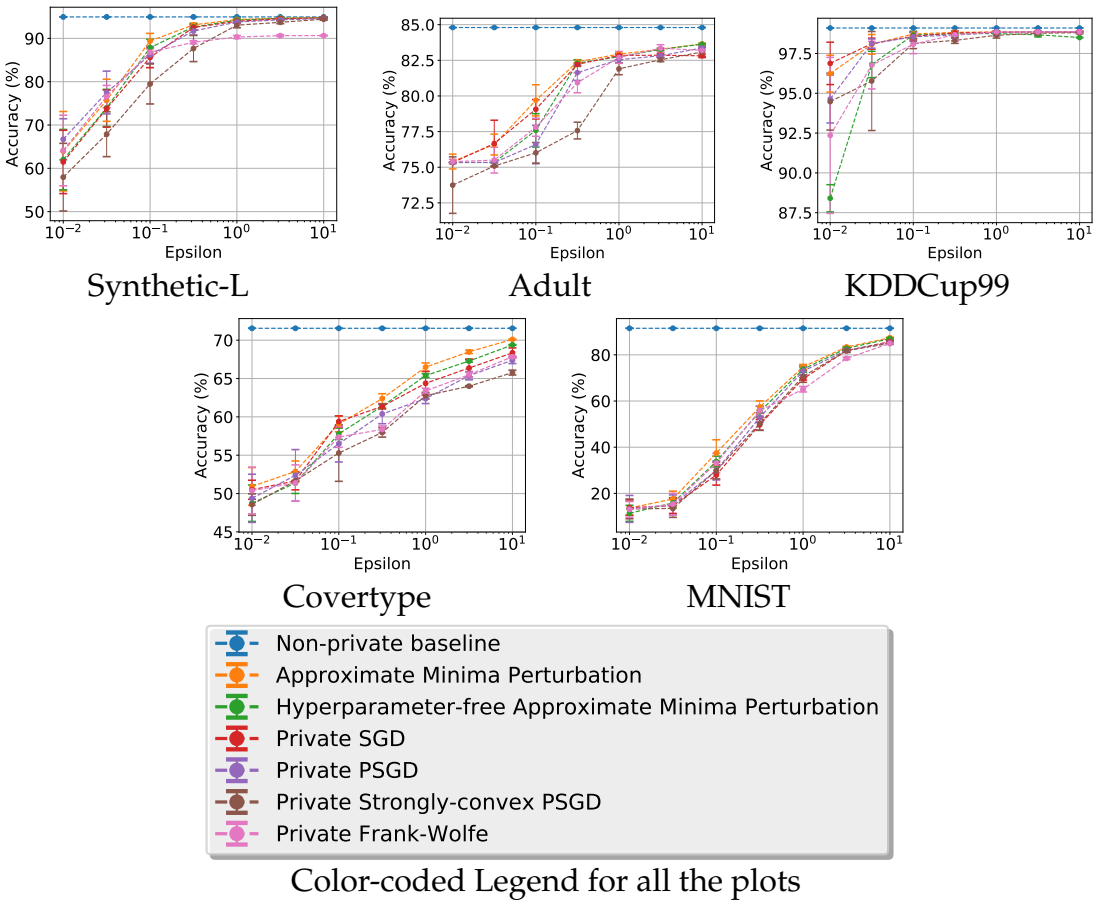

Figure 4.7: Accuracy results for Huber SVM on low-dimensional datasets. Horizontal axis depicts varying values of ϵ; vertical axis shows accuracy (in %) on the testing set.

stant Δ when for each sample x, we have $\|x\| \leq \Delta$.

To ensure that the experiments run to completion for Synthetic-H, we run the experiments on 2000 samples, each consisting of 2000 dimensions. For all the experiments, we obtain the non-private baseline using SciPy's `minimize` procedure with the Huber SVM loss function defined above. Following Wu et al. (2017), we set $h = 0.1$. The results for low-dimensional datasets are shown in Figure 4.7, high-dimensional datasets in Figure 4.8, and industrial datasets in Figure 4.9.

We show more precise results in Figure 4.10. They demonstrate a similar trend

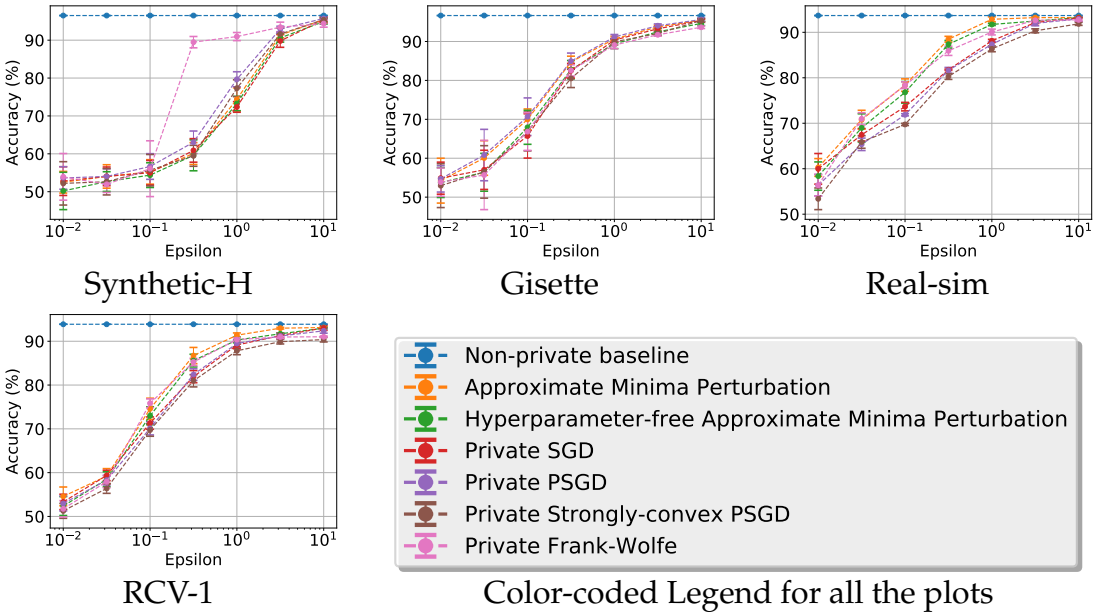

Figure 4.8: Accuracy results for Huber SVM on high-dimensional datasets. Horizontal axis depicts varying values of ϵ; vertical axis shows accuracy (in %) on the testing set.

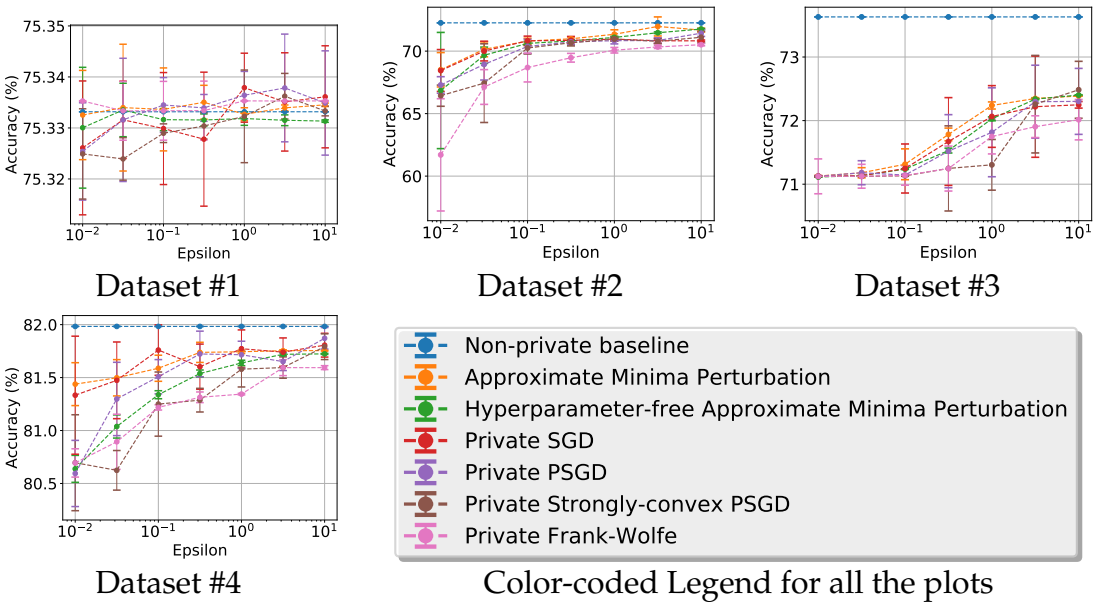

Figure 4.9: Accuracy results for Huber SVM on industrial datasets. Horizontal axis depicts varying values of ϵ; vertical axis shows accuracy (in %) on the testing set.

Dataset	NP baseline	AMP	H-F AMP	P-SGD	P-PSGD	P-SCPSGD	P-FW
Low-Dimensional Binary Datasets ($\epsilon = 0.1$)							
Synthetic-L	94.9	**89.3**	87.8	85.6	86.2	79.4	86.8
Adult	84.8	**79.6**	77.5	79.0	76.5	76.0	77.8
KDDCup99	99.1	98.7	**98.7**[2]	98.5	98.5	98.1	98.0
Low-Dimensional Multi-class Datasets ($\epsilon = 1$)							
Covertype	71.5	**66.4**	65.3	64.3	62.3	62.7	63.3
MNIST	91.5	**74.7**	73.7	69.6	72.9	70.6	65.1
High-Dimensional Datasets ($\epsilon = 0.1$)							
Synthetic-H[3]	96.5	55.2	54.3	55.0	**56.6**	55.6	56.0
Gisette	96.6	69.9	67.9	65.7	**70.6**	66.8	66.8
Real-sim	93.6	**78.3**	76.7	73.6	71.8	69.7	78.3
RCV1[3]	93.8	74.5	72.9	71.3	70.1	69.7	**75.8**
Industrial Datasets ($\epsilon = 0.1$)							
Dataset #1	75.3	75.3	75.3	75.3	**75.3**[4]	75.3	75.3
Dataset #2	72.2	**70.8**	70.6	70.8	70.3	70.2	68.6
Dataset #3	73.6	**71.3**	71.2	71.2	71.1	71.1	71.1
Dataset #4[3]	81.9	81.5	81.3	**81.7**	81.5	81.2	81.2

Figure 4.10: Accuracy results (in %) for Huber SVM. For each dataset, the result in bold represents the DP algorithm with the best accuracy for that dataset. We report the accuracy for $\epsilon = 1$ for multi-class datasets, as compared to $\epsilon = 0.1$ for datasets with binary classification, as multi-class classification is a more difficult task than binary classification. A key for the abbreviations used for the algorithms is provided in Table 4.3.

to the earlier results for logistic regression, with our Approximate Minima Perturbation approach generally providing the highest accuracy. However, the advantage of Approximate Minima Perturbation is less pronounced in this setting.

[2]H-F AMP can outperform AMP when the data-independent strategy provides a better value for the privacy budget fraction f_1 than the specific set of values we consider for tuning in AMP.

[3]The numbers cited here do not reflect the trend for this dataset, as can be seen from Figure 4.10

[4]Slightly outperforms even the NP baseline, as can been seen from Figure 4.9.

4.4.7 Discussion

For large datasets, the cost of privacy is low. Our results confirm the expectation that very accurate differentially private models exist for large datasets. Even for relatively small datasets like Adult and KDDCup99 (where $m < 100,000$), our results show that a differentially private model has accuracy within 6% of the non-private baseline even for a conservative privacy setting of $\epsilon = 0.1$.

For *all* the larger industrial datasets ($m > 1$m), the accuracy of the best differentially private model is within 4% of the non-private baseline even for the most conservative privacy value considered ($\epsilon = 0.01$). For $\epsilon = 0.1$, it is within 2% of the baseline for two of these datasets, essentially identical to the baseline for one of them, and even slightly higher than the baseline for one.

These results suggest that for realistic deployments on large datasets ($m > 1$m, and low-dimensional), a differentially private model can be deployed without much loss in accuracy.

Approximate Minima Perturbation almost always provides the best accuracy, and is easily deployable in practice. Our results in all the experiments demonstrate that among the available algorithms for differentially private convex optimization, our Approximate Minima Perturbation approach almost always produces models with the best accuracy. For four of the five low-dimensional datasets, and all the public high-dimensional datasets we considered, Approximate Minima Perturbation provided consistently better accuracy than the other algorithms. Under some conditions like high-dimensionality of the datasets, and sparsity of the optimal predictive model for it, private Frank-Wolfe does give the best performance. Unlike Approximate Minima Perturbation, however, no hyperparameter-free variant of private Frank-Wolfe exists—and suboptimal hyperparameter values

can reduce accuracy significantly for this algorithm.

As mentioned earlier, Approximate Minima Perturbation also has important properties that enable its practical deployment. It can leverage any off-the-shelf optimizer as a black box, allowing implementations to use existing scalable optimizers (our implementation uses Scipy's `minimize`). None of the other evaluated algorithms have these properties.

Hyperparameter-free Approximate Minima Perturbation provides good utility. As demonstrated by our experimental results, AMP can be deployed without tuning hyperparameters, at little cost to accuracy. Our data-independent approach therefore enables deployment—without significant loss of accuracy—in practical settings where public data may not be available for tuning.

CHAPTER 5

Model-Agnostic Private Learning

In this chapter, we will look at a framework that needs only a black-box access to a non-private learner for obtaining private classifiers when an analyst has a limited amount of unlabelled public data at her disposal. The utility analysis for this framework applies to any sufficiently accurate non-private learner.

5.1 ADDITIONAL PRELIMINARIES

For classification tasks, we use $\mathcal{X}$ to denote the space of feature vectors, and $\mathcal{Y}$ to denote the set of labels. Thus, the data universe $U = \mathcal{X} \times \mathcal{Y}$ in this case, and each data element is denoted as (x, y). First, we provide a definition of PAC learning (used in Section 5.4).

Definition 5.1.1 (Agnostic Probably Approximately Correct (PAC) learner (Valiant (1984); Kearns & Vazirani (1994))). *Let $\mathcal{D}$ be a distribution defined over the space of feature vectors and labels $U = \mathcal{X} \times \mathcal{Y}$. Let $\mathcal{H}$ be a hypothesis class with each $h \in \mathcal{H}$ is a mapping $h : \mathcal{X} \to \mathcal{Y}$. We say an algorithm $\mathcal{A} : U^* \to \mathcal{H}$ is an* Agnostic PAC learner *for $\mathcal{H}$ if it satisfies the following condition: For every $\alpha, \beta \in (0, 1)$, there is a number $m = m(\alpha, \beta) \in \mathbb{N}$ such that when $\mathcal{A}$ is run on a dataset D of m i.i.d. examples from $\mathcal{D}$, then with probability $1 - \beta$ (over the randomness of D) it outputs a hypothesis h_D with $L(h_D; \mathcal{D}) \leq \gamma + \alpha$, where $L(h; \mathcal{D}) \triangleq \mathbb{P}_{(x,y)\sim\mathcal{D}}[h(x) \neq y]$ and $\gamma \triangleq \min_{h\in\mathcal{H}} L(h; \mathcal{D})$.*

We will also use the following parametrized version of the above definition.

Definition 5.1.2 ((α, β, m)-learner for a class $\mathcal{H}$). *Let $\alpha, \beta \in (0, 1)$ and $m \in \mathbb{N}$. An algorithm $\mathcal{A}$ is* (α, β, m)-(agnostic) PAC learner for a class $\mathcal{H}$ *if, given an input dataset D of m i.i.d. examples from the underlying unknown distribution $\mathcal{D}$, with probability*

$1-\beta$, *it outputs a hypothesis* $h_D \in \mathcal{H}$ *with* $L(h_D;\mathcal{D}) \leq \gamma + \alpha$ *(where* γ *is defined as in Definition 5.1.1 above).*

In this chapter, we will use the notion of neighboring under insertion/deletion (Definition 2.1.2) for the guarantee of DP (Definition 2.1.3).

5.1.1 The Sparse Vector Technique

Here, we describe the Sparse vector technique used later in this chapter. It is a common framework for achieving differential privacy, and we provide here the privacy and utility guarantees for it. Sparse vector allows answering a set of queries in an online setting, where a cost for privacy is incurred only if the answer to a query falls near or below a predetermined threshold. We denote the set of queries by $\mathcal{F} = \{f_1, \cdots, f_{\widetilde{m}}\}$, where every $f_i : U^* \to \mathbb{R}$, and has global sensitivity at most one. We provide a pseudocode for the technique in Algorithm 6. Next, we provide the privacy and accuracy guarantees for Algorithm 6.

Algorithm 6 $\mathcal{A}_{\mathsf{sparseVec}}$: Sparse vector technique

Input: dataset: D, query set $\mathcal{F} = \{f_1, \cdots, f_{\widetilde{m}}\}$, privacy parameters $\epsilon, \delta > 0$, unstable query cutoff: T, threshold: w

1: $c \leftarrow 0$, $\lambda \leftarrow \sqrt{32T\log(1/\delta)}/\epsilon$, and $\widehat{w} \leftarrow w + \mathsf{Lap}(\lambda)$
2: **for** $f_i \in \mathcal{F}$ and $c \leq T$ **do**
3: $\quad \widehat{f_i}(D) \leftarrow f_i(D) + \mathsf{Lap}(2\lambda)$
4: $\quad$ **If** $\widehat{f_i}(D) > \widehat{w}$, **then** , output $\top$, **else** output $\perp$, and set $\widehat{w} \leftarrow w + \mathsf{Lap}(\lambda)$, $c \leftarrow c+1$

Theorem 3 (Privacy guarantee (Dwork et al. (2010); Hardt & Rothblum (2010); Dwork et al. (2014a))). *Algorithm 6 is* (ϵ, δ)*-differentially private.*

Theorem 4 (Accuracy guarantee (Dwork et al. (2010); Hardt & Rothblum (2010); Dwork et al. (2014a))). *For* $\alpha = \log(2\widetilde{m}T/\beta)\sqrt{512T\log(1/\delta)}/\epsilon$, *and any set of* $\widetilde{m}$

queries $f_1, \cdots, f_{\tilde{m}}$*, define the set* $L(\alpha) = \{i : f_i(D) \leq w + \alpha\}$*. If* $|L(\alpha)| \leq T$*, then we have the following w.p. at least* $1 - \beta$*:* $\forall i \notin L(\alpha)$*, Algorithm 6 outputs* $\top$*.*

5.2 RELATED WORK

For learning privately via aggregation and knowledge transfer, Hamm et al. (2016) explored a similar technique. However, their construction deviated from the above description. In particular, it was a white-box construction with weak accuracy guarantees; their guarantees also involved making strong assumptions about the learning model and the loss function used in training. Recent work (Papernot et al. (2016, 2018)), of which Papernot et al. (2018) is independent from our work, gave algorithms that follow the knowledge transfer paradigm described above. Their constructions are black-box. However, only empirical evaluations are given for their constructions; no formal utility guarantees are provided. For the query-answering setting, a recent independent work (Dwork & Feldman (2018)) considers the problem of private prediction, but only in the single-query setting, whereas we study the multiple-query setting. The earliest idea of using ensemble classifiers to provide differentially private prediction can be traced to Dwork, Rothblum, and Thakurta from 2013.

5.3 PRIVATELY ANSWERING STABLE ONLINE QUERIES

In this section, we design a generic framework that allows answering a set of queries on a dataset while preserving differential privacy, and only incurs a privacy cost for the queries that are unstable.

5.3.1 The Distance to Instability Framework

First, we describe the distance to instability framework from Smith & Thakurta (2013) that releases the exact value of a function on a dataset while preserving differential privacy, provided the function is sufficiently *stable* on the dataset. We define the notion of stability first, and provide the pseudocode for a private estimator for any function via this framework in Algorithm $\mathcal{A}_{\mathsf{stab}}$ (Algorithm 7).

Algorithm 7 $\mathcal{A}_{\mathsf{stab}}$: Private estimator for f via distance to instability (Smith & Thakurta (2013))

Input: dataset: D, function $f : U^* \to \mathcal{R}$, distance to instability $\mathsf{dist}_f : U^* \to \mathbb{R}$, threshold: Γ, privacy parameter $\epsilon > 0$

1: $\widehat{\mathsf{dist}} \leftarrow \mathsf{dist}_f(D) + \mathsf{Lap}\,(1/\epsilon)$
2: **If** $\widehat{\mathsf{dist}} > \Gamma$, **then** output $f(D)$, **else** output $\perp$

Definition 5.3.1 (k-stability (Smith & Thakurta (2013))). *A function $f : U^* \to \mathcal{R}$ is* k-stable on dataset D *if adding or removing any k elements from D does not change the value of f, that is, $f(D) = f(D')$ for all D' such that $|D \triangle D'| \leq k$. We say f is* stable *on D if it is (at least) 1-stable on D, and* unstable *otherwise.*

The *distance to instability* of a dataset $D \in U^*$ with respect to a function f is the number of elements that must be added to or removed from D to reach a dataset that is not stable. Note that D is k-stable if and only if its distance to instability is at least k.

Theorem 5 (Privacy guarantee for $\mathcal{A}_{\mathsf{stab}}$). *If the threshold $\Gamma = \log(1/\delta)/\epsilon$, and the distance to instability function $dist_f(D) = \arg\max_k [f(D)$ is k-stable$]$, then Algorithm 7 is (ϵ, δ)-differentially private.*

Proof. We prove the above theorem by considering the two possibilities for any D' s.t. $|D \Delta D'| = 1$: either $f(D) = f(D')$, or $f(D) \neq f(D')$. We prove the privacy in these two cases via Lemmas 5.3.2 and 5.3.3.

Lemma 5.3.2. *Let $D \in U^*$ be any fixed dataset. Assume that for any dataset $D' \in U^*$ s.t. $|D \Delta D'| = 1$, we have $f(D) = f(D')$. Then, for any output $s \in \mathcal{R} \cup \{\perp\}$, we have:* $\Pr[\mathcal{A}_{\mathsf{stab}}(D, f) = s] \leq e^{\epsilon} \Pr[\mathcal{A}_{\mathsf{stab}}(D', f) = s]$.

