Title
Towards Efficient and Effective Privacy-Preserving Machine Learning

Author
Wang, Lingxiao

Publication Date
2021

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

Towards Efficient and Effective Privacy-Preserving Machine Learning

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Computer Science

by

Lingxiao Wang

2021

ABSTRACT OF THE DISSERTATION

Towards Efficient and Effective Privacy-Preserving
Machine Learning

by

Lingxiao Wang
Doctor of Philosophy in Computer Science
University of California, Los Angeles, 2021
Professor Quanquan Gu, Chair

The past decade has witnessed the fast growth and tremendous success of machine learning. However, recent studies showed that existing machine learning models are vulnerable to privacy attacks, such as membership inference attacks, and thus pose severe threats to personal privacy. Therefore, one of the major challenges in machine learning is to learn effectively from enormous amounts of sensitive data without giving up on privacy. This dissertation summarizes our contributions to the field of privacy-preserving machine learning, i.e., solving machine learning problems with strong privacy and utility guarantees.

In the first part of the dissertation, we consider the privacy-preserving sparse learning problem. More specifically, we establish a novel differentially private hard-thresholding method as well as a knowledge-transfer framework for solving the sparse learning problem. We show that our proposed methods are not only efficient but can also achieve improved privacy and utility guarantees.

In the second part of the dissertation, we propose novel efficient and effective algorithms for solving empirical risk minimization problems. To be more specific, our proposed algo-

rithms can reduce the computational complexities and improve the utility guarantees for solving nonconvex optimization problems such as training deep neural networks.

In the last part of the dissertation, we study the privacy-preserving empirical risk minimization in the distributed setting. In such a setting, we propose a new privacy-preserving framework by combining the multi-party computation (MPC) protocol and differentially private mechanisms and show that our framework can achieve better privacy and utility guarantees compared with existing methods.

The methods and techniques proposed in this dissertation form a line of researches that deepens our understandings of the trade-off between privacy, utility and efficient in privacy-preserving machine learning, and could also help us develop more efficient and effective private learning algorithms.

The dissertation of Lingxiao Wang is approved.

Stanley Osher

Wei Wang

Amit Sahai

Quanquan Gu, Committee Chair

University of California, Los Angeles

2021

To my family

TABLE OF CONTENTS

1 Introduction **1**

2 Differentially Private Iterative Gradient Hard Thresholding for Sparse Learning **5**

2.1 Introduction 5

2.2 Related Work 7

2.3 Preliminaries 9

2.4 Algorithmic Framework 11

2.5 Main Results 13

2.5.1 Results for Generic Model 13

2.5.2 Implications for Specific Examples 15

2.6 Numerical Experiments 17

2.6.1 Synthetic Data Experiments 19

2.6.2 Real Data Experiments 20

2.7 Additional Results 21

2.8 Proofs of the Main Results 21

2.8.1 Proof of Theorem 2.5.2 21

2.8.2 Proof of Theorem 2.5.4 22

2.8.3 Proof of Corollary 2.7.1 25

2.9 Proof of Specific Examples 25

2.9.1 Proof of Corollary 2.5.6 26

2.9.2 Proof of Corollary 2.5.8 27

2.10 Auxiliary Lemmas 28
2.11 Conclusions 29

3 A Knowledge Transfer Framework for Differentially Private Sparse Learning 30

3.1 Introduction 30
3.1.1 Additional Related Work 35
3.2 Proposed Method 35
3.3 Main Results 37
3.3.1 Results for Generic Models 38
3.3.2 Results for Specific Models 40
3.4 Experiments 43
3.4.1 Numerical Simulations 43
3.4.2 Real Data Experiments 45
3.5 Additional Results 46
3.5.1 Additional Main Results 46
3.6 Proofs of the Main Results 47
3.6.1 Proof of Theorem 3.3.3 47
3.6.2 Proof of Theorem 3.3.5 50
3.6.3 Proof of Corollary 3.3.7 52
3.7 Proofs of Specific Examples 53
3.7.1 Proof of Corollary 3.3.9 53
3.7.2 Proof of Corollary 3.3.13 54
3.7.3 Proof of Corollary 3.3.11 55

3.7.4 Proof of Corollary 3.3.15 55
3.8 Proofs of Additional Lemmas 56
3.8.1 Proof of Lemma 3.6.2 56
3.9 Conclusions and Future Work 58

4 Efficient Privacy-Preserving Stochastic Nonconvex Optimization 59

4.1 Introduction 59
4.2 Related Work 61
4.3 Preliminaries 63
4.4 Algorithm 65
4.5 Main Results 67
4.6 Proof Outline of the Main Results 70
4.6.1 Privacy Guarantee 70
4.6.2 Utility Guarantee 71
4.7 Experiments 72
4.7.1 Nonconvex Logistic Regression 73
4.7.2 Convolutional Neural Networks 74
4.8 Additional experiments 78
4.8.1 Results on *ijcnn1* dataset 78
4.8.2 Additional Results on MNIST and CIFAR-10 datasets 79
4.9 Proof of main results 81
4.9.1 Proof of Theorem 4.5.1 81
4.9.2 Proof of Theorem 4.5.4 85
4.10 Proof of Lemma 4.3.7 90

4.11 Auxiliary Lemmas 91
4.12 Conclusions 91

5 Dp-lssgd: A Stochastic Optimization Method to Lift the Utility in Privacy-Preserving ERM 93

5.1 Introduction 93
5.1.1 Our Contributions 95
5.1.2 Related Work 96
5.1.3 Notation 97
5.1.4 Organization 97
5.2 Problem Setup and Algorithm 98
5.2.1 Laplacian Smoothing Stochastic Gradient Descent (LSSGD) 98
5.2.2 DP-LSSGD 99
5.3 Main Theory 100
5.4 Experiments 103
5.4.1 Logistic Regression for MNIST Classification 103
5.4.2 CNN for MNIST and CIFAR10 Classification 106
5.4.3 CNN for CIFAR10 Classification 108
5.5 Proof of the Main Theorems 108
5.5.1 Privacy Guarantee 108
5.5.2 Utility Guarantee – Convex Optimization 112
5.5.3 Utility Guarantee – Nonconvex Optimization 114
5.6 Calculations of β and γ 116
5.6.1 Calculation of γ 116

5.6.2 Calculation of β . . . 119
5.7 Laplacian Smoothing and Diffusion Equation . . . 120
5.8 Conclusions . . . 121

6 Distributed Learning Without Distress: Privacy-preserving ERM . . . 122
6.1 Introduction . . . 122
6.1.1 Contributions . . . 125
6.2 Background on Differential Privacy and Multi-Party Computation . . . 126
6.2.1 Differential Privacy . . . 127
6.2.2 Secure Multi-Party Computation . . . 128
6.3 Multi-Party Machine Learning . . . 129
6.3.1 Model Aggregation with Output Perturbation . . . 130
6.3.2 Iterative Learning with Gradient Perturbation . . . 132
6.4 Experiments . . . 134
6.5 Proofs of the Main Theorems . . . 138
6.5.1 Proof of Theorem 6.3.2 . . . 138
6.5.2 Proof of Theorem 6.3.3 . . . 140
6.5.3 Proof of Corollary 6.3.5 . . . 140
6.5.4 Proof of Theorem 6.3.6 . . . 141
6.5.5 Proof of Theorem 6.3.7 . . . 142
6.6 More Experimental Results . . . 142
6.6.1 Experiments on *KddCup99* dataset . . . 142
6.6.2 Experiments on *KddCup98* dataset . . . 144
6.6.3 Experiments on *Adult* dataset . . . 144

6.7 Implementation of Secure Aggregation 147

7 Conclusion and Future Work 149

LIST OF FIGURES

2.1 Numerical results for sparse linear regression and sparse logistic regression. . . . 17

3.1 Illustration of the proposed teacher-student framework. 32

3.2 Numerical results for sparse linear and logistic regression. 44

4.1 Results for nonconvex logistic regression on *a9a* dataset. (a), (b) illustrate the objective loss versus the number of epochs. (c), (d) present the gradient norm versus the number of epochs. 75

4.2 Results on MNIST dataset. (a), (b) depict the test error under the privacy budget $\epsilon = 3.0$. (c), (d) illustrate the test error under the privacy budget $\epsilon = 1.2$. 75

4.3 Results for CNN6 on CIFAR-10 dataset. (a), (b) depict the test error under the privacy budget $\epsilon = 2.0$. (c), (d) illustrate the test error under the privacy budget $\epsilon = 4.0$ 75

4.4 Results for nonconvex logistic regression on *ijcnn1* dataset. (a), (b) show the objective loss versus the number of epochs. (c), (d) illustrate the gradient norm versus the number of epochs. 78

4.5 Results for CNN on MNIST and CIFAR-10 datasets. (a), (b) illustrate the results on MNIST dataset. (c), (d) demonstrate the results for CNN6 on CIFAR-10 dataset. (e)-(j) show the results for CNN5 on CIFAR-10 dataset. 80

5.1 Training (left) and validation (middle) losses of the logistic regression on the MNIST trained by DP-SGD with $(\epsilon, \delta = 10^{-5})$-DP guarantee. (right): testing accuracy of a simple CNN on the MNIST trained by DP-SGD with $(\epsilon, \delta = 10^{-5})$-DP guarantee. 94

5.2 Illustration of LS ($\sigma = 10$ for $\mathbf{v}_1$ and $\sigma = 100$ for $\mathbf{v}_2$). (a): 1D signal sampled uniformly from $\sin(x)$ for $x \in [0, 2\pi]$. (b), (c), (d): 2D original, noisy, and Laplacian Smoothed noisy signals sampled uniformly from $\sin(x)\sin(y)$ for $(x, y) \in [0, 2\pi] \times [0, 2\pi]$. 100

5.3 Training and validation losses of the multi-class logistic regression by DP-LSSGD. (a) and (b): training and validation curves with $(0.2, 10^{-5})$-DP guarantee; (c) and (d): training and validation curves with $(0.1, 10^{-5})$-DP guarantee. (Average over 5 runs) 104

5.4 Accuracy of the logistic regression on MNIST when different learning rates are used to train the model. Left: $(0.1, 10^{-5})$-DP; Right: $(0.2, 10^{-5})$-DP. 105

5.5 Performance comparison (validation accuracy) between different DP optimization algorithms in training CNN for MNIST classification with a fixed $\delta = 10^{-5}$. . . . 107

5.6 Performance comparison between different differentially private optimization algorithms in training CNN for CIFAR10 classification with a fixed $\delta = 10^{-5}$. . . . 108

5.7 Left & middle panels: Contrasting performance (validation acc) of DP-SGD and DP-LSSGD with different σ and different learning rate. Right panel: ϵ vs. Testing accuracy of the private models trained by different DP-optimization algorithms with a fixed $\delta = 10^{-5}$. 109

6.1 Optimality Gap and Relative Accuracy Loss Comparison on *KDDCup99* ($m = 1,000$). (All models have privacy budget $\epsilon = 0.5$, except **Rajkumar and Agarwal** which consumers $\epsilon = 0.5$ privacy budget each iteration.) 137

6.2 Optimality Gap and Relative Accuracy Loss Comparison on *KDDCup98* ($m = 1,000$). (As in Figure 6.1, all models have privacy budget $\epsilon = 0.5$, except **Rajkumar and Agarwal**.) 138

6.3 Optimality Gap and Relative Accuracy Loss Comparison on *KddCup99* ($m = 100$) *All models except **Rajkumar and Agarwal** have privacy budget $\epsilon = 0.5$. 143

6.4 Optimality Gap and Relative Accuracy Loss Comparison on *KddCup99* ($m = 50,000$) *All models except **Rajkumar and Agarwal** have privacy budget $\epsilon = 0.5$. 143

6.5 Optimality Gap and Relative Accuracy Loss Comparison on *KddCup98* ($m = 100$) *All models except **Rajkumar and Agarwal** have privacy budget $\epsilon = 0.5$. 144

6.6 Optimality Gap and Relative Accuracy Loss Comparison on *KddCup98* ($m = 50,000$) *All models except **Rajkumar and Agarwal** have privacy budget $\epsilon = 0.5$. 145

6.7 Optimality Gap and Relative Accuracy Loss Comparison on *Adult* ($m = 100$) *All models except **Rajkumar and Agarwal** have privacy budget $\epsilon = 0.5$. . . 146

6.8 Optimality Gap and Relative Accuracy Loss Comparison on *Adult* ($m = 1,000$) *All models except **Rajkumar and Agarwal** have privacy budget $\epsilon = 0.5$. . . 146

6.9 Optimality Gap and Relative Accuracy Loss Comparison on *Adult* ($m = 30,000$) *All models except **Rajkumar and Agarwal** have privacy budget $\epsilon = 0.5$. . . 147

LIST OF TABLES

2.1 Comparison of different (ϵ, δ)-DP algorithms for sparse linear regression. We ignore the $\log(1/\delta)$ term in the utility guarantees. Note that γ is the probability that the differentially private model selection algorithms can successfully recover the true support. 9

2.2 Comparison of different algorithms for various privacy budgets ϵ in terms of MSE on the test data and its corresponding standard error on E2006-TFIDF. 18

2.3 Comparison of different algorithms for various privacy budgets ϵ in terms of test error and its corresponding standard deviation on RCV1 data. 18

3.1 Comparison of different algorithms for sparse linear regression in the setting of (ϵ, δ)-DP. We report the utility bound achieved by the privacy-preserving mechanisms, and ignore the $\log(1/\delta)$ term. Note that $n\epsilon \gg 1$, $\mathbf{x}_i$ denotes the i-th input vector, and υ is the probability that the support selection procedure can successfully recover the true support. 34

3.2 Comparison of different algorithms for various privacy budgets ϵ with $\delta = 10^{-5}$ in terms of MSE (mean $\pm$ std) and its corresponding standard deviation on E2006-TFIDF. 44

3.3 Comparison of different algorithms for various privacy budgets ϵ with $\delta = 10^{-5}$ in terms of test error (mean $\pm$ std) and its corresponding standard deviation on RCV1 data. 45

4.1 Comparison of different (ϵ, δ)-DP algorithms for nonconvex optimization. We report the utility bound in terms of $\mathbb{E}\|\nabla F(\boldsymbol{\theta}^p)\|_2$, where $\boldsymbol{\theta}^p$ is the output of the differentially private algorithm, d is the problem dimension, $\mathbb{E}$ is taken over the randomness of the algorithm. 62

4.2 Comparison of different algorithms on *a9a* dataset when $\epsilon \in \{0.2, 0.5\}$ and $\delta = 10^{-5}$. We use the STORM algorithm (CO19) as the non-private baseline. 74

4.3 Comparison of different algorithms on *ijcnn1* dataset under different privacy budgets $\epsilon \in \{0.2, 0.5\}$ and $\delta = 10^{-5}$. Note that the non-private baseline denotes the test error of the non-private STORM algorithm (CO19). 79

5.1 Utility and Differential Privacy Guarantees. 95

5.2 Testing accuracy of the multi-class logistic regression trained by DP-LSSGD with $(\epsilon, \delta = 10^{-5})$-DP guarantee and different LS parameter σ. Unit: %. (5 runs) . . 105

5.3 The values of γ corresponding to some σ and d. 119

5.4 The values of β corresponding to some σ and d. 120

6.1 Comparison of noise magnitudes for various multi-party differential privacy methods. 135

6.2 Secure Aggregation in MPC 148

ACKNOWLEDGMENTS

I would like to express my special thanks to my advisor Quanquan Gu who gave me the opportunity to start, learn and grow in the past years. He is one of the best advisors I ever met and provides me with invaluable advice and helps me in difficult periods. He always encourages me to challenge myself and explore new areas that help me grow fast. Without him, I could never have completed my Ph.D. study. I'm also grateful that I can have the opportunity to study at UVa and UCLA with Dr. Gu and met so many great researchers.

I also want to thank all my coworkers during the past few years. They are always kindful, easy-going and knowledgeable. I benefited a lot from the collaborations. The collaboration with Prof. David Evans offers me the opportunity to start my journey in privacy-preserving machine learning. His suggestions on research and career are very helpful. The collaboration with Prof. Miryung Kim gives me the precious experience of exploring new adventures in different fields. It's a great pleasure that I can work with and learn from them. I would also like to thank my dissertation committee members, Prof. Amit Sahai, Prof. Wei Wang, and Prof. Stanley Osher, for their helpful feedback and suggestions on my Ph.D. research and dissertation work.

I am really thankful to all the members of Statistical Machine Learning Lab, Pan Xu, Lu Tian, Jinghui Chen, Dongruo Zhou, Difan Zou, Yuan Cao, Jiafan He, Weitong Zhang, Yue Wu, Zixiang Chen. I will definitely miss the time spending with you. I am looking forward to working with you in the future and hope our paths will cross again.

I am also grateful to my mentor/supervisor Dr. Huang during my internship in JD. We build a good collaboration and I gained a lot during my internship. I got in touch with NLP the first time and started showing interest in that research area. I am also fortunate to intern there twice.

I would like to mention the support from my family and my friends. Thank my parents for always give me advice, help me grow and support me to pursue my career. Their love

helps me overcome all the difficulties. Thank my friends who have always been there to accompany me and continue to help me. Finally, I would like to sincerely thanks Mengyao for the extreme support through this journey. Without her love and countless sacrifices, I cannot achieve this point.

VITA

2014	B.S. (Mathematics and Applied Mathematics), University of Science and Technology Beijing.
2016	M.S. (Statistics), University of Washington.
2016-2017	Teaching Assistant, Systems and Information Engineering Department, University of Virginia.
2017-2018	Research Assistant, Computer Science Department, University of Virginia.
2020	Ph.D. Candidate in Computer Science, UCLA.
2018-present	Research Assistant, Computer Science Department, UCLA.
2020-present	Teaching Assistant, Computer Science Department, UCLA.

CHAPTER 1

Introduction

The success of machine learning techniques relies on large-scale data. For many applications such as medical research, it is difficult for them to reap the fruits of machine learning because their data is often highly sensitive such as individual patient records that may not be permitted to share due to privacy expectations or regulations. More importantly, recent studies have found that trained machine learning models can surprisingly leak an individual's sensitive information, which urges us to deal with privacy breaches when we deploy machine learning models. These privacy concerns give rise to the following question: *Is it possible for us to leverage machine learning techniques without giving up on privacy?* The answer is affirmative, and the primary research goal in this dissertation is to develop principled ways to address privacy concerns when using machine learning methods to solve real-world problems.

In the past decade, much work has been done to advance machine learning with differential privacy (DMN06) since differential privacy can provide statistical data privacy for sensitive information and has recently emerged as the new gold standard in data privacy protection. However, design efficient and effective privacy-preserving machine learning methods have many challenges. More specifically, in many modern machine learning applications, the problem dimension can increase with the number of observations, i.e., the high-dimensional setting. Thus, directly applying existing privacy-preserving methods without considering the model structure information such as sparse structure or low-rank structure will often output models with very unsatisfied privacy and utility guarantees. Although, many approaches

(KST12; TS13; JT14; TTZ15) have been proposed to solve differentially private learning problems in the high-dimensional setting, theses methods either have unsatisfactory utility guarantees or are computationally inefficient. Therefore, the natural question to ask here is: can we achieve the best of both worlds, i.e., a strong utility guarantee and high computational efficiency, for solving high-dimensional private learning problems? Two of my works (see Chapter 2 and Chapter 3) aim at addressing this question.

In addition, many machine learning problems such as training deep neural networks are nonconvex optimization problems, which makes designing efficient and effective privacy-preserving machine learning methods much more challenging. For instance, the most popular method to train private deep neural networks is differentially private stochastic gradient descent (DP-SGD) (BST14a; WYX17a; ZZM17b) due to the strong privacy guarantees it can provide. However, DP-SGD is computationally hard for large-scale problems (e.g., large model parameters and training dataset) due to the large variance of the stochastic gradient estimator. Therefore, how to fully utilize the power of non-private machine learning approaches and thus design more efficient algorithms is one of the major challenges in the current privacy-preserving machine learning literature. One of my work (see Chapter 4) aims at address this challenge when we consider the nonconvex optimization problem. Another major issue of existing privacy-preserving machine learning methods lies in their possibly significant degeneration of trained models' utility compared with the model trained using the non-private counterparts. In Chapter 5, we introduce a differentially private Laplacian smoothing stochastic gradient descent method to mitigate this degradation for solving the empirical risk minimization.

In many real-world applications such as medical research and personalized recommendation, data is collected by different organizations and individuals. These parties wish to learn collective models by combining data across institutions to produce more accurate models. However, the privacy concerns become more acute in this distributed/federated learning setting since: (1) different parties do not want to expose their data to others; and (2) the

well-trained collective model should provide protection against inference attacks. Therefore, it is very challenging to design privacy-preserving distributed learning framework which can address these privacy concerns and achieve strong utilities. In Chapter 6, we discuss how to combine the Cryptograph technique and the advanced differentially private mechanism to address these concerns.

The major contributions of my works are summarize as follows. Two of my works aim at developing more efficient and effective privacy-preserving sparse learning methods. More specifically, we propose a differentially private iterative gradient hard thresholding algorithm, which allows us to have strong privacy and utility guarantees. Our algorithm is a synergistic combination of the state-of-the-art privacy-preserving mechanism and sparse learning algorithm, which has a linear convergence rate and can be applied to a broad class of private high-dimensional machine learning with the sparse model structure. Built upon this method, we develop a knowledge transfer framework to further improve the utility guarantee of our method when solving the sparse learning problem. The key insight of the proposed framework is to use a non-private "teacher" model to train a privacy-preserving "student" model while preserving the model structure information of the "teacher" model. We show that our proposed framework can achieve the state-of-the-art utility guarantee compared with exiting methods.