Proof. First, note that with the instantiation in Theorem 5, the function dist_f has a global sensitivity of one. Therefore, by the guarantees of the Laplace mechanism (Lemma 2.1.8), $\widehat{\mathsf{dist}}$ satisfies ϵ-differential privacy. Since the set of possible outputs is the same (i.e., $\{f(D), \perp\}$) for both D and D', and the decision to output $f(D)$ versus $\perp$ depends only on $\widehat{\mathsf{dist}}$, we get the statement of the lemma by the post-processing property of differential privacy (Lemma 2.1.4). □

Lemma 5.3.3. *Let $D \in U^*$ be any fixed dataset. Assume that for any dataset $D' \in U^*$ s.t. $|D \Delta D'| = 1$, we have $f(D) \neq (D')$. Then, for any output $s \in \mathcal{R} \cup \{\perp\}$, we have the following with probability at least $1 - \delta$:* $\mathcal{A}_{\mathsf{stab}}(D, f) = \mathcal{A}_{\mathsf{stab}}(D', f) = \perp$.

Proof. Since $f(D) \neq f(D')$, it follows that $f(D)$ and $f(D')$ are unstable, that is, $\mathsf{dist}_f(D) = \mathsf{dist}_f(D') = 0$. This implies

$$\Pr[\mathcal{A}_{\mathsf{stab}}(D, f) = \perp] = \Pr[\mathcal{A}_{\mathsf{stab}}(D', f) = \perp] = \Pr\left[\mathsf{Lap}\left(\frac{1}{\epsilon}\right) \leq \frac{\log(1/\delta)}{\epsilon}\right].$$

Since the density function for the Laplace distribution $\mathsf{Lap}(\lambda)$ is $\mu(x) = \frac{1}{2\lambda} e^{-|x|/\lambda}$, it follows that $\Pr\left[\mathsf{Lap}\left(\frac{1}{\epsilon}\right) \leq \frac{\log(1/\delta)}{\epsilon}\right] \geq 1 - \delta$. □

We get the statement of Theorem 5 by combining Lemmas 5.3.2 and 5.3.3. □

Theorem 6 (Utility guarantee for $\mathcal{A}_{\mathsf{stab}}$ (Smith & Thakurta (2013)))**.** *If the threshold $\Gamma = \log(1/\delta)/\epsilon$, the distance to instability function is chosen as in Theorem 5, and $f(D)$ is $\left(\left(\log(1/\delta) + \log(1/\beta)\right)/\epsilon\right)$-stable, then Algorithm 7 outputs $f(D)$ with probability at least $1 - \beta$.*

5.3.2 Online Query Release via Distance to Instability

Using Algorithm $\mathcal{A}_{\mathsf{OQR}}$ (Algorithm 8), we show that for a set of $\widetilde{m}$ queries $\mathcal{F} = \{f_1, \cdots, f_{\widetilde{m}}\}$ to be answered on a dataset D, one can *exactly* answer all but T of them while satisfying differential privacy, as long as at most T queries in $\mathcal{F}$ are not k-stable, where $k \approx \log(\widetilde{m})\sqrt{T}/\epsilon$. Notice that the dependence of k on the total number of queries ($\widetilde{m}$) is logarithmic. In contrast, one would achieve a dependence of roughly $\sqrt{\widetilde{m}}$ by using the advanced composition property of differential privacy (Lemma 2.1.6).

Algorithm 8 $\mathcal{A}_{\mathsf{OQR}}$: Online Query Release via distance to instability

Input: dataset: D, query set $\mathcal{F} = \{f_1, \cdots, f_{\widetilde{m}}\}$ chosen online, distance to instability $\mathsf{dist}_{f_i} : U^* \to \mathbb{R}, \forall i \in [\widetilde{m}]$, unstable query cutoff: T, privacy parameters $\epsilon, \delta > 0$

1: $c \leftarrow 0$, $\lambda \leftarrow \sqrt{32T\log(2/\delta)}/\epsilon$, $w \leftarrow 2\lambda \cdot \log(2\widetilde{m}/\delta)$, and $\widehat{w} \leftarrow w + \mathsf{Lap}(\lambda)$.
2: **for** $f \in \mathcal{F}$ and $c \leq T$ **do**
3: $\quad \mathsf{out} \leftarrow \mathcal{A}_{\mathsf{stab}}(D, f, \mathsf{dist}_f, \Gamma = \widehat{w}, \epsilon = 1/2\lambda)$
4: $\quad$ **If** $\mathsf{out} = \perp$, **then** $c \leftarrow c + 1$ and $\widehat{w} \leftarrow w + \mathsf{Lap}(\lambda)$
5: $\quad$ Output out

The main design focus in this section is that the algorithms should be able to handle very generic query classes $\mathcal{F}$ under minimal assumptions. A salient feature of Algorithm $\mathcal{A}_{\mathsf{OQR}}$ is that it only requires the range $\mathcal{R}_i$ of the function $f_i : U^* \to \mathcal{R}_i$, where $f_i \in \mathcal{F}$, to be discrete for all $i \in [\widetilde{m}]$.

We provide the privacy and utility guarantees for Algorithm $\mathcal{A}_{\mathsf{OQR}}$ in Theorem 7 and Corollary 5.3.6, respectively. Surprisingly, the utility guarantee of $\mathcal{A}_{\mathsf{OQR}}$ has *no dependence* on the cardinality of the set $\mathcal{R}_i$, for all $i \in [\widetilde{m}]$.

Theorem 7 (Privacy guarantee for $\mathcal{A}_{\mathsf{OQR}}$)**.** *If for all functions $f \in \mathcal{F}$, the distance to instability function is $dist_f(D) = \arg\max_k [f(D)$ is k-stable$]$, then Algorithm 8 is (ϵ, δ)-differentially private.*

Proof. In our proof, we use ideas from the proof of Theorem 5 and the Sparse vector technique (see Section 5.1 for a background on the technique). For clarity, we split the computation in Algorithm 8 into two logical phases: First, for every query $f \in \mathcal{F}$, $\mathcal{A}_{\mathsf{OQR}}$ either commits to $\top$, or outputs $\perp$ based on the input dataset D. Next, if it commits to $\top$, then it outputs $f(D)$.

Now, let us consider two fictitious algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$, where $\mathcal{A}_1$ outputs the sequence of $\top$ and $\perp$ corresponding to the first phase above, and $\mathcal{A}_2$ is invoked to output $f_i(D)$ only for the queries f_i that $\mathcal{A}_1$ output $\top$. Notice that the combination of $\mathcal{A}_1$ and $\mathcal{A}_2$ is equivalent to $\mathcal{A}_{\mathsf{OQR}}$. Since $\mathcal{A}_1$ is essentially executing the sparse vector technique (Algorithm 6), by Theorem 3, it satisfies $(\epsilon, \delta/2)$-differential privacy. Next, we analyze the privacy for Algorithm $\mathcal{A}_2$.

Consider any particular query $f \in \mathcal{F}$. For any dataset D' s.t. $|D \triangle D'| = 1$, there are two possibilities: either $f(D) = f(D')$, or $f(D) \neq f(D')$. When $f(D) = f(D')$, if $\mathcal{A}_1$ outputs $\perp$, algorithm $\mathcal{A}_2$ is not invoked and hence the privacy guarantee isn't affected. Moreover,if $\mathcal{A}_1$ outputs $\top$, we get the following lemma by the post-processing property of differential privacy (Lemma 2.1.4):

Lemma 5.3.4. *Let $D \in U^*$ be any fixed dataset. Assume that for any dataset $D' \in U^*$ s.t. $|D \triangle D'| = 1$, we have $f(D) = f(D')$. Then, for any output $s \in \mathcal{R}$, we have the following for the invocation of Algorithm $\mathcal{A}_2$:* $\Pr[\mathcal{A}_2(D, f) = s] = \Pr[\mathcal{A}_2(D', f) = s]$.

When $f(D) \neq f(D')$, by Lemma 5.3.3, $\mathcal{A}_1$ outputs $\perp$ with probability at least $1 - \delta/2\widetilde{m}$. Therefore, we get that:

Lemma 5.3.5. *Let $D \in U^*$ be any fixed dataset. Assume that for any dataset $D' \in U^*$ s.t. $|D \triangle D'| = 1$, we have $f(D) \neq f(D')$. Then, Algorithm $\mathcal{A}_2$ is never invoked to output $f(D)$ with probability at least $1 - \delta/2\widetilde{m}$.*

Now, consider the sequence of queries $f_1, \cdots, f_{\widetilde{m}}$. Let $\mathcal{F}_1$ be the set of queries

where, for every $f \in \mathcal{F}_1$, we have $f(D) = f(D')$. Let $\mathcal{F}_2 = \mathcal{F} \backslash \mathcal{F}_1$. Since Algorithm $\mathcal{A}_1$ is $(\epsilon, \delta/2)$-differentially private for all queries in $\mathcal{F}$, it is also $(\epsilon, \delta/2)$-differentially private for all queries in $\mathcal{F}_1$. Now since $|\mathcal{F}_2| \leq \widetilde{m}$, using Lemma 5.3.5 and taking an union bound over all the queries in $|\mathcal{F}_2|$, Algorithm $\mathcal{A}_2$ is never invoked for queries in $\mathcal{F}_2$ with probability at least $1 - \delta/2$. By the basic composition property of DP (Lemma 2.1.5), this implies (ϵ, δ)-differential privacy for the overall algorithm $\mathcal{A}_{\mathsf{OQR}}$. □

Corollary 5.3.6 (Utility guarantee for $\mathcal{A}_{\mathsf{OQR}}$)**.** *For any set of $\widetilde{m}$ adaptively chosen queries $\mathcal{F} = \{f_1, \cdots, f_{\widetilde{m}}\}$, let $dist_{f_i}(D) = \arg\max_k [f_i(D)$ is k-stable$]$ for each f_i. Also, define $L(\alpha) = \{i : \mathsf{dist}_{f_i}(D) < \alpha\}$ for $\alpha = 32 \cdot \log(4\widetilde{m}T/\min(\delta, \beta))\sqrt{2T\log(2/\delta)}/\epsilon$. If $|L(\alpha)| \leq T$, then we have the following w.p. at least $1 - \beta$: $\forall i \notin L(\alpha)$, Algorithm $\mathcal{A}_{\mathsf{OQR}}$ (Algorithm 8) outputs $f_i(D)$.*

Proof. The proof follows directly from Theorem 4. To see this, note that Algorithm $\mathcal{A}_{\mathsf{OQR}}$ follows the same lines of Algorithm $\mathcal{A}_{\mathsf{sparseVec}}$ (Algorithm 6) with slight adjustments. In particular, $\top$ in $\mathcal{A}_{\mathsf{sparseVec}}$ is replaced with $f(D)$ in $\mathcal{A}_{\mathsf{OQR}}$; δ in the setting of λ in $\mathcal{A}_{\mathsf{sparseVec}}$ is replaced with $\delta/2$ in the setting of λ in $\mathcal{A}_{\mathsf{OQR}}$; w which is left arbitrary in $\mathcal{A}_{\mathsf{sparseVec}}$ is set to $2\lambda \log(2\widetilde{m}/\delta)$ in $\mathcal{A}_{\mathsf{OQR}}$; q in $\mathcal{A}_{\mathsf{sparseVec}}$ is replaced with $\mathsf{dist}_f(D)$ in $\mathcal{A}_{\mathsf{stab}}$; and $\widehat{q}$ in $\mathcal{A}_{\mathsf{sparseVec}}$ is replaced with $\widehat{\mathsf{dist}}$ in $\mathcal{A}_{\mathsf{stab}}$. Putting these together with Theorem 4 and the premise in the corollary statement (i.e., $|\{i : \mathsf{dist}_{f_i}(D) < \alpha\}| \leq T$) immediately proves the corollary with the specified value of α. Note that by comparing Theorem 4 with the premise in the corollary, we can see that the value of α in the corollary is obtained by adding the value of w as set in $\mathcal{A}_{\mathsf{OQR}}$ and the value of α as set in Theorem 4.

□

5.3.3 Instantiation: Online Query Release via Subsample and Aggregate

While Algorithm $\mathcal{A}_{\mathsf{OQR}}$ has the desired property in terms of generality, it falls short in two critical aspects: i) it relies directly on the distance to instability framework (Algorithm $\mathcal{A}_{\mathsf{stab}}$ in Section 5.3.1) which does not provide an efficient way to compute the distance to instability for a given function, and ii) given a function class $\mathcal{F}$, it is unclear which functions from $\mathcal{F}$ satisfy the desired property of α-stability.

In Algorithm $\mathcal{A}_{\mathsf{subSamp}}$ (Algorithm 9), we address both of these concerns by instantiating the distance to instability function in Algorithm $\mathcal{A}_{\mathsf{OQR}}$ with the subsample and aggregate framework (as done in Smith & Thakurta (2013)). We provide the privacy and accuracy guarantees for $\mathcal{A}_{\mathsf{subSamp}}$ in Corollary 5.3.7, and Theorem 8, respectively. In Section 5.4, we show how Algorithm $\mathcal{A}_{\mathsf{subSamp}}$ can be used for classification problems without relying too much on the underlying learning model (e.g., convex versus non-convex models).

Algorithm 9 $\mathcal{A}_{\mathsf{subSamp}}$: Online Query Release via subsample and aggregate

Input: dataset: D, query set $\mathcal{F} = \{f_1, \cdots, f_{\widetilde{m}}\}$ chosen online, range of the queries: $\{\mathcal{R}_1, \cdots, \mathcal{R}_{\widetilde{m}}\}$, unstable query cutoff: T, privacy parameters $\epsilon, \delta > 0$, failure probability: β

1: $b \leftarrow 136 \cdot \log\left(4\widetilde{m}T/\min\left(\delta, \beta/2\right)\right)\sqrt{T\log(2/\delta)}/\epsilon$

2: Arbitrarily split D into b non-overlapping chunks of size m/b. Call them $D_1, \cdots, D_b$

3: **for** $i \in [\widetilde{m}]$ **do**

4: Let $\mathcal{S}_i = \{f_i(D_1), \cdots, f_i(D_b)\}$, and for every $r \in \mathcal{R}_i$, let $\mathsf{ct}(r) = \#$ times r appears in $\mathcal{S}_i$

5: $\widehat{f}_i(D) \leftarrow \arg\max_{r \in \mathcal{R}_i} [\mathsf{ct}(r)]$

6: $\mathsf{dist}_{\widehat{f}_i} \leftarrow \max\left\{0, \left(\max_{r \in \mathcal{R}_i}[\mathsf{ct}(r)] - \max_{r \in \mathcal{R}_i \setminus \widehat{f}_i(D)}[\mathsf{ct}(r)]\right) - 1\right\}$

7: Output $\mathcal{A}_{\mathsf{OQR}}\left(D, \left\{\widehat{f}_1, \cdots, \widehat{f}_{\widetilde{m}}\right\}, \left\{\mathsf{dist}_{\widehat{f}_1}, \cdots, \mathsf{dist}_{\widehat{f}_{\widetilde{m}}}\right\}, T, \epsilon, \delta\right)$

The key idea in $\mathcal{A}_{\mathsf{subSamp}}$ is as follows: i) First, arbitrarily split the dataset D into

b subsamples of equal size, $D_1, \cdots, D_b$, ii) For each query $f_i \in \mathcal{F}$, where $i \in [\widetilde{m}]$, and each $r \in \mathcal{R}_i$, compute $\mathsf{ct}(r)$, which is the number of subsamples D_j, where $j \in [b]$, for which $f_i(D_j) = r$, iii) Define $\widehat{f_i}(D)$ to be the $r \in \mathcal{R}_i$ with the largest ct, and the distance to instability function $\mathsf{dist}_{\widehat{f_i}}$ to correspond to the the difference between the largest ct and the second largest ct among all $r \in \mathcal{R}_i$, iv) Invoke $\mathcal{A}_{\mathsf{OQR}}$ with $\widehat{f_i}$ and $\mathsf{dist}_{\widehat{f_i}}$. Now, note that $\mathsf{dist}_{\widehat{f_i}}$ is *always* efficiently computable. Furthermore, Theorem 8 shows that if D is a dataset of m i.i.d. samples drawn from some distribution $\mathcal{D}$, and f_i on a dataset of m/b i.i.d. samples drawn from $\mathcal{D}$ matches some $r \in \mathcal{R}_i$ w.p. at least $3/4$, then with high probability $\widehat{f_i}(D)$ is a stable query.

Corollary 5.3.7 (Privacy guarantee for $\mathcal{A}_{\mathsf{subSamp}}$)**.** *Algorithm 9 is (ϵ, δ)-differentially private.*

The proof of Corollary 5.3.7 follows immediately from the privacy guarantee for $\mathcal{A}_{\mathsf{OQR}}$ (Algorithm 8).

Theorem 8 (Utility guarantee for $\mathcal{A}_{\mathsf{subSamp}}$)**.** *Let $\mathcal{F}$ denote any set of $\widetilde{m}$ adaptively chosen queries, and D be a dataset of m samples drawn i.i.d. from a fixed distribution $\mathcal{D}$. For $b = 136 \cdot \log\left(4\widetilde{m}T/\min\left(\delta, \beta/2\right)\right) \cdot \sqrt{T\log(2/\delta)}/\epsilon$, let $\bar{L} \subseteq \mathcal{F}$ be a set of queries s.t. for every $f \in \bar{L}$, there exists some x_f for which $f(\widehat{D}) = x_f$ w.p. at least $3/4$ over drawing a dataset $\widehat{D}$ of m/b i.i.d. data samples from $\mathcal{D}$. If $|\bar{L}| \geq \widetilde{m} - T$, then w.p. at least $1 - \beta$ over the randomness of Algorithm $\mathcal{A}_{\mathsf{subSamp}}$ (Algorithm 9), we have the following: $\forall f \in \bar{L}$, Algorithm $\mathcal{A}_{\mathsf{subSamp}}$ outputs x_f. Here, (ϵ, δ) are the privacy parameters.*

Proof. For a given query $f \in \mathcal{F}$, let $X_f^{(i)}$ be the random variable that equals to 1 if $f(D_i)$ in Algorithm $\mathcal{A}_{\mathsf{subSamp}}$ equals x_f, and 0 otherwise. Thus, we have $\Pr[X_f^{(i)} = 1] \geq 3/4$ by assumption. By the standard Chernoff-Hoeffding bound, we get $\sum_{i=1}^{b} X_f^{(i)} \geq 3b/4 - \sqrt{b\log(2\widetilde{m}/\beta)/2}$ with probability at least $1 - \beta/2\widetilde{m}$. If

we set $b \geq 72 \log(2\widetilde{m}/\beta)$, then the previous expression is at least $2b/3$. By the union bound, this implies that with probability at least $1-\beta/2$, we have $\mathsf{dist}_{\widehat{f}} \geq b/3$ for every $f \in \bar{L}$. Furthermore, to satisfy the distance to instability condition in Corollary 5.3.6, we need $b/3 \geq 32 \cdot \log\left(4\widetilde{m}T/\min\left(\delta, \beta/2\right)\right) \sqrt{2T\log(2/\delta)}/\epsilon$. Both the conditions on b are satisfied by setting $b = 136 \cdot \log\left(4\widetilde{m}T/\min\left(\delta, \beta/2\right)\right) \sqrt{T\log(2/\delta)}/\epsilon$. Using Corollary 5.3.6 along with this value of b, we get the statement of the theorem. □

5.4 PRIVATELY ANSWERING CLASSIFICATION QUERIES

In this section, we instantiate the distance to instability framework (Algorithm 7) with the subsample and aggregate framework (Algorithm 9), and then combine it with the Sparse vector technique (Algorithm 6) to obtain a construction for privately answering classification queries with a conservative use of the privacy budget (Algorithm 10 below). We consider here the case of binary classification for simplicity. However, we note that one can easily extend the construction (and obtain analogous guarantees) for multi-class classification.

A private training set, denoted by D, is a set of m private binary-labeled data points $\{(x_1, y_1), \ldots, (x_m, y_m)\} \in (\mathcal{X} \times \mathcal{Y})^m$ drawn i.i.d. from some (arbitrary unknown) distribution $\mathcal{D}$ over $U = \mathcal{X} \times \mathcal{Y}$. We will refer to the induced marginal distribution over $\mathcal{X}$ as $\mathcal{D}_{\mathcal{X}}$. We consider a sequence of (online) classification queries defined by a sequence of $\widetilde{m}$ unlabeled points from $\mathcal{X}$, namely $\mathcal{Q} = \{\tilde{x}_1, \cdots, \tilde{x}_{\widetilde{m}}\} \in \mathcal{X}^{\widetilde{m}}$, drawn i.i.d. from $\mathcal{D}_{\mathcal{X}}$, and let $\{\tilde{y}_1, \cdots, \tilde{y}_{\widetilde{m}}\} \in \{0,1\}^{\widetilde{m}}$ be the corresponding true unknown labels. Algorithm 10 has oracle access to a non-private learner $\mathcal{A}$ for a hypothesis class $\mathcal{H}$. We will consider both realizable and non-realizable cases of the standard PAC model. In particular, $\mathcal{A}$ is assumed to be an (agnostic) PAC learner for $\mathcal{H}$.

Algorithm 10 $\mathcal{A}_{\mathsf{binClas}}$: Private Online Binary Classification via subsample and aggregate, and sparse vector

Input: Private dataset: D, sequence of online unlabeled public data (defining the classification queries) $\mathcal{Q} = \{\tilde{x}_1, \cdots, \tilde{x}_{\widetilde{m}}\}$, oracle access to a non-private learner $\mathcal{A} : U^* \to \mathcal{H}$ for a hypothesis class $\mathcal{H}$, cutoff parameter: T, privacy parameters $\epsilon, \delta > 0$, failure probability: β

$c \leftarrow 0$, $\lambda \leftarrow \sqrt{32T\log(2/\delta)/\epsilon}$, and $b \leftarrow 34\sqrt{2}\lambda \cdot \log\left(4\widetilde{m}T/\min\left(\delta, \beta/2\right)\right)$

$w \leftarrow 2\lambda \cdot \log(2\widetilde{m}/\delta)$, and $\widehat{w} \leftarrow w + \mathsf{Lap}(\lambda)$

Arbitrarily split D into b non-overlapping chunks of size m/b. Call them $D_1, \cdots, D_b$

for $j \in [b]$, train $\mathcal{A}$ on D_j to get a classifier $h_j \in \mathcal{H}$

for $i \in [\widetilde{m}]$ and $c \leq T$ **do**

  Let $\mathcal{S}_i = \{h_1(x_i), \cdots, h_b(x_i)\}$, and for $y \in \{0, 1\}$, let $\mathsf{ct}(y) = \#$ times y appears in $\mathcal{S}_i$

  $\widehat{q}_{x_i}(D) \leftarrow \arg\max_{y \in \{0,1\}} [\mathsf{ct}(y)]$, $\mathsf{dist}_{\widehat{q}_{x_i}} \leftarrow \max\left\{0, \mathsf{ct}\left(\widehat{q}_{x_i}\right) - \mathsf{ct}\left(1 - \widehat{q}_{x_i}\right) - 1\right\}$

  $\mathsf{out}_i \leftarrow \mathcal{A}_{\mathsf{stab}}\left(D, \widehat{q}_{x_i}, \mathsf{dist}_{\widehat{q}_{x_i}}, \Gamma = \widehat{w}, \epsilon = 1/2\lambda\right)$

  If $\mathsf{out}_i = \perp$, **then** $c \leftarrow c + 1$ and $\widehat{w} \leftarrow w + \mathsf{Lap}(\lambda)$

  Output out_i

Theorem 9 (Privacy guarantee for $\mathcal{A}_{\mathsf{binClas}}$)**.** *Algorithm $\mathcal{A}_{\mathsf{binClas}}$ (Algorithm 10) is (ϵ, δ)-DP.*

The proof of this theorem follows from combining the guarantees of the distance to instability framework (Smith & Thakurta (2013)), and the sparse vector technique (Dwork et al. (2014a)). The idea is that in each round of query response, if the algorithm outputs a label in $\{0, 1\}$, then there is "no loss in privacy" in terms of ϵ (as there is sufficient consensus). However, when the output is $\perp$, there is a loss of privacy. This argument is formalized via the distance to instability framework. Sparse vector helps account for the privacy loss across all the $\widetilde{m}$ queries.

Theorem 10. *Let $\alpha, \beta \in (0, 1)$, and $\gamma \triangleq \min_{h \in \mathcal{H}} L(h; \mathcal{D})$. (Note that in the realizable case $\gamma = 0$). In Algorithm $\mathcal{A}_{\mathsf{binClas}}$ (Algorithm 10), suppose we set the cutoff parameter as $T = 3\left((\gamma + \alpha)\widetilde{m} + \sqrt{(\gamma + \alpha)\widetilde{m}\log(\widetilde{m}/\beta)/2}\right)$. If $\mathcal{A}$ is an $(\alpha, \beta/b, m/b)$-agnostic PAC learner (Definition 5.1.2), where b is as defined in $\mathcal{A}_{\mathsf{binClas}}$, then i) with probability at least*

$1-2\beta$, *$\mathcal{A}_{\mathsf{binClas}}$ does not halt before answering all the $\widetilde{m}$ queries in $\mathcal{Q}$, and outputs $\perp$ for at most T queries; and ii) the misclassification rate of $\mathcal{A}_{\mathsf{binClas}}$ is at most $T/\widetilde{m} = O(\gamma+\alpha)$.*

Proof. First, notice that $\mathcal{A}$ is an $(\alpha, \beta/b, m/b)$-agnostic PAC learner, hence w.p. $\geq 1-\beta$, the misclassification rate of h_j for all $j \in [b]$ is at most $\gamma+\alpha$. So, by the standard Chernoff's bound, with probability at least $1-\beta$ none of the h_j's misclassify more than $(\gamma+\alpha)\widetilde{m} + \sqrt{(\gamma+\alpha)\widetilde{m}\log(\widetilde{m}/\beta)/2} \triangleq B$ queries in $\mathcal{Q}$. Now, we use the following Markov-style counting argument (Lemma 5.4.1) to bound the number of queries for which at least $b/3$ classifiers in the ensemble $\{h_1, \ldots, h_b\}$ result in a misclassification.

Lemma 5.4.1. *Consider a set of $\{(\tilde{x}_1, \tilde{y}_1), \ldots, (\tilde{x}_{\widetilde{m}}, \tilde{y}_{\widetilde{m}})\} \in (\mathcal{X}\times\mathcal{Y})^{\widetilde{m}}$, and b binary classifiers $h_1, \ldots, h_b$, where each classifier is guaranteed to make at most B mistakes in predicting the $\widetilde{m}$ labels $\{\tilde{y}_1, \ldots, \tilde{y}_{\widetilde{m}}\}$. For any $\xi \in (0, 1/2]$,*

$$\left|\left\{i \in [\widetilde{m}] : |\{j \in [b] : h_j(\tilde{x}_i) \neq \tilde{y}_i\}| > \xi b\right\}\right| < B/\xi.$$

Therefore, there are at most $3B$ queries $\tilde{x}_i \in \mathcal{Q}$, where the votes of the ensemble $\{h_1(\tilde{x}_i), \ldots, h_b(\tilde{x}_i)\}$ has number of ones (or, zeros) $> b/3$ (i.e., they significantly disagree). Now, to prove part (i) of the theorem, observe that to satisfy the distance to instability condition (in Theorem 6) for the remaining $\widetilde{m}-3B$ queries, it would suffice to have $b/3 \geq 32\log\left(4\widetilde{m}T/\min\left(\delta, \beta/2\right)\right)\sqrt{2T\log(2/\delta)}/\epsilon$ (taking into account the noise in the threshold passed to $\mathcal{A}_{\mathsf{stab}}$ in Step 8 of $\mathcal{A}_{\mathsf{binClas}}$). This condition on b is satisfied by the setting of b in $\mathcal{A}_{\mathsf{binClas}}$. For part (ii), note that by the same lemma above, w.p. $1-\beta$, there are at least $2b/3$ classifiers that output the correct label in each of the remaining $\widetilde{m}-3B$ queries. Hence, w.p. $\geq 1-2\beta$, Algorithm $\mathcal{A}_{\mathsf{binClas}}$ will correctly classify such queries. This completes the proof. □

Remark 11. *A natural question for using Theorem 10 in the agnostic case is that how*

would one know the value of γ in practice, in order to set the right value for T? One simple approach is to set aside half the training dataset, and compute the empirical misclassification rate with differential privacy to get a sufficiently accurate estimate for $\gamma + \alpha$ (as in standard validation techniques Shalev-Shwartz & Ben-David (2014)), and use it to set T. Since the sensitivity of misclassification rate is small, the amount of noise added would not affect the accuracy of the estimation. Furthermore, with a large enough training dataset, the asymptotics of Theorem 10 would not change either.

Explicit misclassification rate: In Theorem 10, it might seem that there is a circular dependency of the following terms: $T \to \alpha \to b \to T$. However, the number of independent relations is equal to the number of parameters, and hence, we can set them meaningfully to obtain non-trivial misclassification rates.

We now obtain an explicit misclassification rate for $\mathcal{A}_{\mathsf{binClas}}$ in terms of the VC-dimension of $\mathcal{H}$. Let V denote the VC-dimension of $\mathcal{H}$. First, we consider the realizable case ($\gamma = 0$). Our result for this case is formally stated in the following theorem.

Theorem 12 (Misclassification rate in the realizable case)**.** *For any $\beta \in (0, 1)$, there exists $M = \widetilde{\Omega}(\epsilon\, m/V)$, and a setting for $T = \widetilde{O}\left(\bar{m}^2\, V^2/\epsilon^2\, m^2\right)$, where $\bar{m} \triangleq \max(M, \widetilde{m})$, such that w.p. $\geq 1 - \beta$, $\mathcal{A}_{\mathsf{binClas}}$ yields the following misclassification rate: (i) $\widetilde{O}(V/\epsilon\, m)$ for up to M queries, and (ii) $\widetilde{O}\left(\widetilde{m} V^2/\epsilon^2\, m^2\right)$ for $\widetilde{m} > M$ queries.*

Proof. By standard uniform convergence arguments (Shalev-Shwartz & Ben-David (2014)), there is an $(\alpha, \beta, m/b)$-PAC learner with a misclassification rate of $\alpha = \widetilde{O}\left(bV/m\right)$. Setting T as in Theorem 10 with the aforementioned setting of α, and setting b as in Algorithm $\mathcal{A}_{\mathsf{binClas}}$ gives the setting of T in the theorem statement. For up to $\widetilde{m} = \widetilde{\Omega}(\epsilon\, m/V)$ queries, the setting of T becomes $T = O(1)$, and hence Theorem 10 implies $\mathcal{A}_{\mathsf{binClas}}$ yields a misclassification rate $\widetilde{O}(V/\epsilon\, m)$, which is es-

sentially the same as the optimal non-private rate. Beyond $\widetilde{\Omega}(\epsilon\, m/V)$ queries, $T = \widetilde{O}(\widetilde{m}^2\, V^2/\epsilon^2\, m^2)$, and hence, Theorem 10 implies that the misclassification rate of $\mathcal{A}_{\mathsf{binClas}}$ is $\widetilde{O}\left(\widetilde{m} V^2/\epsilon^2\, m^2\right)$. □

We note that the attainable misclassification rate is significantly smaller than the rate of $\widetilde{O}\left(\sqrt{\widetilde{m}} V/\epsilon\, m\right)$ implied by a direct application of the advanced composition theorem of differential privacy. Next, we provide analogous statement for the non-realizable case ($\gamma > 0$).

Theorem 13 (Misclassification rate in the non-realizable case)**.** *For any $\beta \in (0,1)$, there exists $M = \widetilde{\Omega}\left(\min\left\{1/\gamma, \sqrt{\epsilon\, m/V}\right\}\right)$, and $T = O(\bar{m}\gamma) + \widetilde{O}\left(\bar{m}^{4/3}\, V^{2/3}/\epsilon^{2/3}\, m^{2/3}\right)$, where $\bar{m} \triangleq \max\{M, \widetilde{m}\}$, such that w.p. $\geq 1-\beta$, Algorithm $\mathcal{A}_{\mathsf{binClas}}$ yields the following misclassification rate: (i) $O(\gamma) + \widetilde{O}\left(\sqrt{V/\epsilon\, m}\right)$ for up to M queries, and (ii) $O(\gamma) + \widetilde{O}\left(\widetilde{m}^{1/3}\, V^{2/3}/\epsilon^{2/3}\, m^{2/3}\right)$ for $\widetilde{m} > M$ queries.*

Proof. Again, by a standard argument, $\mathcal{A}$ is $(\alpha, \beta, m/b)$-agnostic PAC learner with $\alpha = \widetilde{O}\left(\sqrt{bV/m}\right)$, and hence, it has a misclassification rate of $\approx \gamma + \widetilde{O}\left(\sqrt{bV/m}\right)$ when trained on a dataset of size m/b. Setting T as in Theorem 10 with this value of α, and setting b as in $\mathcal{A}_{\mathsf{binClas}}$, and then solving for T in the resulting expression, we get the setting of T as in the theorem statement (it would help here to consider the cases where $\gamma > \alpha$ and $\gamma \leq \alpha$ separately). For up to $M = \widetilde{\Omega}\left(\min\left\{1/\gamma, \sqrt{\epsilon\, m/V}\right\}\right)$ queries, the setting of T becomes $T = O(1)$, and hence Theorem 10 implies $\mathcal{A}_{\mathsf{binClas}}$ yields a misclassification rate of $O(\gamma) + \widetilde{O}\left(\sqrt{V/\epsilon\, m}\right)$, which is essentially the same as the optimal non-private rate. Beyond M queries, we have that $T = O(\widetilde{m}\gamma) + \widetilde{O}\left(\widetilde{m}^{4/3}\, V^{2/3}/\epsilon^{2/3}\, m^{2/3}\right)$. Hence, Theorem 10 implies that the misclassification rate of $\mathcal{A}_{\mathsf{binClas}}$ is $O(\gamma) + \widetilde{O}\left(\widetilde{m}^{1/3}\, V^{2/3}/\epsilon^{2/3}\, m^{2/3}\right)$. □

5.5 ANSWERING QUERIES TO MODEL-AGNOSTIC PRIVATE LEARNING

In this section, we build on our algorithm and results in Section 5.4 to achieve a stronger objective. In particular, we bootstrap from our previous algorithm an (ϵ, δ)-differentially private learner that publishes a final classifier. The idea is based on a knowledge transfer technique: we use our private construction above to generate labels for sufficient number of unlabeled domain points. Then, we use the resulting labeled set as a new training set for any standard (non-private) learner, which in turn outputs a classifier. We prove explicit sample complexity bounds for the final private learner in both PAC and agnostic PAC settings.

Our final construction can also be viewed as a private learner in the less restrictive setting of label-private learning where the learner is only required to protect the privacy of the labels in the training set. Note that any construction for our original setting can be used as a label-private learner simply by splitting the training set into two parts and throwing away the labels of one of them.

Let h_{priv} denote the mapping defined by $\mathcal{A}_{\mathsf{binClas}}$ (Algorithm 10) on a single query (unlabeled data point). That is, for $x \in \mathcal{X}$, let $h_{\mathsf{priv}}(x) \in \{0, 1, \perp\}$ denote the output of $\mathcal{A}_{\mathsf{binClas}}$ on a single input query x. Note that w.l.o.g., we can view h_{priv} as a binary classifier by replacing $\perp$ with a uniformly random label in $\{0, 1\}$. Our private learner is described in Algorithm 11 below.

Algorithm 11 $\mathcal{A}_{\mathsf{Priv}}$: Private Learner

Input: Unlabeled set of $\widetilde{m}$ i.i.d. feature vectors: $\mathcal{Q} = \{\tilde{x}_1, \ldots, \tilde{x}_{\widetilde{m}}\}$, oracle access to our private classifier h_{priv}, oracle access to an agnostic PAC learner $\mathcal{A}$ for a class $\mathcal{H}$.

1: **for** $t = 1, \ldots, \widetilde{m}$ **do**
2: $\quad \hat{y}_t \leftarrow h_{\mathsf{priv}}(\tilde{x}_t)$
3: **Output** $\hat{h} \leftarrow \mathcal{A}(\tilde{D})$, where $\tilde{D} = \{(\tilde{x}_1, \hat{y}_1), \ldots, (\tilde{x}_{\widetilde{m}}, \hat{y}_{\widetilde{m}})\}$

Note that since differential privacy is closed under post-processing, $\mathcal{A}_{\mathsf{Priv}}$ is (ϵ, δ)-DP w.r.t. the original dataset (input to $\mathcal{A}_{\mathsf{binClas}}$). Note also that the mapping h_{priv} is independent of $\mathcal{Q}$; it only depends on the input training set D (in particular, on $h_1, \ldots, h_b$), and the internal randomness of $\mathcal{A}_{\mathsf{binClas}}$. We now make the following claim about h_{priv}.

Claim 5.5.1. *Let $0 < \beta \leq \alpha < 1$, and $\widetilde{m} \geq 4\log(1/\alpha\beta)/\alpha$. Suppose that $\mathcal{A}$ in $\mathcal{A}_{\mathsf{binClas}}$ (Algorithm 10) is an $(\alpha, \beta/b, m/b)$-(agnostic) PAC learner for the hypothesis class $\mathcal{H}$. Then, with probability at least $1 - 2\beta$ (over the randomness of the private training set D, and the randomness in $\mathcal{A}_{\mathsf{binClas}}$), we have $L(h_{\mathsf{priv}}; \mathcal{D}) \leq 3\gamma + 7\alpha = O(\gamma + \alpha)$, where $\gamma = \min_{h \in \mathcal{H}} L(h; \mathcal{D})$.*

Proof. The proof largely relies on the proof of Theorem 10. First, note that w.p. $\geq 1 - \beta$ (over the randomness of the input dataset D), for all $j \in [b]$, we have $L(h_j; \mathcal{D}) \leq \alpha$. For the remainder of the proof, we will condition on this event. Let $\tilde{x}_1, \ldots, \tilde{x}_{\widetilde{m}}$ be a sequence of i.i.d. domain points, and $\tilde{y}_1, \ldots, \tilde{y}_{\widetilde{m}}$ be the corresponding (unknown) labels. Now, define $v_t \triangleq \mathbf{1}\left(|\{j \in [b] : h_j(\tilde{x}_t) \neq \tilde{y}_t\}| > b/3\right)$ for every $t \in [\widetilde{m}]$. Note that since $(\tilde{x}_1, \tilde{y}_1), \ldots, (\tilde{x}_{\widetilde{m}}, \tilde{y}_{\widetilde{m}})$ are i.i.d., it follows that $v_1, \ldots, v_{\widetilde{m}}$ are i.i.d. (this is true conditioned on the original dataset D). As in the proof of Theorem 10, we have: $\mathop{\mathbb{P}}_{\tilde{x}_1, \ldots, \tilde{x}_{\widetilde{m}}}\left[\frac{1}{\widetilde{m}}\sum_{t=1}^{\widetilde{m}} v_t > 3\left(\alpha + \gamma + \sqrt{\frac{\log(\widetilde{m}/\beta)}{2\widetilde{m}(\alpha+\gamma)}}\right)\right] < \beta$. Hence, for any $t \in [\widetilde{m}]$, we have $\mathop{\mathbb{E}}_{\tilde{x}_t}[v_t] = \mathop{\mathbb{E}}_{\tilde{x}_1, \ldots, \tilde{x}_{\widetilde{m}}}\left[\frac{1}{\widetilde{m}}\sum_{t=1}^{\widetilde{m}} v_t\right] < \beta + 3\left(\alpha + \gamma + \sqrt{\frac{\log(\widetilde{m}/\beta)}{2\widetilde{m}(\alpha+\gamma)}}\right) \leq 7\alpha + 3\gamma$. Let $\bar{v}_t = 1 - v_t$. Using the same technique as in the proof of Theorem 10, we can show that w.p. at least $1 - \beta$ over the internal randomness in Algorithm 10, we have $\bar{v}_t = 1 \Rightarrow h_{\mathsf{priv}}(\tilde{x}_t) = \tilde{y}_t$. Hence, conditioned on this event, we have $\mathop{\mathbb{P}}_{\tilde{x}_t}[h_{\mathsf{priv}}(\tilde{x}_t) \neq \tilde{y}_t] \leq \mathop{\mathbb{P}}_{\tilde{x}_t}[v_t = 1] = \mathop{\mathbb{E}}_{\tilde{x}_t}[v_t] \leq 7\alpha + 3\gamma$. □

We now state and prove the main results of this section. Let V denote the VC-dimension of $\mathcal{H}$.

Theorem 14 (Sample complexity bound in the realizable case)**.** *Let $0 < \beta \leq \alpha < 1$. Let $\widetilde{m}$ be such that $\mathcal{A}$ is an $(\alpha, \beta, \widetilde{m})$-agnostic PAC learner of $\mathcal{H}$, i.e., $\widetilde{m} = O\left(\frac{V+\log(1/\beta)}{\alpha^2}\right)$. Let the parameter T of $\mathcal{A}_{\mathsf{binClas}}$ (Algorithm 10) be set as in Theorem 12. There exists $m = \widetilde{O}\left(V^{3/2}/\epsilon\,\alpha^{3/2}\right)$ for the size of the private dataset such that, w.p. $\geq 1 - 3\beta$, the output hypothesis $\hat{h}$ of $\mathcal{A}_{\mathsf{Priv}}$ (Algorithm 11) satisfies $L(\hat{h}; \mathcal{D}) = O(\alpha)$.*

Proof. Let $h^* \in \mathcal{H}$ denote the true labeling hypothesis. We will denote the true distribution $\mathcal{D}$ as $(\mathcal{D}_{\mathcal{X}}, h^*)$. Note that since T is set as in Theorem 12, and given the value of $\widetilde{m}$ in the theorem statement, we get $b = \widetilde{O}(V^2/\epsilon^2\,\alpha^2 m)$. Hence, there is a setting $m = \widetilde{O}\left(V^{3/2}/\epsilon\,\alpha^{3/2}\right)$ such that $\mathcal{A}$ is an $(\alpha, \beta/b, m/b)$-PAC learner for $\mathcal{H}$ (in particular, sample complexity in the realizable case $= m/b = \widetilde{O}(V/\alpha)$). Hence, by Claim 5.5.1, w.p. $\geq 1 - 2\beta$, we get $L(h_{\mathsf{priv}}; \mathcal{D}) \leq 7\alpha$. For the remainder of the proof, we will condition on this event. Let $\tilde{D} = \{(x_1, \hat{y}_1), \ldots, (x_{\widetilde{m}}, \hat{y}_{\widetilde{m}})\}$ be the *new* training set generated by $\mathcal{A}_{\mathsf{Priv}}$ (Algorithm 11), where $\widetilde{m}$ is set as in the theorem statement. Note that each $(\tilde{x}_t, \hat{y}_t),\ t \in [\widetilde{m}]$, is drawn independently from $(\mathcal{D}_{\mathcal{X}}, h_{\mathsf{priv}})$. Now, since $\mathcal{A}$ is also an $(\alpha, \beta, \widetilde{m})$-agnostic PAC learner for $\mathcal{H}$, w.p. $\geq 1 - \beta$ (over the *new* set $\tilde{D}$), the output hypothesis $\hat{h}$ satisfies

$$L(\hat{h}; (\mathcal{D}_{\mathcal{X}}, h_{\mathsf{priv}})) - L(h^*; (\mathcal{D}_{\mathcal{X}}, h_{\mathsf{priv}})) \leq L(\hat{h}; (\mathcal{D}_{\mathcal{X}}, h_{\mathsf{priv}})) - \min_{h \in \mathcal{H}} L(h; (\mathcal{D}_{\mathcal{X}}, h_{\mathsf{priv}})) \leq \alpha.$$

Observe that

$$\begin{aligned} L(h^*; (\mathcal{D}_{\mathcal{X}}, h_{\mathsf{priv}})) &= \mathop{\mathbb{E}}_{x \sim \mathcal{D}_{\mathcal{X}}} \left[\mathbf{1}\left(h^*(x) \neq h_{\mathsf{priv}}(x)\right)\right] = L(h_{\mathsf{priv}}; (\mathcal{D}_{\mathcal{X}}, h^*)) = L(h_{\mathsf{priv}}; \mathcal{D}) \\ &\leq 7\alpha \end{aligned}$$

where the last inequality follows from Claim 5.5.1 (with $\gamma = 0$). Hence, we get