I also study the differentially private nonconvex optimization problem. I develop an efficient privacy-preserving stochastic nonconvex optimization algorithm and prove that our method can reduce the computational complexity and achieves state-of-the-art privacy privacy and utility guarantees compared with DP-SGD. The key insight of our method consists of two parts: (1) leveraging the historical gradient information to consistently reduce the accumulated variance of the gradient estimator; and (2) an adaptive step size to reduce the sensitivity, which is the key to provide strong privacy guarantees, of the gradient estimator. Furthremore, we develop a differentially private Laplacian smoothing stochastic gradientdescent method to improve the utility guarantee of existing methoods. The core of

our method is the Laplacian smoothing, which smooths out the random noise used in the privacy-preserving mechanisms. We show that our new approach could improve the trained private models' utility, both numerically and theoretically.

To address the privacy concern and achieve strong utilities in the distributed setting, we proposed solutions that combines differential privacy with secure multi-party computation (ZE15a). Our solution's key insight comes from two sources: Firstly, we combine different parties' local information within a secure computation, which allows us to protect each party's private records. Secondly, we add the required differential privacy noise inside the secure computation after aggregation, which can reduce the amount of random noise in the privacy-preserving mechanisms. With these two insights, our proposed framework can not only address the privacy concerns in the distributed/federated learning setting but also significantly improve the utilities in privacy-preserving empirical risk minimization.

The rest of the dissertation is organize as follows. We first discuss our efforts in solving differentially private sparse learning problem. In Chapter 2, we introduce our proposed efficient private learning algorithm: differentially private iterative gradient hard thresholding (DP-IGHT). We prove that our proposed algorithm can achieve a linear convergence rate. Based on the proposed algorithm, we introduce a knowledge transfer framework for private sparse learning problem in Chapter 3. Our proposed framework not only enjoys the fast convergence rate of DP-IGHT but also significantly improve the utility guarantees compared with existing methods. We then study the differentially private nonconvex optimization in Chapter 4 and Chapter 5. In Chapter 4, we propose an efficient learning algorithm for training deep neural networks with privacy guarantees. In Chapter 5, we develop an effective method to improve the utility guarantees in private deep learning. Next, we introduce a novel privacy-preserving framework for distributed empirical risk minimization in Chapter 6. Lastly, we conclude the dissertation by elaborating several future research directions in Chapter 7.

CHAPTER 2

Differentially Private Iterative Gradient Hard Thresholding for Sparse Learning

2.1 Introduction

In modern high-dimensional data analytics, where the problem dimension can increase with the number of observations, sparse learning has emerged as a prominent method to alleviate overfitting and provide statistically reliable results. Consequently, many sparse learning algorithms such as ℓ_1 convex relaxation based methods (Tib96; Gee08; NYW09) have been proposed in the past two decades. Compared with ℓ_1 convex relaxation based sparse learning algorithms, ℓ_0 constrained sparse learning algorithms (Zha11; YLZ14; JTK14; CG16) received increasing attention due to its small estimation bias. In specific, the ℓ_0 constrained sparse learning is formulated as follows

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^d} L_S(\boldsymbol{\theta}) := \frac{1}{n}\sum_{i=1}^{n} \ell(\boldsymbol{\theta}; \mathbf{z}_i) \;\; \text{subject to} \;\; \|\boldsymbol{\theta}\|_0 \leq s, \tag{2.1.1}$$

where $S = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n\}$ denotes the training dataset with $\mathbf{z}_i = (\mathbf{x}_i, y_i)$, L_S is the empirical loss function, $\|\boldsymbol{\theta}\|_0$ denotes the number of nonzero entries in $\boldsymbol{\theta}$, s is a parameter for tuning the sparsity level of $\boldsymbol{\theta}$, and we assume that the data are generated from some underlying statistical model with sparse parameter vector $\boldsymbol{\theta}^* \in \mathbb{R}^d$ such that $\|\boldsymbol{\theta}^*\|_0 = s^*$. The goal of sparse learning is to recover $\boldsymbol{\theta}^*$.

In many applications, the data used for sparse learning are sensitive datasets, such as financial records or genomic data, raising a big concern that the adversaries may be able

to infer the private information from the trained model. This privacy concern necessitates the private-preserving algorithms for learning sparse models. The prerequisite for developing such algorithms is a rigorous privacy definition. In recent years, differential privacy (DMN06) has been served as the most widely adopted notion of statistical data privacy and has been applied to many real world applications (EPK14; DKY17). The formal definition of differential privacy is as follows.

Definition 2.1.1 (Differential privacy (DMN06))**.** A randomized mechanism $\mathcal{M} : \mathcal{S}^n \to \mathcal{R}$ satisfies (ϵ, δ)-differential privacy if for any two adjacent data sets $S, S' \in \mathcal{S}^n$ differing by one example, and any output subset $O \subseteq \mathcal{R}$, it holds that

$$\mathbb{P}[\mathcal{M}(S) \in O] \leq e^{\epsilon} \cdot \mathbb{P}[\mathcal{M}(S') \in O] + \delta.$$

According to the definition, differential privacy requires that datasets differing by one example lead to similar distributions on the output of a randomized algorithm. This implies that an adversary will draw essentially the same conclusions about an individual whether or not that individual's data was used even if many records are known a priori to the adversary.

There exist several studies (KST12; TS13; TTZ15) trying to develop differentially private algorihthms for solving sparse learning problems. However, they only consider sparse linear regression, and the convergence rates and utility guarantees of these methods are suboptimal. In order to overcome the limitations of existing differentially private sparse learning algorithms, we propose a differentially private iterative gradient hard thresholding (DP-IGHT) algorithm for solving the sparsity constrained learning problem (2.1.1), which is not only very efficient but also has comparable or even better utility guarantees than the state-of-the-art methods. We summarize the contributions of our work as follows

- Compared with existing work that is limited to sparse linear regression, our differentially private sparse learning algorithm is generic enough that it can be applied to a broad family of loss functions that satisfy the restricted strong convexity and

smoothness conditions (BRT09; NYW09), and each component function is Lipschitz continuous. We demonstrate the superiority of our framework through two concrete examples: sparse linear regression and sparse logistic regression.

- We prove the linear convergence rate for our DP-IGHT algorithm, which outperforms the sub-linear convergence rate of Frank-Wolfe based method (TTZ15), and does not rely on any computationally intractable support selection algorithm as required by (KST12).

- We establish strong utility guarantee for our DP-IGHT algorithm. Specifically, it achieves the best known utility guarantee (KST12) for sparse linear regression while not requiring any extra support selection procedure. Our approach also provides the first utility guarantee for sparse logistic regression.

Notation. For a d-dimensional vector $\mathbf{x} = [x_1, ..., x_d]^\top$, we use $\|\mathbf{x}\|_2 = (\sum_{i=1}^d |x_i|^2)^{1/2}$ to denote its ℓ_2-norm, and use $\|\mathbf{x}\|_\infty = \max_i |x_i|$ to denote its ℓ_∞-norm. We let $\text{supp}(\mathbf{x})$ be the index set of nonzero entries of $\mathbf{x}$, and $\text{supp}(\mathbf{x}, s)$ be the index set of the top s entries of $\mathbf{x}$ in terms of magnitude. We use $\mathcal{S}^n$ to denote the input space with n examples and $\mathcal{R}$ to denote the output space. Given two sequences $\{a_n\}$ and $\{b_n\}$, if there exists a constant $0 < C < \infty$ such that $a_n \leq Cb_n$, we write $a_n = O(b_n)$, and we use $\widetilde{O}(\cdot)$ to hide the logarithmic factors. We denote the d by d identity matrix by $\mathbf{I}_d$. For simplicity, we use $\ell_i(\cdot)$ to denote $\ell(\cdot; \mathbf{z}_i)$ throughout the paper.

2.2 Related Work

To develop differentially private algorithms, the commonly used methods include output perturbation (CM09), objective perturbation (CM09), and gradient (iterative) perturbation (BST14b). More specifically, output perturbation adds random noise to the output of a non-private algorithm. Objective perturbation perturbs the objective function

of learning algorithms by random noise before learning. And the idea of gradient perturbation is to introduce random noise into the intermediate steps of the learning algorithm. Although these approaches have been extensively studied for empirical risk minimization (CM09; CMS11b; KST12; BST14b; ZZM17b; WYX17b; WGX18; JWE18b) in classical setting, their applications to sparse learning in the high-dimensional regime remain understudied.

There exist several ad hoc approaches (KST12; TS13; JT14; TTZ15) to solving differentially private (sparse) learning in the high-dimensional setting. For example, (JT14) proposed a differentially private algorithm with the dimension independent utility guarantee for empirical risk minimization. Nevertheless, their method only works for specific loss functions and the utility guarantee is sub-optimal in terms of other parameters. The most relevant studies to ours are (KST12; TS13; TTZ15), which studied differentially private sparse linear regression. In detail, (KST12; TS13) proposed to first perform some differentially private model selection algorithms to estimate the support set of sparse model parameter vector, and then run the objective perturbation algorithm to estimate the parameter vector with its support restricted to the estimated subset. While, their method can achieve $O\big(s^{*2}\log(2/\gamma)/(n\epsilon)^2\big)$ utility guarantee, where ϵ is the privacy budget and γ is the probability that the model selection algorithms can successfully select the true support, the model selection algorithms, such as exponential mechanism, may be computational inefficient or even intractable in practice. In addition, the privacy and utility guarantees of their algorithm only holds for the exact optimal solution to the perturbed optimization problem. Later on, (TTZ15) developed a differentially private Frank-Wolfe algorithm, which is based on the gradient perturbation, for sparse learning. They showed that their algorithm can achieve $O\big(\omega(\mathcal{C})^{2/3}\log n/(n\epsilon)^{3/2}\big)$ utility guarantee, where $\omega(\mathcal{C})$ is the Gaussian width of the constraint set $\mathcal{C}$. However, their approach only has a sublinear convergence rate and the Gaussian width $\omega(\mathcal{C})$ can only be estimated for some specific convex set such as ℓ_1-norm ball.

Different from the aforementioned methods for sparse learning, our proposed DP-IGHT algorithm does not require exactly solving the optimization problem or the extra model selection procedure. Therefore, it is able to attain better empirical performances and more preferable in practice. In addition, our algorithm enjoys a linear convergence rate, which is more efficient than previous methods. The detailed comparisons of different algorithms for sparse linear regression are summarized in Table 2.1.

Algorithm	Method	Utility	Convergence	RSC/RSS	Support selection
Frank-Wolfe (TTZ15)	Iterative	$O\Big(\frac{\log(nd)}{(n\epsilon)^{2/3}}\Big)$	Sub-linear	No	No
Two stage (KST12)	Objective	$O\Big(\frac{s^{*2}\log(2/\gamma)}{(n\epsilon)^2}\Big)$	NA	Yes	Yes
DP-IGHT This paper	Iterative	$O\Big(\frac{s^{*2}\log d}{(n\epsilon)^2}\Big)$	Linear	Yes	No

Table 2.1: Comparison of different (ϵ, δ)-DP algorithms for sparse linear regression. We ignore the $\log(1/\delta)$ term in the utility guarantees. Note that γ is the probability that the differentially private model selection algorithms can successfully recover the true support.

2.3 Preliminaries

In this section, we present some definitions that will be used throughout our paper. We first introduce several classes of functions that we considered in our work.

Definition 2.3.1 (G-Lipschitz continuous)**.** A function $f : \mathbb{R}^d \to \mathbb{R}$ is G-Lipschitz continuous, if the following inequality holds for all $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \mathrm{dom} f$

$$|f(\boldsymbol{\theta}_1) - f(\boldsymbol{\theta}_2)| \leq G\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2.$$

Note that for a differentiable function f, G-Lipschitz continuous implies that the gradient norm is bounded, i.e., $\|\nabla f(\boldsymbol{\theta})\|_2 \leq G$ for all $\boldsymbol{\theta} \in \mathrm{dom} f$.

Definition 2.3.2 (Sparse eigenvalue condition)**.** A twice differentiable function $f : \mathbb{R}^d \to \mathbb{R}$ satisfies sparse eigenvalue condition with parameters $\mu > 0$ and $\beta > 0$, if the following holds for the Hessian of f for all $\boldsymbol{\theta} \in \mathrm{dom} f$,

$$\mu = \inf_{\mathbf{v}} \left\{ \mathbf{v}^\top \nabla^2 f(\boldsymbol{\theta}) \mathbf{v} \mid \|\mathbf{v}\|_0 \leq s,\ \|\mathbf{v}\|_2 = 1 \right\},$$
$$\beta = \sup_{\mathbf{v}} \left\{ \mathbf{v}^\top \nabla^2 f(\boldsymbol{\theta}) \mathbf{v} \mid \|\mathbf{v}\|_0 \leq s,\ \|\mathbf{v}\|_2 = 1 \right\}.$$

For sparse learning problems, sparse eigenvalue condition (BRT09) implies the restricted strong convexity and smoothness conditions (NYW09; LW13), which guarantee the objective function behaves like a strongly convex and smooth function over a sparse domain even the function is general convex in its entire domain. In the following discussion, we denote κ by β/μ.

Zero-concentrated differential privacy. Although the notion of (ϵ, δ)-DP, i.e., Definition 2.1.1, is widely used for the analysis of the output and objective perturbation methods, it is not suitable for the gradient perturbation method since it will give loose composition results. We propose to use the notion of zero-concentrated differential privacy (BS16b), which has a sharp composition result and thus is a better choice for gradient perturbation method.

Definition 2.3.3 (Zero-concentrated differential privacy)**.** A randomized mechanism $\mathcal{M} : \mathcal{S}^n \to \mathcal{R}$ satisfies ρ-zero-concentrated differential privacy (ρ-zCDP) if for any two adjacent datasets $S, S' \in \mathcal{S}^n$ differing by one example, it holds that for all $\alpha \in (1, \infty)$

$$D_\alpha(\mathcal{M}(S) || \mathcal{M}(S')) \leq \rho\alpha, \tag{2.3.1}$$

where $D_\alpha(\mathcal{M}(S) || \mathcal{M}(S'))$ is the α-Renyi divergence[1] between two distributions $\mathcal{M}(S)$ and $\mathcal{M}(S')$.

[1]The formal definition can be found in (Ren61).

Note that ρ-zCDP can be converted to (ϵ, δ)-DP through the following lemma, which is established in (BS16b).

Lemma 2.3.4. If a randomized mechanism $\mathcal{M} : \mathcal{S}^n \to \mathcal{R}$ satisfies ρ-zCDP, then it satisfies $(\rho + 2\sqrt{\rho \log(1/\delta)}, \delta)$-differential privacy for any $\delta > 0$.

Next, we introduce the definition of ℓ_2-sensitivity, which is used to control the variance of the Gaussian Mechanism for ensuring ρ-zCDP.

Definition 2.3.5 (ℓ_2-sensitivity (DMN06))**.** For two adjacent datasets $S, S' \in \mathcal{S}^n$ differing by one example, the ℓ_2-sensitivity $\Delta_2(q)$ of a function $q : \mathcal{S}^n \to \mathbb{R}^d$ is defined as $\Delta_2(q) = \sup_{S,S'} \|q(S) - q(S')\|_2$.

Based on ℓ_2-sensitivity, we can use Gaussian mechanism to make our algorithms satisfy ρ-zCDP.

Lemma 2.3.6 (Gaussian mechanism (BS16b))**.** Given a function $q : \mathcal{S}^n \to \mathbb{R}^d$, the Gaussian Mechanism $\mathcal{M}(S) = q(S) + \mathbf{u}$, where $\mathbf{u} \sim N(0, \sigma^2 \mathbf{I})$, satisfies $\Delta_2(q)^2/(2\sigma^2)$-zCDP.

ρ-zCDP has the invariant property of post-processing and the property of composition as follows.

Lemma 2.3.7 ((BS16b))**.** For two randomized mechanisms $\mathcal{M}_1 : \mathcal{S}^n \to \mathbb{R}^d$, $\mathcal{M}_2 : \mathcal{S}^n \times \mathbb{R}^d \to \mathbb{R}^d$. If $\mathcal{M}_1$ satisfies ρ_1-zCDP and $\mathcal{M}_2$ satisfies ρ_2-zCDP, then $\mathcal{M}_2(S, M_1(S))$ satisfies $(\rho_1+\rho_2)$-zCDP.

2.4 Algorithmic Framework

In this section, we present our differentially private iterative gradient hard thresholding (DP-IGHT) algorithm, which is illustrated in Algorithm 1, for solving the sparsity constrained optimization problem (2.1.1).

Algorithm 1 Differentially Private Iterative Gradient Hard Thresholding (DP-IGHT)

Require: loss function L_S, thresholding parameters s, step size η, iteration number T, initial estimator $\boldsymbol{\theta}_0$, privacy budget ρ, Lipschitz constant G

for $t = 1, 2, 3, \ldots, T$ **do**

$\boldsymbol{\theta}_t = \mathcal{H}_s\big(\boldsymbol{\theta}_{t-1} - \eta(\nabla L_S(\boldsymbol{\theta}_{t-1}) + \mathbf{u})\big)$, where $\mathbf{u} \sim N(0, \sigma^2 \mathbf{I}_d)$ with $\sigma^2 = TG^2/(n^2\rho)$

end for

Ensure: $\boldsymbol{\theta}_T$

At the core of our proposed Algorithm 1 is the gradient perturbation procedure at each iteration, which ensures the differential privacy. More specifically, we perturb the gradient with Gaussian noise at each iteration and make use of the composition and post-processing properties of differential privacy to characterize the upper bound on the total privacy loss. Compared with the objective perturbation based approaches (CM09; CMS11b; KST12), our algorithm does not require the optimization problem to be solved exactly in order to achieve the privacy and utility guarantees. In addition, since it is very hard to characterize the sensitivity of the optimization problem with the sparsity constraint (XCM12), we do not pursue the output perturbation based approaches (CMS11b; ZZM17b).

According to Algorithm 1, to enforce the sparsity constraint, we use the iterative gradient hard thresholding (IGHT) algorithm, which has been shown to have a linear rate of convergence (JTK14; YLZ14; CG16). Note that if we set $\sigma^2 = 0$ in Algorithm 1, it will reduce to the original IGHT algorithm. The hard thresholding operator $\mathcal{H}_s(\cdot)$ in Algorithm 1 is defined as follows:

$$[\mathcal{H}_s(\boldsymbol{\theta})]_i = \begin{cases} \theta_i, & \text{if } i \in \text{ supp}(\boldsymbol{\theta}, s) \\ 0, & \text{otherwise}\,. \end{cases} \tag{2.4.1}$$

$\mathcal{H}_s(\boldsymbol{\theta})$ preserves the top s elements in $\boldsymbol{\theta}$ in terms of magnitude and set others to be zero. We will show later that the proposed DP-IGHT algorithm also enjoys a linear convergence rate, and therefore is more efficient than existing methods.

2.5 Main Results

In this section, we first present the main theoretical properties of Algorithm 1 for generic models, and then show its applications to two specific examples: sparse linear regression and sparse logistic regression.

Recall that the goal of sparse learning is to estimate the underlying sparse parameter vector $\boldsymbol{\theta}^*$ of a statistical model. Thus, we impose a high probability upper bound on the gradient of the objective function at $\boldsymbol{\theta}^*$, which is used to characterize the statistical error of different statistical models.

Condition 2.5.1. For a given sample size n and tolerance parameter $\gamma \in (0, 1)$, let $\varepsilon(n, \gamma)$ be the smallest scalar such that with probability at least $1 - \gamma$, we have

$$\|\nabla L_S(\boldsymbol{\theta}^*)\|_\infty \leq \varepsilon(n, \gamma),$$

where $\varepsilon(n, \gamma)$ depends on the sample size n and γ.

Equipped with this condition, we are ready to establish the main theoretical results of Algorithm 1.

2.5.1 Results for Generic Model

We first present the privacy guarantee of Algorithm 1 for solving sparse learning problem (2.1.1) under ρ-zCDP.

Theorem 2.5.2. Suppose each component function ℓ_i of L_S is G-Lipschitz continuous, the output $\boldsymbol{\theta}_T$ of Algorithm 1 satisfies ρ-zCDP after T iterations if $\sigma^2 = TG^2/(n^2\rho)$.

Remark 2.5.3. According to Lemma 6.2.6, we can also derive that the output $\boldsymbol{\theta}_T$ of Algorithm 1 satisfies (ϵ, δ)-DP if $\sigma^2 = TG^2/\big(n\big(\sqrt{\log(1/\delta) + \epsilon} - \sqrt{\log(1/\delta)}\big)\big)^2$. Furthermore, if $\epsilon \leq \log(1/\delta)$, we can get $\sigma^2 \leq 6TG^2 \log(1/\delta)/(n\epsilon)^2$, which matches the bound of the noise

variance for gradient perturbation methods (WYX17b). Note that for the Lipschitz parameter G, we can exactly calculate a tight upper bound for sparse linear regression and sparse logistic regression. However, for general loss functions, one practical approach to choose G is to use gradient clipping (ACG16b).

Next, we provide the utility guarantee of Algorithm 1 for solving sparse learning problem (2.1.1).

Theorem 2.5.4. Suppose the loss function L_S satisfies sparse eigenvalue condition with parameters μ, β, and Condition 2.5.1, and each component function ℓ_i is G-Lipschitz continuous. There exist constants $\{C_i\}_{i=1}^5$ such that if $\sigma^2 = TG^2/(n^2\rho)$, $\eta = C_1/(\beta+\mu)$, and $s \geq C_2\kappa^2 s^*$, then $\boldsymbol{\theta}_T$ converges to $\boldsymbol{\theta}^*$ at a linear rate. In addition, if we choose $T = C_3\kappa \log\big(\rho n^2\mu^2\|\boldsymbol{\theta}^*\|_2^2/(\kappa^2 G^2 s\log d)\big)$, the following holds with probability at least $1-\gamma$

$$\mathbb{E}\|\boldsymbol{\theta}_T - \boldsymbol{\theta}^*\|_2^2 \leq C_4\frac{\kappa^2 s^*}{\mu^2}\varepsilon(n,\gamma)^2 + C_5\frac{\kappa^3 G^2 s^*\log d}{n^2\mu^2\rho}\cdot\log\frac{\rho n\mu\|\boldsymbol{\theta}^*\|_2}{s^*\kappa G}, \tag{2.5.1}$$

where the expectation is taken over the randomness of the Gaussian noises in Algorithm 1.

Remark 2.5.5. The utility bound in (2.5.1) consists of two terms: the first one denotes the statistical error, while the second term corresponds to the error introduced by the Gaussian mechanism. It is worth noting that the error term caused by the Gaussian mechanism depends on $s^*\log d$ instead of d comparing with the previous differentially private learning algorithms (BST14b). According to Lemma 6.2.6, we can also derive the following utility guarantee under (ϵ,δ)-DP

$$O\bigg(\frac{s^*\kappa^2\varepsilon(n,\gamma)^2}{\mu^2} + \frac{\kappa^3 G^2 s^*\log d\log(1/\delta)}{n^2\epsilon^2\mu^2}\bigg),$$

and we defer such result to the supplemental material.

2.5.2 Implications for Specific Examples

In this subsection, we demonstrate the implications of the main theory for Algorithm 1 when it is applied to specific examples. Note that here we directly spell out the utility results under (ϵ, δ)-DP for the ease of comparison.

2.5.2.1 Sparse Linear Regression

The first example we considered is the linear regression problem in the high-dimensional regime $y_i = \langle \mathbf{x}_i, \boldsymbol{\theta}^* \rangle + \xi_i$, where $\mathbf{y} = [y_1, \ldots, y_n] \in \mathbb{R}^n$ denotes the response vector, $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]^\top \in \mathbb{R}^{n \times d}$ is the design matrix, $\boldsymbol{\xi} = [\xi_1, \ldots, \xi_n] \in \mathbb{R}^n$ is a noise vector, and $\boldsymbol{\theta}^* \in \mathbb{R}^d$ with $\|\boldsymbol{\theta}^*\|_0 = s^*$ is the underlying sparse regression coefficient vector that we want to recover. In the high-dimensional regime, we have $n \ll d$. In order to estimate the sparse parameter vector $\boldsymbol{\theta}^*$, according to (2.1.1), we consider the following sparsity constrained optimization problem, which has been studied in many previous works (Zha11; YLZ14; JTK14; CG16)

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^d} F(\boldsymbol{\theta}) := \frac{1}{2n} \|\mathbf{X}\boldsymbol{\theta} - \mathbf{y}\|_2^2 \;\; \text{subject to} \;\; \|\boldsymbol{\theta}\|_0 \leq s, \tag{2.5.2}$$

where we have each component function as $\ell_i(\boldsymbol{\theta}) = \big(\langle \mathbf{x}_i, \boldsymbol{\theta} \rangle - y_i\big)^2/2$. The next corollary provides the privacy and utility guarantees of Algorithm 1 for solving (2.5.2).

Corollary 2.5.6. Suppose each row of the design matrix $\mathbf{x}_i$ is an independent sub-Gaussian random vector with $\|\mathbf{x}_i\|_2 \leq K$, and the noise vector $\boldsymbol{\xi} \sim N(0, \nu^2 \mathbf{I}_n)$. For a given privacy budget $\epsilon > 0$ and a constant $\delta \in (0, 1)$, there exist constants $\{C_i\}_{i=1}^3$ such that if $n \geq C_1 s \log d$, and we choose $\sigma^2 = 2\lambda T K^2 (\sqrt{2s}\|\boldsymbol{\theta}^*\|_2 + \nu \log n)^2 \log(1/\delta)/(n^2\epsilon^2)$, appropriate η, large enough s, then for $T = C_2 \kappa \log\big(n^2\epsilon^2/(sK^2 \log d \log(1/\delta))\big)$, the output $\boldsymbol{\theta}_T$ of Algorithm 1 satisfies (ϵ, δ)-DP. In addition, with probability at least $1 - \exp(-C_3 n)$, we have

$$\mathbb{E}\|\boldsymbol{\theta}_T - \boldsymbol{\theta}^*\|_2^2 \leq C_4 \nu^2 K^2 \frac{s^* \log d}{n} + C_5 K^2 \big(\|\boldsymbol{\theta}^*\|_2^2 + \nu^2\big) \frac{s^{*2} \log d \log(1/\delta)}{n^2 \epsilon^2},$$

where C_4, C_5 are some constants depending on log terms, which are small constants.

Remark 2.5.7. Corollary 2.5.6 implies that $O\big(s^* \log d/n + s^{*2} \log d \log(1/\delta)/(n^2\epsilon^2)\big)$ utility guarantee can be achieved by our algorithm in the setting of (ϵ, δ)-DP. The term $O(s^* \log d/n)$ denotes the statistical error for sparse vector estimation, which matches the minimax lower bound (RWY11). The term $O(s^{*2} \log d \log(1/\delta)/(n^2\epsilon^2))$ corresponds to the error introduced by the Gaussian mechanism, which matches the best known result (KST12).

2.5.2.2 Sparse Logistic Regression

For logistic regression, we assume that each observation y_i is drawn from the following Bernoulli distribution $\mathbb{P}(y_i = 1|\mathbf{x}_i, \boldsymbol{\theta}^*) = \exp\big(\langle \boldsymbol{\theta}^*, \mathbf{x}_i\rangle - \log\big(1+\exp(\langle \boldsymbol{\theta}^*, \mathbf{x}_i\rangle)\big)\big)$, where $\mathbf{x}_i \in \mathbb{R}^d$ is the predictive vector, $\boldsymbol{\theta}^* \in \mathbb{R}^d$ with $\|\boldsymbol{\theta}^*\|_0 = s^*$ is the underlying parameter vector we want to recover. According to (2.1.1), we propose to solve the following sparsity constrained maximum likelihood estimation problem (YLZ14; LAL16; CG16)

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^d} L_S(\boldsymbol{\theta}) := -\frac{1}{n}\sum_{i=1}^{n}\big[y_i\langle \boldsymbol{\theta}, \mathbf{x}_i\rangle - \log\big(1+\exp(\langle \boldsymbol{\theta}, \mathbf{x}_i\rangle)\big)\big] \quad \text{subject to} \quad \|\boldsymbol{\theta}\|_0 \le s, \quad (2.5.3)$$

where we have each component function as $\ell_i(\boldsymbol{\theta}) = \log\big(1+\exp(\langle \boldsymbol{\theta}, \mathbf{x}_i\rangle)\big) - y_i\langle \boldsymbol{\theta}, \mathbf{x}_i\rangle$. We have the following theoretical guarantees for sparse logistic regression.

Corollary 2.5.8. Suppose each row of the design matrix $\mathbf{x}_i$ is independent sub-Gaussian random vector and $\|\mathbf{x}_i\|_2 \le K$. For a given privacy budget $\epsilon > 0$ and a constant $\delta \in (0, 1)$, there exist constants $\{C_i\}_{i=1}^4$ such that if $n \ge C_1 s \log d$, and we choose $\sigma^2 = TK^2 \log(1/\delta)/(n^2\epsilon^2)$, appropriate η, large enough s, then for $T = C_2\kappa \log\big(n^2\epsilon^2/(sK^2 \log d \log(1/\delta))\big)$, the output $\boldsymbol{\theta}_T$ of Algorithm 1 is (ϵ, δ)-DP. In addition, with probability at least $1 - \exp(-C_3 n) - C_4/d$, we have

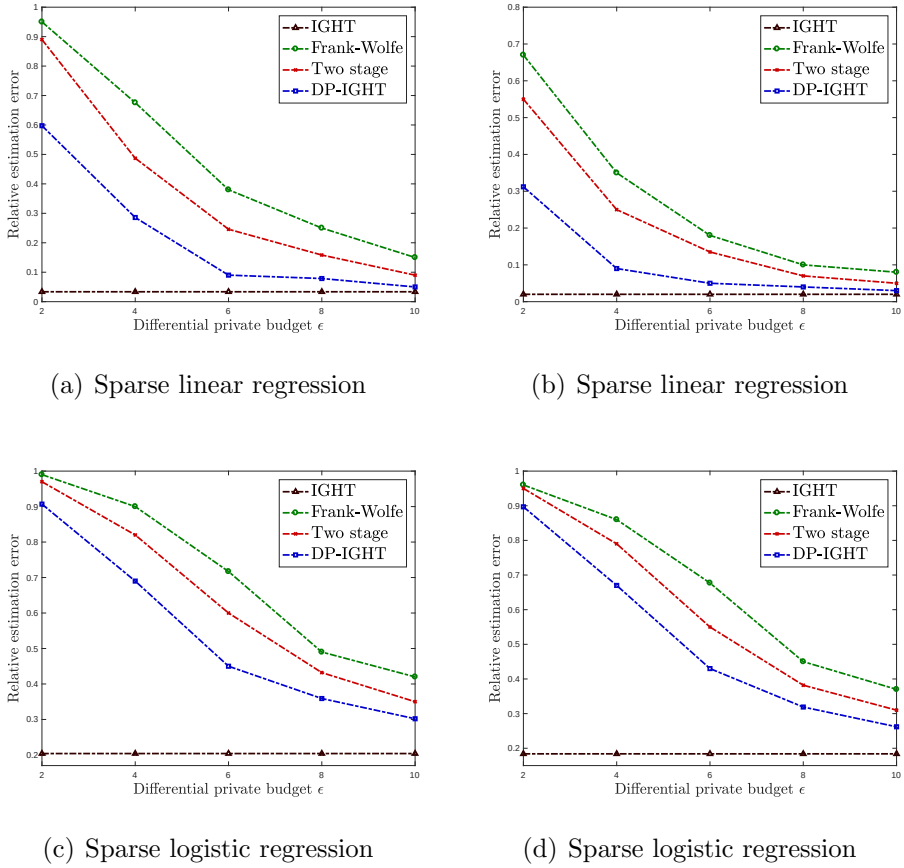
$$\mathbb{E}\|\boldsymbol{\theta}_T - \boldsymbol{\theta}^*\|_2^2 \le C_5 K^2 \frac{s^* \log d}{n} + C_6 K^2 \frac{s^* \log d}{n^2\epsilon^2} \log(1/\delta),$$

where C_5, C_6 are some constants depending on log terms, which are small constants.

Remark 2.5.9. In the setting of (ϵ, δ)-DP, our proposed algorithm can achieve $O(s^* \log d/n + s^* \log d \log(1/\delta)/(n^2\epsilon^2))$ utility guarantee after $T = O(\log\big(n^2\epsilon^2/s\big))$ iterations. In particular,

the term $O(s^* \log d/n)$ corresponds to the statistical error, while the term $O(s^* \log d \log(1/\delta)/(n^2\epsilon^2))$ denotes the error caused by the Gaussian mechanism. To the best of our knowledge, this is the first utility guarantee for sparse logistic regression.

2.6 Numerical Experiments



(a) Sparse linear regression

(b) Sparse linear regression

(c) Sparse logistic regression

(d) Sparse logistic regression

Figure 2.1: Numerical results for sparse linear regression and sparse logistic regression.

In this section, we present experimental results of our proposed algorithm on both synthetic and real datasets. We compare our algorithm with Two stage (KST12) and Frank-Wolfe (TTZ15) methods. Although these two approaches were originally proposed for sparse linear regression, and have no theoretical guarantees for sparse logistic regression, they can

Method	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 6$	$\epsilon = 8$	$\epsilon = 10$
IGHT	0.785	0.785	0.785	0.785	0.785
Frank-Wolfe	1.514 (0.093)	1.320 (0.090)	1.210 (0.084)	1.105 (0.079)	1.094 (0.071)
Two stage	1.286 (0.112)	1.072 (0.101)	1.042 (0.082)	0.997 (0.080)	0.986 (0.075)
DP-IGHT	1.057 (0.107)	0.890 (0.081)	0.854 (0.073)	0.823 (0.070)	0.810 (0.066)

Table 2.2: Comparison of different algorithms for various privacy budgets ϵ in terms of MSE on the test data and its corresponding standard error on E2006-TFIDF.

Method	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 6$	$\epsilon = 8$	$\epsilon = 10$
IGHT	0.0625	0.0625	0.0625	0.0625	0.0625
Frank-Wolfe	0.1271 (0.0043)	0.1034 (0.0037)	0.0938 (0.0034)	0.0852 (0.0036)	0.0807 (0.0031)
Two stage	0.1213 (0.0041)	0.0989 (0.0039)	0.0893 (0.0035)	0.0810 (0.0033)	0.0791 (0.0034)
DP-IGHT	0.1168 (0.0038)	0.0956 (0.0035)	0.0841 (0.0037)	0.0797 (0.0030)	0.0762 (0.0032)

Table 2.3: Comparison of different algorithms for various privacy budgets ϵ in terms of test error and its corresponding standard deviation on RCV1 data.

still be applied to sparse logistic regression and produce reasonable empirical results. Thus we also include them as two baselines for sparse logistic regression. For all the experiments, we choose the variance of the random noise of different methods as suggested by their theoretical guarantees, and select other parameters, such as the step size, iteration number, and thresholding parameter by five-fold cross-validation. Note that we use the non-private iterative gradient hard thresholding method as the non-private baseline. In contrast to DP-IGHT, the non-private IGHT does not add any noise in the gradient step.

2.6.1 Synthetic Data Experiments

We first investigate the performances of different methods on synthetic datasets for sparse linear and logistic regression.

Sparse Linear Regression. For sparse linear regression, the underlying sparse vector $\boldsymbol{\theta}^*$ has s^* nonzero entries that are drawn independently from a uniform distribution over the interval $(-1, 1)$. We generate the design matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$ such that each element of $\mathbf{X}$ follows i.i.d. uniform distribution over the interval $(-2, 2)$, then we scale each row $\mathbf{x}_i$ such that $\|\mathbf{x}_i\|_2 \leq 2s^*$. The observation is generated according to $\mathbf{y} = \mathbf{X}^\top\boldsymbol{\theta}^* + \boldsymbol{\xi}$, where the noise vector $\boldsymbol{\xi} \sim N(0, \nu^2\mathbf{I})$ with $\nu^2 = 0.1$. We consider the following settings: (i) $n = d = 1000, s^* = 10$; (ii) $n = d = 5000, s^* = 30$. We set $\delta = 0.01$ and vary the privacy budget ϵ from 2 to 10. Note that due to the hardness of the problem itself, we choose relatively large privacy budgets compared with the low-dimensional problem to ensure meaningful results. Figure 2.1(a) and 2.1(b) illustrate the relative estimation error $\|\widehat{\boldsymbol{\theta}} - \boldsymbol{\theta}^*\|_2/\|\boldsymbol{\theta}^*\|_2$ versus privacy budget of different methods over 10 trails. We can see that the relative estimation errors of our method are close to the non-private baseline (IGHT), and are better than existing private methods.

Sparse Logistic Regression. For sparse logistic regression, we generate the underlying sparse vector $\boldsymbol{\theta}^*$ and the design matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$ in the same way as sparse linear regression. Each observation y_i is generated from the following logistic distribution

$$y_i = \begin{cases} 1, & \text{with probability } 1/\big(1 + \exp(\langle \mathbf{x}_i, \boldsymbol{\theta}^*\rangle)\big), \\ 0, & \text{with probability } 1 - 1/\big(1 + \exp(\langle \mathbf{x}_i, \boldsymbol{\theta}^*\rangle)\big) . \end{cases}$$

We also consider the following two settings: (i) $n = 1000, d = 1000, s^* = 10$; (ii) $n = 5000, d = 5000, s^* = 30$. In addition, we choose the privacy budget ϵ from 2 to 10, and set $\delta = 0.01$. We demonstrate the relative estimation error $\|\widehat{\boldsymbol{\theta}} - \boldsymbol{\theta}^*\|_2/\|\boldsymbol{\theta}^*\|_2$ versus privacy budget ϵ of different methods in Figure 2.1(c) and 2.1(d). The results show that our method

can output accurate estimators when we have relative large privacy budget. In addition, it consistently outperforms the baseline algorithms under different privacy budget.

2.6.2 Real Data Experiments

In this experiment, we use two real datasets, E2006-TFIDF dataset (KLR09) and RCV1 dataset (LYR04), for the evaluation of sparse linear regression and sparse logistic regression, respectively.

E2006-TFIDF Data. For sparse linear regression problem, we use E2006-TFIDF dataset, which consists of financial risk data from thousands of U.S. companies. In detail, it contains 16087 training examples, 3308 testing examples, and we randomly sample 50000 features for this experiment. In addition, we set $s^* = 2000$, $\delta = 0.01$, $\epsilon \in [2, 10]$. Table 2.2 reports the mean square error (MSE) on the test data of different methods for various privacy budgets over 10 trails. In specific, MSE on the test data is defined as follows: $\|\mathbf{X}_{\text{test}}^\top \widehat{\boldsymbol{\theta}} - \mathbf{y}_{\text{test}}\|_2^2/(2n_{\text{test}})$, where $\{\mathbf{X}_{\text{test}}, \mathbf{y}_{\text{test}}\}$ are the test data, n_{test} is the number of test examples, and $\widehat{\boldsymbol{\theta}}$ is the estimator learned on the training data. The results in Table 2.2 show that the performance of our algorithm is close to the non-private baseline (i.e., IGHT), and is much better than Frank-Wolfe and Two stage.

RCV1 Data. In order to compare different algorithms for sparse logistic regression, we use RCV1 dataset, which is a Reuters Corpus Volume I data set for text categorization research. More specifically, RCV1 is an archive of over 800000 manually categorized newswire stories made available by Reuters, Ltd. for research purposes. It contains 20242 training examples, 677399 testing examples and 47236 features. We use the whole training dataset and a subset of the test dataset, which contains 20000 testing examples for our experiment. In detail, we set $s^* = 500$, $\delta = 0.01$, $\epsilon \in [2, 10]$. We compare all algorithms in terms of their classification error on the test set over 10 replications, which is summarized in Table 2.3. It is obvious

that our algorithm achieves the lowest test error among private algorithms on RCV1 dataset, which demonstrates the superiority of our algorithm.

2.7 Additional Results

In this section, we present the utility guarantee of Algorithm 1 in the setting of (ϵ, δ)-DP.

Corollary 2.7.1. Suppose the loss function satisfies the same conditions as in Theorem 2.5.4. There exist constants $\{C_i\}_{i=1}^5$, if $\sigma^2 = TG^2/(n(\sqrt{\log(1/\delta) + \epsilon} - \sqrt{\log(1/\delta)}))^2$, $\eta = C_1/(\beta + \mu)$, $s \geq C_2\kappa^2 s^*$, then for $T = C_3\kappa \log\big(n^2\mu^2 \log(1/\delta)/(\kappa s G^2 \log d)\big)$, the following holds with probability at least $1 - \gamma$

$$\mathbb{E}\|\boldsymbol{\theta}_T - \boldsymbol{\theta}^*\|_2^2 \leq C_4 \frac{\kappa^2 s^*}{\mu^2}\varepsilon^2 + C_5 \frac{\kappa^3 G^2 s^* \log d}{n^2\mu^2\epsilon^2} \cdot \log \frac{n\mu\|\boldsymbol{\theta}^*\|_2 \log(1/\delta)}{s^*\kappa G}.$$

Remark 2.7.2. The utility bound in (2.7.1) consists of two terms: the first one denotes the statistical error, while the other one corresponds to the error introduced by the Gaussian mechanism.

2.8 Proofs of the Main Results

In this section, we lay out the proofs of our main results.

2.8.1 Proof of Theorem 2.5.2

We first prove the privacy guarantee of Algorithm 1.

Proof. Here we will derive the concentrated differential privacy of Algorithm 1. As mentioned before, the key to provide the privacy guarantee is to characterize the ℓ_2-sensitivity of the proposed mechanism. For our proposed algorithm, we propose to add Gaussian noise at each iteration. Thus we will first to upper bound the ℓ_2-sensitivity of the gradient at

each iteration, and use the composition and post-processing properties of the concentrated differential privacy to derive the privacy guarantee.

Recall that we have the following update rule at $(t+1)$-th iteration $\boldsymbol{\theta}_{t+1} = \mathcal{H}_s\big(\boldsymbol{\theta}_t - \eta\big(\nabla L_S(\boldsymbol{\theta}_t) + \mathbf{u}\big)\big)$. We consider the following query function $\mathbf{q}_t = \nabla L_S(\boldsymbol{\theta}_t)$. For two adjacent datasets S, S' differing by one example indexed by i and i', the ℓ_2-sensitivity $\Delta(\mathbf{q}_t)$ of $\mathbf{q}_t$ can be characterized as follows:

$$\Delta(\mathbf{q}_t) = \frac{1}{n}\|\nabla \ell_i(\boldsymbol{\theta}_t) - \nabla \ell_{i'}(\boldsymbol{\theta}_t)\|_2 \leq \frac{2G}{n},$$

where the last inequality is due to the G-Lipschitz continuous of each component function. For the query function $\mathbf{q}_t$, we consider the following Gaussian mechanism $\mathcal{M}_t' = \mathbf{q}_t + \mathbf{u}$, where $\mathbf{u} \sim N(0, \sigma^2\mathbf{I})$. According to Lemma 2.3.6, the Gaussian mechanism $\mathcal{M}_t'$ will satisfy $G^2/(\sigma^2 n^2)$-zCDP. Therefore, according to Lemma 2.3.7, the mechanism $\mathcal{M}_t = \mathcal{H}_s\big(\boldsymbol{\theta}_t - \eta\mathcal{M}_t'\big)$ will still satisfy $G^2/(\sigma^2 n^2)$-zCDP. Since we will run Algorithm 1 for T iterations, according to the composition property, i.e., Lemma 2.3.7, we can obtain that Algorithm 1 satisfies $TG^2/(\sigma^2 n^2)$-zCDP.

□

2.8.2 Proof of Theorem 2.5.4

In this subsection, we provide the utility guarantee of Algorithm 1.