$L(\hat{h}; (\mathcal{D}_{\mathcal{X}}, h_{\mathsf{priv}})) \leq 8\alpha$. Furthermore, observe that

$$\begin{aligned} L(\hat{h}; \mathcal{D}) &= \mathop{\mathbb{E}}_{x \sim \mathcal{D}_{\mathcal{X}}} \left[\mathbf{1}(\hat{h}(x) \neq h^*(x)) \right] \leq \mathop{\mathbb{E}}_{x \sim \mathcal{D}_{\mathcal{X}}} \left[\mathbf{1}(\hat{h}(x) \neq h_{\mathsf{priv}}(x)) + \mathbf{1}(h_{\mathsf{priv}}(x) \neq h^*(x)) \right] \\ &= L(\hat{h}; (\mathcal{D}_{\mathcal{X}}, h_{\mathsf{priv}})) + L(h_{\mathsf{priv}}; \mathcal{D}) \leq 15\alpha. \end{aligned}$$

Hence, w.p. $\geq 1 - 3\beta$, we have $L(\hat{h}; \mathcal{D}) \leq 15\alpha$. □

Remark 15. *In Theorem 14, if $\mathcal{A}$ is an ERM learner, then the value of $\widetilde{m}$ can be reduced to $\widetilde{O}(V/\alpha)$. Hence, the resulting sample complexity would be $m = \widetilde{O}(V^{3/2}/\epsilon\,\alpha)$, saving us a factor of $\frac{1}{\sqrt{\alpha}}$. This is because the disagreement rate in the labels produced by $\mathcal{A}_{\mathsf{binClas}}$ is $\approx \alpha$, and agnostic learning with such a low disagreement rate can be done using $\widetilde{O}(V/\alpha)$ if the learner is an ERM (Boucheron et al., 2005, Corollary 5.2).*

Remark 16. *Our result involves using an agnostic PAC learner $\mathcal{A}$. Agnostic PAC learners with optimal sample complexity can be computationally inefficient. One way to give an efficient construction in the realizable case (with a slightly worse sample complexity) is to use a PAC learner (rather than an agnostic one) in $\mathcal{A}_{\mathsf{Priv}}$ with target accuracy α (and hence, $\widetilde{m} = \widetilde{O}(V/\alpha)$), but then train the PAC learner in $\mathcal{A}_{\mathsf{binClas}}$ towards a target accuracy $1/\widetilde{m}$. Hence, the misclassification rate of $\mathcal{A}_{\mathsf{binClas}}$ can be driven to zero. This yields a sample complexity bound $m = \widetilde{O}(V^2/\epsilon\,\alpha)$.*

Theorem 17 (Sample complexity bound in the non-realizable case)**.** *Let $0 < \beta \leq \alpha < 1$, and $\widetilde{m} = O\left(\frac{V + \log(1/\beta)}{\alpha^2}\right)$. Let T be set as in Theorem 13. There exists $m = \widetilde{O}\left(V^{3/2}/\epsilon\,\alpha^{5/2}\right)$ such that, w.p. $\geq 1 - 3\beta$, the output hypothesis $\hat{h}$ of (Algorithm 11) satisfies $L(\hat{h}; \mathcal{D}) = O(\alpha + \gamma)$.*

Proof. The proof is similar to the proof of Theorem 14. □

5.6 DISCUSSION

Implications, and comparison to prior work on label privacy: Our results also apply to the setting of label-private learning, where the learner is only required to protect the privacy of the labels in the training set. That is, in this setting, all unlabeled features in the training set can be viewed as public information. This is a less restrictive setting than the setting we consider in this chapter. In particular, our construction can be directly used as a label-private learner simply by splitting the training set into two parts and discarding the labels in one of them. The above theorems give sample complexity upper bounds that are only a factor of $\widetilde{O}\left(\sqrt{V/\alpha}\right)$ worse than the optimal non-private sample complexity bounds. We note, however, that our sample complexity upper bound for the agnostic case has a suboptimal dependency (by a small constant factor) on $\gamma \triangleq \min_{h\in\mathcal{H}} L(h;\mathcal{D})$.
Label-private learning has been considered before in Chaudhuri & Hsu (2011) and Beimel et al. (2016). Both works have only considered pure, i.e., $(\epsilon, 0)$, differentially private learners for those settings, and the constructions in both works are white-box, i.e., they do not allow for modular construction based on a black-box access to a non-private learner. The work of Chaudhuri & Hsu (2011) gave upper and lower bounds on the sample complexity in terms of the doubling dimension. Their upper bound involves a smoothness condition on the distribution of the features $\mathcal{D}_{\mathcal{X}}$. The work of Beimel et al. (2016) showed that the sample complexity (of pure differentially label-private learners) can be characterized in terms of the VC dimension. They proved an upper bound on the sample complexity for the realizable case. The bound of Beimel et al. (2016) is only a factor of $O(1/\alpha)$ worse than the optimal non-private bound for the realizable case.

Beyond standard PAC learning with binary loss: In this chapter, we used our algorithmic framework to derive sample complexity bounds for the standard (agnostic) PAC model with the binary 0-1 loss. However, it is worth pointing out that our framework is applicable in more general settings. In particular, if a surrogate loss (e.g., hinge loss or logistic loss) is used instead of the binary loss, then our framework can be instantiated with any non-private learner with respect to that loss. That is, our construction does not necessarily require an (agnostic) PAC learner. However, in such case, the accuracy guarantees of our construction will be different from what we have here for the standard PAC model. In particular, in the surrogate loss model, one often needs to invoke some weak assumptions on the data distribution in order to bound the optimization error (Shalev-Shwartz & Ben-David (2014)). One can still provide meaningful accuracy guarantees since our framework allows for transferring the classification error guarantee of the underlying non-private learner to a classification error guarantee for the final private learner.

CHAPTER 6

Private Matrix Completion

In this chapter, we will look at the first provably differentially private algorithm with formal utility guarantees for the problem of user-level privacy-preserving matrix completion.

6.1 ADDITIONAL PRELIMINARIES

6.1.1 Notions of Privacy

Let $D = \{d_1, \cdots, d_m\}$ be a dataset of m entries. Each entry d_i lies in a fixed domain $\mathcal{T}$, and belongs to an individual i, whom we refer to as an *agent* in this chapter. Furthermore, d_i encodes potentially sensitive information about agent i. Let $\mathcal{A}$ be an algorithm that operates on dataset D, and produces a vector of m outputs, one for each agent i, from a set of possible outputs $\mathcal{S}$. Formally, let $\mathcal{A} : \mathcal{T}^m \to \mathcal{S}^m$. Let D_{-i} denote the dataset D without the entry of the i-th agent, and similarly $\mathcal{A}_{-i}(D)$ be the set of outputs without the output for the i-th agent. Also, let $(d_i; D_{-i})$ denote the dataset obtained by adding data entry d_i to the dataset D_{-i}. In this chapter, we will use the notion of neighboring under modification (Definition 2.1.1) for the guarantee of privacy.

At a high-level, an algorithm $\mathcal{A}$ is (ϵ, δ)-standard DP (Definition 2.1.3) if for any agent i and dataset D, the output $\mathcal{A}(D)$ and D_{-i} do not reveal "much" about her data entry d_i. For reasons mentioned in Section 3.4, our matrix completion algorithms provide a privacy guarantee based on a relaxed notion of DP, called *joint differential privacy* , which was initially proposed in Kearns et al. (2014).

Definition 18 (Joint differential privacy (Kearns et al. (2014))). *An algorithm $\mathcal{A}$ sat-*

isfies (ϵ, δ)-joint differential privacy *if for any agent* i*, any two possible values of data entry* $d_i, d_i' \in \mathcal{T}$ *for agent* i*, any tuple of data entries for all other agents,* $D_{-i} \in \mathcal{T}^{m-1}$*, and any output* $S \in \mathcal{S}^{m-1}$*,*

$$\Pr_{\mathcal{A}}\left[\mathcal{A}_{-i}\left(d_i; D_{-i}\right) \in S\right] \leq e^{\epsilon} \Pr_{\mathcal{A}}\left[\mathcal{A}_{-i}\left(d_i'; D_{-i}\right) \in S\right] + \delta.$$

Intuitively, an algorithm $\mathcal{A}$ preserves (ϵ, δ)-joint differential privacy if for any agent i and dataset D, the output of $\mathcal{A}$ for the other $(m-1)$ agents (denoted by $\mathcal{A}_{-i}(D)$) and D_{-i} do not reveal "much" about her data entry d_i. Such a relaxation is necessary for matrix completion because an accurate completion of the row of an agent can reveal a lot of information about her data entry. However, it is still a very strong privacy guarantee for an agent even if every other agent colludes against her, as long as she does not make the predictions made to her public.

In this chapter, we consider the privacy parameter ϵ to be a small constant (≈ 0.1), and $\delta < 1/m$. There are semantic reasons for such choice of parameters (Kasiviswanathan & Smith (2008)), but that is beyond the scope of this chapter.

6.1.2 The Frank-Wolfe Algorithm

We use the classic Frank-Wolfe algorithm (Frank & Wolfe (1956)) as one of the optimization building blocks for our differentially private algorithms. In Algorithm 12, we state the Frank-Wolfe method to solve the following convex optimization problem:

$$\widehat{Y} = \arg \min_{\|Y\|_{\text{nuc}} \leq k} \frac{1}{2|\Omega|} \left\|\mathsf{P}_{\Omega}\left(Y - Y^{*}\right)\right\|_F^2. \tag{6.1}$$

In this chapter, we use the approximate version of the algorithm from Jaggi (2013). The only difference is that, instead of using an exact minimizer to the linear opti-

Algorithm 12 Approximate Frank-Wolfe algorithm

Input: Set of revealed entries: Ω, operator: P_Ω, matrix: $\mathsf{P}_\Omega(Y^*) \in \mathbb{R}^{m\times n}$, nuclear norm constraint: k, time bound: T, slack parameter: γ
$Y^{(0)} \leftarrow \{0\}^{m\times n}$
for $t \in [T]$ **do**
  $W^{(t-1)} \leftarrow \frac{1}{|\Omega|}\mathsf{P}_\Omega\left(Y^{(t-1)} - Y^*\right)$
  Get $Z^{(t-1)}$ with $\left\|Z^{(t-1)}\right\|_{\mathsf{nuc}} \leq k$ s.t. $\left(\langle W^{(t-1)}, Z^{(t-1)}\rangle - \min_{\|\Theta\|_{\mathsf{nuc}}\leq k}\langle W^{(t-1)}, \Theta\rangle\right) \leq \gamma$
  $Y^{(t)} \leftarrow \left(1 - \frac{1}{T}\right) Y^{(t-1)} + \frac{Z^{(t-1)}}{T}$
Return $Y^{(T)}$

mization problem, Line 12 of Algorithm 12 uses an oracle that minimizes the problem up to a slack of γ. In the following, we provide the convergence guarantee for Algorithm 12.

Note: Observe that the algorithm converges at the rate of $O(1/T)$ even with an error slack of γ. While such a convergence rate is sufficient for us to prove our utility guarantees, we observe that this rate is rather slow in practice.

Theorem 19 (Utility guarantee)**.** *Let γ be the slack in the linear optimization oracle in Line 12 of Algorithm 12. Then, following is true for $Y^{(T)}$:*

$$L\left(Y^{(T)};\Omega\right) - \min_{\|Y\|_{\mathsf{nuc}}\leq k} L\left(Y;\Omega\right) \leq \frac{k^2}{|\Omega|T} + \gamma.$$

Proof (Adapted from Jaggi (2013)). Let $\mathcal{D} \in \mathbb{R}^{m\times n}$ some fixed domain. We will define the curvature parameter C_f of any differentiable function $f : \mathcal{D} \to \mathbb{R}$ to be the following:

$$C_f = \max_{\substack{x,s\in\mathcal{D},\mu\in[0,1]:\\ y=x+\mu(s-x)}} \frac{2}{\mu^2}\left(f(y) - f(x) - \langle y - x, \nabla f(x)\rangle\right).$$

In the optimization problem in (6.1), let $f(Y) = \frac{1}{2|\Omega|}\left\|\mathsf{P}_\Omega\left(Y - Y^*\right)\right\|_F^2$, and

$G^{(t-1)} = \arg\min_{\|\Theta\|_{\mathsf{nuc}} \leq k} \langle W^{(t-1)}, \Theta \rangle$, where $W^{(t-1)}$ is as defined in Line 3 of Algorithm 12. We now have the following due to smoothness:

$$\begin{aligned} f\left(Y^{(t)}\right) &= f\left(Y^{(t-1)} + \frac{1}{T}\left(Z^{(t-1)} - Y^{(t-1)}\right)\right) \\ &\leq f\left(Y^{(t-1)}\right) + \frac{1}{2T^2} C_f + \frac{1}{T} \langle Z^{(t-1)} - Y^{(t-1)}, \bigtriangledown f\left(Y^{(t-1)}\right) \rangle. \end{aligned} \tag{6.2}$$

Now, by the γ-approximation property in Line 4 of Algorithm 12, we have:

$$\langle Z^{(t-1)} - Y^{(t-1)}, \bigtriangledown f\left(Y^{(t-1)}\right) \rangle \leq \langle G^{(t-1)} - Y^{(t-1)}, \bigtriangledown f\left(Y^{(t-1)}\right) \rangle + \gamma.$$

Therefore, we have the following from Equation (6.2):

$$f\left(Y^{(t)}\right) \leq f\left(Y^{(t-1)}\right) + \frac{C_f}{2T^2}\left(1 + \frac{2T\gamma}{C_f}\right) + \frac{1}{T} \langle G^{(t-1)} - Y^{(t-1)}, \bigtriangledown f\left(Y^{(t-1)}\right) \rangle. \tag{6.3}$$

Recall the definition of $\widehat{Y}$ from (6.1), and let $h(\Theta) = f(\Theta) - f(\widehat{Y})$. By convexity, we have the following (also called the duality gap):

$$\langle Y^{(t)} - G^{(t)}, \bigtriangledown f\left(Y^{(t)}\right) \rangle \geq h\left(Y^{(t)}\right). \tag{6.4}$$

Therefore, from (6.3) and (6.4), we have the following:

$$\begin{aligned} h\left(Y^{(T)}\right) &\leq h\left(Y^{(T-1)}\right) - \frac{h\left(Y^{(T-1)}\right)}{T} + \frac{C_f}{2T^2}\left(1 + \frac{2T\gamma}{C_f}\right) \\ &= \left(1 - \frac{1}{T}\right) h\left(Y^{(T-1)}\right) + \frac{C_f}{2T^2}\left(1 + \frac{2T\gamma}{C_f}\right) \\ &\leq \frac{C_f}{2T^2}\left(1 + \frac{2T\gamma}{C_f}\right) \cdot \left(1 + \left(1 - \frac{1}{T}\right) + \left(1 - \frac{1}{T}\right)^2 + \cdots\right) \end{aligned}$$

$$\leq \frac{C_f}{2T}\left(1+\frac{2T\gamma}{C_f}\right) = \frac{C_f}{2T}+\gamma$$

$$\Leftrightarrow f\left(Y^{(T)}\right) - f\left(\widehat{Y}\right) \leq \frac{C_f}{2T}+\gamma. \tag{6.5}$$

With the above equation in hand, we bound the term C_f for the stated $f(\Theta)$ to complete the proof. Notice that $\frac{2k^2}{|\Omega|}$ is an upper bound on the curvature constant C_f (See Lemma 1 from Shalev-Shwartz et al. (2011), or Section 2 of Clarkson (2010), for a proof). Therefore, from (6.5), we get:

$$f\left(Y^{(T)}\right) - f\left(\widehat{Y}\right) \leq \frac{k^2}{|\Omega|T}+\gamma,$$

which completes the proof. □

6.2 PRIVATE MATRIX COMPLETION VIA FRANK-WOLFE

Recall that the objective is to solve the matrix completion problem (defined in Section 3.4.1) under Joint DP. A standard modeling assumption is that Y^* is nearly low-rank, leading to the following empirical risk minimization problem (Keshavan et al. (2010); Jain et al. (2013); Jin et al. (2016)): $\min_{\mathsf{rank}(Y)\leq k} \underbrace{\frac{1}{2|\Omega|}\|\mathsf{P}_\Omega(Y-Y^*)\|_F^2}_{L(Y;\Omega)}$, where $k \ll \min(m,n)$. As this is a challenging non-convex optimization problem, a popular approach is to relax the rank constraint to a nuclear-norm constraint, i.e., $\min_{\|Y\|_{\mathsf{nuc}}\leq k} L(Y;\Omega)$.

To this end, we use the FW algorithm (Algorithm 12) as our building block. FW is a popular conditional gradient algorithm in which the current iterate is updated as: $Y^{(t)} \leftarrow (1-\eta)Y^{(t-1)} + \eta \cdot G$, where η is the step size, and G is given by: $\arg\min_{\|G\|_{\mathsf{nuc}}\leq k} \langle G, \nabla_{Y^{(t-1)}} L(Y;\Omega)\rangle$. Note that the optimal solution to the above problem is $G = -k\mathbf{u}\mathbf{v}^\top$, where ($\lambda$, $\mathbf{u}$, $\mathbf{v}$) are the top singular components of

$A^{(t-1)} = \mathsf{P}_\Omega(Y^{(t-1)} - Y^*)$. Also, the optimal G is a rank-one matrix.

Algorithmic ideas: In order ensure Joint DP and still have strong error guarantees, we develop the following ideas. These ideas have been formally compiled into Algorithm 13. Notice that both the functions $\mathcal{A}_{\mathsf{global}}$ and $\mathcal{A}_{\mathsf{local}}$ in Algorithm 13 are parts of the Private FW technique, where $\mathcal{A}_{\mathsf{global}}$ consists of the global component, and each user runs $\mathcal{A}_{\mathsf{local}}$ at her end to carry out a local update. Throughout this discussion, we assume that $\max_{i\in[m]} \|\mathsf{P}_\Omega(Y_i^*)\|_2 \leq \Delta$.

Splitting the update into global and local components: One can equivalently write the Frank-Wolfe update as follows: $Y^{(t)} \leftarrow (1-\eta)Y^{(t-1)} - \eta \cdot \frac{k}{\lambda} A^{(t-1)}\mathbf{v}\mathbf{v}^\top$, where $A^{(t-1)}, \mathbf{v}$, and λ are defined as above. Note that $\mathbf{v}$ and λ^2 can also be obtained as the top right eigenvector and eigenvalue of $A^{(t-1)^\top}A^{(t-1)} = \sum_{i=1}^{m} {A_i}^{(t-1)^\top} {A_i}^{(t-1)}$, where ${A_i}^{(t-1)} = \mathsf{P}_\Omega({Y_i}^{(t-1)} - {Y_i}^*)$ is the i-th row of $A^{(t-1)}$. We will use the *global component* $\mathcal{A}_{\mathsf{global}}$ in Algorithm 13 to compute $\mathbf{v}$ and λ. Using the output of $\mathcal{A}_{\mathsf{global}}$, each user (row) $i \in [m]$ can compute her *local update* (using $\mathcal{A}_{\mathsf{local}}$) as follows:

$$ {Y_i}^{(t)} = (1-\eta){Y_i}^{(t-1)} - \frac{\eta k}{\lambda}\mathsf{P}_\Omega(Y^{(t-1)} - Y^*)_i \mathbf{v}\mathbf{v}^\top. \tag{6.6} $$

A block schematic of this idea is presented in Figure 6.1.

Noisy rank-one update: Observe that $\mathbf{v}$ and λ, the statistics computed in each iteration of $\mathcal{A}_{\mathsf{global}}$, are aggregate statistics that use information from all rows of Y^*. This ensures that they are noise tolerant. Hence, adding sufficient noise can ensure standard DP (Definition 2.1.3) for $\mathcal{A}_{\mathsf{global}}$. The second term in computing $\widehat{\lambda}'$ in Algorithm 13 is due to a bound on the spectral norm of the Gaussian noise matrix. We use this bound to control the error introduced in the computation of $\hat{\lambda}$. Since the final objective is to satisfy Joint DP (Definition 18), the local component

Algorithm 13 Private Frank-Wolfe algorithm

function Global Component $\mathcal{A}_{\mathsf{global}}$ **(Input-** privacy parameters: (ϵ, δ) s.t. $\epsilon \leq 2\log(1/\delta)$, total number of iterations: T, bound on $\|\mathsf{P}_\Omega(Y_i^*)\|_2$: Δ, failure probability: β, number of users: m, number of items: n**)**

$\sigma \leftarrow \Delta^2\sqrt{64 \cdot T\log(1/\delta)}/\epsilon$, $\widehat{\mathbf{v}} \leftarrow \{0\}^n$, $\widehat{\lambda} \leftarrow 0$

for $t \in [T]$ **do**

$W^{(t)} \leftarrow \{0\}^{n\times n}$, $\widehat{\lambda}' \leftarrow \widehat{\lambda} + \sqrt{\sigma\log(n/\beta)}n^{1/4}$

for $i \in [m]$ **do** $W^{(t)} \leftarrow W^{(t)} + \mathcal{A}_{\mathsf{local}}(i, \widehat{\mathbf{v}}, \widehat{\lambda}', T, t, \Delta)$

$\widehat{W}^{(t)} \leftarrow W^{(t)} + N^{(t)}$, where $N^{(t)} \in \mathbb{R}^{n\times n}$ is a matrix with i.i.d. entries from $\mathcal{N}(0, \sigma^2)$

$(\widehat{\mathbf{v}}, \widehat{\lambda}^2) \leftarrow$ Top eigenvector and eigenvalue of $\widehat{W}^{(t)}$

function Local Update $\mathcal{A}_{\mathsf{local}}$ **(Input-** user number: i, top right singular vector: $\widehat{\mathbf{v}}$, top singular value: $\widehat{\lambda}'$, total number of iterations: T, current iteration: t, bound on $\|\mathsf{P}_\Omega(Y_i^*)\|_2$: Δ, private true matrix row: $\mathsf{P}_\Omega(Y_i^*)$**)**

${Y_i}^{(0)} \leftarrow \{0\}^n$, ${A_i}^{(t-1)} \leftarrow \mathsf{P}_\Omega({Y_i}^{(t-1)} - Y_i^*)$

$\widehat{u}_i \leftarrow ({A_i}^{(t-1)} \cdot \widehat{\mathbf{v}})/\widehat{\lambda}'$

Define $\Pi_{\Delta,\Omega}(M)_{i,j} = \min\left\{\frac{\Delta}{\|\mathsf{P}_\Omega(M_i)\|_2}, 1\right\} \cdot M_{i,j}$

${Y_i}^{(t)} \leftarrow \Pi_{\Delta,\Omega}\left(\left(1 - \frac{1}{T}\right){Y_i}^{(t-1)} - \frac{k}{T}\widehat{u}_i(\widehat{\mathbf{v}})^T\right)$

${A_i}^{(t)} \leftarrow \mathsf{P}_\Omega\left({Y_i}^{(t)} - Y_i^*\right)$

if $t = T$, Output ${Y_i}^{(T)}$ as prediction to user i and **stop**

else Return ${{A_i}^{(t)}}^\top {A_i}^{(t)}$ to $\mathcal{A}_{\mathsf{global}}$

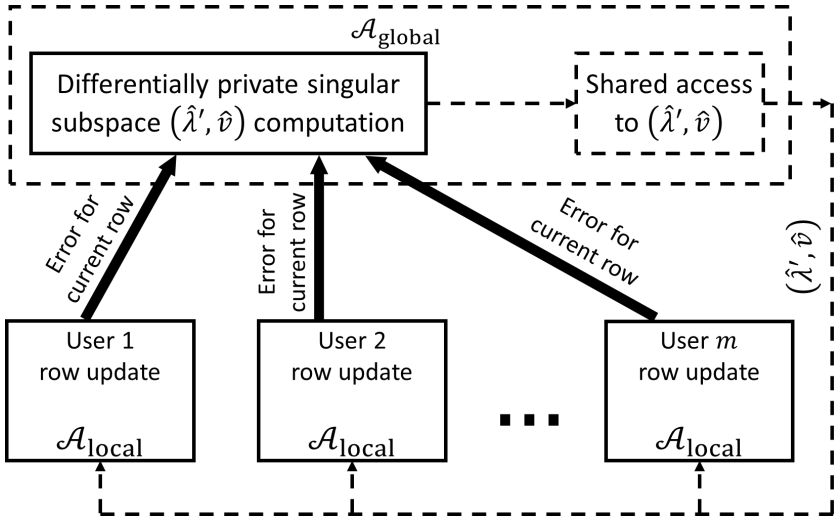

Figure 6.1: Block schematic describing the two functions $\mathcal{A}_{\text{local}}$ and $\mathcal{A}_{\text{global}}$ of Algorithm 13. The solid boxes and arrows represent computations that are privileged and without external access, and the dotted boxes and arrows represent the unprivileged computations.

$\mathcal{A}_{\text{local}}$ can compute the update for each user (corresponding to (6.6)) without adding any noise.

Controlling norm via projection: In order to control the amount of noise needed to ensure DP, any individual data entry (here, any row of Y^*) should have a bounded effect on the aggregate statistic computed by $\mathcal{A}_{\text{global}}$. However, each intermediate computation $Y_i^{(t)}$ in (6.6) can have high L_2-norm even if $\|\mathsf{P}_\Omega(Y_i^*)\|_2 \leq \Delta$. We address this by applying a projection operator $\Pi_{\Delta,\Omega}$ (defined below) to $Y_i^{(t)}$, and compute the local update as $\Pi_{\Delta,\Omega}\left(Y_i^{(t)}\right)$ in place of (6.6). $\Pi_{\Delta,\Omega}$ is defined as follows: For any matrix M, $\Pi_{\Delta,\Omega}$ ensures that any row of the "zeroed out" matrix $\mathsf{P}_\Omega(M)$ does not have L_2-norm higher than Δ. Formally, $\Pi_{\Delta,\Omega}(M)_{i,j} = \min\left\{\frac{\Delta}{\|\mathsf{P}_\Omega(M_i)\|_2}, 1\right\} \cdot M_{i,j}$ for all entries (i,j) of M. In our analysis, we show that this projection operation does not increase the error.

6.2.1 Privacy and Utility Analysis

Theorem 20. *Algorithm 13 satisfies (ϵ, δ)-joint DP.*

Proof. Note that we require $\epsilon \leq 2\log\left(\frac{1}{\delta}\right)$ for inputs (ϵ, δ) in Algorithm 13. Assuming this is satisfied, let us consider the matrix sequence $W^{(1)}, \cdots, W^{(T)}$ produced by function $\mathcal{A}_{\mathsf{global}}$. Notice that if every user $i \in [m]$ knows this sequence, then she can construct her updates $Y_i^{(1)}, \cdots, Y_i^{(T)}$ by herself *independent* of any other user's data. Therefore, by post-processing (Lemma 2.1.4), it follows that as long as function $\mathcal{A}_{\mathsf{global}}$ satisfies (ϵ, δ)-differential privacy, one can ensure (ϵ, δ)-joint differential privacy for Algorithm 13, i.e., the combined pair of functions $\mathcal{A}_{\mathsf{global}}$ and $\mathcal{A}_{\mathsf{local}}$. (Recall that Lemma 2.1.4 states that any operation performed on the output of a differentially private algorithm, without accessing the raw data, remains differentially private with the same level of privacy.) Hence, Lemma 6.2.1 completes the proof of privacy.

Lemma 6.2.1. *For input parameters (ϵ, δ) such that $\epsilon \leq 2\log\left(\frac{1}{\delta}\right)$, let $W^{(t)}$ be the output in every iteration $t \in [T]$ of function $\mathcal{A}_{\mathsf{global}}$ in Algorithm 13. Then, $\mathcal{A}_{\mathsf{global}}$ is (ϵ, δ)-differentially private.*

Proof. We are interested in the function $\mathsf{Cov}(A^{(t)}) = {A^{(t)}}^\top A^{(t)}$, where we have $A^{(t)} = \mathsf{P}_\Omega\left(Y^{(t)} - Y^*\right)$. Since $\|\mathsf{P}_\Omega\left(Y^{(t)}\right)_i\|_2 \leq \Delta$ and $\|\mathsf{P}_\Omega\left(Y^*\right)_i\|_2 \leq \Delta$ for all rows $i \in [m]$, we have that the L_2-sensitivity of $\mathsf{Cov}(A^{(t)})$ is $4\Delta^2$. Recall that the L_2-sensitivity of Cov corresponds to the maximum value of $\|\mathsf{Cov}(A) - \mathsf{Cov}(A')\|_F$ for any two matrices A, A' in the domain, and differing in exactly one row. Using the zCDP guarantee of the Gaussian mechanism (Lemma 2.1.14), the composition property of zCDP (Lemma 2.1.12), and the relation between zCDP and approximate DP (Lemma 2.1.13), it follows that adding Gaussian noise with standard deviation $\sigma = \frac{\Delta^2\sqrt{64 \cdot T\log(1/\delta)}}{\epsilon}$ in each iteration of the global component of private Frank-Wolfe

(function $\mathcal{A}_{\mathsf{global}}$) ensures (ϵ, δ)-differential privacy for $\epsilon \leq 2\log(1/\delta)$. □

□

We now show that the empirical risk of our algorithm is close to the optimal as long as the number of users m is "large".

Theorem 21 (Excess empirical risk guarantee)**.** *Let Y^* be a matrix with $\|Y^*\|_{\mathsf{nuc}} \leq k$, and $\max_{i\in[m]} \|\mathsf{P}_\Omega(Y^*)_i\|_2 \leq \Delta$. Let $Y^{(T)}$ be a matrix, with its rows being $Y_i^{(T)}$ for all $i \in [m]$, computed by function $\mathcal{A}_{\mathsf{local}}$ in Algorithm 13 after T iterations. If $\epsilon \leq 2\log\left(\frac{1}{\delta}\right)$, then with probability at least 2/3 over the outcomes of Algorithm 13, the following is true:*

$$L\left(Y^{(T)};\Omega\right) = O\left(\frac{k^2}{|\Omega|T} + \frac{kT^{1/4}\Delta\sqrt{n^{1/2}\log^{1/2}(1/\delta)\log n}}{|\Omega|\sqrt{\epsilon}}\right).$$

Furthermore, if $T = \tilde{O}\left(\frac{k^{4/5}\epsilon^{2/5}}{n^{1/5}\Delta^{4/5}}\right)$, then $L\left(Y^{(T)};\Omega\right) = \tilde{O}\left(\frac{k^{6/5}n^{1/5}\Delta^{4/5}}{|\Omega|\epsilon^{2/5}}\right)$ after hiding poly-logarithmic terms.

Proof. Recall that in function $\mathcal{A}_{\mathsf{global}}$ of Algorithm 13, the matrix $\widehat{W}^{(t)}$ captures the total error covariance corresponding to all the users at a given time step t, i.e., ${A^{(t)}}^\top A^{(t)} = \sum_{i\in[m]} {A_i^{(t)}}^\top A_i^{(t)}$. Spherical Gaussian noise of appropriate scale is added to ensure that $\widehat{W}^{(t)}$ is computed under the constraint of differential privacy. Let $\widehat{\mathbf{v}}$ be the top eigenvector of $\widehat{W}^{(t)}$, and let $\widehat{\lambda}^2$ be the corresponding eigenvalue. In Lemma 6.2.2, we first show that $\widehat{\lambda}$ is a reasonable approximation to the energy of $A^{(t)}$ captured by $\widehat{\mathbf{v}}$, i.e., $\|A^{(t)}\widehat{\mathbf{v}}\|_2$. Furthermore, in Lemma 6.2.3 we show that $\widehat{\mathbf{v}}$ captures sufficient energy of the matrix $A^{(t)}$. Hence, we can conclude that one can use $\widehat{\mathbf{v}}$ as a proxy for the top right singular vector of $A^{(t)}$.

Lemma 6.2.2. *With probability at least* $1-\beta$*, the following is true:*

$$\|A^{(t)}\widehat{\mathbf{v}}\|_2 \leq \widehat{\lambda} + O\left(\sqrt{\sigma \log(n/\beta)\sqrt{n}}\right).$$

Proof. Let $E = \widehat{W}^{(t)} - {A^{(t)}}^\top A^{(t)}$, where the matrix $\widehat{W}^{(t)}$ is computed in iteration t of the function $\mathcal{A}_{\mathsf{global}}$. We have,

$$\begin{aligned}
\|A^{(t)}\widehat{\mathbf{v}}\|_2^2 &= \widehat{\mathbf{v}}^\top {A^{(t)}}^\top A^{(t)}\widehat{\mathbf{v}} \\
&= \widehat{\mathbf{v}}^\top \left({A^{(t)}}^\top A^{(t)} + E\right)\widehat{\mathbf{v}} - \widehat{\mathbf{v}}^\top E\,\widehat{\mathbf{v}} \\
&\leq \widehat{\lambda}^2 + \|E\|_2 \\
&\leq \widehat{\lambda}^2 + O\left(\sigma \log(n/\beta)\sqrt{n}\right) \text{ w.p. } \geq 1-\beta \qquad (6.7)
\end{aligned}$$

Inequality (6.7) follows from the spectral norm bound on the Gaussian matrix E drawn i.i.d. from $\mathcal{N}(0,\sigma^2)$. (See Corollary 2.3.5 in Tao (2012) for a proof). The statement of the lemma follows from inequality (6.7). □

Lemma 6.2.3 (Follows from Theorem 3 of Dwork et al. (2014b))**.** *Let* $A \in \mathbb{R}^{m\times n}$ *be a matrix and let* $\widehat{W} = A^\top A + E$*, where* $E \sim \mathcal{N}(0,\sigma^2\mathbb{I}_{n\times n})$*. Let* $\mathbf{v}$ *be the top right singular vector of* A*, and let* $\widehat{\mathbf{v}}$ *be the top eigenvector of* $\widehat{W}$*. The following is true with probability at least* $1-\beta$*:*

$$\|A\widehat{\mathbf{v}}\|_2^2 \geq \|A\mathbf{v}\|_2^2 - O\left(\sigma \log(n/\beta)\sqrt{n}\right).$$

Now, one can compactly write the update equation of $Y^{(t)}$ in function $\mathcal{A}_{\mathsf{local}}$ of

Algorithm 13 for all the users as:

$$Y^{(t)} \leftarrow \Pi_{\Delta,\Omega}\left(\left(1-\frac{1}{T}\right)Y^{(t-1)} - \frac{k}{T}\widehat{\mathbf{u}}\widehat{\mathbf{v}}^\top\right), \tag{6.8}$$

where $\widehat{\mathbf{u}}$ corresponds to the set of entries $\widehat{u}_i$ in function $\mathcal{A}_{\mathsf{local}}$ represented as a vector. Also, by Lemma 6.2.2, we can conclude that $\|\widehat{\mathbf{u}}\|_2 \leq 1$. Hence, $Y^{(t)}$ is in the set $\{Y : \|Y\|_{\mathsf{nuc}} \leq k\}$ for all $t \in [T]$.

In the following, we incorporate the noisy estimation in the analysis of original Frank-Wolfe (stated in Section 6.1.2). In order to do so, we need to ensure a couple of properties: i) We need to obtain an appropriate bound on the slack parameter γ in Algorithm 12, and ii) we need to ensure that the projection operator $\Pi_{\Delta,\Omega}$ in function $\mathcal{A}_{\mathsf{local}}$ does not introduce additional error. We do this via Lemma 6.2.4 and 6.2.5 respectively.

Lemma 6.2.4. *For the noise variance σ used in function $\mathcal{A}_{\mathsf{global}}$ of Algorithm 13, w.p. at least $1-\beta$, the slack parameter γ in the linear optimization step of Frank-Wolfe algorithm is at most $O\left(\frac{k}{|\Omega|}\sqrt{\sigma\log(n/\beta)\sqrt{n}}\right)$.*

Proof. Recall that $\widehat{\lambda}^2$ corresponds to the maximum eigenvalue of $W^{(t)}$, and notice that $A^{(t)}$ is the scaled gradient of the loss function $L(\theta;\Omega)$ at $\theta = \Pi_{\Delta,\Omega}\left(Y^{(t)}\right)$. Essentially, we need to compute the difference between the values of $\langle\frac{1}{|\Omega|}A^{(t)}, k\mathbf{u}\mathbf{v}^\top\rangle$ and $\langle\frac{1}{|\Omega|}A^{(t)}, k\widehat{\mathbf{u}}\widehat{\mathbf{v}}^\top\rangle$. Let $\alpha = \langle\frac{1}{|\Omega|}A^{(t)}, k\mathbf{u}\mathbf{v}^\top\rangle$, and $\widehat{\alpha} = \langle\frac{1}{|\Omega|}A^{(t)}, k\widehat{\mathbf{u}}\widehat{\mathbf{v}}^\top\rangle$. Now, we have the following w.p. at least $1-\beta$:

$$\begin{aligned}\widehat{\alpha} &= \frac{k\widehat{\mathbf{v}}^\top {A^{(t)}}^\top \widehat{\mathbf{u}}}{|\Omega|} = \frac{k\widehat{\mathbf{v}}^\top {A^{(t)}}^\top A^{(t)}\widehat{\mathbf{v}}}{|\Omega|\left(\widehat{\lambda}+\Theta\left(\sqrt{\sigma\log(n/\beta)\sqrt{n}}\right)\right)} \\ &= \frac{k\|A^{(t)}\widehat{\mathbf{v}}\|_2^2}{|\Omega|\left(\widehat{\lambda}+\Theta\left(\sqrt{\sigma\log(n/\beta)\sqrt{n}}\right)\right)}\end{aligned}$$

$$\geq \frac{k\left(\|A^{(t)}\mathbf{v}\|_2^2 - O\left(\sigma\log(n/\beta)\sqrt{n}\right)\right)}{|\Omega|\left(\widehat{\lambda} + \Theta\left(\sqrt{\sigma\log(n/\beta)\sqrt{n}}\right)\right)}$$
$$= \frac{k\left(\frac{|\Omega|\lambda}{k}\alpha - O\left(\sigma\log(n/\beta)\sqrt{n}\right)\right)}{|\Omega|\left(\widehat{\lambda} + \Theta\left(\sqrt{\sigma\log(n/\beta)\sqrt{n}}\right)\right)}, \tag{6.9}$$

where λ^2 is the maximum eigenvalue of ${A^{(t)}}^\top A^{(t)}$, the second equality follows from the definition of $\widehat{\mathbf{u}}$, and the inequality follows from Lemma 6.2.3. One can rewrite (6.9) as:

$$\alpha - \widehat{\alpha} \leq \underbrace{\left(1 - \frac{\lambda}{\left(\widehat{\lambda} + \Theta\left(\sqrt{\sigma\log(n/\beta)\sqrt{n}}\right)\right)}\right)\alpha}_{E_1} + \underbrace{O\left(\frac{k\sigma\log(n/\beta)\sqrt{n}}{|\Omega|\left(\widehat{\lambda} + \Theta\left(\sqrt{\sigma\log(n/\beta)\sqrt{n}}\right)\right)}\right)}_{E_2}. \tag{6.10}$$

We will analyze E_1 and E_2 in (6.10) separately. One can write E_1 in (6.10) as follows:

$$E_1 = \left(\frac{\left(\widehat{\lambda} + O\left(\sqrt{\sigma\log(n/\beta)\sqrt{n}}\right)\right) - \lambda}{\left(\widehat{\lambda} + \Theta\left(\sqrt{\sigma\log(n/\beta)\sqrt{n}}\right)\right)}\right)\alpha$$
$$= \frac{k}{|\Omega|}\left(\frac{\left(\widehat{\lambda} + O\left(\sqrt{\sigma\log(n/\beta)\sqrt{n}}\right)\right) - \lambda}{\left(\widehat{\lambda} + \Theta\left(\sqrt{\sigma\log(n/\beta)\sqrt{n}}\right)\right)}\right)\lambda. \tag{6.11}$$

By Weyl's inequality for eigenvalues, and the fact that w.p. at least $1-\beta$, we have $\|\widehat{W}^{(t)} - {A^{(t)}}^\top A^{(t)}\|_2 = O\left(\sigma\log(n/\beta)\sqrt{n}\right)$ because of spectral properties of random Gaussian matrices (Corollary 2.3.5 in Tao (2012)), it follows that $\left|\widehat{\lambda} - \lambda\right| = O\left(\sqrt{\sigma\log(n/\beta)\sqrt{n}}\right)$. Therefore, one can conclude from (6.11) that $E_1 = O\left(\frac{k}{|\Omega|}\sqrt{\sigma\log(n/\beta)\sqrt{n}}\right)$. Now, we will bound the term E_2 in (6.10). Since $\widehat{\lambda} \geq 0$, it follows that $E_2 = O\left(\frac{k}{|\Omega|}\sqrt{\sigma\log(n/\beta)\sqrt{n}}\right)$. Therefore, the slack parameter $\alpha - \hat{\alpha} = E_1 + E_2 = O\left(\frac{k}{|\Omega|}\sqrt{\sigma\log(n/\beta)\sqrt{n}}\right)$. □

Lemma 6.2.5. *Define the operators* P_Ω *and* $\Pi_{\Delta,\Omega}$ *as described in function* $\mathcal{A}_{\mathsf{local}}$ *in Section 6.2. Let* $L(Y;\Omega) = \frac{1}{2|\Omega|}\|\mathsf{P}_\Omega(Y - Y^*)\|_F^2$ *for any matrix* $Y \in \mathbb{R}^{m\times n}$*. The following is true for all* $Y \in \mathbb{R}^{m\times n}$*:* $L\left(\Pi_{\Delta,\Omega}\left(Y\right);\Omega\right) \leq L\left(\mathsf{P}_\Omega\left(Y\right);\Omega\right)$*.*

Proof. First, notice that for any matrix $M = \left[m_1^\top, \cdots, m_m^\top\right]$ (where $m_i^\top$ corresponds to the i-th row of M), we have $\|M\|_F^2 = \sum_i \|m_i\|_2^2$. Let Π_Δ be the L_2 projector onto a ball of radius Δ, and $\mathbb{B}_\Delta^n$ be a ball of radius Δ in n-dimensions, centered at the origin. Then, for any pair of vectors, $v_1 \in \mathbb{R}^n$ and $v_2 \in \mathbb{B}_\Delta^n$, $\|\Pi_\Delta\left(v_1\right) - v_2\|_2 \leq \|v_1 - v_2\|_2$. This follows from the contraction property of L_2-projection. Hence, by the above two properties, and the fact that each row of the matrix $\mathsf{P}_\Omega\left(Y^*\right) \in \mathbb{B}_\Delta^n$, we can conclude $L\left(\Pi_\Delta(\mathsf{P}_\Omega(Y));\Omega\right) \leq L\left(\mathsf{P}_\Omega(Y);\Omega\right)$ for any $Y \in \mathbb{R}^{m\times n}$. This concludes the proof. □

This means we can still use Theorem 19. Hence, we can conclude that, w.p. $\geq 1 - \beta$:

$$L\left(Y^{(T)};\Omega\right) = O\left(\frac{k^2}{|\Omega|T} + \frac{k}{|\Omega|}\sqrt{\sigma\log(n/\beta)\sqrt{n}}\right)$$

Here we used the fact that the curvature parameter C_f from Theorem 19 is at most $k^2/|\Omega|$ (see Jaggi et al. (2010) for a proof). Setting $\beta = 1/3$ completes the proof. □

Remark 22. *We further illustrate our empirical risk bound by considering a simple setting: let* Y^* *be a rank-one matrix with* $Y^*_{ij} \in [-1,1]$ *and* $|\Omega| = m\sqrt{n}$*. Then* $k = O(\sqrt{mn})$*, and* $\Delta = O(n^{1/4})$*, implying an error of* $\widetilde{O}\left(\sqrt{n}m^{-2/5}\right)$ *hiding the privacy parameter* ϵ*; in contrast, a trivial solution like* $Y = 0$ *leads to* $O(1)$ *error. Naturally, the error increases with* n *as there is more information to be protected. However, it decreases with a larger number of users* m *as the presence/absence of a user has lesser effect on the solution with increasing* m*. We leave further investigation into the dependency of the error on* m *for future work.*

Remark 23. *Our analysis does not require an upper bound on* $\|Y^*\|_{\text{nuc}}$ *(as stated in Theorem 21); we would instead incur an additional error of* $\min_{\|Y\|_{\text{nuc}} \leq k} \frac{1}{|\Omega|} \|\mathsf{P}_\Omega(Y^* - Y)\|_F^2$*. Moreover, consider a similar scenario as in Remark 22, but* $|\Omega| = mn$*, i.e., all the entries of* Y^* *are revealed. In such a case,* $\Delta = O(\sqrt{n})$*, and the problem reduces to that of standard low-rank matrix approximation of* Y^**. Note that our result here leads to an error bound of* $\widetilde{O}\left(n^{1/5}m^{-2/5}\right)$*, while the state-of-the-art result by Hardt & Roth (2013) leads to an error bound of* $O(1)$ *due to being in the much stricter standard DP model.*

6.2.1.1 Generalization error guarantee

We now present a generalization error (defined in Equation 3.1) bound which shows that our approach provides accurate prediction over *unknown* entries. For obtaining our bound, we use the following result from Srebro & Shraibman (2005).