Proof. In order to provide the utility guarantee of our proposed method, we need to characterize the effect of Gaussian noise we add at each iteration. Recall that, at $(t+1)$-th iteration, we have $\boldsymbol{\theta}_{t+1} = \mathcal{H}_s\big(\boldsymbol{\theta}_t - \eta(\nabla L_S(\boldsymbol{\theta}_t) + \mathbf{u})\big)$, where $\mathbf{u} \sim N(0, \sigma^2\mathbf{I})$. According to Theorem 2.5.2, we have $\sigma^2 = TG^2/(n^2\rho)$, which implies $\mathbb{E}\|\mathbf{u}\|_\infty^2 \leq TG^2 \log d/(n^2\rho)$ by Lemma 2.10.1.

Let $\Omega = \text{ supp}(\boldsymbol{\theta}_t) \cup \text{ supp}(\boldsymbol{\theta}_{t+1}) \cup \text{ supp}(\boldsymbol{\theta}^*)$ and $\widetilde{\boldsymbol{\theta}}_{t+1} = \mathcal{P}_\Omega\big(\boldsymbol{\theta}_t - \eta(\nabla L_S(\boldsymbol{\theta}_t) + \mathbf{u})\big)$, where $\mathcal{P}_\Omega(\boldsymbol{\theta})$ means that we main the elements of $\boldsymbol{\theta}$ in Ω and set others to zero. It implies that

$\boldsymbol{\theta}_{t+1} = \mathcal{H}_s(\widetilde{\boldsymbol{\theta}}_{t+1})$. According to Lemma 3.3 in (LAL16), we have

$$\|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}^*\|_2^2 \leq \left(1 + \frac{2\sqrt{s^*}}{\sqrt{s - s^*}}\right)\|\widetilde{\boldsymbol{\theta}}_{t+1} - \boldsymbol{\theta}^*\|_2^2.$$

Therefore, if we can establish the convergence as $\|\widetilde{\boldsymbol{\theta}}_{t+1} - \boldsymbol{\theta}^*\|_2^2 \leq \rho\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|_2^2$, we can obtain that $\|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}^*\|_2^2 \leq \alpha_1\rho\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|_2^2$, where $\alpha_1 = 1 + 2\sqrt{s^*}/\sqrt{s - s^*}$. For $\widetilde{\boldsymbol{\theta}}_{t+1}$, we have

$$\begin{aligned}\|\widetilde{\boldsymbol{\theta}}_{t+1} - \boldsymbol{\theta}^*\|_2^2 &= \big\|\mathcal{P}_\Omega\big(\boldsymbol{\theta}_t - \eta(\nabla L_S(\boldsymbol{\theta}_t) + \mathbf{u})\big) - \boldsymbol{\theta}^*\big\|_2^2 \\ &= \Big\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^* - \eta\mathcal{P}_\Omega\big(\nabla L_S(\boldsymbol{\theta}^*) + \big(\mathbf{H}(\gamma)\big)_{*\Omega}(\boldsymbol{\theta}_t - \boldsymbol{\theta}^*) + \mathbf{u}\big)\Big\|_2^2 \end{aligned} \tag{2.8.1}$$

where $\mathbf{H}(\gamma) = \int_0^1 \nabla^2 L_S(\boldsymbol{\theta}^* + \gamma(\boldsymbol{\theta} - \boldsymbol{\theta}^*))d\gamma$, the last equation is due to the fundamental theorem of calculus, and $\mathbf{H}(\gamma)_{*\Omega}$ denotes that we restrict columns of $\mathbf{H}(\gamma)$ to the support Ω. Furthermore, taking expectation over $\mathbf{u}$ conditioned on $\boldsymbol{\theta}_t$, we have

$$\begin{aligned}\mathbb{E}\|\widetilde{\boldsymbol{\theta}}_{t+1} - \boldsymbol{\theta}^*\|_2^2 &= \mathbb{E}\big\|\mathbf{A}(\boldsymbol{\theta}_t - \boldsymbol{\theta}^*) - \eta\mathcal{P}_\Omega\big(\nabla L_S(\boldsymbol{\theta}^*) + \mathbf{u}\big)\big\|_2^2 \\ &\leq \mathbb{E}\big(\|\mathbf{A}\|_2^2 \cdot \|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|_2^2\big) + \eta^2\mathbb{E}\big\|\mathcal{P}_\Omega\big(\nabla L_S(\boldsymbol{\theta}^*) + \mathbf{u}\big)\big\|_2^2 \\ &\quad - 2\eta\mathbb{E}\langle\mathbf{A}(\boldsymbol{\theta}_t - \boldsymbol{\theta}^*), \mathcal{P}_\Omega\big(\nabla L_S(\boldsymbol{\theta}^*)\big)\rangle - 2\eta\mathbb{E}\langle\mathbf{A}(\boldsymbol{\theta}_t - \boldsymbol{\theta}^*), \mathcal{P}_\Omega(\mathbf{u})\rangle,\end{aligned}$$

where the first equality is due to the definition of $\mathcal{P}_\Omega$ and $\mathbf{A} = \mathbf{I} - \eta\big(\mathbf{H}(\gamma)\big)_{\Omega\Omega}$. Thus by Young's inequality, we can obtain

$$-2\eta\mathbb{E}\langle\mathbf{A}(\boldsymbol{\theta}_t - \boldsymbol{\theta}^*), \mathcal{P}_\Omega(\mathbf{u})\rangle \leq \frac{2\eta\mu}{7}\mathbb{E}\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|_2^2 + \frac{14\eta}{\mu}\mathbb{E}\big(\|\mathbf{A}\|_2^2 \cdot \big\|\mathcal{P}_\Omega(\mathbf{u})\|_2^2\big).$$

By the same argument, we can further get

$$-2\eta\mathbb{E}\langle\mathbf{A}(\boldsymbol{\theta}_t - \boldsymbol{\theta}^*), \mathcal{P}_\Omega\big(\nabla L_S(\boldsymbol{\theta}^*)\big)\rangle \leq \frac{2\eta\mu}{7}\mathbb{E}\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|_2^2 + \frac{14\eta}{\mu}\mathbb{E}\big(\|\mathbf{A}\|_2^2 \cdot \big\|\mathcal{P}_\Omega\big(\nabla L_S(\boldsymbol{\theta}^*)\big)\big\|_2^2\big).$$

Plugging these two results into (2.8.1), we can obtain

$$\begin{aligned}\mathbb{E}\|\widetilde{\boldsymbol{\theta}}_{t+1} - \boldsymbol{\theta}^*\|_2^2 &\leq \mathbb{E}\big(\|\mathbf{A}\|_2^2 \cdot \|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|_2^2\big) + \eta^2\mathbb{E}\big\|\mathcal{P}_\Omega\big(\nabla L_S(\boldsymbol{\theta}^*) + \mathbf{u}\big)\big\|_2^2 \\ &\quad + \frac{4\eta\mu}{7}\mathbb{E}\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|_2^2 + \frac{14\eta}{\mu}\mathbb{E}\big(\|\mathbf{A}\|_2^2 \cdot \big\|\mathcal{P}_\Omega\big(\nabla L_S(\boldsymbol{\theta}^*)\big)\big\|_2^2\big) + \frac{14\eta}{\mu}\mathbb{E}\big(\|\mathbf{A}\|_2^2 \cdot \big\|\mathcal{P}_\Omega(\mathbf{u})\big\|_2^2\big) \\ &\leq \Big(1 - \frac{3\eta\mu}{7}\Big)\mathbb{E}\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|_2^2 + \Big(\frac{14\eta}{\mu} - 14\eta^2\Big)\Big(\mathbb{E}\big\|\mathcal{P}_\Omega\big(\nabla L_S(\boldsymbol{\theta}^*)\big)\big\|_2^2 + \mathbb{E}\big\|\mathcal{P}_\Omega(\mathbf{u})\big\|_2^2\Big),\end{aligned}$$

where the last inequality is due to the sparse eigenvalue condition of L. As we discussed before, we can get

$$\mathbb{E}\|\boldsymbol{\theta}_{t+1}-\boldsymbol{\theta}^*\|_2^2 \leq \alpha_1\Big(1-\frac{3\eta\mu}{7}\Big)\mathbb{E}\|\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\|_2^2+\alpha_1(2s+s^*)\Big[\Big(\frac{14\eta}{\mu}-14\eta^2\Big)\big(\|\nabla L_S(\boldsymbol{\theta}^*)\|_\infty^2+\mathbb{E}\|\mathbf{u}\|_\infty^2\big)\Big],$$

Since we have $\eta = 2/(\beta+\mu)$, $s \geq (4\kappa^2+1)s^*$, where $\kappa = \beta/\mu$, we can get

$$\begin{aligned}\mathbb{E}\|\boldsymbol{\theta}_{t+1}-\boldsymbol{\theta}^*\|_2^2 &\leq \rho\mathbb{E}\|\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\|_2^2+C_1\frac{\kappa^2 s^*(\beta-\mu)}{(\beta+\mu)^2\mu}\big(\|\nabla L_S(\boldsymbol{\theta}^*)\|_\infty^2+\mathbb{E}\|\mathbf{u}\|_\infty^2\big)\\ &\leq \rho\mathbb{E}\|\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\|_2^2+C_2\frac{\kappa^2 s^*(\beta-\mu)}{(\beta+\mu)^2\mu}(\varepsilon^2+TG^2\log d/(n^2\rho)), \qquad (2.8.2)\end{aligned}$$

where the last inequality is due to Condition 2.5.1 and Lemma 2.10.1, and $\rho = 1-1/(7\kappa)$, C_1, C_2 are absolute constants. Thus taking sum of (2.8.2) over $t = 0, 1, \ldots, T-1$ and taking expectation with respect to all t's, we can get

$$\begin{aligned}\mathbb{E}\|\boldsymbol{\theta}_T-\boldsymbol{\theta}^*\|_2^2 &\leq \rho^T\|\boldsymbol{\theta}^*\|_2^2+C_2\frac{\kappa^2 s^*(\beta-\mu)}{(\beta+\mu)^2\mu(1-\rho)}(\varepsilon^2+TG^2\log d/(n^2\rho))\\ &= \rho^T\|\boldsymbol{\theta}^*\|_2^2+C_3\frac{\kappa^3 s^*(\beta-\mu)}{(\beta+\mu)^2\mu}(\varepsilon^2+TG^2\log d/(n^2\rho))\\ &\leq \rho^T\|\boldsymbol{\theta}^*\|_2^2+C_3\frac{\kappa^2 s^*}{\mu^2}(\varepsilon^2+TG^2\log d/(n^2\rho)), \qquad (2.8.3)\end{aligned}$$

which implies that

$$\mathbb{E}\|\boldsymbol{\theta}_T-\boldsymbol{\theta}^*\|_2^2 \leq \rho^T\|\boldsymbol{\theta}^*\|_2^2+C_3\frac{\kappa^2 s^*}{\mu^2}\varepsilon^2+C_5\frac{s^*T\kappa^2G^2}{n^2\mu^2\rho}\log d,$$

where C_4, C_5 are absolute constants. Thus if we take T such that $\rho^T\|\boldsymbol{\theta}^*\|_2^2 \leq C_5 s^* k^2 G^2\log d/(n^2\mu^2\rho)$, i.e.,

$$T = C_6\kappa\log\frac{\rho n^2\mu^2\|\boldsymbol{\theta}^*\|_2^2}{s^*\kappa^2G^2\log d}, \qquad (2.8.4)$$

where C_6 is an absolute constant, we have the following inequality

$$\mathbb{E}\|\boldsymbol{\theta}_T-\boldsymbol{\theta}^*\|_2^2 \leq C_3\frac{\kappa^2 s^*}{\mu^2}\varepsilon^2+C_7\frac{s^*\kappa^3G^2}{n^2\mu^2\rho}\log d\cdot\log\frac{\rho n\mu\|\boldsymbol{\theta}^*\|_2}{s^*\kappa G}.$$

where C_7 is an absolute constant. □

2.8.3 Proof of Corollary 2.7.1

Proof. According to Lemma 6.2.6, we have that if Algorithm 1 satisfies ρ-zCDP, then it satisfies $(\rho + 2\sqrt{\rho \log(1/\delta)}, \delta)$-DP for any $\delta > 0$. This implies if we want Algorithm 1 satisfies (ϵ, δ)-DP, we only need $\epsilon = \rho + 2\sqrt{\rho \log(1/\delta)}$, which gives us $\rho = (\sqrt{\log(1/\delta) + \epsilon} - \sqrt{\log(1/\delta)})^2$. According to Theorem 2.5.4, we have $\sigma^2 = TG^2/(n^2\rho)$. Therefore, plugging the above relationship between ϵ and ρ into Theorem 2.5.4, we can obtain

$$\sigma^2 = \frac{TG^2}{n^2\big(\sqrt{\log(1/\delta) + \epsilon} - \sqrt{\log(1/\delta)}\big)^2}, \quad T = C_1 \kappa \log\left(\frac{\log(1/\delta) n^2 \mu^2 \|\boldsymbol{\theta}^*\|_2^2}{\kappa^2 G^2 s^* \log d}\right),$$

and we have the following utility guarantee in the setting of (ϵ, δ)-DP

$$\mathbb{E}\|\boldsymbol{\theta}_T - \boldsymbol{\theta}^*\|_2^2 \leq C_2 \frac{\kappa^2 s^*}{\mu^2}\varepsilon^2 + C_3 \frac{\kappa^3 G^2 s^* \log d \log(1/\delta)}{n^2 \mu^2 \epsilon^2} \cdot \log \frac{n\mu \|\boldsymbol{\theta}^*\|_2 \log(1/\delta)}{s^* \kappa G},$$

where C_1, C_2, C_3 are absolute constants. □

2.9 Proof of Specific Examples

In this section, we prove the results for different examples. To make use of the general results, we only need to verify the required conditions for specific examples. To this end, we need the following lemmas, which has been previously proved for many common examples of sub-Gaussian random design (RWY11).

Lemma 2.9.1. Suppose each row of the design matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ are independent isotropic sub-Gaussian random vector with sub-Gaussian parameter α, there exist some constants $\{C_i\}_{i=1}^2$ such that for all $\mathbf{v} \in \mathbb{R}^d$ with at most s nonzero entries, if $n \geq C_1 s\alpha^2 \log d$, with probability at least $1 - \exp(-C_2 n)$, we have

$$\frac{4}{5}\|\mathbf{v}\|_2^2 \leq \frac{\|\mathbf{X}\mathbf{v}\|_2^2}{n} \leq \frac{6}{5}\|\mathbf{v}\|_2^2.$$

The second one, which has been proved in (LW13), provides the statistical error of sparse learning problems.

Lemma 2.9.2. For a Gaussian random vector $\boldsymbol{\xi} \in \mathbb{R}^n$ with zero mean and variance $\nu^2 \mathbf{I}_n$, if each row of $\mathbf{X} \in \mathbb{R}^{n \times d}$ are independent sub-Gaussian random vector with sub-Gaussian parameter α, we have with probability at least $1 - \exp(-C_3 n)$

$$\left\| \frac{1}{n} \mathbf{X}^\top \boldsymbol{\xi} \right\|_\infty \leq C_4 \nu \alpha \sqrt{\frac{\log d}{n}},$$

where C_3, C_4 are absolute constants.

2.9.1 Proof of Corollary 2.5.6

Now, we are ready to prove the results for sparse linear regression.

Proof. According to the objective function in (2.5.2), we have the following close form of gradient and Hessian for sparse linear regression

$$\nabla L_S(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}_i^\top \boldsymbol{\theta} - y_i) \mathbf{x}_i, \quad \nabla^2 L_S(\boldsymbol{\theta}) = \frac{\mathbf{X}^\top \mathbf{X}}{n},$$

where $\mathbf{x}_i$ is the i-th row of the design matrix $\mathbf{X}$. According to Lemma 2.9.1, for all $\mathbf{v} \in \mathbb{R}^d$ with at most $C_1 s$ nonzero entries, as long as $n \geq C_2 s K^2 \log d$, we have with probability at least $1 - \exp(-C_3 n)$,

$$\mu \|\mathbf{v}\|_2^2 \leq \mathbf{v}^\top \nabla^2 L_S(\boldsymbol{\theta}) \mathbf{v} \leq \beta \|\mathbf{v}\|_2^2,$$

where $\mu = 4/5, \beta = 6/5$, and $\{C_i\}_{i=1}^3$ are absolute constants. This implies sparse eigenvalue condition of L with parameters μ and β as defined above. In addition, we have $\|\nabla \ell_i(\boldsymbol{\theta})\|_2 \leq |\mathbf{x}_i^\top \boldsymbol{\theta} - y_i| \cdot \|\mathbf{x}_i\|_2 \leq \big(\sqrt{2s} \|\boldsymbol{\theta}^*\|_2 + \nu \log n\big) K$ where the inequality is due to the fact that $\boldsymbol{\theta} = \boldsymbol{\theta}_t$ with $|\operatorname{supp}(\boldsymbol{\theta}_t)| \leq s$, and the contraction property $\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|_2 \leq \|\boldsymbol{\theta}^*\|_2$. Thus we have G-Lipschitz continuous condition holds for sparse linear regression with parameter $G = \big(\sqrt{2s} \|\boldsymbol{\theta}^*\|_2 + \nu \log n\big)$. Furthermore, we have $\nabla L_S(\boldsymbol{\theta}^*) = \mathbf{X}^\top \boldsymbol{\xi} / n$. According to Lemma 2.9.2, we have $\|\nabla L_S(\boldsymbol{\theta}^*)\|_\infty \leq C_5 \nu K \sqrt{\log d / n}$ holds with probability at least $1 - \exp(-C_6 n)$, where C_5, C_6 are absolute constants. Thus we have Condition 2.5.1 holds for sparse linear

regression. Thus, plugging all the parameters of sparse linear regression in Corollary 2.7.1, we can get

$$\mathbb{E}\|\boldsymbol{\theta}_T - \boldsymbol{\theta}^*\|_2^2 \leq C_7\nu^2K^2\frac{s^*\log d}{n} + C_8K^2(\|\boldsymbol{\theta}^*\|_2^2 + \nu^2)\frac{s^{*2}\log d\log(1/\delta)}{n^2\epsilon^2},$$

holds with probability at least $1 - \exp(-C_5 n) - \exp(-C_6 n)$. □

2.9.2 Proof of Corollary 2.5.8

Now, we will prove the results for sparse logistic regression.

Proof. According to the loss function in (2.5.3), we can obtain

$$\nabla L_S(\boldsymbol{\theta}) = -\frac{1}{n}\sum_{i=1}^{n}\big(y_i - \psi(\boldsymbol{\theta}^\top\mathbf{x}_i)\big)\mathbf{x}_i, \quad \nabla^2 L_S(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n}\psi'(\boldsymbol{\theta}^\top\mathbf{x}_i)\mathbf{x}_i\mathbf{x}_i^\top,$$

where $\psi(x) = \exp(x)/(1+\exp(x))$ and $\psi'(x) = \exp(x)/(1+\exp(x))^2$. In addition, for all $\mathbf{v} \in \mathbb{R}^d$, we have

$$\mathbf{v}^\top\nabla^2 L_S(\boldsymbol{\theta})\mathbf{v} = \frac{1}{n}\sum_{i=1}^{n}\psi'(\boldsymbol{\theta}^\top\mathbf{x}_i)\mathbf{v}^\top\mathbf{x}_i\mathbf{x}_i^\top\mathbf{v}.$$

Since we have $\boldsymbol{\theta}^\top\mathbf{x}_i$ is bounded, we have $\psi'(x)$ is upper and lower bounded by some constants C_1, C_2. Therefore, according to Lemma 2.9.1, for all $\mathbf{v} \in \mathbb{R}^d$ with at most $C_2 s$ nonzero entries, as long as $n \geq C_3 sK^2\log d$, we have with probability at least $1 - \exp(-C_4 n)$,

$$\mu\|\mathbf{v}\|_2^2 \leq \mathbf{v}^\top\nabla^2 L_S(\boldsymbol{\theta})\mathbf{v} \leq \beta\|\mathbf{v}\|_2^2,$$

where $\mu = 4/5C_1, \beta = 6/5C_2$, and $\{C_i\}_{i=1}^4$ are absolute constants, which implies sparse eigenvalue condition. Furthermore, we have

$$\|\nabla\ell_i(\boldsymbol{\theta})\|_2 = \big\|\big(y_i - \psi(\boldsymbol{\theta}^\top\mathbf{x}_i)\big)\mathbf{x}_i\big\|_2 \leq \|\mathbf{x}_i\|_2 \leq K,$$

where the first inequality is due the the fact that $y_i \in \{0,1\}$ and $\psi(x) \in (0,1)$. Thus we have G-Lipschitz continuous condition holds for sparse logistic regression with parameter $G = K$.

In addition, we have $\nabla L_S(\boldsymbol{\theta}^*) = \frac{1}{n}\sum_{i=1}^n b_i \mathbf{x}_i$, where $b_i = y_i - \psi(\boldsymbol{\theta}^{*\top}\mathbf{x}_i)$. Therefore, according to Corollary 2 in (LW13), we have $\|\nabla L_S(\boldsymbol{\theta}^*)\|_\infty \leq C_5 K\sqrt{\log d/n}$ holds with probability at least $1 - C_6/d$, where C_5, C_6 are absolute constants. Therefore, plugging these parameters into Corollary 2.7.1, we can obtain

$$\mathbb{E}\|\boldsymbol{\theta}_T - \boldsymbol{\theta}^*\|_2^2 \leq C_7 K^2 \frac{s^* \log d}{n} + C_8 K^2 \frac{s^* \log d}{n^2\epsilon^2}\log(1/\delta),$$

where C_7, C_8 are absolute constants. □

2.10 Auxiliary Lemmas

Lemma 2.10.1. For a random vector $\mathbf{u} \sim N(0, \sigma^2\mathbf{I}_d)$, we have $\mathbb{E}\|\mathbf{u}\|_\infty^2 \leq C\sigma^2 \log d$, where C is a universal constant.

Proof. Since $\mathbf{u} \sim N(0, \sigma^2\mathbf{I}_d)$, we have coordinates u_i are i.i.d. $N(0, \sigma^2)$. Therefore, let $X_i = |u_i|$, we have X_i is a folded normal distribution. In addition, we have $\mathbb{E}\|\mathbf{u}\|_\infty^2 = \mathbb{E}(\max_i |u_i|)^2$. Let $Z = \max_i X_i$, we want to bound $\mathbb{E}Z^2$. For any $t > 0$, we have

$$\exp\{t^2\mathbb{E}Z^2\} \leq \mathbb{E}\exp\{t^2 Z^2\} \leq \mathbb{E}\max_i \exp\{t^2 X_i^2\} \leq 2\sum_{i=1}^d \mathbb{E}\exp tX_i \leq 4d\exp\Big\{\frac{t^2\sigma^2}{2}\Big\},$$

where the first inequality is due to the Jensen's inequality, the second inequality is due to the fact that $(\max_i X_i)^2 \leq \max_i X_i^2$, and the last one comes from the moment generating function of the folded normal distribution. Thus we have

$$\mathbb{E}Z^2 \leq \frac{\log 4d}{t^2} + \frac{\sigma^2}{2}.$$

Plugging $t = 1/\sigma$ into the above inequality, we can get

$$\mathbb{E}Z^2 \leq 2\sigma^2 \log 4d.$$

□

2.11 Conclusions

In this paper, we proposed a privacy preserving iterative gradient hard thresholding algorithm for sparse learning. We establish a linear convergence rate and strong utility guarantee of our algorithm. Experiments on both synthetic and real world data demonstrate the superiority of our algorithm.

CHAPTER 3

A Knowledge Transfer Framework for Differentially Private Sparse Learning

3.1 Introduction

In the Big Data era, sensitive data such as genomic data and purchase history data, are ubiquitous, which necessitates learning algorithms that can protect the privacy of each individual data record. A rigorous and standard notion for privacy guarantees is differential privacy (DMN06). By adding random noise to the model parameters (output perturbation), some intermediate steps of the learning algorithm (gradient perturbation), or the objective function of learning algorithms (objective perturbation), differentially private algorithms ensure that the trained models can learn the statistical information of the population without leaking any information about the individuals. In the last decade, a surge of differentially private learning algorithms (CM09; CMS11b; KST12; BST14b; TTZ15; ZZM17b; WYX17b; WGX18; JWE18b; WJE19) for empirical risk minimization have been developed. However, most of these studies only consider the classical setting, where the problem dimension is fixed. In the modern high-dimensional setting where the problem dimension can increase with the number of observations, all these empirical risk minimization algorithms fail. A common and effective approach to address these issues is to assume the model has a certain structure such as sparse structure or low-rank structure. In this prospectus, we consider high-dimensional models with sparse structure. Given a dataset $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$ are the input vector and response of the i-th example, our goal is to

estimate the underlying sparse parameter vector $\boldsymbol{\theta}^* \in \mathbb{R}^d$, which has s^* nonzero entries, by solving the following ℓ_2-norm regularized optimization problem with the sparsity constraint

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^d} \bar{L}_S(\boldsymbol{\theta}) := L_S(\boldsymbol{\theta}) + \lambda \|\boldsymbol{\theta}\|_2^2/2 \quad \text{subject to} \quad \|\boldsymbol{\theta}\|_0 \leq s, \tag{3.1.1}$$

where $L_S(\boldsymbol{\theta}) := n^{-1} \sum_{i=1}^n \ell(\boldsymbol{\theta}; \mathbf{x}_i, y_i)$ is the empirical loss on the training data, $\ell(\boldsymbol{\theta}; \mathbf{x}_i, y_i)$ is the loss function defined on the training example $(\mathbf{x}_i, y_i)$, $\lambda \geq 0$ is a regularization parameter, $\|\boldsymbol{\theta}\|_0$ counts the number of nonzero entries in $\boldsymbol{\theta}$, and s controls the sparsity of $\boldsymbol{\theta}$. The reason we add an extra ℓ_2 regularizer to (3.1.1) is to ensure the strong convexity of the objective function without making any assumption on the data.

In order to achieve differential privacy for sparse learning, a line of research (KST12; TS13; JT14; TTZ15; WG19a) studied differentially private learning problems in the high-dimensional setting, where the problem dimension can be larger than the number of observations. For example, (JT14) provided a differentially private algorithm with the dimension independent utility guarantee. However, their approach only considers the case when the underlying parameter lies in a simplex. For sparse linear regression, (KST12; TS13) proposed a two-stage approach to ensure differentially privacy. In detail, they first estimate the support set of the sparse model parameter vector using some differentially private model selection algorithm, and then estimate the parameter vector with its support restricted to the estimated subset using the objective perturbation approach (CM09). Nevertheless, the support selection algorithm, like exponential mechanism, is computational inefficient or even intractable in practice. (TTZ15) proposed a differentially private algorithm for sparse linear regression by combining the Frank-Wolfe method (FW56) and the exponential mechanism. Although their utility guarantee is worse than (KST12; WG19a), it does not depend on the restricted strong convexity (RSC) and smoothness (RSS) conditions (NYW09). Recently, (WG19a) developed a differentially private iterative gradient hard thresholding (IGHT) (JTK14; YLZ14) based framework for sparse learning problems by injecting Gaussian noise into the intermediate gradients. However, all the aforementioned methods either have unsatisfactory utility guarantees or are computationally inefficient. For example, the utility guarantees provided

by (KST12; TS13; WG19a) depend on the ℓ_2-norm bound of the input vector, which can be in the order of $O(\sqrt{d})$ and grows as d increases in the worse case. While the utility guarantee of the algorithm proposed by (TTZ15) only depends on the ℓ_∞-norm bound of the input vector, it has a worse utility guarantee, and its convergence rate is sub-linear.

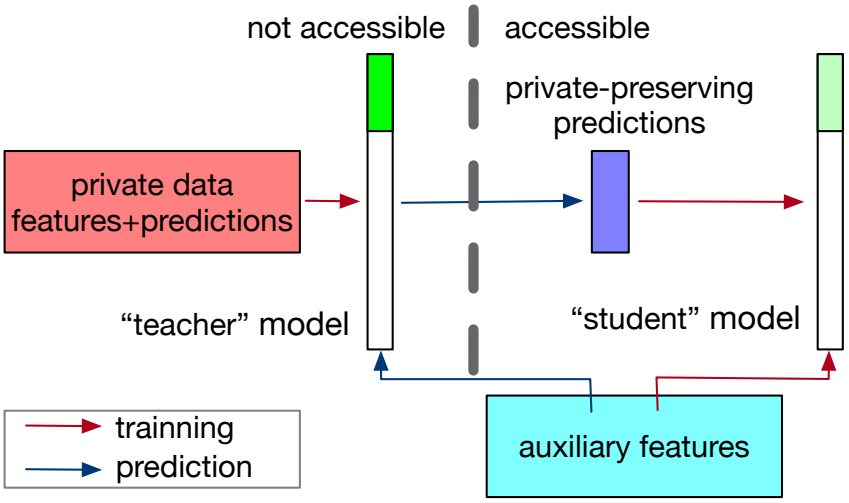


Figure 3.1: Illustration of the proposed teacher-student framework.

Therefore, a natural question is whether we can achieve the best of both worlds: a strong utility guarantee and high computational efficiency. To this end, we propose to make use of the idea of knowledge distillation (BCN06; HVD15), which is a knowledge transfer technique originally introduced as a mean of model compression. The original motivation of using knowledge distillation is to use a large and complex "teacher" model to train a small "student" model, while maintaining its accuracy. For the differentially private sparse learning problem, similar idea can be applied here: we can use a non-private "teacher" model to train a differentially private "student" model, while preserving the sparse information of the "teacher" model. We notice that several knowledge transfer approaches have been recently investigated in the differentially private classification problem (HCB16; PAE16; BTT18; YJS19). Nevertheless, the application of knowledge distillation to the generic differentially private high-dimensional sparse learning problem is new and has never been studied before.

In this prospectus, we propose a knowledge transfer framework for solving the high-dimensional sparse learning problem on a private dataset, which is illustrated in Figure 3.1. Our proposed algorithm is not only very efficient but also has improved utility guarantees compared with the state-of-the-art methods. More specifically, we first train a non-private "teacher" model using IGHT from the private dataset. Based on this "teacher" model, we then construct a privacy-preserving dataset using some auxiliary inputs, which are drawn from some given distributions or public datasets. Finally, by training a "student" model using IGHT again based on the newly generated dataset, we can obtain a differentially private sparse estimator. Table 3.1 summarizes the detailed comparisons of different methods for sparse linear regression, and we summarize the contributions of our work as follows

- Our proposed differentially private framework can be applied to any smooth loss function, which covers a broad family of sparse learning problems. In particular, we showcase the application of our framework to sparse linear regression and sparse logistic regression.

- We prove a better utility guarantee and establish a liner convergence rate for our proposed method. For example, for sparse linear regression, our method achieves $O\big(K^2 s^{*2}\sqrt{\log d}/(n\epsilon)\big)$ utility guarantee, where K is the ℓ_∞-norm bound of the input vectors, and ϵ is the privacy budget. Compared with the best known utility bound $O\big(\widetilde{K}^2 s^{*2}\log d/(n^2\epsilon^2)\big)$ (KST12; WG19a) ($\widetilde{K}$ is the ℓ_2-norm bound of the input vectors), our utility guarantee is better than it by a factor of $O\big(\widetilde{K}^2\sqrt{\log d}/(K^2 n\epsilon)\big)$. Considering that $\widetilde{K}$ can be $\sqrt{d}$ times larger than K, the improvement factor can be as large as $O\big(d\sqrt{\log d}/(n\epsilon)\big)$. Similar improvement is achieved for sparse logistic regression.

- With the extra sparse eigenvalue condition (BRT09) on the private data, our method can achieve $O\big(K^2 s^{*3}\log d/(n^2\epsilon^2)\big)$ utility guarantee for sparse linear regression. It is better than the best known result (KST12; WG19a) $O\big(\widetilde{K}^2 s^{*2}\log d/(n^2\epsilon^2)\big)$ by a factor of $O\big(\widetilde{K}^2/(K^2 s^*)\big)$, which can be as large as $O\big(d/s^*\big)$. Similar improvement is also

Table 3.1: Comparison of different algorithms for sparse linear regression in the setting of (ϵ, δ)-DP. We report the utility bound achieved by the privacy-preserving mechanisms, and ignore the $\log(1/\delta)$ term. Note that $n\epsilon \gg 1$, $\mathbf{x}_i$ denotes the i-th input vector, and υ is the probability that the support selection procedure can successfully recover the true support.

Algorithm	Data Assumption	Utility	Convergence Rate	Utility Assumption
Frank-Wolfe (TTZ15)	$\max_{i\in[n]} \|\mathbf{x}_i\|_\infty \leq 1$	$O\Big(\frac{\log(nd)}{(n\epsilon)^{2/3}}\Big)$	Sub-linear	No
Two Stage (KST12)	$\max_{i\in[n]} \|\mathbf{x}_i\|_2 \leq \widetilde{K}$	$O\Big(\frac{\widetilde{K}^2 s^{*2}\log(2/\upsilon)}{(n\epsilon)^2}\Big)$	NA	RSC/RSS
DP-IGHT (WG19a)	$\max_{i\in[n]} \|\mathbf{x}_i\|_2 \leq \widetilde{K}$	$O\Big(\frac{\widetilde{K}^2 s^{*2}\log d}{(n\epsilon)^2}\Big)$	Linear	RSC/RSS
DPSL-KT $\lambda > 0$	$\max_{i\in[n]} \|\mathbf{x}_i\|_\infty \leq K$	$O\Big(\frac{K^2 s^{*2}\sqrt{\log d}}{n\epsilon}\Big)$	Linear	No
DPSL-KT $\lambda = 0$	$\max_{i\in[n]} \|\mathbf{x}_i\|_\infty \leq K$ RSC/RSS	$O\Big(\frac{K^2 s^{*3}\log d}{(n\epsilon)^2}\Big)$	Linear	RSC/RSS

achieved for sparse logistic regression.

Notation. For a d-dimensional vector $\mathbf{x} = [x_1, ..., x_d]^\top$, we use $\|\mathbf{x}\|_2 = (\sum_{i=1}^d |x_i|^2)^{1/2}$ to denote its ℓ_2-norm, and use $\|\mathbf{x}\|_\infty = \max_i |x_i|$ to denote its ℓ_∞-norm. We let $\mathrm{supp}(\mathbf{x})$ be the index set of nonzero entries of $\mathbf{x}$, and $\mathrm{supp}(\mathbf{x}, s)$ be the index set of the top s entries of $\mathbf{x}$ in terms of magnitude. We use $\mathcal{S}^n$ to denote the input space with n examples and $\mathcal{R}, \mathcal{R}'$ to denote the output space. Given two sequences $\{a_n\}, \{b_n\}$, if there exists a constant $0 < C < \infty$ such that $a_n \leq Cb_n$, we write $a_n = O(b_n)$, and we use $\widetilde{O}(\cdot)$ to hide the logarithmic factors. We use $\mathbf{I}_d \in \mathbb{R}^{d\times d}$ to denote the identity matrix. Throughout the prospectus, we use

$\ell_i(\cdot)$ as the shorthand notation for $\ell(\cdot; \mathbf{x}_i, y_i)$, and $\boldsymbol{\theta}_{\min}$ to denote the minimizer of problem (3.1.1).

3.1.1 Additional Related Work

To further enhance the privacy guarantee for training data, there has emerged a fresh line of research (HCB16; PAE16; BTT18; YJS19) that studies the knowledge transfer techniques for the differentially private classification problem. More specifically, these methods propose to first train an ensemble of "teacher" models based on disjoint subsets of the private dataset, and then train a "student" model based on the private aggregation of the ensemble. However, their approaches only work for the classification task, and cannot be directly applied to general sparse learning problems. Moreover, their sub-sample and aggregate framework may not be suitable for the high-dimensional sparse learning problem since each "teacher" model is trained on a subset of the private dataset, which makes the "large d, small n" scenario even worse. In contrast to their sub-sample and aggregate based knowledge transfer approach, we propose to use the distillation based method (BCN06; HVD15), which is more suitable for the high-dimensional sparse learning problem.

3.2 Proposed Method

We first lay out several definitions, which will be used throughout this chapter.

Definition 3.2.1. A function $f : \mathbb{R}^d \to \mathbb{R}$ is λ-strongly convex, if for any $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \mathbb{R}^d$,

$$f(\boldsymbol{\theta}_1) - f(\boldsymbol{\theta}_2) - \langle \nabla f(\boldsymbol{\theta}_2), \boldsymbol{\theta}_1 - \boldsymbol{\theta}_2 \rangle \geq \lambda \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2^2 / 2.$$

Definition 3.2.2. A function $f : \mathbb{R}^d \to \mathbb{R}$ is $\bar{\beta}$-smooth, if for any $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \mathbb{R}^d$,

$$f(\boldsymbol{\theta}_1) - f(\boldsymbol{\theta}_2) - \langle \nabla f(\boldsymbol{\theta}_2), \boldsymbol{\theta}_1 - \boldsymbol{\theta}_2 \rangle \leq \bar{\beta} \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2^2 / 2.$$

Next we present the definition of sub-Gaussian distribution (Ver10).

Definition 3.2.3. We say $\boldsymbol{X} \in \mathbb{R}^d$ is a sub-Gaussian random vector with parameter $\alpha > 0$, if $(\mathbb{E}|\mathbf{u}^\top \boldsymbol{X}|^p)^{1/p} \leq \alpha\sqrt{p}$ for all $p \geq 1$ and all unit vector $\mathbf{u}$ with $\|\mathbf{u}\|_2 = 1$.

Next, we present our differentially private sparse learning framework, which is illustrated in Algorithm 2. Note that Algorithm 2 will call IGHT algorithm (YLZ14; JTK14) in Algorithm 3. IGHT enjoys linear convergence rate and is widely used for sparse learning. Note that for the sparsity constraint, i.e., $\|\boldsymbol{\theta}\|_0 \leq s$, the hard thresholding operator $\mathcal{H}_s(\boldsymbol{\theta})$ is defined as follows: $[\mathcal{H}_s(\boldsymbol{\theta})]_i = \theta_i$ if $i \in \text{ supp}(\boldsymbol{\theta}, s)$ and $[\mathcal{H}_s(\boldsymbol{\theta})]_i = 0$ otherwise, for $i \in [d]$. It preserves the largest s entries of $\boldsymbol{\theta}$ in magnitude. Equipped with IGHT, our framework also has a linear convergence rate for solving high-dimensional sparsity constrained problems.

Algorithm 2 Differentially Private Sparse Learning via Knowledge Transfer (DPSL-KT)

input Loss function $\bar{L}_S$, distribution $\widetilde{\mathcal{D}}$, IGHT parameters $s, \eta_1, \eta_2, T_1, T_2$, function f, $\boldsymbol{\theta}_0$, σ

1: $\widehat{\boldsymbol{\theta}} = \text{IGHT}(\boldsymbol{\theta}_0, \bar{L}_S, s, \eta_1, T_1)$

2: Generate training set: $S^{\text{p}} = \{(\widetilde{\mathbf{x}}_i, y_i^{\text{p}})\}_{i=1}^m$, where $y_i^{\text{p}} = \langle \widehat{\boldsymbol{\theta}}, \widetilde{\mathbf{x}}_i \rangle + \xi_i$, $\widetilde{\mathbf{x}}_i \sim \widetilde{\mathcal{D}}$, $\xi_i \sim N(0, \sigma^2)$

3: Constructing the new task: $\widetilde{L}(\boldsymbol{\theta}) = (2m)^{-1} \sum_{i=1}^m \big(y_i^{\text{p}} - \langle \boldsymbol{\theta}, \widetilde{\mathbf{x}}_i \rangle\big)^2$

4: $\boldsymbol{\theta}^{\text{p}} = \text{IGHT}(\boldsymbol{\theta}_0, \widetilde{L}, s, \eta_2, T_2)$

output $\boldsymbol{\theta}^{\text{p}}$

Algorithm 3 Iterative Gradient Hard Thresholding (IGHT)

input Loss function L_S, parameters s, η, T, $\boldsymbol{\theta}_0$

1: **for** $t = 1, 2, 3, \ldots, T$ **do**

2: $\quad \boldsymbol{\theta}_t = \mathcal{H}_s\big(\boldsymbol{\theta}_{t-1} - \eta \nabla L_S(\boldsymbol{\theta}_{t-1})\big)$

3: **end for**

output $\boldsymbol{\theta}_T$

There are two key ingredients in our framework: (1) an efficient problem solver, i.e., iterative gradient hard thresholding (IGHT) algorithm (YLZ14; JTK14), and (2) the knowledge transfer procedure. In detail, we first solve the optimization problem (3.1.1) using IGHT,

which is demonstrated in Algorithm 3, to get a non-private "teacher" estimator $\widehat{\boldsymbol{\theta}}$. The next step is the knowledge transfer procedure: we draw some synthetic features $\{\widetilde{\mathbf{x}}_i\}_{i=1}^m$ from a given distribution $\widetilde{\mathcal{D}}$, and output the corresponding private-preserving responses $\{y_i^{\mathrm{p}}\}_{i=1}^m$ using the Gaussian mechanism: $y_i^{\mathrm{p}} = \langle \widehat{\boldsymbol{\theta}}, \widetilde{\mathbf{x}}_i \rangle + \xi_i$, where ξ_i is the Gaussian noise to protect the private information contained in $\widehat{\boldsymbol{\theta}}$. Finally, by solving a new sparsity constrained learning problem $\widetilde{L}$ using the privacy-preserving synthetic dataset $S^{\mathrm{p}} = \{(\widetilde{\mathbf{x}}_i, y_i^{\mathrm{p}})\}_{i=1}^m$, we can get a differentially private "student" estimator $\boldsymbol{\theta}^{\mathrm{p}}$.

Our proposed knowledge transfer framework can achieve both strong privacy and utility guarantees. Intuitively speaking, the newly constructed learning problem can reduce the utilization of the privacy budget since we only require the generated responses to preserve the privacy of original training sample, which in turn leads to a strong privacy guarantee. In addition, this new learning problem contains the knowledge of the "teacher" estimator, which preserves the sparsity information of the underlying parameter. As a result, the "student" estimator can also have a strong utility guarantee.

3.3 Main Results

In this section, we will present the privacy and utility guarantees for Algorithm 2. We start with two conditions, which will be used in the result for generic models. Later, when we apply our result to specific models, these conditions will be verified explicitly.

The first condition is about the upper bound on the gradient of the function L_S, which will be used to characterize the statistical error of generic sparse models.

Condition 3.3.1. For a given sample size n and tolerance parameter $\zeta \in (0, 1)$, let $\varepsilon(n, \zeta)$ be the smallest scalar such that with probability at least $1 - \zeta$, we have $\|\nabla L_S(\boldsymbol{\theta}^*)\|_\infty \leq \varepsilon(n, \zeta)$.

To derive the utility guarantee, we also need the sparse eigenvalue condition (Zha10) on the function L_S, which directly implies the restricted strong convex and smooth properties

(NYW09; LW13) of the function L_S.

Condition 3.3.2. The empirical loss L_S on the training data satisfies the sparse eigenvalue condition, if for all $\boldsymbol{\theta}$, there exist positive numbers μ and β such that

$$\mu = \inf_{\mathbf{v}} \left\{ \mathbf{v}^\top \nabla^2 L_S(\boldsymbol{\theta}) \mathbf{v} \mid \|\mathbf{v}\|_0 \leq s,\ \|\mathbf{v}\|_2 = 1 \right\},$$
$$\beta = \sup_{\mathbf{v}} \left\{ \mathbf{v}^\top \nabla^2 L_S(\boldsymbol{\theta}) \mathbf{v} \mid \|\mathbf{v}\|_0 \leq s,\ \|\mathbf{v}\|_2 = 1 \right\}.$$

3.3.1 Results for Generic Models

We first present the privacy guarantee of Algorithm 2 in the setting of (ϵ, δ)-DP.