Theorem 24. *Let* Y^* *be a hidden matrix, and the data samples in* Ω *be drawn uniformly at random from* $[m] \times [n]$*. Let* $A \in \mathbb{R}^{m \times n}$ *be a matrix with* $\mathsf{rank}(A) \leq r$*, and let each entry of* A *be bounded by a constant. Then, the following holds with probability at least 2/3 over choosing* Ω*:*

$$|L(A) - L(A;\Omega)| = \tilde{O}\left(\sqrt{\frac{r \cdot (m+n)}{|\Omega|}}\right).$$

The $\tilde{O}(\cdot)$ *hides poly-logarithmic terms in* m *and* n*.*

Also, the output of Private FW (Algorithm 13) has rank at most T, where T is the number of iterations. Thus, replacing T from Theorem 21, we get the following:

Corollary 6.2.1 (Generalization Error)**.** *Let* $\|Y^*\|_{\text{nuc}} \leq k$ *for a hidden matrix* Y^**, and* $\|\mathsf{P}_\Omega(Y_i^*)\|_2 \leq \Delta$ *for every row* i *of* Y^**. If we choose the number of rounds in Algorithm 13 to be* $O\left(\frac{k^{4/3}}{(|\Omega|(m+n))^{1/3}}\right)$*, the data samples in* Ω *are drawn u.a.r. from* $[m] \times [n]$*, and* $\epsilon \leq 2\log\left(\frac{1}{\delta}\right)$*, then with probability at least 2/3 over the outcomes of the algorithm and*

choosing Ω, the following is true for the final completed matrix Y:

$$L(Y) = \tilde{O}\left(\frac{k^{4/3}\Delta n^{1/4}}{\sqrt{\epsilon|\Omega|^{13/6}(m+n)^{1/6}}} + \left(\frac{k\sqrt{m+n}}{|\Omega|}\right)^{2/3}\right).$$

The $\tilde{O}(\cdot)$ hides poly-logarithmic terms in $m, n, |\Omega|$ and δ.

Remark 25. *We further illustrate our bound using a setting similar to the one considered in Remark 22. Let Y^* be a rank-one matrix with $Y^*_{ij} \in [-1, 1]$ for all i, j; let $|\Omega| \geq m\sqrt{n}\cdot\mathsf{polylog}(n)$, i.e., the fraction of movies rated by each user is arbitrarily small for larger n. For this setting, our generalization error is $o(1)$ for $m = \omega(n^{5/4})$. This is slightly higher than the bound in the non-private setting by Shalev-Shwartz et al. (2011), where $m = \omega(n)$ is sufficient to get generalization error $o(1)$. Also, as the first term in the error bound pertains to DP, it decreases with a larger number of users m, and increases with n as it has to preserve privacy of a larger number of items. In contrast, the second term is the matrix completion error decreases with n. This is intuitive, as a larger number of movies enables more sharing of information between users, thus allowing a better estimation of preferences Y^*. However, just increasing m may not always lead to a more accurate solution (for example, consider the case of $n = 1$).*

Remark 26. *The guarantee in Corollary 6.2.1 is for uniformly random Ω, but using the results of Shamir & Shalev-Shwartz (2011), it is straightforward to extend our results to any i.i.d. distribution over Ω. Moreover, we can extend our results to handle strongly convex and smooth loss functions instead of the squared loss considered in this chapter.*

6.2.2 Efficient PCA via Oja's Algorithm

Algorithm 13 requires computing the top eigenvector of $\widehat{W}^{(t)} = W^{(t)} + N^{(t)}$, where $W^{(t)} = \sum_i \left(A_i{}^{(t)}\right)^\top A_i{}^{(t)}$ and $N^{(t)}$ is a random noise matrix. However, this can be a bottleneck for computation as $N^{(t)}$ itself is a dense $n \times n$ matrix, implying

a space complexity of $\Omega(n^2 + mk)$, where k is the maximum number of ratings provided by a user. Similarly, standard eigenvector computation algorithms will require $O(mk^2 + n^2)$ time (ignoring factors relating to rate of convergence), which can be prohibitive for practical recommendation systems with large n. We would like to stress that this issue plagues even standard DP PCA algorithms (Dwork et al. (2014c)), which have quadratic space-time complexity in the number of dimensions.

We tackle this by using a stochastic algorithm for the top eigenvector computation that significantly reduces both space and time complexity while preserving privacy. In particular, we use Oja's algorithm (Jain et al. (2016)), which computes top eigenvectors of a matrix with a stochastic access to the matrix itself. That is, if we want to compute the top eigenvector of $W^{(t)}$, we can use the following updates:

$$\widehat{\mathbf{v}}_\tau = (I + \eta X_\tau)\widehat{\mathbf{v}}_{\tau-\mathbf{1}}, \qquad \widehat{\mathbf{v}}_\tau = \widehat{\mathbf{v}}_\tau / \|\widehat{\mathbf{v}}_\tau\|_2 \tag{6.12}$$

where $\mathbb{E}[X_\tau] = W^{(t)}$. For example, we can update $\widehat{\mathbf{v}}_\tau$ using $X_\tau = W^{(t)} + N_\tau^{(t)}$ where each entry of $N_\tau^{(t)}$ is sampled i.i.d. from a Gaussian distribution calibrated to ensure DP. Even this algorithm in its current form does not decrease the space or time complexity as we need to generate a dense matrix ${N_\tau}^{(t)}$ in each iteration. However, by observing that ${N_\tau}^{(t)} v = g_\tau \sim \mathcal{N}(0, \sigma^2 \mathbf{1}^n)$ where v is independent of ${N_\tau}^{(t)}$, we can now replace the generation of ${N_\tau}^{(t)}$ by the generation of a vector g_τ, thus reducing both the space and time complexity of our algorithm. The computation of each update is significantly *cheaper* as long as $mk \ll n^2$, which is the case for practical recommendation systems as k tends to be fairly small there (typically on the order of $\sqrt{n}$).

Algorithm 14 provides a pseudocode of the eigenvector computation method.

Algorithm 14 Private Oja's algorithm

Input: An $m \times n$ matrix A s.t. each row $\|A_i\|_2 \leq \Delta$, privacy parameters: (ϵ, δ) s.t. $\epsilon \leq 2\log(1/\delta)$, total number of iterations: Γ
$\sigma \leftarrow \Delta^2\sqrt{256 \cdot \Gamma \log(2/\delta)}/\epsilon, \widehat{\mathbf{v}}_{\mathbf{0}} \sim \mathcal{N}(0, \sigma^2 I)$
for $\tau \in [\Gamma]$ **do**
  $\eta = \frac{1}{\Gamma\sigma\sqrt{n}}, g_\tau \sim \mathcal{N}(0, \sigma^2 \mathbf{1}^n)$
  $\widehat{\mathbf{v}}_\tau \leftarrow \widehat{\mathbf{v}}_{\tau-\mathbf{1}} + \eta\left(A^T A \widehat{\mathbf{v}}_{\tau-\mathbf{1}} + g_\tau\right), \widehat{\mathbf{v}}_\tau \leftarrow \widehat{\mathbf{v}}_\tau / \|\widehat{\mathbf{v}}_\tau\|_2$
Return $\widehat{\mathbf{v}}_{\mathbf{\Gamma}}, \left(\widehat{\lambda}_\Gamma^2 \leftarrow ||A \cdot \widehat{\mathbf{v}}_{\mathbf{\Gamma}}||_2^2 + \mathcal{N}(0, \sigma^2)\right)$

The computation of the approximate eigenvector $\widehat{\mathbf{v}}_{\mathbf{\Gamma}}$ and the eigenvalue $\widehat{\lambda}_\Gamma^2$ in it is DP (directly follows via the proof of Theorem 20). The next natural question is how well can $\widehat{\mathbf{v}}_{\mathbf{\Gamma}}$ approximate the behavior of the top eigenvector of the non-private covariance matrix $W^{(t)}$? To this end, we provide Theorem 27 below that analyzes Oja's algorithm, and shows that the *Rayleigh quotient* of the approximate eigenvector is close to the top eigenvalue of $W^{(t)}$. In particular, using Theorem 27 along with the fact that in our case, $\mathcal{V} = \sigma^2 n$, we have $\|A^{(t)}\|_2^2 \leq \|A^{(t)}\widehat{\mathbf{v}}_{\mathbf{\Gamma}}\|_2^2 + O\left(\sigma\sqrt{n}\log(\eta/\beta)\right)$ with high probability (w.p. $\geq 1 - \beta^2$)), where $\widehat{\mathbf{v}}_{\mathbf{\Gamma}}$ is the output of Algorithm 14, $\Gamma = \Omega\left(\min\left\{\frac{1}{\beta}, \frac{\|A^{(t)}\|^2}{\sigma\sqrt{n}}\right\}\right)$, and $\eta = \frac{1}{\Gamma \cdot \sigma\sqrt{n}}$.

Note that the above given bound is exactly the bound required in the proof of Theorem 19 in Section 6.1.2. Hence, computing the top eigenvector privately using Algorithm 14 does not change the utility bound of Theorem 21.

Theorem 27 (Based on Theorem 3 (Allen-Zhu & Li (2017))). *Let $X_1, X_2, \ldots X_\Gamma$ be sampled i.i.d. such that for each $i \in [\Gamma]$, we have $\mathbb{E}(X_i) = W = A^T A$. Moreover, let $\mathcal{V} = \max\{\|\mathbb{E}(X_i - W)^T(X_i - W)\|, \|\mathbb{E}(X_i - W)(X_i - W)^T\|\}$, and $\eta = \frac{1}{\sqrt{\mathcal{V}\Gamma}}$. Then, the Γ-th iterate of Oja's Algorithm (Update (6.12)) , i.e., $\widehat{\mathbf{v}}_{\mathbf{\Gamma}}$, satisfies (w.p. $\geq 1 - 1/\mathsf{poly}(\Gamma)$):* $\widehat{\mathbf{v}}_{\mathbf{\Gamma}}^T W \widehat{\mathbf{v}}_{\mathbf{\Gamma}} \geq \|W\|_2 - O\left(\sqrt{\frac{\mathcal{V}}{\Gamma}} + \frac{\|W\|_2}{\Gamma}\right)$.

Comparison with Private Power Iteration (PPI) method (Hardt & Roth (2013)): Private PCA via PPI provides utility guarantees dependent on the gap between the

top and the kth eigenvalue of the input matrix A for some $k > 1$, whereas private Oja's utility guarantee is gap-independent.

6.3 PRIVATE MATRIX COMPLETION VIA SINGULAR VALUE DECOMPOSITION

In this section, we study a simple SVD-based algorithm for differentially private matrix completion. Our SVD-based algorithm for matrix completion just computes a low-rank approximation of $\mathsf{P}_\Omega(Y^*)$, but still provides *reasonable* error guarantees (Keshavan et al. (2010)). Moreover, the algorithm forms a foundation for more sophisticated algorithms like alternating minimization (Hardt & Wootters (2014)), singular value projection (Jain et al. (2010)) and singular value thresholding (Cai et al. (2010)). Thus, similar ideas may be used to extend our approach.

Algorithmic idea: At a high level, given rank r, Algorithm 15 first computes a differentially private version of the top-r right singular subspace of $\mathsf{P}_\Omega(Y^*)$, denoted by V_r. Each user projects her data record onto V_r (after appropriate scaling) to complete her row of the matrix. Since each user's completed row depends on the other users via the global computation which is performed under differential privacy, the overall agorithm satisfies joint differential privacy . In principle, this is the same as in Section 6.2, except now it is a direct rank-r decomposition instead of an iterative rank-1 decomposition. Also, our overall approach is similar to that of McSherry & Mironov (2009), except that each user in McSherry & Mironov (2009) uses a nearest neighbor algorithm in the local computation phase (see Algorithm 15). Additionally, in contrast to McSherry & Mironov (2009), we provide a formal generalization guarantee.

Algorithm 15 Private Matrix Completion via SVD

Input: Privacy parameters: (ϵ, δ), matrix dimensions: (m, n), uniform L_2-bound on the rows of $\mathsf{P}_\Omega(Y^*)$: Δ, and rank bound: r

Global computation: Compute the top-r subspace V_r for the matrix $\widehat{W} \leftarrow \sum_{i=1}^{m} W_i + N$, where $W_i = \Pi_\Delta\left(\mathsf{P}_\Omega\left(Y_i^*\right)\right)^\top \Pi_\Delta\left(\mathsf{P}_\Omega\left(Y_i^*\right)\right)$, Π_Δ is the projection onto the L_2-ball of radius Δ, $N \in \mathbb{R}^{n\times n}$ corresponds to a matrix with i.i.d. entries from $\mathcal{N}(0, \sigma^2)$, and $\sigma \leftarrow \Delta^2\sqrt{64\log(1/\delta)}/\epsilon$

Local computation: Each user i computes the i-th row of the private approximation $\widehat{Y}$: $\widehat{Y}_i \leftarrow \frac{mn}{|\Omega|}\mathsf{P}_\Omega\left(Y_i^*\right) V_r V_r^\top$

6.3.1 Privacy and Utility Analysis

We now present the privacy and generalization guarantees for the above algorithm.

Theorem 28. *Algorithm 15 satisfies (ϵ, δ)-joint differential privacy .*

The proof of privacy for Algorithm 15 follows immediately from the proof of Theorem 20, as the key step of computing the top eigenvectors of the W matrix remains the same.

For the generalization error bound for Algorithm 15, we use the standard low-rank matrix completion setting, i.e., entries are sampled i.i.d., and the underlying matrix Y^* is incoherent (Definition 29). Intuitively, incoherence ensures that the left and right singular subspaces of a matrix have a low correlation with the standard basis. The scale of μ is $[0, \max\{m, n\}]$. Since, we are assuming $m \geq n$, we have $\mu \in [0, m]$.

Definition 29 (μ-incoherence (Jin et al. (2016))). *Let $Y \in \mathbb{R}^{m\times n}$ be a matrix of rank at most r, and let $U \in \mathbb{R}^{m\times r}$ and $V \in \mathbb{R}^{r\times n}$ be the left and right singular subspaces of Y.*

Then, the incoherence μ is the following:

$$\mu = \max\left\{\frac{m}{r}\max_{1\le i\le m}\|UU^T e_i\|_2, \frac{n}{r}\max_{1\le i\le n}\|VV^T f_i\|_2\right\}.$$

Here, $e_i \in \mathbb{R}^m$ and $f_i \in \mathbb{R}^n$ are the i-th standard basis vectors in m and n dimensions, respectively.

Under the above set of assumptions, we get:

Theorem 30. *Let $Y^* \in \mathbb{R}^{m\times n}$ be a rank-r, μ-incoherent matrix with condition number $\kappa = \|Y^*\|_2/\lambda_r(Y^*)$, where $\lambda_r(\cdot)$ corresponds to the r-th largest singular value. Also, let the set of known entries Ω be sampled uniformly at random s.t. $|\Omega| \ge c_0\kappa^2\mu mr\log m$ for a large constant $c_0 > 0$. Let $\|\mathsf{P}_\Omega(Y^*)_i\|_2 \le \Delta$ for every row i of Y^*. Then, with probability at least $2/3$ over the outcomes of the algorithm, the following holds for $\widehat{Y}$ estimated by Algorithm 15:*

$$L(\widehat{Y};\Omega) = O\left(\frac{\Delta^4\kappa^4 m^3 n^4\cdot r\cdot\log(1/\delta)}{|\Omega|^4\|Y^*\|_2^2\epsilon^2} + \frac{\mu\|Y^*\|_2^2\cdot r^2\log m}{n\cdot|\Omega|}\right).$$

Using $\Delta \le \|Y^*\|_2$, we get:

$$L(\widehat{Y}) = O\left(\frac{\min\left(\Delta^2, \frac{\mu^2 rn}{m}\right)\kappa^4 m^3 n^4 r\log(1/\delta)}{|\Omega|^4\epsilon^2} + \frac{\mu\|Y^*\|_2^2\cdot r^2\log m}{n\cdot|\Omega|}\right).$$

The $O(\cdot)$ hides only universal constants.

Proof. Let $B = \frac{1}{p}\mathsf{P}_\Omega(Y^*)$ where $p = |\Omega|/(m\cdot n)$ and let V_r be the top-r right singular subspace of B. Suppose $\Pi_r = V_r V_r^\top$ be the projector onto that subspace. Recall that $\widehat{V}_r$ is the right singular subspace defined in Algorithm 15 and let $\widehat{\Pi}_r = \widehat{V}_r\widehat{V}_r^T$ be the corresponding projection matrix.

Then, using the triangular inequality, we have:

$$\begin{aligned}\|B\widehat{\Pi}_r - Y^*\|_2 &\leq \|B\Pi_r - Y^*\|_2 + \|B\widehat{\Pi}_r - B\Pi_r\|_2 \\ &\leq c_1\|Y^*\|_2\sqrt{\frac{\mu m r \log m}{|\Omega|}} + \|B\widehat{\Pi}_r - B\Pi_r\|_2, \end{aligned} \tag{6.13}$$

where the second inequality follows from the following standard result (Lemma 6.3.1) from the matrix completion literature, and holds w.p. $\geq 1 - 1/m^{10}$.

Lemma 6.3.1 (Follows from Lemma A.3 in Jin et al. (2016))**.** *Let M be an $m \times n$ matrix with $m \geq n$, rank r, and incoherence μ, and Ω be a subset of i.i.d. samples from M. There exists universal constants c_1 and c_0 such that if $|\Omega| \geq c_0 \mu m r \log m$, then with probability at least $1 - 1/m^{10}$, we have:*

$$\|M - \frac{mn}{|\Omega|}\mathsf{P}_\Omega(M)\|_2 \leq c_1\|M\|_2\sqrt{\frac{\mu \cdot m \cdot r \log m}{|\Omega|}}.$$

Using Theorem 6 of Dwork et al. (2014b), the following holds with probability at least $2/3$,

$$\|\widehat{\Pi}_r - \Pi_r\|_2 = O\left(\frac{\Delta^2\sqrt{n\log(1/\delta)}}{(\alpha_r^2 - \alpha_{r+1}^2)\epsilon}\right), \tag{6.14}$$

where α_i is the i-th singular value of $\mathsf{P}_\Omega(Y^*) = p \cdot B$.

Recall that $\kappa = \|Y^*\|_2/\lambda_r(Y^*)$, where λ_r is the r-th singular value of Y^*. Let $|\Omega| \geq c_0\kappa^2\mu m r \log m$ with a large constant $c_0 > 0$. Then, using Lemma 6.3.1 and Weyl's inequality, we have (w.p. $\geq 1 - 1/m^{10}$):

$$\alpha_r \geq 0.9 \cdot p\frac{1}{\kappa}\|Y^*\|_2, \qquad \text{and} \qquad \alpha_{r+1} \leq c_1 p \cdot \|Y^*\|_2\sqrt{\frac{\mu \cdot m \cdot r \log m}{|\Omega|}} \leq 0.1 \cdot \alpha_r \tag{6.15}$$

Similarly,

$$\|B\|_2 \leq 2\|Y^*\|_2, w.p. \geq 1 - 1/m^{10}. \tag{6.16}$$

Using (6.13), (6.14), (6.15), and (6.16), we have w.p. $\geq \frac{2}{3} - \frac{5}{m^{10}}$:

$$\|B\widehat{\Pi}_r - Y^*\|_2 \leq 8\|Y^*\|_2 \cdot \frac{\Delta^2\kappa^2\sqrt{n\log(1/\delta)}}{p^2\|Y^*\|_2^2\epsilon} + c_1\|Y^*\|_2\sqrt{\frac{\mu \cdot m \cdot r\log m}{|\Omega|}}.$$

Recall that $\widehat{Y} = \frac{1}{p}\mathsf{P}_\Omega(Y^*)\widehat{\Pi}_r = B\widehat{\Pi}_r$. Hence:

$$\frac{\|B\widehat{\Pi}_r - Y^*\|_2^2}{mn} \leq O\left(\frac{\Delta^4\kappa^4 n\log(1/\delta)}{mn \cdot p^4\|Y^*\|_2^2\epsilon^2}\right) + c_1\|Y^*\|_2^2\frac{\mu \cdot m \cdot r\log m}{mn \cdot |\Omega|}.$$

The theorem now follows by using $\|A\|_F^2 \leq r\|A\|_2^2$, where r is the rank of A. □

Remark 31. *Let Y^* be a rank one incoherent matrix with $Y^*_{ij} = \Theta(1)$, $|\Omega| = m\sqrt{n}$, $\Delta = O(n^{1/4})$, and $\mu = O(1)$. Notice that the spectral norm $\|Y^*\|_2 \approx \sqrt{mn}$. Hence, the first term in the bound reduces to $O\left(\frac{n^2}{m^2}\right)$ and the second error term is $O\left(\frac{1}{\sqrt{n}}\right)$, whereas a trivial solution of $Y = 0$ leads to $O(1)$ error. Similar to the behavior in Remark 25, the first term above increases with n, and decreases with increasing m due to the noise added, while the second term decreases with increasing n due to more sharing between users.*

Remark 32. *Under the assumptions of Theorem 30, the second term can be arbitrarily small for other standard matrix completion methods like the FW-based method (Algorithm 13) studied in Section 6.2 above. However, the first error term for such methods can be significantly larger. For example, the error of Algorithm 13 in the setting of Remark 31 is $\approx O\left(\frac{n^{13/24}}{m^{5/12}}\right)$ as the second term in Corollary 6.2.1 vanishes in this setting; in contrast, the error of the SVD-based method (Algorithm 15) is $O\left(\frac{n^2}{m^2} + \frac{1}{\sqrt{n}}\right)$. On the other hand, if the data does not satisfy the assumptions of Theorem 30, then the error incurred by Algorithm 15 can be significantly larger (or even trivial) when compared to that of Algorithm 13.*

6.4 EXPERIMENTAL EVALUATION

We now present empirical results for Private FW (Algorithm 13) on several benchmark datasets, and compare its performance to state-of-the-art methods like McSherry & Mironov (2009), and private as well as non-private variant of the Projected Gradient Descent (PGD) method (Cai et al. (2010); Bassily et al. (2014a); Abadi et al. (2016)). In all our experiments, we see that private FW provides accuracy very close to that of the non-private baseline, and almost always significantly outperforms both the private baselines.