Theorem 3.3.3. Suppose the loss function on each training example satisfies $\|\nabla \ell_i(\boldsymbol{\theta}_{\min})\|_\infty \leq \gamma$, and $\widetilde{\mathcal{D}}$ is a sub-Gaussian distribution with parameter $\widetilde{\alpha}$ and the covariance matrix $\|\widetilde{\boldsymbol{\Sigma}}\|_2 \leq \widetilde{\beta}$, and $m \geq C_1 \widetilde{\alpha} s \log d$ for some absolute constant C_1. Given a privacy budget ϵ and a constant $\delta \in (0, 1)$, the output $\boldsymbol{\theta}^{\mathrm{p}}$ of Algorithm 2 satisfies (ϵ, δ)-DP if $\sigma^2 = 8m\widetilde{\beta} s \gamma^2 \log(2.5/\delta)/(n^2 \epsilon^2 \lambda^2)$.

Remark 3.3.4. Theorem 3.3.3 suggests that in order to ensure the privacy guarantee, the only condition on the private data is the ℓ_∞-norm bound on the gradient of the loss function on each training example. This is in contrast to the ℓ_2-norm bound required by many previous work (KST12; TTZ15; WG19a) for sparse learning problems. We remark that ℓ_∞-norm bound is a milder condition than ℓ_2-norm bound, and gives a better utility guarantee that only depends on the ℓ_∞-norm of the input data vectors instead of their ℓ_2-norm.

Next, we provide the linear convergence rate and the utility guarantee of Algorithm 2.

Theorem 3.3.5. Suppose that the loss function $\bar{L}_S$ is $\bar{\beta}$-smooth and L_S satisfies Condition 3.3.1 with parameter $\varepsilon(n, \zeta)$. Under the same conditions of Theorem 3.3.3 on ℓ_i, $\widetilde{\mathcal{D}}$, σ^2, there exist constants $\{C_i\}_{i=1}^8$ such that if $n = m \geq C_1 \widetilde{\alpha} s \log d$, $s \geq C_2 \kappa^2 s^*$ with $\kappa = \bar{\beta}/\lambda$, the stepsize $\eta_1 = C_3 \lambda / \bar{\beta}^2, \eta_2 = C_4 / \widetilde{\beta}$, then $\boldsymbol{\theta}^{\mathrm{p}}$ converges to $\boldsymbol{\theta}^*$ at a linear rate. In addition, if we choose $\lambda^2 = C_5 \gamma \sqrt{s^* \log d \log(1/\delta)}/(n\epsilon)$, for large enough T_1, T_2, with probability at least

$1-\zeta-C_6/d$, the output $\boldsymbol{\theta}^{\mathrm{p}}$ of Algorithm 2 satisfies

$$\|\boldsymbol{\theta}^{\mathrm{p}}-\boldsymbol{\theta}^*\|_2^2 \leq C_7\frac{s^*}{\bar{\beta}^2}\varepsilon(n,\zeta)^2 + C_8\big(1/\bar{\beta}^2+\widetilde{\alpha}^2/\widetilde{\beta}\big)\frac{\gamma\sqrt{s^{*3}\log d\log(1/\delta)}}{n\epsilon}.$$

Remark 3.3.6. The utility bound of our method consists of two terms: the first term denotes the statistical error of generic sparse models, while the second one corresponds to the error introduced by the Gaussian mechanism, and is the dominating term. Therefore, the utility bound is of order $O\big(\gamma\sqrt{s^{*3}\log d\log(1/\delta)}/(n\epsilon)\big)$, which depends on the true sparsity s^* rather than the dimension of the problem d, and therefore is desirable for sparse learning.

The following corollary shows that if L_S further satisfies Condition 3.3.2, our method can achieve an improved utility guarantee.

Corollary 3.3.7. Suppose that L_S satisfies Condition 3.3.2 with parameters μ, β. Under the same conditions of Theorem 3.3.5 on $L_S, \ell_i, \widetilde{\mathcal{D}}$, the output $\boldsymbol{\theta}^{\mathrm{p}}$ of Algorithm 2 satisfies (ϵ,δ)-DP if we set $\lambda = 0$ and $\sigma^2 = 8m\widetilde{\beta}s\gamma^2\log(2.5/\delta)/(n^2\epsilon^2\mu^2)$. In addition, there exist constants $\{C_i\}_{i=1}^7$ such that if $n = m \geq C_1\widetilde{\alpha}s\log d$, $s \geq C_2\kappa^2 s^*$ with $\kappa = \beta/\mu$, step size $\eta_1 = C_3\mu/\beta^2, \eta_2 = C_4/\widetilde{\beta}$, for large enough T_1, T_2, with probability at least $1-\zeta-C_5/d$, the output $\boldsymbol{\theta}^{\mathrm{p}}$ of Algorithm 2 satisfies

$$\|\boldsymbol{\theta}^{\mathrm{p}}-\boldsymbol{\theta}^*\|_2^2 \leq C_6\frac{s^*}{\beta^2}\varepsilon(n,\zeta)^2 + C_7\widetilde{\alpha}^2\frac{\gamma^2 s^{*2}\log d\log(1/\delta)}{\widetilde{\beta}\mu^2 n^2\epsilon^2}.$$

Remark 3.3.8. Corollary 3.3.7 shows that if the training loss on the private data satisfies the sparse eigenvalue condition, Algorithm 2 can achieve $\widetilde{O}\big(\gamma^2 s^{*2}/(n\epsilon)^2\big)$ utility guarantee by setting $\lambda = 0$ and the variance σ^2 accordingly. It improves the utility without the sparse eigenvalue condition $\widetilde{O}\big(\gamma s^{*3/2}/(n\epsilon)\big)$ in Theorem 3.3.5 by a factor of $\widetilde{O}\big(n\epsilon/\gamma\sqrt{s^*}\big)$. Note that sparse eigenvalue condition has been verified for many sparse models (NYW09) including sparse linear regression and sparse logistic regression.

3.3.2 Results for Specific Models

In this subsection, we demonstrate the results of our framework for specific models. Note that the privacy guarantee has been established in Theorem 3.3.3, and we only present the utility guarantees.

3.3.2.1 Sparse linear regression

We consider the following linear regression problem in the high-dimensional regime (Tib96): $\mathbf{y} = \mathbf{X}\boldsymbol{\theta}^* + \boldsymbol{\xi}$, where $\mathbf{y} \in \mathbb{R}^n$ is the response vector, $\mathbf{X} \in \mathbb{R}^{n\times d}$ denotes the design matrix, $\boldsymbol{\xi} \in \mathbb{R}^n$ is a noise vector, and $\boldsymbol{\theta}^* \in \mathbb{R}^d$ with $\|\boldsymbol{\theta}^*\|_0 \leq s^*$ is the underlying sparse coefficient vector that we want to recover. In order to estimate the sparse vector $\boldsymbol{\theta}^*$, we consider the following sparsity constrained estimation problem, which has been studied in many previous work (Zha11; FR13; YLZ14; JTK14; CG16)

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^d} \frac{1}{2n}\|\mathbf{X}\boldsymbol{\theta} - \mathbf{y}\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{\theta}\|_2^2 \;\text{ subject to }\; \|\boldsymbol{\theta}\|_0 \leq s. \tag{3.3.1}$$

The utility guarantee of Algorithm 2 for solving (3.3.1) can be implied by Theorem 3.3.5. Here we only need to verify Condition 3.3.2 for the sparse linear regression model. In specific, we can show that $\nabla L_S(\boldsymbol{\theta}^*) = \mathbf{X}^\top\boldsymbol{\xi}/n$, and we can prove that $\|\nabla L_S(\boldsymbol{\theta}^*)\|_\infty \leq C_1\nu\sqrt{\log d/n}$ holds with probability at least $1-\exp(-C_2 n)$, where C_1, C_2 are absolute constants. Therefore, we have $\zeta = 1-\exp(-C_2 n)$, $\varepsilon(n,\zeta) = C_1\nu\sqrt{\log d/n}$. By substituting these quantities into Theorem 3.3.5, we can obtain the following corollary.

Corollary 3.3.9. Suppose that each row of the design matrix satisfies $\max_{i\in[n]}\|\mathbf{x}_i\|_\infty \leq K$, and the noise vector $\boldsymbol{\xi} \sim N(0,\nu^2\mathbf{I}_n)$. Under the same conditions of Theorem 3.3.5 on $\widetilde{\mathcal{D}}, \sigma^2, \eta_1, \eta_2, s$, there exist constants $\{C_i\}_{i=1}^5$ such that if $m = n \geq C_1 s\log d$, $\lambda^2 = C_2K^2s^*\sqrt{\log d\log(1/\delta)}/(n\epsilon)$, with probability at least $1-C_3/d$, the output $\boldsymbol{\theta}^{\mathrm{p}}$ of Algorithm 2 satisfies

$$\|\boldsymbol{\theta}^{\mathrm{p}} - \boldsymbol{\theta}^*\|_2^2 \leq C_4\nu^2K^2\frac{s^*\log d}{n} + C_5\widetilde{\alpha}^2K^2\frac{s^{*2}\sqrt{\log d\log(1/\delta)}}{\widetilde{\beta}n\epsilon}.$$

Remark 3.3.10. Corollary 3.3.9 suggests that $O\big(s^* \log d/n + K^2 s^{*2}\sqrt{\log d \log(1/\delta)}/(n\epsilon)\big)$ utility guarantee can be achieved by our algorithm. The term $O(s^* \log d/n)$ denotes the statistical error for sparse vector estimation, which matches the minimax lower bound (RWY11). While the term $\widetilde{O}(K^2 s^{*2}/(n\epsilon))$ corresponds to the error introduced by the privacy-preserving mechanism, and is the dominating term. Compared with the best-known result (KST12; WG19a) $\widetilde{O}(\widetilde{K}^2 s^{*2}/(n^2\epsilon^2))$, where $\|\mathbf{x}_i\|_2 \leq \widetilde{K}$ for all $i \in [n]$, our utility guarantee does not require the sparse eigenvalue condition and is better than their results by a factor of $\widetilde{O}\big(\widetilde{K}^2/(K^2 n\epsilon)\big)$. Since we have $\widetilde{K} \leq \sqrt{d}K$ in the worst case, the improvement factor can be as large as $\widetilde{O}\big(d/(n\epsilon)\big)$. Compared with the utility guarantee $\widetilde{O}\big(1/(n\epsilon)^{2/3}\big)$ obtained by (TTZ15), our method improves their result by a factor of $\widetilde{O}\big((n\epsilon)^{1/3}/(Ks^*)^2\big)$, which demonstrates the advantage of our framework.

Next, we present the theoretical guarantees of our methods under the extra sparse eigenvalue condition for sparse linear regression.

Corollary 3.3.11. Suppose that each row $\mathbf{x}_i$ of the design matrix satisfies $\mathbf{x}_i \sim N(0, \mathbf{\Sigma})$, $\max_{i\in[n]} \|\mathbf{x}_i\|_\infty \leq K$, and the noise vector $\boldsymbol{\xi} \sim N(0, \nu^2 \mathbf{I}_n)$. For a given ϵ, δ, under the same conditions of Corollary 3.3.7 on $\widetilde{\mathcal{D}}, \sigma^2, \lambda, \eta_1, \eta_2, s$, there exist constants $\{C_i\}_{i=1}^4$ such that if $m = n \geq C_1 s \log d$, the output of Algorithm 2 satisfies (ϵ, δ)-DP. In addition, with probability at least $1 - C_2/d$, we have

$$\|\boldsymbol{\theta}^{\mathrm{p}} - \boldsymbol{\theta}^*\|_2^2 \leq C_3 \nu^2 K^2 \frac{s^* \log d}{n} + C_4 \widetilde{\alpha}^2 K^2 \frac{s^{*3} \log d \log(1/\delta)}{\widetilde{\beta} n^2 \epsilon^2}.$$

Remark 3.3.12. According to Corollary 3.3.11, the output of Algorithm 2 will satisfy (ϵ, δ)-DP with the utility guarantee $\widetilde{O}\big(K^2 s^{*3}/(n^2\epsilon^2)\big)$, which improves the result in Corollary 3.3.9 by a factor of $\widetilde{O}\big(n\epsilon/s^*\big)$.

3.3.2.2 Sparse logistic regression

For high-dimensional logistic regression, we assume the label of each example follows an i.i.d. Bernoulli distribution conditioned on the input vector $\mathbb{P}(y = 1|\mathbf{x}, \boldsymbol{\theta}^*) = \exp\big(\boldsymbol{\theta}^{*\top}\mathbf{x} -$

$\log\big(1+\exp(\boldsymbol{\theta}^{*\top}\mathbf{x})\big)\big)$, where $\mathbf{x}\in\mathbb{R}^d$ is the input vector, $\boldsymbol{\theta}^*\in\mathbb{R}^d$ with $\|\boldsymbol{\theta}^*\|_0\le s^*$ is the sparse parameter vector we would like to estimate. Given observations $\{(\mathbf{x}_i,y_i)\}_{i=1}^n$, we consider the following maximum likelihood estimation problem with sparsity constraints (YLZ14; CG16)

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^d} -\frac{1}{n}\sum_{i=1}^n\big[y_i\boldsymbol{\theta}^\top\mathbf{x}_i-\log\big(1+\exp(\boldsymbol{\theta}^\top\mathbf{x}_i)\big)\big]+\frac{\lambda}{2}\|\boldsymbol{\theta}\|_2^2 \ \text{ subject to } \ \|\boldsymbol{\theta}\|_0\le s. \tag{3.3.2}$$

The utility guarantee of Algorithm 2 for solving (3.3.2) is shown in the following corollary.

Corollary 3.3.13. Under the same conditions of Corollary 3.3.9 on $\mathbf{x}_i,\widetilde{\mathcal{D}},\sigma^2,\eta_1,\eta_2,s$, there exist constants $\{C_i\}_{i=1}^5$ such that if $m=n\ge C_1 s\log d$, $\lambda^2=C_2K\sqrt{s^*\log d\log(1/\delta)}/(n\epsilon)$, with probability at least $1-C_3/d$, the output $\boldsymbol{\theta}^{\mathrm{p}}$ of Algorithm 2 satisfies

$$\|\boldsymbol{\theta}^{\mathrm{p}}-\boldsymbol{\theta}^*\|_2^2\le C_4K^2\frac{s^*\log d}{n}+C_5\widetilde{\alpha}^2K\frac{\sqrt{s^{*3}\log d\log(1/\delta)}}{\widetilde{\beta}n\epsilon}.$$

Remark 3.3.14. Corollary 3.3.13 suggests that $O\big(s^*\log d/n+K\sqrt{s^{*3}\log d\log(1/\delta)}/(n\epsilon)\big)$ utility guarantee can be obtained by our algorithm for sparse logistic regression. The term $\widetilde{O}\big(Ks^{*3/2}/(n\epsilon)\big)$ caused by the Gaussian mechanism is the dominating term and does not depend on the sparse eigenvalue condition, and is better than the best-known result (WG19a) $\widetilde{O}\big(\widetilde{K}^2s^{*2}/(n^2\epsilon^2)\big)$ by a factor of $\widetilde{O}\big(\widetilde{K}^2s^{*1/2}/(Kn\epsilon)\big)$. The improvement factor can be as large as $\widetilde{O}\big(dK/(n\epsilon)\big)$ since $\widetilde{K}\le\sqrt{d}K$.

If we have the extra sparse eigenvalue condition, our method can achieve an improved utility guarantee for sparse logistic regression as follows.

Corollary 3.3.15. Suppose that each row $\mathbf{x}_i$ of the design matrix satisfies $\mathbf{x}_i\sim N(0,\boldsymbol{\Sigma})$, $\max_{i\in[n]}\|\mathbf{x}_i\|_\infty\le K$. For a given ϵ,δ, under the same conditions of Corollary 3.3.7 on $\widetilde{\mathcal{D}},\sigma^2,\lambda,\eta_1,\eta_2,s$, there exist constants $\{C_i\}_{i=1}^4$ such that if $m=n\ge C_1s\log d$, the output of Algorithm 2 satisfies (ϵ,δ)-DP. In addition, with probability at least $1-C_2/d$, we have the following utility for $\boldsymbol{\theta}^{\mathrm{p}}$

$$\|\boldsymbol{\theta}^{\mathrm{p}}-\boldsymbol{\theta}^*\|_2^2\le C_3K^2\frac{s^*\log d}{n}+C_4\widetilde{\alpha}^2K^2s^{*2}\frac{\log d\log(1/\delta)}{\widetilde{\beta}n^2\epsilon^2}.$$

Remark 3.3.16. Corollary 3.3.15 shows that our method can obtain an improved utility guarantee $\widetilde{O}\big(K^2 s^{*2}/(n\epsilon)^2\big)$ for sparse logistic regression under the extra sparse eigenvalue assumption.

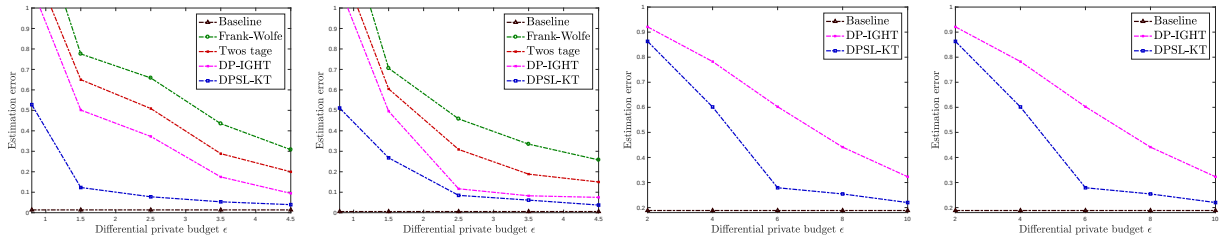
3.4 Experiments

In this section, we present experimental results of our proposed algorithm on both synthetic and real datasets. For sparse linear regression, we compare our framework with Two stage (KST12), Frank-Wolfe (TTZ15), and DP-IGHT (WG19a) algorithms. For sparse logistic regression, we compare our framework with DP-IGHT (WG19a) algorithm. For all of our experiments, we choose the parameters of different methods according to the requirements of their theoretical guarantees. More specifically, on the synthetic data experiments, we assume s^* is known for all the methods. On the real data experiments, s^* is unknown, neither our method or the competing methods has the knowledge of s^*. So we simply choose a sufficiently large s as a surrogate of s^*. Given s, for the parameter λ in our method, according to Theorem 4.5, we choose λ from a sequence of values $c_1\sqrt{s\log d\log(1/\delta)}/(n\epsilon)$, where $c_1 \in \{10^{-6}, 10^{-5}, \ldots, 10^{1}\}$, by cross-validation. For competing methods, given s, we choose the iteration number of Frank-Wolfe from a sequence of values $c_2 s$, where $c_2 \in \{0.5, 0.6, \ldots, 1.5\}$, and the regularization parameter in the objective function of Two Stage from a sequence of values $c_3 s/\epsilon$, where $c_3 \in \{10^{-3}, 10^{-2}, \ldots, 10^{2}\}$, by cross-validation. For DP-IGHT, we choose its stepsize from the grid $\{1/2^0, 1/2^1, \ldots, 1/2^6\}$ by cross-validation. For the non-private baseline, we use the non-private IGHT (YLZ14).

3.4.1 Numerical Simulations

In this subsection, we investigate our framework on synthetic datasets for sparse linear and logistic regression. In both problems, we generate the design matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$ such that each entry is drawn i.i.d. from a uniform distribution $U(-1, 1)$, and the underlying sparse

vector $\boldsymbol{\theta}^*$ has s nonzero entries that are randomly generated. In addition, we consider the following two settings: (i) $n = 800, d = 1000, s^* = 10$; (ii) $n = 4000, d = 5000, s^* = 50$. We choose $\widetilde{\mathcal{D}}$ to be a uniform distribution $U(-1, 1)$, which implies $\widetilde{\beta} = 1/3$.



(a) Linear regression (b) Linear regression (c) Logistic regression (d) Logistic regression

Figure 3.2: Numerical results for sparse linear and logistic regression.

Table 3.2: Comparison of different algorithms for various privacy budgets ϵ with $\delta = 10^{-5}$ in terms of MSE (mean $\pm$ std) and its corresponding standard deviation on E2006-TFIDF.

Method	$\epsilon = 0.8$	$\epsilon = 1.5$	$\epsilon = 2.5$	$\epsilon = 3.5$	$\epsilon = 4.5$
IGHT	0.8541	0.8541	0.8541	0.8541	0.8541
Frank-Wolfe	4.471 (0.239)	2.004 (0.155)	1.535 (0.140)	1.206 (0.095)	1.099 (0.082)
Two stage	4.022 (0.159)	1.803 (0.141)	1.326 (0.093)	1.107 (0.103)	1.053 (0.069)
DP-IGHT	3.731 (0.207)	1.687 (0.126)	1.304 (0.035)	1.067 (0.051)	0.968 (0.062)
DPSL-KT	1.227 (0.110)	1.178 (0.056)	1.065 (0.054)	0.971 (0.031)	0.952 (0.010)

Sparse linear regression For sparse linear regression, the observations are generated according to the linear regression model $\mathbf{y} = \mathbf{X}^\top \boldsymbol{\theta}^* + \boldsymbol{\xi}$, where the noise vector $\boldsymbol{\xi} \sim N(0, \nu^2 \mathbf{I})$ with $\nu^2 = 0.1$. In our experiments, we set $\delta = 0.01$ and vary the privacy budget ϵ from 0.8 to 5. Note that due to the hardness of the problem itself, we choose relatively large privacy budgets compared with the low-dimensional problem to ensure meaningful results. Figure 4.2(a) and 4.2(b) illustrate the estimation error $\|\widehat{\boldsymbol{\theta}} - \boldsymbol{\theta}^*\|_2 / \|\boldsymbol{\theta}^*\|_2$ of different methods averaged over 10 trails. The results show that the estimation error of our method is close to

Table 3.3: Comparison of different algorithms for various privacy budgets ϵ with $\delta = 10^{-5}$ in terms of test error (mean $\pm$ std) and its corresponding standard deviation on RCV1 data.