Datasets: As we want to preserve privacy of every user, and the output for each user is n-dimensional, we can expect the private recommendations to be accurate only when $m \gg n$ (see Theorem 20). Due to this constraint, we conduct experiments on the following datasets:

1. *Synthetic:* We generate a random rank-one matrix $Y^* = uv^T$ with unit L_∞-norm, $m = 500$k, and $n = 400$.

2. *Jester:* This dataset contains $n = 100$ jokes, and $m \approx 73$k users. We rescale the ratings to be from 0 to 5.

3. *MovieLens10M (Top 400):* We pick the $n = 400$ most rated movies from the Movielens10M dataset, resulting in $m \approx 70$k users of the ≈ 71k users in the dataset.

4. *Netflix (Top 400):* We pick the $n = 400$ most rated movies from the Netflix prize dataset, resulting in $m \approx 474$k users of the ≈ 480k users in the dataset.

5. *Yahoo! Music (Top 400):* We pick the $n = 400$ most rated songs from the Yahoo! music dataset, resulting in $m \approx 995$k users of the ≈ 1m users in the dataset.

We rescale the ratings to be from 0 to 5.

Procedure: For all datasets, we randomly sample 1% of the given ratings for measuring the test error. For experiments with privacy, for all datasets except Jester, we randomly select at most $\xi = 80$ ratings per user to get $\mathsf{P}_\Omega(Y^*)$. We vary the privacy parameter $\epsilon \in [0.1, 5]$, but keep $\delta = 10^{-6}$, thus ensuring that $\delta < \frac{1}{m}$ for all datasets. Moreover, we report results averaged over 10 independent runs. The requirement in Algorithm 13 that $\epsilon \leq 2\log(1/\delta)$ is satisfied by all the values of ϵ considered for the experiments.

Note that the privacy guarantee is user-level, which effectively translates to an entry-level guarantee of $\epsilon_{entry} = \frac{\epsilon_{user}}{\xi}$, i.e., $\epsilon_{entry} \in [0.00125, 0.0625]$ as $\epsilon_{user} \in [0.1, 5]$.

For the experiments with private Frank-Wolfe (Algorithm 13), we normalize the data as $\hat{r}_{i,j} = r_{i,j} - u_i$ for all $i \in [m], j \in [n]$, where $r_{i,j}$ is user i's rating for item j, and u_i is the average rating of user i. Note that each user can safely perform such a normalization at her end without incurring any privacy cost. Regarding the parameter choices for private FW, we cross-validate over the nuclear norm bound k, and the number of iterations T for each dataset. For k, we set it to the actual nuclear norm for the synthetic dataset, and choose from $\{20\text{k}, 25\text{k}\}$ for Jester, $\{120\text{k}, 130\text{k}\}$ for Netflix, $\{30\text{k}, 40\text{k}\}$ for MovieLens10M, and $\{130\text{k}, 150\text{k}\}$ for the Yahoo! Music dataset. We choose T from various values in $[5, 50]$. Consequently, the rank of the prediction matrix for all the private FW experiments is at most 50. For faster training, we calibrate the scale of the noise in every iteration according to the number of iterations that the algorithm has completed, while still ensuring the overall DP guarantee.

Non-private baseline: For the non-private baseline, we normalize the training data for the experiments with non-private Frank-Wolfe by removing the per-user

and per-movie averages (as in Jaggi et al. (2010)), and we run non-private FW for 400 iterations. For non-private PGD, we tune the step size schedule. We find that non-private FW and non-private PGD converge to the same accuracy after tuning, and hence, we use this as our baseline.

Private baselines: To the best of our knowledge, only McSherry & Mironov (2009) and Liu et al. (2015) address the user-level DP matrix completion problem. While we present an empirical evaluation of the 'SVD after cleansing method' from the former, we refrain from comparing to the latter as the exact privacy parameters (ϵ and δ) for the Stochastic Gradient Langevin Dynamics based algorithm in Liu et al. (2015) (correspondigly, in Wang et al. (2015)) are unclear. They use a Markov chain based sampling method; to obtain quantifiable (ϵ, δ), the sampled distribution is required to converge (non-asymptotically) to a DP preserving distribution in L_1 distance, for which we are not aware of any analysis. We also provide a comparison with private PGD (Algorithm 16).

For the 'SVD after cleansing method' from McSherry & Mironov (2009), we set $\delta = 10^{-6}$, and select ϵ appropriately to ensure a fair comparison. We normalize the data by removing the private versions of the global average rating and the per-movie averages. We tune the shrinking parameters β_m and β_p from various values in $[5, 15]$, and β from $[5, 25]$. For private PGD, we tune T from various values in $[5, 50]$, and the step size schedule from $\left\{t^{-1/2}, t^{-1}, 0.05, 0.1, 0.2, 0.5\right\}$ for $t \in [T]$. We set the nuclear norm constraint k equal to the nuclear norm of the hidden matrix, and for faster training, we calibrate the scale of the noise as in our private FW experiments.

Results: Figure 6.2 shows the results of our experiments. Even though all the considered private algorithms satisfy Joint DP, our private FW method almost al-

Algorithm 16 Private Projected Gradient Descent

Input: Set of revealed entries: Ω, operator: P_Ω, matrix: $\mathsf{P}_\Omega(Y^*) \in \mathbb{R}^{m\times n}$, bound on $\|\mathsf{P}_\Omega(Y_i^*)\|_2$: Δ, nuclear norm constraint: k, time bound: T, step size schedule: η_t for $t \in [T]$, privacy parameters: (ϵ, δ)
$\sigma \leftarrow \Delta^2\sqrt{64 \cdot T\log(1/\delta)}/\epsilon$
$Y^{(0)} \leftarrow \{0\}^{m\times n}$
for $t \in [T]$ **do**
  $Y^{(t)} \leftarrow Y^{(t-1)} - \eta_t \cdot \mathsf{P}_\Omega\left(Y^* - Y^{(t)}\right)$
  $W^{(t)} \leftarrow {Y^{(t)}}^\top Y^{(t)} + N^{(t)}$, where $N^{(t)} \in \mathbb{R}^{n\times n}$ corresponds to a matrix with i.i.d. entries from $\mathcal{N}(0, \sigma^2)$
  $\widehat{V} \leftarrow$ Eigenvectors of $W^{(t)}$, $\widehat{\Lambda}^2 \leftarrow$ Diagonal matrix containing the n eigenvalues of $W^{(t)}$
  $\widehat{U} \leftarrow Y^{(t)}\widehat{V}\widehat{\Lambda}^{-1}$
  if $\sum_{i\in[n]} \widehat{\Lambda}_{i,i} > k$ **then**
    Find a diagonal matrix Z s.t. $\sum_{i\in[n]} Z_{i,i} = k$, and $\exists\tau$ s.t. $\forall i \in [n], Z_{i,i} = \max\left(0, \widehat{\Lambda}_{i,i} - \tau\right)$
  else
    $Z \leftarrow \widehat{\Lambda}$
  $Y^{(t)} \leftarrow \widehat{U}Z\widehat{V}^\top$
Return $Y^{(T)}$

ways incurs a significantly lower test RMSE than the two private baselines. Note that although non-private PGD provides similar empirical accuracy as non-private FW, the difference in performance for their private versions can be attributed to the noise being calibrated to a rank-one update for our private Frank-Wolfe. In all our experiments, the implementation of private FW with Oja's method (Algorithm 14) did not suffer any perceivable loss of accuracy as compared to the variant in Algorithm 13; all the plots in Figure 6.2 remain identical.

6.4.1 Additional Experimental Evaluation

Here, we provide the empirical results for our private Frank-Wolfe algorithm (Algorithm 13) as well as the 'SVD after cleansing method' of McSherry & Mironov

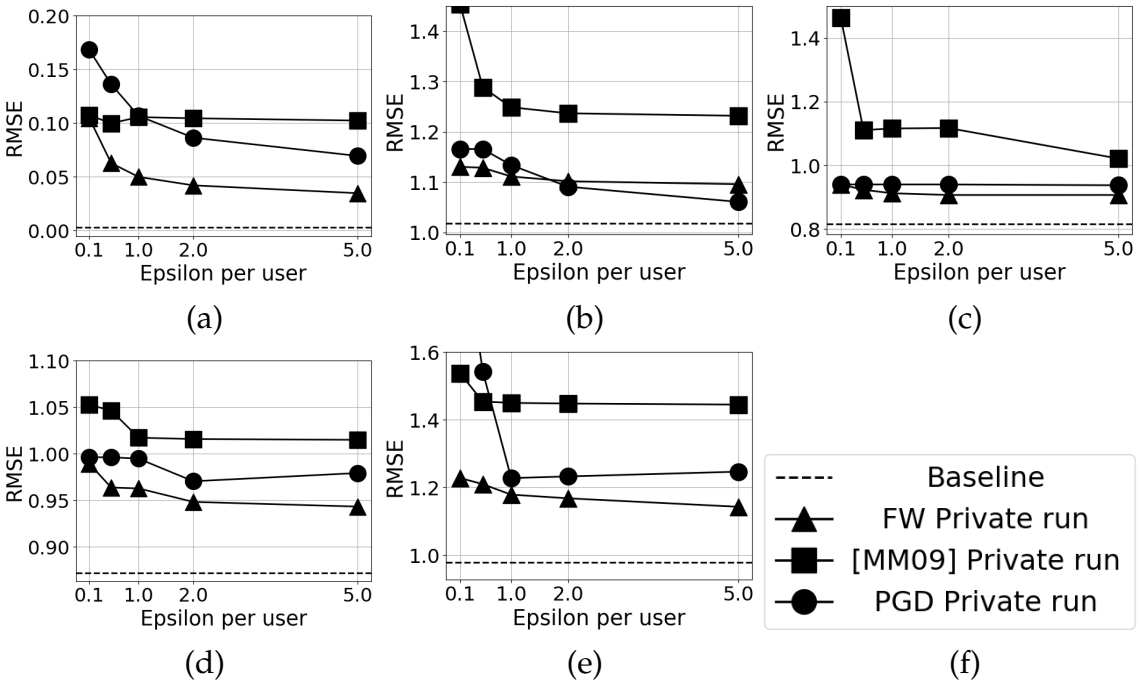

Figure 6.2: Root mean squared error (RMSE) vs. ϵ, on (a) synthetic, (b) Jester, (c) MovieLens10M, (d) Netflix, and (e) Yahoo! Music datasets, for $\delta = 10^{-6}$. A legend for all the plots is given in (f).

(2009) for $n = 900$ with all the above considered datasets (except Jester). We see that private PGD takes too long to complete for $n = 900$; we present an evaluation for the other algorithms for the following additional datasets:

1. *Synthetic-900:* We generate a random rank-one matrix $Y^* = uv^T$ with unit L_∞-norm, $m = 500$k, and $n = 900$.

2. *MovieLens10M (Top 900):* We pick the $n = 900$ most rated movies from the Movielens10M dataset, which has $m \approx 70$k users of the ≈ 71k users in the dataset.

3. *Netflix (Top 900):* We pick the $n = 900$ most rated movies from the Netflix prize dataset, which has $m \approx 477$k users of the ≈ 480k users in the dataset.

4. *Yahoo! Music (Top 900):* We pick the $n = 900$ most rated songs from the Yahoo! music dataset, which has $m \approx 998$k users of the ≈ 1m users in the dataset. We rescale the ratings to be from 0 to 5.

We follow the same experimental procedure as above. For the nuclear norm bound k, we set it to the actual nuclear norm for Synthetic-900 dataset, and choose from $\{150\text{k}, 160\text{k}\}$ for Netflix, $\{50\text{k}, 60\text{k}\}$ for MovieLens10M, and $\{260\text{k}, 270\text{k}\}$ for the Yahoo! Music dataset. We choose T from various values in $[5, 50]$.

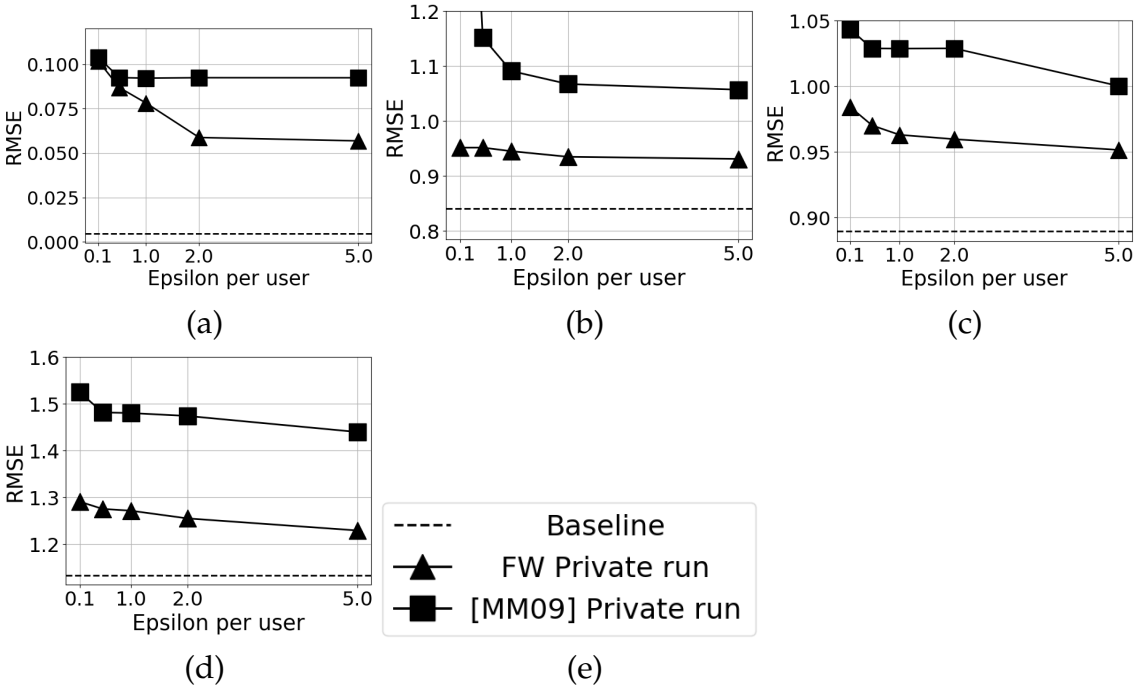

Figure 6.3: Root mean squared error (RMSE) vs. ϵ, on (a) Synthetic-900, (b) MovieLens10M, (c) Netflix, and (d) Yahoo! Music datasets, for $\delta = 10^{-6}$. A legend for all the plots is given in (e).

In Figure 6.3, we show the results of our experiments on the Synthetic-900 dataset in plot (a), MovieLens10M (Top 900) in plot (b), Netflix (Top 900) in plot (c), and Yahoo! Music (Top 900) in plot (d). In all the plots, we see that the test RMSE for private Frank-Wolfe almost always incurs a significantly lower error than the

method of McSherry & Mironov (2009).

CHAPTER 7

Conclusions and Open Problems

The main contribution of our work has been to design scalable private learning techniques that provide generalization guarantees comparable to the best possible non-private one within the class of interest. Additionally, the techniques have been designed to be widely applicable and easy to implement. For future work, there are several directions that we think will be interesting.

Private Convex Optimization: We developed Approximate Minima Perturbation, a practical algorithm for private convex optimization that can leverage any off-the-shelf optimizer, and has a competitive hyperparameter-free variant that can be used for supervised learning. We have also performed an extensive empirical evaluation of state-of-the-art approaches for differentially private convex optimization. This benchmark provides a standard point of comparison for further advances in differentially private convex optimization.

The utility guarantee of our AMP technique (Theorem 2) applies when the model space is $\mathbb{R}^n$ (i.e., unconstrained optimization). It will be interesting to understand AMP's utility when the model space is an n-dimensional ball with a fixed diameter (i.e., constrained optimization). Another important direction to explore is whether the assumptions we make, namely convexity and smoothness of the loss function, are necessary for methods similar to Objective Perturbation (including our method AMP) to provide a privacy guarantee.

Model-agnostic Private Learning: We designed an algorithm with formal utility guarantees for obtaining private classifiers, in the presence of a limited amount of opt-in data, while requiring only a black-box access to a non-private learner.

An important direction is to extend this framework beyond classification tasks, for example, to regression. Moreover, our algorithm is designed to aggregate classification labels, which are discrete scalars. It will be interesting to see if there are effective techniques for aggregating gradients, which are continuous vectors. Such techniques can widen the applicability of the framework.

Private Matrix Completion: We designed the Private Frank-Wolfe algorithm for private matrix completion that provides strong user-level privacy guarantees along with formal utility guarantees and a strong empirical performance. We also gave an optimal differentially private algorithm for singular vector computation, that provides significant savings in terms of space and time when operating on sparse matrices.

It will be interesting to understand the optimal dependence of the generalization error for our Private Frank-Wolfe technique on the number of users and the number of items. Extending our designed techniques to other popular matrix completion methods, like alternating minimization, is another promising direction.

BIBLIOGRAPHY

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., & Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org, http://tensorflow.org/.

Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., & Zhang, L. (2016). Deep learning with differential privacy. In *Proceedings of the 2016 Association for Computing Machinery (ACM) SIGSAC Conference on Computer and Communications Security*, CCS '16, (pp. 308–318). New York, NY, USA: Association for Computing Machinery (ACM).

Allen-Zhu, Z., & Li, Y. (2017). First efficient convergence for streaming k-pca: A global, gap-free, and near-optimal rate. In *58th Institute of Electrical and Electronics Engineers (IEEE) Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, (pp. 487–492).

Bassily, R., Smith, A., & Thakurta, A. (2014a). Private empirical risk minimization: Efficient algorithms and tight error bounds. In *Foundations of Computer Science (FOCS), 2014 Institute of Electrical and Electronics Engineers (IEEE) 55th Annual Symposium on*, (pp. 464–473). Institute of Electrical and Electronics Engineers (IEEE).

Bassily, R., Smith, A. D., & Thakurta, A. (2014b). Private empirical risk minimization, revisited. *Computing Research Repository (CoRR)*, *abs/1405.7085*.

Bassily, R., Thakurta, A. G., & Thakkar, O. (2018). Model-agnostic private learning. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, (pp. 7102–7112).

Beimel, A., Kasiviswanathan, S. P., & Nissim, K. (2010). Bounds on the Sample Complexity for Private Learning and Private Data Release. In *Theory of Cryptography Conference (TCC)*, (pp. 437–454). Springer.

Beimel, A., Nissim, K., & Stemmer, U. (2013). Characterizing the sample complexity of private learners. In *Innovations in Theoretical Computer Science (ITCS)*, (pp. 97–110). Association for Computing Machinery (ACM).

Beimel, A., Nissim, K., & Stemmer, U. (2016). Private learning and sanitization: Pure vs. approximate differential privacy. *Theory of Computing*, *12*(1), 1–61.

Bennett, J., & Lanning, S. (2007). The netflix prize. In *In KDD Cup and Workshop in conjunction with KDD*.

Billingsley, P. (1995). *Probability and Measure*. Wiley Series in Probability and Statistics. Wiley.

Blum, A., Dwork, C., McSherry, F., & Nissim, K. (2005). Practical privacy: the SuLQ framework. In *Proceedings of the twenty-fourth Association for Computing Machinery (ACM) SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, (pp. 128–138). Association for Computing Machinery (ACM).

Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., Ramage, D., Segal, A., & Seth, K. (2017). Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 Association for Computing Machinery (ACM) Special Interest Group on Security, Audit and Control (SIGSAC) Conference on Computer and Communications Security*, CCS '17, (pp. 1175–1191). New York, NY, USA: ACM. http://doi.acm.org/10.1145/3133956.3133982.

Boucheron, S., Bousquet, O., & Lugosi, G. (2005). Theory of classification: A survey of some recent advances. *European Series in Applied and Industrial Mathematics (ESAIM): probability and statistics*, *9*, 323–375.

Boyd, S., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge university press.

Breiman, L. (1996). Bagging predictors. *Machine learning*, *24*(2), 123–140.

Bubeck, S., et al. (2015). Convex optimization: Algorithms and complexity. *Foundations and Trends® in Machine Learning*, *8*(3-4), 231–357.

Bun, M., Nissim, K., Stemmer, U., & Vadhan, S. P. (2015). Differentially private release and learning of threshold functions. In *Institute of Electrical and Electronics Engineers (IEEE) 56th Annual Symposium on Foundations of Computer Science (FOCS) 2015, Berkeley, CA, USA, 17-20 October, 2015*, (pp. 634–649).

Bun, M., & Steinke, T. (2016). Concentrated differential privacy: Simplifications, extensions, and lower bounds. In *Theory of Cryptography Conference (TCC)*, (pp. 635–658).

Bun, M., Ullman, J., & Vadhan, S. (2014). Fingerprinting codes and the price of approximate differential privacy. In *Proceedings of the Forty-sixth Annual Association for Computing Machinery (ACM) Symposium on Theory of Computing*, STOC '14, (pp. 1–10). New York, NY, USA: Association for Computing Machinery (ACM).

Cai, J.-F., Candès, E. J., & Shen, Z. (2010). A singular value thresholding algorithm for matrix completion. *Society for Industrial and Applied Mathematics (SIAM) Journal on Optimization*, (pp. 1956–1982).

Calandrino, J. A., Kilzer, A., Narayanan, A., Felten, E. W., & Shmatikov, V. (2011). "You Might Also Like": Privacy risks of collaborative filtering. In *Institute of Electrical and Electronics Engineers (IEEE) Symposium on Security and Privacy*, (pp. 231–246).

Candes, E., & Recht, B. (2012). Exact matrix completion via convex optimization. *Communications of the Association for Computing Machinery (ACM)*, (pp. 111–119).

Carlini, N., Liu, C., Kos, J., Erlingsson, Ú., & Song, D. (2018). The secret sharer: Measuring unintended neural network memorization & extracting secrets. *Computing Research Repository (CoRR), abs/1802.08232*.

Chan, T.-H. H., Shi, E., & Song, D. (2011). Private and continual release of statistics. *Association for Computing Machinery (ACM) Transactions on Information and System Security*, *14*(3), 26:1–26:24.