Method	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 6$	$\epsilon = 8$
IGHT	0.0645	0.0645	0.0645	0.0645
Frank-Wolfe	0.1381 (0.0045)	0.1134 (0.0041)	0.0978 (0.0032)	0.0882 (0.0033)
Two stage	0.1272 (0.0044)	0.1061(0.0038)	0.0949 (0.0035)	0.0866 (0.0031)
DP-IGHT	0.1179 (0.0035)	0.1026 (0.0036)	0.0922 (0.0032)	0.0824 (0.0029)
DPSL-KT	0.1105 (0.0038)	0.0974 (0.0035)	0.0885 (0.0029)	0.0787(0.0031)

the non-private baseline, and is significantly better than other private baselines. Even when we have a small privacy budget (i.e., $\epsilon = 0.8$), our method can still recover the underlying sparse vector with reasonably small estimation error, while others fail.

Sparse logistic regression For sparse logistic regression, each label is generated from the logistic distribution $\mathbb{P}(y = 1) = 1/\big(1 + \exp(\mathbf{x}_i^\top \boldsymbol{\theta}^*)\big)$. In this problem, we vary the privacy budget ϵ from 2 to 10, and set $\delta = 0.01$. We present the estimation error versus privacy budget ϵ of different methods in Figure 4.2(c) and 4.2(d). The results show that our method can output accurate estimators when we have relative large privacy budget, and it consistently outperforms the private baseline.

3.4.2 Real Data Experiments

For real data experiments, we use E2006-TFIDF dataset (KLR09) and RCV1 dataset (LYR04), for the evaluation of sparse linear regression and sparse logistic regression, respectively.

E2006-TFIDF data For sparse linear regression problem, we use E2006-TFIDF dataset, which consists of financial risk data from thousands of U.S. companies. In detail, it contains 16087 training examples, 3308 testing examples, and we randomly sample 25000 features for this experiment. In order to validate our proposed framework, we randomly divide

the original dataset into two datasets: private dataset and public dataset. For the private dataset, it contains 8044 training examples, and we assume that this dataset contains the sensitive information that we want to protect. For the public dataset, it contains 8043 training examples. We set $s = 2000$, $\delta = 10^{-5}$, $\epsilon \in [0.8, 5]$. We estimate $\widetilde{\beta}$ by the sample covariance matrix. Table 3.2 reports the mean square error (MSE) on the test data of different methods for various privacy budgets over 10 trails. The results show that the performance of our algorithm is close to the non-private baseline even when we have small private budgets, and is much better than existing methods.

RCV1 data For sparse logistic regression, we use a Reuters Corpus Volume I (RCV1) data set for text categorization research. RCV1 is released by Reuters, Ltd. for research purposes, and consists of over 800000 manually categorized newswire stories. It contains 20242 training examples, 677399 testing examples and 47236 features. As before, we randomly divide the original dataset into two datasets with equal size serving as the private and publice datasets. In addition, we randomly choose 10000 test examples and 20000 features, and set $s = 500$, $\delta = 10^{-5}$, $\epsilon \in [2, 8]$. We estimate $\widetilde{\beta}$ by the sample covariance matrix. We compare all algorithms in terms of their classification error on the test set over 10 replications, which is summarized in Table 3.3. Evidently our algorithm achieves the lowest test error among all private algorithms on RCV1 dataset, which demonstrates the superiority of our algorithm.

3.5 Additional Results

In this section, we present the additional theoretical guarantees of our methods under the extra sparse eigenvalue conditions for sparse linear and logistic regression.

3.5.1 Additional Main Results

Corollary 3.5.1. Suppose that each row $\mathbf{x}_i$ of the design matrix satisfies $\mathbf{x}_i \sim N(0, \mathbf{\Sigma})$, $\max_{i\in[n]} \|\mathbf{x}_i\|_\infty \leq K$, and the noise vector $\boldsymbol{\xi} \sim N(0, \nu^2 \mathbf{I}_n)$. For a given ϵ, δ, under the same

conditions of Corollary 3.3.7 on $\widetilde{\mathcal{D}}, \sigma^2, \lambda, \eta_1, \eta_2, s$, there exist constants $\{C_i\}_{i=1}^4$ such that if $m = n \geq C_1 s \log d$, the output of Algorithm 2 satisfies (ϵ, δ)-DP. In addition, with probability at least $1 - C_2/d$, we have

$$\|\boldsymbol{\theta}^{\mathrm{p}} - \boldsymbol{\theta}^*\|_2^2 \leq C_3 \nu^2 K^2 \frac{s^* \log d}{n} + C_4 \widetilde{\alpha}^2 K^2 \frac{s^{*3} \log d \log(1/\delta)}{\widetilde{\beta} n^2 \epsilon^2}.$$

Remark 3.5.2. According to Corollary 3.5.1, we can achieve an improved utility guarantee $\widetilde{O}\big(K^2 s^{*3}/(n\epsilon)^2\big)$ for sparse linear regression if we have further assumption, i.e., Gaussian distribution, on the private data $\mathbf{x}_i$.

Corollary 3.5.3. Suppose that each row $\mathbf{x}_i$ of the design matrix satisfies $\mathbf{x}_i \sim N(0, \boldsymbol{\Sigma})$, $\max_{i \in [n]} \|\mathbf{x}_i\|_\infty \leq K$. For a given ϵ, δ, under the same conditions of Corollary 3.3.7 on $\widetilde{\mathcal{D}}, \sigma^2, \lambda, \eta_1, \eta_2, s$, there exist constants $\{C_i\}_{i=1}^4$ such that if $m = n \geq C_1 s \log d$, the output of Algorithm 2 satisfies (ϵ, δ)-DP. In addition, with probability at least $1 - C_2/d$, we have the following utility for $\boldsymbol{\theta}^{\mathrm{p}}$

$$\|\boldsymbol{\theta}^{\mathrm{p}} - \boldsymbol{\theta}^*\|_2^2 \leq C_3 K^2 \frac{s^* \log d}{n} + C_4 \widetilde{\alpha}^2 K^2 s^{*2} \frac{\log d \log(1/\delta)}{\widetilde{\beta} n^2 \epsilon^2}.$$

Remark 3.5.4. Corollary 3.5.3 shows that if we have further assumption, i.e., Gaussian distribution, on the private data $\mathbf{x}_i$, we can obtain an improved utility guarantee $\widetilde{O}\big(K^2 s^{*2}/(n\epsilon)^2\big)$ for sparse linear logistic regression.

3.6 Proofs of the Main Results

3.6.1 Proof of Theorem 3.3.3

In this subsection, we will derive the differential privacy of Algorithm 2. First, we need the following lemma to characterize the properties of the generated samples. It has been previously proved for many common examples of sub-Gaussian random design (RWY11; ANW10; RZ12).

Lemma 3.6.1. Suppose each row of the design matrix $\widetilde{\mathbf{X}} \in \mathbb{R}^{m\times d}$ follows sub-Gaussian distribution with parameter $\widetilde{\alpha}$, and the covariance matrix $\|\widetilde{\mathbf{\Sigma}}\|_2 \leq \widetilde{\beta}$, there exist some constants $\{C_i\}_{i=1}^2$ such that for all $\mathbf{v} \in \mathbb{R}^d$ with at most s nonzero entries, if $m \geq C_1 s\alpha^2 \log d$, with probability at least $1 - \exp(-C_2 m)$, we have

$$\psi_1 \widetilde{\beta}\|\mathbf{v}\|_2^2 \leq \frac{\|\widetilde{\mathbf{X}}\mathbf{v}\|_2^2}{m} \leq \psi_2 \widetilde{\beta}\|\mathbf{v}\|_2^2,$$

where $\psi_1 = 4/5$ and $\psi_2 = 6/5$.

Proof of Theorem 3.3.3. Note that there is no privacy issue with respect to the newly generated features $\widetilde{\mathbf{x}}_i \in \mathbb{R}^d$ for $i = 1, \ldots, m$. We only need to prove that the generated predictions $y_1^{\mathrm{p}}, \ldots, y_m^{\mathrm{p}}$ satisfy differential privacy. Thus by the post-processing property, i.e., Lemma 6.2.5, we can show that the output $\boldsymbol{\theta}^{\mathrm{p}}$ of Algorithm 2 satisfies differential privacy.

According to Algorithm 2, we generate the new training set S^{p} with i-th example as $(y_i^{\mathrm{p}}, \widetilde{\mathbf{x}}_i)$, where $y_i^{\mathrm{p}} = \langle \widehat{\boldsymbol{\theta}}, \widetilde{\mathbf{x}}_i \rangle + \xi_i$, $\widetilde{\mathbf{x}}_i \sim \widetilde{\mathcal{D}}, \xi_i \sim N(0, \sigma^2)$. Consider the following function $\mathbf{q} : \mathcal{S}^n \to \mathbb{R}^m$ such that the i-th coordinate of $\mathbf{q}(S)$ is $\langle \widehat{\boldsymbol{\theta}}_S, \widetilde{\mathbf{x}}_i \rangle$, where $\widehat{\boldsymbol{\theta}}_S$ is trained on the training set S using IGHT, i.e., Algorithm 3. Thus for the function $\mathbf{q}$, we can characterize its sensitivity as follows: for two adjacent training sets S, S' with one different example indexed by i, we have

$$\begin{aligned}
\Delta(\mathbf{q}) &= \sqrt{\sum_{i=1}^m \left(\langle \widehat{\boldsymbol{\theta}}_S, \widetilde{\mathbf{x}}_i \rangle - \langle \widehat{\boldsymbol{\theta}}_{S'}, \widetilde{\mathbf{x}}_i \rangle\right)^2} \\
&= \sqrt{\sum_{i=1}^m \langle \widehat{\boldsymbol{\theta}}_S - \widehat{\boldsymbol{\theta}}_{S'}, \widetilde{\mathbf{x}}_i \rangle^2} \\
&\leq \sqrt{2m\widetilde{\beta}} \left\| \widehat{\boldsymbol{\theta}}_S - \widehat{\boldsymbol{\theta}}_{S'} \right\|_2, \qquad (3.6.1)
\end{aligned}$$

where the last inequality is due to the Lemma 3.6.1. Note that the inequality (3.6.1) holds with probability at least $1-\exp(-C_2 m)$. We will show in next that how this high probability can be absorbed into the definition of (ϵ, δ)-DP. Let us define the event E: inequality (3.6.1) holds, and we have $\mathbb{P}[\bar{E}] \leq \delta_2$, where $\delta_2 = \exp(-C_2 m)$. As long as we have $m \geq C_3 \log(2/\delta)$,

we can get $\delta_2 \leq \delta/2$. Given the event E holds, we can proceed to derive the privacy guarantee of our method as follows.

For two adjacent training sets S and S', we define $\boldsymbol{\theta}_S^{\min}$ and $\boldsymbol{\theta}_{S'}^{\min}$ as follows

$$\boldsymbol{\theta}_S^{\min} = \underset{\boldsymbol{\theta}\in\mathbb{R}^d}{\operatorname{argmin}} \bar{L}_S(\boldsymbol{\theta}) := L_S(\boldsymbol{\theta}) + \frac{\lambda}{2}\|\boldsymbol{\theta}\|_2^2 \quad \text{subject to} \quad \|\boldsymbol{\theta}\|_0 \leq s,$$
$$\boldsymbol{\theta}_{S'}^{\min} = \underset{\boldsymbol{\theta}\in\mathbb{R}^d}{\operatorname{argmin}} \bar{L}_{S'}(\boldsymbol{\theta}) := L_{S'}(\boldsymbol{\theta}) + \frac{\lambda}{2}\|\boldsymbol{\theta}\|_2^2 \quad \text{subject to} \quad \|\boldsymbol{\theta}\|_0 \leq s.$$

Therefore, we can obtain

$$\begin{aligned}\big\|\widehat{\boldsymbol{\theta}}_S - \widehat{\boldsymbol{\theta}}_{S'}\big\|_2 &\leq \big\|\widehat{\boldsymbol{\theta}}_S - \boldsymbol{\theta}_S^{\min}\big\|_2 + \big\|\widehat{\boldsymbol{\theta}}_{S'} - \boldsymbol{\theta}_{S'}^{\min}\big\|_2 + \big\|\boldsymbol{\theta}_S^{\min} - \boldsymbol{\theta}_{S'}^{\min}\big\|_2 \\ &\leq \varrho^T\big\|\boldsymbol{\theta}_S^{\min}\big\|_2 + \varrho^T\big\|\boldsymbol{\theta}_{S'}^{\min}\big\|_2 + \big\|\boldsymbol{\theta}_S^{\min} - \boldsymbol{\theta}_{S'}^{\min}\big\|_2, \end{aligned} \tag{3.6.2}$$

where $\varrho < 1$ and the last inequality is due to the convergence guarantee (YLZ14) of IGHT for $\bar{L}_S, \bar{L}_{S'}$. Since $\bar{L}_S$ is strongly convex with parameter λ, we have

$$\langle \nabla \bar{L}_S(\boldsymbol{\theta}_S^{\min}) - \nabla \bar{L}_S(\boldsymbol{\theta}_{S'}^{\min}), \boldsymbol{\theta}_S^{\min} - \boldsymbol{\theta}_{S'}^{\min}\rangle \geq \lambda\big\|\boldsymbol{\theta}_S^{\min} - \boldsymbol{\theta}_{S'}^{\min}\big\|_2^2.$$

In addition, we have $\langle \nabla \bar{L}_S(\boldsymbol{\theta}_S^{\min}), \boldsymbol{\theta}_{S'}^{\min} - \boldsymbol{\theta}_S^{\min}\rangle \geq 0$, $\langle \nabla \bar{L}_{S'}(\boldsymbol{\theta}_{S'}^{\min}), \boldsymbol{\theta}_S^{\min} - \boldsymbol{\theta}_{S'}^{\min}\rangle \geq 0$, which implies

$$\langle \nabla \bar{L}_{S'}(\boldsymbol{\theta}_{S'}^{\min}) - \nabla \bar{L}_S(\boldsymbol{\theta}_{S'}^{\min}), \boldsymbol{\theta}_S^{\min} - \boldsymbol{\theta}_{S'}^{\min}\rangle \geq \lambda\big\|\boldsymbol{\theta}_S^{\min} - \boldsymbol{\theta}_{S'}^{\min}\big\|_2^2.$$

Thus we can obtain

$$\begin{aligned}\lambda\big\|\boldsymbol{\theta}_S^{\min} - \boldsymbol{\theta}_{S'}^{\min}\big\|_2 &\leq \sqrt{2s}\big\|\nabla \bar{L}_{S'}(\boldsymbol{\theta}_{S'}^{\min}) - \nabla \bar{L}_S(\boldsymbol{\theta}_{S'}^{\min})\big\|_\infty \\ &= \frac{\sqrt{2s}}{2n}\big\|\nabla \ell(\boldsymbol{\theta}_{S'}^{\min}; \mathbf{x}_i) - \nabla \ell(\boldsymbol{\theta}_{S'}^{\min}; \mathbf{x}_{i'})\big\|_\infty. \end{aligned} \tag{3.6.3}$$

Since we have $\|\nabla \ell(\boldsymbol{\theta}_{S'}^{\min}; \mathbf{x}_i)\|_\infty \leq \gamma$ for all $\mathbf{x}_i$, we can get

$$\|\boldsymbol{\theta}_S^{\min} - \boldsymbol{\theta}_{S'}^{\min}\|_2 \leq \frac{\sqrt{2s}\gamma}{n\lambda}. \tag{3.6.4}$$

As a result, combining (3.6.1), (3.6.2), and (3.6.4), for large enough T, we can obtain

$$\Delta(\mathbf{q}) \leq 2\sqrt{ms\widetilde{\beta}}\frac{\gamma}{n\lambda}. \tag{3.6.5}$$

As a result, according to Lemma 6.2.4, to ensure $(\epsilon, \delta/2)$-DP, we need to add the zero mean Gaussian vector with the variance parameter

$$\sigma^2 = \frac{8m\widetilde{\beta}s\gamma^2}{n^2\epsilon^2\lambda^2}\log(2.5/\delta). \tag{3.6.6}$$

We use $\mathcal{M}$ to denote our mechanism, i.e., Algorithm 2. Given E happens, $\mathcal{M}$ satisfies $(\epsilon, \delta/2)$-DP. Now, we are ready show that $\mathcal{M}$ satisfies (ϵ, δ)-DP. According to Remark 3.1.2 in (DMN06), we need to prove that

$$\max_{O\in\mathcal{R}} \log \frac{\mathbb{P}[\mathcal{M}(S) \in O] - \delta}{\mathbb{P}[\mathcal{M}(S') \in O]} \leq \epsilon.$$

Since we have for all $O \in \mathcal{R}$

$$\begin{aligned}
\mathbb{P}[\mathcal{M}(S) \in O] &= \mathbb{P}[\mathcal{M}(S) \in O \mid E] \cdot \mathbb{P}[E] + \mathbb{P}[\mathcal{M}(S) \in O \mid \bar{E}] \cdot \mathbb{P}[\bar{E}] \\
&\leq \left(e^{\epsilon}\mathbb{P}[\mathcal{M}(S') \in O \mid E] + \delta/2\right) \cdot \mathbb{P}[E] + \delta/2 \\
&\leq e^{\epsilon}\mathbb{P}[\mathcal{M}(S') \in O] + \delta/2 + \delta/2,
\end{aligned}$$

where the second inequality is due to the $(\epsilon, \delta/2)$-DP of our method given inequality (3.6.1) holds, and the fact that $\mathbb{P}[\bar{E}] \leq \delta/2$. Therefore, we can obtain that

$$\max_{O\in\mathcal{R}} \log \frac{\mathbb{P}[\mathcal{M}(S) \in O] - \delta}{\mathbb{P}[\mathcal{M}(S') \in O]} \leq \max_{O\in\mathcal{R}} \log \frac{e^{\epsilon}\mathbb{P}[\mathcal{M}(S') \in O] + \delta/2 + \delta/2 - \delta}{\mathbb{P}[\mathcal{M}(S') \in O]} = \epsilon,$$

which implies Algorithm 2 satisfies (ϵ, δ)-DP. And the conditions we need are: $\widetilde{\mathbf{x}}_i$ are i.i.d. sub-Gaussian random vector with parameter α, the generated sample size $m \geq \max\{C_1 s\widetilde{\alpha}^2 \log d, C_3 \log(2/\delta)\}$, where C_1, C_3 are absolute constants. □

3.6.2 Proof of Theorem 3.3.5

In this subsection, we establish the utility guarantee of Algorithm 2. In order to prove the utility guarantee of our method, we need the following lemmas.

Lemma 3.6.2. Consider the sparsity constrained problem (3.1.1). Suppose that $\bar{L}_S$ is $\bar{\beta}$-smooth, and L_S satisfies Condition 3.3.1 with parameter ε. There exist constants $\{C_i\}_{i=1}^5$

such that if $\eta = C_1\lambda/\bar{\beta}^2$, $s \geq C_2\kappa^2 s^*$, where $\kappa = \bar{\beta}/\lambda$, the output $\widehat{\boldsymbol{\theta}}$ of Algorithm 3 satisfies the following with probability at least $1-\rho$

$$\|\boldsymbol{\theta}_T - \boldsymbol{\theta}^*\|_2^2 \leq \varrho^T \|\boldsymbol{\theta}_0 - \boldsymbol{\theta}^*\|_2^2 + C_4 \frac{s^*}{\bar{\beta}^2}(\varepsilon^2 + \lambda^2 \|\boldsymbol{\theta}^*\|_\infty^2),$$

where $\varrho = 1 - 1/(7\kappa)$. If T is large enough, we have $\|\boldsymbol{\theta}_T - \boldsymbol{\theta}^*\|_2^2 \leq C_5 s^*(\varepsilon^2 + \lambda^2\|\boldsymbol{\theta}^*\|_\infty^2)/\bar{\beta}^2$.

The next lemma, which has been proved in (LW13), provides the statistical error of sparse linear regression, which will be used to characterize the statistical error of our newly constructed learning problem.

Lemma 3.6.3. For a Gaussian random vector $\boldsymbol{\epsilon} \in \mathbb{R}^n$ with zero mean and variance $\nu^2\mathbf{I}_n$, if each row of $\mathbf{X} \in \mathbb{R}^{n\times d}$ are independent sub-Gaussian random vector with sub-Gaussian parameter α, we have with probability at least $1 - \exp(-C_6 n)$

$$\left\| \frac{1}{n}\mathbf{X}^\top \boldsymbol{\epsilon} \right\|_\infty \leq C_7 \nu\alpha \sqrt{\frac{\log d}{n}},$$

where C_6, C_7 are absolute constants.