Chaudhuri, K., & Hsu, D. J. (2011). Sample complexity bounds for differentially private learning. In *COLT 2011 - The 24th Annual Conference on Learning Theory, June 9-11, 2011, Budapest, Hungary*, (pp. 155–186). http://proceedings.mlr.press/v19/chaudhuri11a/chaudhuri11a.pdf.

Chaudhuri, K., Monteleoni, C., & Sarwate, A. D. (2011). Differentially private empirical risk minimization. *Journal of Machine Learning Research*, *12*(Mar), 1069–1109.

Chaudhuri, K., & Vinterbo, S. (2013). A stability-based validation procedure for differentially private machine learning. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, (pp. 2652–2660). USA: Curran Associates Inc.

Clarkson, K. L. (2010). Coresets, sparse greedy approximation, and the frank-wolfe algorithm. *Association for Computing Machinery (ACM) Transactions on Algorithms (TALG)*, (pp. 63:1–63:30).

Dasgupta, S., & Schulman, L. (2007). A probabilistic analysis of EM for mixtures of separated, spherical gaussians. *Journal of Machine Learning Research (JMLR)*, (pp. 203–226).

Dinur, I., & Nissim, K. (2003). Revealing information while preserving privacy. In *Proceedings of the Twenty-Second Association for Computing Machinery (ACM) Special Interest Group on Algorithms and Computation Theory (SIGACT)-Special Interest*

Group on Management of Data (SIGMOD)-Special Interest Group on Artificial Intelligence (SIGART) Symposium on Principles of Database Systems, June 9-12, 2003, San Diego, CA, USA, (pp. 202–210).

Duchi, J. C., Jordan, M. I., & Wainwright, M. J. (2013). Local privacy and statistical minimax rates. In *Foundations of Computer Science (FOCS), 2013 Institute of Electrical and Electronics Engineers (IEEE) 54th Annual Symposium on*, (pp. 429–438). Institute of Electrical and Electronics Engineers (IEEE).

Dwork, C., & Feldman, V. (2018). Privacy-preserving prediction. In *Conference On Learning Theory*, (pp. 1693–1702).

Dwork, C., Feldman, V., Hardt, M., Pitassi, T., Reingold, O., & Roth, A. (2015a). Generalization in adaptive data analysis and holdout reuse. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, (pp. 2350–2358). Cambridge, MA, USA: MIT Press.

Dwork, C., Kenthapadi, K., McSherry, F., Mironov, I., & Naor, M. (2006a). Our data, ourselves: Privacy via distributed noise generation. In *EUROCRYPT*, (pp. 486–503).

Dwork, C., McSherry, F., Nissim, K., & Smith, A. (2006b). Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography Conference*, (pp. 265–284). Springer.

Dwork, C., Naor, M., Reingold, O., Rothblum, G., & Vadhan, S. (2009). On the complexity of differentially private data release: efficient algorithms and hardness results. In *Symposium on Theory of Computing (STOC)*, (pp. 381–390).

Dwork, C., Roth, A., et al. (2014a). The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, *9*(3-4), 211–407.

Dwork, C., & Rothblum, G. N. (2016). Concentrated differential privacy. *Computing Research Repository (CoRR)*, *abs/1603.01887*.

Dwork, C., Rothblum, G. N., & Vadhan, S. P. (2010). Boosting and differential privacy. In *Foundations of Computer Science (FOCS)*, (pp. 51–60).

Dwork, C., Smith, A., Steinke, T., Ullman, J., & Vadhan, S. (2015b). Robust traceability from trace amounts. In *2015 Institute of Electrical and Electronics Engineers (IEEE) 56th Annual Symposium on Foundations of Computer Science*, (pp. 650–669).

Dwork, C., Talwar, K., Thakurta, A., & Zhang, L. (2014b). Analyze gauss: optimal bounds for privacy-preserving principal component analysis. In *Proceedings of the 46th Annual Association for Computing Machinery (ACM) Symposium on Theory of Computing*, (pp. 11–20). Association for Computing Machinery (ACM).

Dwork, C., Talwar, K., Thakurta, A., & Zhang, L. (2014c). Randomized response strikes back: Private singular subspace computation with (nearly) optimal error guarantees. In *Symposium on Theory of Computing (STOC)*.

Evans, D., Kolesnikov, V., & Rosulek, M. (2018). A pragmatic introduction to secure multi-party computation. *Foundations and Trends in Privacy and Security*, *2*(2-3), 70–246.

Feldman, V. (2016). Generalization of erm in stochastic convex optimization: The dimension strikes back. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, & R. Garnett (Eds.) *Advances in Neural Information Processing Systems 29*, (pp. 3576–3584). Curran Associates, Inc.

Feldman, V., Mironov, I., Talwar, K., & Thakurta, A. (2018). Privacy amplification by iteration. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, (pp. 521–532). https://doi.org/10.1109/FOCS.2018.00056.

Frank, M., & Wolfe, P. (1956). An algorithm for quadratic programming. *Naval research logistics quarterly*, *3*(1-2), 95–110.

Ganta, S. R., Kasiviswanathan, S. P., & Smith, A. (2008). Composition attacks and auxiliary information in data privacy. In *Proceedings of the 14th Association for Computing Machinery (ACM) Special Interest Group on Knowledge Discovery and Data Mining (SIGKDD) International Conference on Knowledge Discovery and Data mining*, (pp. 265–273). Association for Computing Machinery (ACM).

Goldberg, K., Roeder, T., Gupta, D., & Perkins, C. (2001). Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, *4*(2), 133–151.

Hamm, J., Cao, Y., & Belkin, M. (2016). Learning privately from multiparty data. In *International Conference on Machine Learning*, (pp. 555–563).

Hardt, M., & Roth, A. (2012). Beating randomized response on incoherent matrices. In *Symposium on Theory of Computing (STOC)*, (pp. 1255–1268).

Hardt, M., & Roth, A. (2013). Beyond worst-case analysis in private singular vector computation. In *Symposium on Theory of Computing (STOC)*, (pp. 331–340).

Hardt, M., & Rothblum, G. N. (2010). A multiplicative weights mechanism for privacy-preserving data analysis. In *Foundations of Computer Science (FOCS)*, (pp. 61–70).

Hardt, M., & Wootters, M. (2014). Fast matrix completion without the condition number. In *Computational Learning Theory (COLT)*, (pp. 638–678).

Harper, F. M., & Konstan, J. A. (2015). The movielens datasets: History and context. *Association for Computing Machinery (ACM) Transactions on Intelligent Systems*, (pp. 19:1–19:19).

Homer, N., Szelinger, S., Redman, M., Duggan, D., Tembe, W., Muehling, J., Pearson, J. V., Stephan, D. A., Nelson, S. F., & Craig, D. W. (2008). Resolving individuals contributing trace amounts of dna to highly complex mixtures using high-density snp genotyping microarrays. *PLoS genetics*, *4*(8), e1000167.

Iyengar, R., Near, J. P., Song, D., Thakkar, O., Thakurta, A., & Wang, L. (2019a). Differentially private convex optimization benchmark. https://github.com/sunblaze-ucb/dpml-benchmark.

Iyengar, R., Near, J. P., Song, D., Thakkar, O., Thakurta, A., & Wang, L. (2019b). Towards practical differentially private convex optimization. In *Proceedings of the 40th Institute of Electrical and Electronics Engineers (IEEE) Symposium on Security and Privacy (SP)*, (pp. 1–18).

Jaggi, M. (2013). Revisiting frank-wolfe: projection-free sparse convex optimization. In *Proceedings of the 30th International Conference on International Conference on Machine Learning-Volume 28*, (pp. 427–435). JMLR. org.

Jaggi, M., Sulovsk, M., et al. (2010). A simple algorithm for nuclear norm regularized problems. In *ICML*, (pp. 471–478).

Jain, P., Jin, C., Kakade, S. M., Netrapalli, P., & Sidford, A. (2016). Streaming PCA: Matching matrix bernstein and near-optimal finite sample guarantees for Oja's algorithm. In *Conference on Learning Theory*, (pp. 1147–1164).

Jain, P., Kothari, P., & Thakurta, A. (2012). Differentially private online learning. In *Conference on Learning Theory (COLT)*, vol. 23, (pp. 24.1–24.34).

Jain, P., Meka, R., & Dhillon, I. S. (2010). Guaranteed rank minimization via singular value projection. In *NIPS*, (pp. 937–945).

Jain, P., Netrapalli, P., & Sanghavi, S. (2013). Low-rank matrix completion using alternating minimization. In *Proceedings of the forty-fifth annual Association for Computing Machinery (ACM) symposium on Theory of computing*, (pp. 665–674). Association for Computing Machinery (ACM).

Jain, P., Thakkar, O., & Thakurta, A. (2018). Differentially private matrix completion revisited. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, (pp. 2220–2229).

Jain, P., & Thakurta, A. (2013). Differentially private learning with kernels. In *International Conference on Machine Learning (ICML)*, (pp. 118–126).

Jain, P., & Thakurta, A. (2014). (Near) dimension independent risk bounds for differentially private learning. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, (pp. I–476–I–484).

Ji, Z., Lipton, Z. C., & Elkan, C. (2014). Differential privacy and machine learning: a survey and review. *Computing Research Repository (CoRR)*, *abs/1412.7584*.

Jin, C., Kakade, S. M., & Netrapalli, P. (2016). Provable efficient online matrix completion via non-convex stochastic gradient descent. In *Neural Information Processing Systems (NIPS)*, (pp. 4520–4528).

Kairouz, P., Oh, S., & Viswanath, P. (2017). The composition theorem for differential privacy. *Institute of Electrical and Electronics Engineers (IEEE) Transactions on Information Theory*, *63*(6), 4037–4049.

Kapralov, M., & Talwar, K. (2013). On differentially private low rank approximation. In *Symposium on Discrete Algorithms (SODA)*, (pp. 1395–1414).

Kasiviswanathan, S. P., Lee, H. K., Nissim, K., Raskhodnikova, S., & Smith, A. (2008). What can we learn privately? In *FOCS*, (pp. 531–540). Institute of Electrical and Electronics Engineers (IEEE) Computer Society.

Kasiviswanathan, S. P., & Smith, A. (2008). A note on differential privacy: Defining resistance to arbitrary side information. *Computing Research Repository (CoRR)*, *arXiv:0803.39461 [cs.CR]*.

Kearns, M., Pai, M., Roth, A., & Ullman, J. (2014). Mechanism design in large games: Incentives and privacy. In *Innovations in Theoretical Computer Science (ITCS)*, (pp. 403–410).

Kearns, M. J., & Vazirani, U. V. (1994). *An Introduction to Computational Learning Theory*. Cambridge, MA, USA: MIT Press.

Keshavan, R. H., Montanari, A., & Oh, S. (2010). Matrix completion from a few entries. *Institute of Electrical and Electronics Engineers (IEEE) Transactions on Information Theory*, (pp. 2980–2998).

Kifer, D., Smith, A., & Thakurta, A. (2012). Private convex empirical risk minimization and high-dimensional regression. *Journal of Machine Learning Research*, *1*, 25.1–25.40.

Konecný, J., McMahan, H. B., Yu, F. X., Richtárik, P., Suresh, A. T., & Bacon, D. (2016). Federated learning: Strategies for improving communication efficiency. *Computing Research Repository (CoRR)*, *abs/1610.05492*.

Konečný, J., McMahan, H. B., Ramage, D., & Richtárik, P. (2016). Federated optimization: Distributed machine learning for on-device intelligence. *ArXiv*, *abs/1610.02527*.

Koren, Y., & Bell, R. M. (2015). Advances in collaborative filtering. In *Recommender Systems Handbook*, (pp. 77–118). Springer.

Korolova, A. (2010). Privacy violations using microtargeted ads: A case study. In *2010 Institute of Electrical and Electronics Engineers (IEEE) International Conference on Data Mining Workshops*, (pp. 474–482). Institute of Electrical and Electronics Engineers (IEEE).

Lacoste-Julien, S. (2016). Convergence rate of frank-wolfe for non-convex objectives. *Computing Research Repository (CoRR)*, *abs/1607.00345*.

Lacoste-Julien, S., & Jaggi, M. (2013). An Affine Invariant Linear Convergence Analysis for Frank-Wolfe Algorithms. *arXiv e-prints*, (p. arXiv:1312.7864).

Lacoste-Julien, S., & Jaggi, M. (2015). On the global linear convergence of Frank-Wolfe optimization variants. In *Advances in Neural Information Processing Systems*, (pp. 496–504).

Lin, Z., Chen, M., & Ma, Y. (2010). The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *Computing Research Repository (CoRR)*, *abs/1009.5055*.

Lindell, Y., & Pinkas, B. (2008). Secure multiparty computation for privacy-preserving data mining. *International Association for Cryptologic Research (IACR) Cryptology ePrint Archive*, *2008*, 197.

Liu, Z., Wang, Y.-X., & Smola, A. (2015). Fast differentially private matrix factorization. In *Proceedings of the 9th Association for Computing Machinery (ACM) Conference on Recommender Systems*, (pp. 171–178).

McMahan, B., Moore, E., Ramage, D., Hampson, S., & y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA*, (pp. 1273–1282). http://proceedings.mlr.press/v54/mcmahan17a.html.

McMahan, B., & Ramage, D. (2017). Federated learning: Collaborative machine learning without centralized training data. *Google Research Blog*, *3*.

McSherry, F., & Mironov, I. (2009). Differentially private recommender systems: building privacy into the net. In *Symp. Knowledge Discovery and Datamining (KDD)*, (pp. 627–636). Association for Computing Machinery (ACM) New York, NY, USA.

Melis, L., Song, C., De Cristofaro, E., & Shmatikov, V. (2018). Exploiting Unintended Feature Leakage in Collaborative Learning. *arXiv e-prints*, (p. arXiv:1805.04049).

Narayanan, A., & Shmatikov, V. (2010). Myths and fallacies of "personally identifiable information". *Communications of the Association for Computing Machinery (ACM)*, *53*(6), 24–26.

Nikolov, A., Talwar, K., & Zhang, L. (2013). The geometry of differential privacy: The sparse and approximate cases. In *Proceedings of the Forty-fifth Annual Association for Computing Machinery (ACM) Symposium on Theory of Computing*, STOC '13, (pp. 351–360). New York, NY, USA: Association for Computing Machinery (ACM).

Nissim, K., Raskhodnikova, S., & Smith, A. (2007). Smooth sensitivity and sampling in private data analysis. In *Symposium on Theory of Computing (STOC)*, (pp. 75–84).

Papernot, N., Abadi, M., Erlingsson, U., Goodfellow, I., & Talwar, K. (2016). Semi-supervised knowledge transfer for deep learning from private training data. *arXiv preprint arXiv:1610.05755*.

Papernot, N., Song, S., Mironov, I., Raghunathan, A., Talwar, K., & Erlingsson, Ú. (2018). Scalable private learning with pate. *arXiv preprint arXiv:1802.08908*.

Paulavičius, R., & Žilinskas, J. (2006). Analysis of different norms and corresponding lipschitz constants for global optimization. *Ukio Technologinis ir Ekonominis Vystymas*, *12*(4), 301–306.

Recht, B. (2011). A simpler approach to matrix completion. *Journal of Machine Learning Research*, (pp. 3413–3430).

Reyzin, L., Smith, A. D., & Yakoubov, S. (2018). Turning HATE into LOVE: homomorphic ad hoc threshold encryption for scalable MPC. *The International Association for Cryptologic Research (IACR) Cryptology ePrint Archive*, *2018*, 997.

Sankararaman, S., Obozinski, G., Jordan, M. I., & Halperin, E. (2009). Genomic privacy and limits of individual detection in a pool. *Nature genetics*, *41*(9), 965.

Shalev-Shwartz, S., & Ben-David, S. (2014). *Understanding machine learning: From theory to algorithms*. Cambridge university press.

Shalev-Shwartz, S., Gonen, A., & Shamir, O. (2011). Large-scale convex minimization with a low-rank constraint. *arXiv preprint arXiv:1106.1622*.

Shamir, O., & Shalev-Shwartz, S. (2011). Collaborative filtering with the trace norm: Learning, bounding, and transducing. In *Conference on Learning Theory (COLT)*, (pp. 661–678).

Shokri, R., & Shmatikov, V. (2015). Privacy-preserving deep learning. In *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, (pp. 909–910).

Shokri, R., Stronati, M., Song, C., & Shmatikov, V. (2017). Membership inference attacks against machine learning models. In *2017 Institute of Electrical and Electronics Engineers (IEEE) Symposium on Security and Privacy (SP)*, (pp. 3–18).

Smith, A., & Thakurta, A. (2013). Differentially private feature selection via stability arguments, and the robustness of the lasso. In *Conference on Learning Theory (COLT)*, (pp. 819–850).

Song, S., Chaudhuri, K., & Sarwate, A. D. (2013). Stochastic gradient descent with differentially private updates. In *Global Conference on Signal and Information Processing (GlobalSIP), 2013 Institute of Electrical and Electronics Engineers (IEEE)*, (pp. 245–248). Institute of Electrical and Electronics Engineers (IEEE).

Srebro, N., & Shraibman, A. (2005). Rank, trace-norm and max-norm. In *International Conference on Computational Learning Theory*, (pp. 545–560).

Sweeney, L. (2002). K-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, *10*(5), 557–570.

Talwar, K., Thakurta, A., & Zhang, L. (2014). Private empirical risk minimization beyond the worst case: The effect of the constraint set geometry. *Computing Research Repository (CoRR)*, *abs/1411.5417*.

Talwar, K., Thakurta, A., & Zhang, L. (2015). Nearly optimal private lasso. In *Neural Information Processing Systems (NIPS)*, (pp. 3025–3033).

Tao, T. (2012). *Topics in random matrix theory*, vol. 132. American Mathematical Society.

Tewari, A., Ravikumar, P. K., & Dhillon, I. S. (2011). Greedy algorithms for structurally constrained high dimensional problems. In *Neural Information Processing Systems (NIPS)*, (pp. 882–890).

Thakurta, A. G., & Smith, A. (2013). (Nearly) optimal algorithms for private online learning in full-information and bandit settings. In C. Burges, L. Bottou, M. Welling, Z. Ghahramani, & K. Weinberger (Eds.) *Advances in Neural Information Processing Systems 26*, (pp. 2733–2741). Curran Associates Inc.

Valiant, L. G. (1984). A theory of the learnable. *Communications of the Association for Computing Machinery (ACM)*, *27*(11), 1134–1142.

Wang, Y.-X., Fienberg, S., & Smola, A. (2015). Privacy for free: Posterior sampling and stochastic gradient monte carlo. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, (pp. 2493–2502).

Wu, X., Fredrikson, M., Jha, S., & Naughton, J. F. (2016). A methodology for formalizing model-inversion attacks. In *2016 Institute of Electrical and Electronics Engineers (IEEE) 29th Computer Security Foundations Symposium (CSF)*, (pp. 355–370).

Wu, X., Fredrikson, M., Wu, W., Jha, S., & Naughton, J. F. (2015). Revisiting differentially private regression: Lessons from learning theory and their consequences. *Computing Research Repository (CoRR)*, *abs/1512.06388*.

Wu, X., Li, F., Kumar, A., Chaudhuri, K., Jha, S., & Naughton, J. (2017). Bolt-on differential privacy for scalable stochastic gradient descent-based analytics. In *Proceedings of the 2017 Association for Computing Machinery (ACM) International Conference on Management of Data*, SIGMOD '17, (pp. 1307–1322). New York, NY, USA: Association for Computing Machinery (ACM).

Yahoo (2011). C15 - Yahoo! music user ratings of musical tracks, albums, artists and genres, version 1.0. *Webscope*.

Yu, H.-F., Jain, P., Kar, P., & Dhillon, I. (2014). Large-scale multi-label learning with missing labels. In *International Conference on Machine Learning (ICML)*, (pp. 593–601).

Zhang, J., Zhang, Z., Xiao, X., Yang, Y., & Winslett, M. (2012). Functional mechanism: Regression analysis under differential privacy. *Proceedings of the Very Large Database (VLDB) Endowment*, *5*(11), 1364–1375.

Zhang, L., Yang, T., & Jin, R. (2017). Empirical risk minimization for stochastic convex optimization: $O(1/n)$- and $O(1/n^2)$-type of risk bounds. In S. Kale, & O. Shamir (Eds.) *Proceedings of the 2017 Conference on Learning Theory*, vol. 65 of *Proceedings of Machine Learning Research*, (pp. 1954–1979). Amsterdam, Netherlands.

CURRICULUM VITAE

Om Dipakbhai Thakkar

August 16, 2019

omthkkr@bu.edu www.omthakkar.com

Academic Training:

09/2019 (expected)	Ph.D. (Computer Science), Boston University, MA, USA
05/2014	B.Tech. (Information and Communication Technology), Dhirubhai Ambani Institute of Information and Communication Technology, Gujarat, India

Doctoral Research:

Title:	Advances in Privacy-preserving Machine Learning
Thesis advisor:	Dr. Adam Smith
Defense date:	August 1, 2019
Summary:	In this work, we design differentially private learning algorithms with performance comparable to the best possible non-private ones. We begin by presenting a technique for practical differentially private convex optimization that can leverage any off-the-shelf optimizer as a black box. Next, we give a learning algorithm that outputs a private classifier when given black-box access to a non-private learner and a limited amount of unlabeled public data. Lastly, we provide the first algorithm for matrix completion with provable user-level privacy and accuracy guarantees, which can also be used to design private recommendation systems.

Original, Peer Reviewed Publications (newest first):

1. Roger Iyengar, Joseph P. Near, Dawn Song, Om Thakkar, Abhradeep Thakurta and Lun Wang. Towards Practical Differentially Private Convex Optimization. *Security and Privacy (S&P), 2019*.

2. Raef Bassily, Om Thakkar, and Abhradeep Thakurta. Model-Agnostic Private Learning. *Neural Information Processing Systems (NeurIPS), 2018*. (Accepted for an oral presentation)

3. Prateek Jain, Om Thakkar, and Abhradeep Thakurta. Differentially Private Matrix Completion Revisited. *International Conference on Machine Learning (ICML), 2018*. (Accepted for a long talk)

4. Ryan Rogers, Aaron Roth, Adam Smith, and Om Thakkar. Max-Information, Differential Privacy, and Post-Selection Hypothesis Testing. *Foundations of Computer Science (FOCS), 2016*.