Proof of Theorem 3.3.5. According to Lemma 3.6.2, we can obtain that

$$\|\widehat{\boldsymbol{\theta}} - \boldsymbol{\theta}^*\|_2^2 \leq C_1 \frac{s^*}{\bar{\beta}^2}(\varepsilon^2 + \lambda^2\|\boldsymbol{\theta}^*\|_\infty^2), \tag{3.6.7}$$

where C_1 is a universal constant. According to Algorithm 2, we have

$$\widetilde{L}(\boldsymbol{\theta}) = \frac{1}{2m}\sum_{i=1}^m \left(y_i^{\mathrm{p}} - \langle \boldsymbol{\theta}, \widetilde{\mathbf{x}}_i\rangle\right)^2.$$

Note that according to Lemma 3.6.1, $\widetilde{L}$ satisfies Condition 3.3.2 with parameters $\psi_1\widetilde{\beta}, \psi_2\widetilde{\beta}$, where $\psi_1 = 4/5, \psi_2 = 6/5$. In addition, according to Lemma 3.6.3, we have $\|\nabla\widetilde{L}(\widehat{\boldsymbol{\theta}})\|_\infty = \|\widetilde{\mathbf{X}}^\top\boldsymbol{\xi}/n\|_\infty = \widetilde{\varepsilon} \leq C_2\sigma\widetilde{\alpha}\sqrt{\log d/m}$ holds with probability at least $1 - \exp(-C_3 m)$. As a result, according to Lemma 3.6.2, we can get

$$\|\boldsymbol{\theta}^{\mathrm{p}} - \widehat{\boldsymbol{\theta}}\|_2^2 \leq C_4 \frac{s^*}{\widetilde{\beta}^2}\widetilde{\varepsilon}^2, \tag{3.6.8}$$

where C_2, C_3, C_4 are universal constants. As a result, combining (3.6.7) and (3.6.8), we can obtain

$$\begin{aligned}\|\boldsymbol{\theta}^{\mathrm{p}} - \boldsymbol{\theta}^*\|_2^2 &\leq 2\|\widehat{\boldsymbol{\theta}}^{\mathrm{p}} - \widehat{\boldsymbol{\theta}}\|_2^2 + 2\|\widehat{\boldsymbol{\theta}} - \boldsymbol{\theta}^*\|_2^2 \\ &\leq 2C_1\frac{s^*}{\bar{\beta}^2}(\varepsilon^2 + \lambda^2\|\boldsymbol{\theta}^*\|_\infty^2) + 2C_2\frac{s^*}{\widetilde{\beta}^2}\widetilde{\varepsilon}^2 \\ &\leq C_5\frac{s^*}{\bar{\beta}^2}(\varepsilon^2 + \lambda^2\|\boldsymbol{\theta}^*\|_\infty^2) + C_6\widetilde{\alpha}^2\frac{s^*}{\widetilde{\beta}^2}\cdot\frac{\log d}{m}\sigma^2,\end{aligned}$$

where C_5, C_6 are absolute constants. Plugging the definition of σ^2 in (3.6.6), we can get

$$\|\widehat{\boldsymbol{\theta}}^{\mathrm{p}} - \boldsymbol{\theta}^*\|_2^2 \leq C_5\frac{s^*}{\bar{\beta}^2}(\varepsilon^2 + \lambda^2\|\boldsymbol{\theta}^*\|_\infty^2) + C_7\widetilde{\alpha}^2\frac{\widetilde{\beta}\widetilde{s}^{*2}}{\widetilde{\beta}^2}\frac{\gamma^2\log d}{n^2\epsilon^2\lambda^2}\log(2.5/\delta).$$

Let $\lambda^2 = C_8\gamma\sqrt{s^*\log d\log(1/\delta)}/(n\epsilon)$, we can get

$$\|\widehat{\boldsymbol{\theta}}^{\mathrm{p}} - \boldsymbol{\theta}^*\|_2^2 \leq C_9\frac{s^*}{\bar{\beta}^2}\varepsilon^2 + C_{10}\left(\frac{1}{\bar{\beta}^2} + \frac{\widetilde{\alpha}^2}{\widetilde{\beta}}\right)\frac{\gamma\sqrt{s^{*3}\log d\log(1/\delta)}}{n\epsilon},$$

where C_7, C_8, C_9, C_{10} are absolute constants. Note that according to Lemma 3.6.2, Algorithm 2 has a linear convergence rate. □

3.6.3 Proof of Corollary 3.3.7

In this subsection we show that if L_S further satisfies Condition 3.3.2, our method can achieve an improved utility guarantee.

Proof of Corollary 3.3.7. We first prove the privacy guarantee of our method. The proof is similar to the proof of Theorem 3.3.3. Since we have that L satisfies Condition 3.3.2 with parameters μ, β, we can get the sensitivity of our method according to (3.6.5) as follows

$$\Delta(\mathbf{q}) \leq 2\sqrt{ms\widetilde{\beta}}\frac{\gamma}{n\mu}.$$

Therefore, according to (3.6.6), if we add the noise with the following variance

$$\sigma^2 = \frac{8m\widetilde{\beta}s\gamma^2}{n^2\epsilon^2\mu^2}\log(2.5/\delta),$$

we can ensure that Algorithm 2 satisfies (ϵ, δ)-DP.

Next, we establish the utility guarantee of our method. According to (3.6.7), we have

$$\|\widehat{\boldsymbol{\theta}} - \boldsymbol{\theta}^*\|_2^2 \leq C_1 \frac{s^*}{\beta^2} \varepsilon^2. \tag{3.6.9}$$

In addition, according to (3.6.8), we have

$$\|\widehat{\boldsymbol{\theta}}^{\mathrm{p}} - \widehat{\boldsymbol{\theta}}\|_2^2 \leq C_2 \widetilde{\alpha}^2 \frac{\widetilde{\beta}\widetilde{s}^{*2}}{\widetilde{\beta}^2} \frac{\gamma^2 \log d}{n^2 \epsilon^2 \mu^2} \log(2.5/\delta). \tag{3.6.10}$$

Combining (3.6.9) and (3.6.10), we can get

$$\|\widehat{\boldsymbol{\theta}}^{\mathrm{p}} - \boldsymbol{\theta}^*\|_2^2 \leq C_3 \frac{s^*}{\beta^2} \varepsilon^2 + C_4 \widetilde{\alpha}^2 \frac{s^{*2} \gamma^2 \log d}{\widetilde{\beta} n^2 \epsilon^2 \mu^2} \log(2.5/\delta),$$

where C_1, C_2, C_3, C_4 are absolute constants. This completes the proof. □

3.7 Proofs of Specific Examples

In this section, we only establish the utility guarantees of our proposed method for different problems, including sparse linear regression and sparse logistic regression since the privacy guarantee of Algorithm 2 has been proved in Theorem 3.3.3. For the ease of presentation, we use L to denote L_S in the following discussion.

3.7.1 Proof of Corollary 3.3.9

In order to prove Corollary 3.3.9, we only need to verify Condition 3.3.1 for L, the upper bound γ of ℓ_i.

Proof of Corollary 3.3.9. According to the objective function in (3.3.1), we have the following close form of gradient and Hessian for L

$$\nabla L(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}_i^\top \boldsymbol{\theta} - y_i) \mathbf{x}_i, \quad \nabla^2 L(\boldsymbol{\theta}) = \frac{\mathbf{X}^\top \mathbf{X}}{n},$$

where $\mathbf{x}_i$ is the i-th row of the design matrix $\mathbf{X}$. First, we verify that $\bar{L}$ is $\bar{\beta}$-smooth. According to the proof of Lemma 3.6.2, we only need to show the upper bound of $\nabla^2 L(\boldsymbol{\theta})$ restricted to some $3s$ sparse support Ω. As a result, we have $\left\|\left(\nabla^2 L(\boldsymbol{\theta})\right)_{\Omega,\Omega}\right\|_2 \leq 3sK^2$, which implies that $\bar{\beta} = 3sK^2 + \lambda$. In addition, we have $\nabla L(\boldsymbol{\theta}^*) = \mathbf{X}^\top \boldsymbol{\varepsilon}/n$. According to the proof of Corollary 2 in (LW13), we have $\|\nabla L(\boldsymbol{\theta}^*)\|_\infty \leq C_1 \nu K \sqrt{\log d/n}$ holds with probability at least $1 - \exp(-C_2 n)$, where C_1, C_2 are absolute constants. Thus we have Condition 3.3.1 holds for L. Next, we are going to estimate the parameter γ for our utility guarantee. For the loss function on each training example, we have $\ell_i(\boldsymbol{\theta}) = (\langle \mathbf{x}_i, \boldsymbol{\theta}\rangle - y_i)^2/2$, which implies $\nabla \ell_i(\boldsymbol{\theta}) = (\langle \mathbf{x}_i, \boldsymbol{\theta}\rangle - y_i)\mathbf{x}_i$. According to (3.6.3), we need to verify $\|\nabla \ell_i(\boldsymbol{\theta}_{\min})\|_\infty \leq \gamma$, where $\boldsymbol{\theta}_{\min}$ is the minimizer of (3.1.1). Since we have $\|\nabla \ell_i(\boldsymbol{\theta}_{\min})\|_\infty = \|(\langle \mathbf{x}_i, \boldsymbol{\theta}_{\min}\rangle - y_i)\mathbf{x}_i\|_\infty \leq C_3\sqrt{s}K^2$, which implies that $\gamma \leq C_3\sqrt{s}K^2$.

Finally, plugging these results into Theorem 3.3.5, we have if $\lambda^2 = C_4 K^2 s^* \sqrt{\log d \log(1/\delta)}/\big(n\epsilon\big)$, we can get

$$\|\widehat{\boldsymbol{\theta}}^{\mathrm{p}} - \boldsymbol{\theta}^*\|_2^2 \leq C_5 \nu^2 K^2 \frac{s \log d}{n} + C_6 \widetilde{\alpha}^2 \frac{K^2 s^{*2} \sqrt{\log d \log(1/\delta)}}{\widetilde{\beta} n \epsilon}.$$

□

3.7.2 Proof of Corollary 3.3.13

In this subsection, we prove the results for sparse logistic regression, and we only need to verify Conditions 3.3.1 for L, the upper bound γ of ℓ_i.

Proof of Corollary 3.3.13. According to the loss function in (3.3.2), we can obtain

$$\nabla L(\boldsymbol{\theta}) = -\frac{1}{n}\sum_{i=1}^n \big(y_i - \psi(\boldsymbol{\theta}^\top \mathbf{x}_i)\big)\mathbf{x}_i, \quad \nabla^2 L(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^n \psi'(\boldsymbol{\theta}^\top \mathbf{x}_i)\mathbf{x}_i \mathbf{x}_i^\top,$$

where $\psi(x) = \exp(x)/(1+\exp(x))$ and $\psi'(x) = \exp(x)/(1+\exp(x))^2$. Since we have $\psi'(x) \leq 1$, following the same proof procedure as before, we can get $\bar{L}$ is $\bar{\beta}$-smooth with $\bar{\beta} = 3sK + \lambda$. In addition, we have $\nabla L(\boldsymbol{\theta}^*) = \frac{1}{n}\sum_{i=1}^n b_i \mathbf{x}_i$, where $b_i = y_i - \psi(\boldsymbol{\theta}^{*\top}\mathbf{x}_i)$. Thus, according to the

proof of Corollary 2 in (LW13), we have $\|\nabla L(\boldsymbol{\theta}^*)\|_\infty \leq C_1 K\sqrt{\log d/n}$ holds with probability at least $1 - C_2/d$, where C_1, C_2 are absolute constants. In addition, we have

$$\|\nabla \ell_i(\boldsymbol{\theta}_{\min})\|_\infty = \big\|\big(y_i - \psi(\boldsymbol{\theta}_{\min}^\top \mathbf{x}_i)\big)\mathbf{x}_i\big\|_\infty \leq K,$$

where the inequality is due the the fact that $y_i \in \{0, 1\}$, $\psi(x) \in (0, 1)$, and $\|\mathbf{x}_i\|_\infty \leq K$. Thus we have $\gamma = K$ for sparse logistic regression.

Finally, plugging these results into Theorem 3.3.5, we have if $\lambda^2 = C_6 K\sqrt{s^* \log d \log(1/\delta)}/\big(n\epsilon\big)$, we can get

$$\|\widehat{\boldsymbol{\theta}}^{\mathrm{p}} - \boldsymbol{\theta}^*\|_2^2 \leq C_7 \nu^2 K^2 \frac{s \log d}{n} + C_8 \widetilde{\alpha}^2 \frac{K\sqrt{s^{*3} \log d \log(1/\delta)}}{\widetilde{\beta} n\epsilon}.$$

□

3.7.3 Proof of Corollary 3.3.11

To prove this result, we only need to verify that L satisfies the sparse eigenvalue condition since other conditions has been previously verified in the proof of Corollary 3.3.9.

Proof of Corollary 3.3.11. Since we have $\nabla^2 L(\boldsymbol{\theta}) = \mathbf{X}^\top \mathbf{X}/n$, according to Proposition 1 in (ANW10), we can obtain that L satisfies Condition 3.3.2 with parameters $\beta = 6/5$ and $\mu = 4/5$ with probability at least $1 - \exp(-C_1 n)$ if we have $n \geq C_2 s \log d$, where C_1, C_2 are absolute constants. Therefore, following the same proof procedure as in the proof of Theorem 3.3.3, this high probability can be absorbed into the δ term in the (ϵ, δ)-DP. As a results, we complete the proof. □

3.7.4 Proof of Corollary 3.3.15

To prove this result, we only need to verify that L satisfies the sparse eigenvalue condition since other conditions has been previously verified in the proof of Corollary 3.3.13.

Proof of Corollary 3.3.15. Since we have $\nabla^2 L(\boldsymbol{\theta}) = (n)^{-1}\sum_{i=1}^{n}\psi'(\boldsymbol{\theta}^\top \mathbf{x}_i)\mathbf{x}_i\mathbf{x}_i^\top$, and $\psi'(\boldsymbol{\theta}^\top \mathbf{x}_i$ is upper and lower bounded by some constants C_1, C_2, we can follow the same procedure as in the proof of Corollary 3.3.11 to show that L satisfies Condition 3.3.2 with parameters $\beta = 6/5C_1$ and $\mu = 4/5C_2$. As a results, we complete the proof. □

3.8 Proofs of Additional Lemmas

In this section, we prove the additional lemmas used in the proofs of the main results. For the ease of presentation, we use L to denote L_S.

3.8.1 Proof of Lemma 3.6.2

Proof. According to Algorithm 3, we have

$$\boldsymbol{\theta}_{t+1} = \mathcal{H}_s\big(\boldsymbol{\theta}_t - \eta\nabla\bar{L}(\boldsymbol{\theta}_t)\big).$$

We denote $\Omega = \text{ supp}(\boldsymbol{\theta}_t) \cup \text{ supp}(\boldsymbol{\theta}_{t+1}) \cup \text{ supp}(\boldsymbol{\theta}^*)$, and we have $s \leq |\Omega| \leq (2s + s^*)$. In addition, we denote $\widetilde{\boldsymbol{\theta}}_{t+1}$ by $\mathcal{P}_\Omega\big(\boldsymbol{\theta}_t - \eta\nabla\bar{L}(\boldsymbol{\theta}_t)\big)$, thus we have $\boldsymbol{\theta}_{t+1} = \mathcal{H}_s(\widetilde{\boldsymbol{\theta}}_{t+1})$. Furthermore, we have the following

$$\begin{aligned}
\|\widetilde{\boldsymbol{\theta}}_{t+1} - \boldsymbol{\theta}^*\|_2^2 &= \big\|\mathcal{P}_\Omega\big(\boldsymbol{\theta}_t - \eta\nabla\bar{L}(\boldsymbol{\theta}_t)\big) - \boldsymbol{\theta}^*\big\|_2^2 \\
&= \big\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^* - \eta\mathcal{P}_\Omega\big(\nabla\bar{L}(\boldsymbol{\theta}_t)\big)\big\|_2^2 \\
&= \Big\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^* - \eta\mathcal{P}_\Omega\big(\nabla\bar{L}(\boldsymbol{\theta}^*) + \big(\mathbf{H}(\gamma)\big)_{*\Omega}(\boldsymbol{\theta}_t - \boldsymbol{\theta}^*)\big)\Big\|_2^2,
\end{aligned}$$

where the last equation is due to the fundamental theorem of calculus, $\mathbf{H}(\gamma) = \int_0^1 \nabla^2\bar{L}(\boldsymbol{\theta}^* + \gamma(\boldsymbol{\theta} - \boldsymbol{\theta}^*))d\gamma$, and $\mathbf{H}(\gamma)_{*\Omega}$ denotes that we restrict columns of $\mathbf{H}(\gamma)$ to the support Ω. Therefore, according to the definition of $\mathcal{P}_\Omega$, we can further obtain

$$\begin{aligned}
\|\widetilde{\boldsymbol{\theta}}_{t+1} - \boldsymbol{\theta}^*\|_2^2 &= \big\|\mathbf{A}(\boldsymbol{\theta}_t - \boldsymbol{\theta}^*) - \eta\mathcal{P}_\Omega\big(\nabla\bar{L}(\boldsymbol{\theta}^*)\big)\big\|_2^2 \\
&\leq \|\mathbf{A}\|_2^2 \cdot \|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|_2^2 + \eta^2\big\|\mathcal{P}_\Omega\big(\nabla\bar{L}(\boldsymbol{\theta}^*)\big)\big\|_2^2 - 2\eta\langle\mathbf{A}(\boldsymbol{\theta}_t - \boldsymbol{\theta}^*), \mathcal{P}_\Omega\big(\nabla\bar{L}(\boldsymbol{\theta}^*)\big)\rangle,
\end{aligned}$$

where we have $\mathbf{A} = \mathbf{I} - \eta\big(\mathbf{H}(\gamma)\big)_{\Omega\Omega}$. Thus by Young's inequality, we can obtain

$$-2\eta\langle \mathbf{A}(\boldsymbol{\theta}_t - \boldsymbol{\theta}^*), \mathcal{P}_\Omega\big(\nabla \bar{L}(\boldsymbol{\theta}^*)\big)\rangle \leq \frac{2\eta\bar{\beta}}{7}\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|_2^2 + \frac{14\eta}{\bar{\beta}}\big(\|\mathbf{A}\|_2^2 \cdot \big\|\mathcal{P}_\Omega\big(\nabla \bar{L}(\boldsymbol{\theta}^*)\big)\big\|_2^2\big).$$

Therefore, we can get

$$\begin{aligned}\|\widetilde{\boldsymbol{\theta}}_{t+1} - \boldsymbol{\theta}^*\|_2^2 &\leq \|\mathbf{A}\|_2^2 \cdot \|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|_2^2 + \eta^2\big\|\mathcal{P}_\Omega\big(\nabla \bar{L}(\boldsymbol{\theta}^*)\big)\big\|_2^2 + \frac{2\eta\bar{\beta}}{7}\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|_2^2 + \frac{14\eta}{\bar{\beta}}\big(\|\mathbf{A}\|_2^2 \cdot \big\|\mathcal{P}_\Omega\big(\nabla \bar{L}(\boldsymbol{\theta}^*)\big)\big\|_2^2\big)\\ &\leq \Big(1 - \frac{5\eta\bar{\beta}}{7}\Big)\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|_2^2 + \Big(\frac{14\eta}{\bar{\beta}} - 14\eta^2\Big)\big\|\mathcal{P}_\Omega\big(\nabla \bar{L}(\boldsymbol{\theta}^*)\big)\big\|_2^2,\end{aligned}$$

where the last inequality is due to the Condition 3.3.2.

In addition, according to Lemma 3.3 in (LAL16), we have

$$\|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}^*\|_2^2 \leq \left(1 + \frac{2\sqrt{s^*}}{\sqrt{s - s^*}}\right)\|\widetilde{\boldsymbol{\theta}}_{t+1} - \boldsymbol{\theta}^*\|_2^2, \tag{3.8.1}$$

which implies that

$$\|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}^*\|_2^2 \leq \alpha\Big(1 - \frac{5\eta\bar{\beta}}{7}\Big)\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|_2^2 + \alpha(2s + s^*)\Big[\Big(\frac{14\eta}{\bar{\beta}} - 14\eta^2\Big)\big(\|\nabla \bar{L}(\boldsymbol{\theta}^*)\|_\infty^2\big)\Big],$$

where $\alpha = 1 + 2\sqrt{s^*}/\sqrt{s - s^*}$. Since we have $\eta = 2\lambda/\bar{\beta}^2$, as long as $s \geq (4\kappa^2 + 1)s^*$, where $\kappa = \bar{\beta}/\lambda$, we can get

$$\|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}^*\|_2^2 \leq \varrho\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|_2^2 + C_1\frac{s^*\lambda}{\bar{\beta}^3}\|\nabla \bar{L}(\boldsymbol{\theta}^*)\|_\infty^2,$$

where the we have $\varrho \leq 1 - 1/(7\kappa) < 1$.

In addition, we have $\nabla \bar{L}(\boldsymbol{\theta}^*) = \nabla L(\boldsymbol{\theta}^*) + \lambda\boldsymbol{\theta}^*$. According to Condition 3.3.1, we have

$$\|\nabla \bar{L}(\boldsymbol{\theta}^*)\|_\infty = \|\nabla L(\boldsymbol{\theta}^*) + \lambda\boldsymbol{\theta}^*\|_\infty \leq \|\nabla L(\boldsymbol{\theta}^*)\|_\infty + \lambda\|\boldsymbol{\theta}^*\|_\infty \leq \varepsilon + \lambda\|\boldsymbol{\theta}^*\|_\infty.$$

As long as we choose $\lambda = O(\varepsilon/\|\boldsymbol{\theta}^*\|_\infty)$, we can get

$$\|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}^*\|_2^2 \leq \varrho\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|_2^2 + C_2\frac{s^*\lambda}{\bar{\beta}^3}(\varepsilon^2 + \lambda^2\|\boldsymbol{\theta}^*\|_\infty^2). \tag{3.8.2}$$

Thus taking sum of (3.8.2) over $t = 0, 1, \ldots, T-1$, we can get

$$\begin{aligned}\|\boldsymbol{\theta}_T - \boldsymbol{\theta}^*\|_2^2 &\leq \varrho^T\|\boldsymbol{\theta}^*\|_2^2 + C_2\frac{s^*\lambda}{\bar{\beta}^3(1-\varrho)}(\varepsilon^2 + \lambda^2\|\boldsymbol{\theta}^*\|_\infty^2)\\ &\leq \varrho^T\|\boldsymbol{\theta}^*\|_2^2 + C_2\frac{s^*}{\bar{\beta}^2}(\varepsilon^2 + \lambda^2\|\boldsymbol{\theta}^*\|_\infty^2).\end{aligned} \tag{3.8.3}$$

Therefore, if we have

$$T \geq C_3 \kappa \log \frac{\bar{\beta}\|\boldsymbol{\theta}^* - \boldsymbol{\theta}_0\|_2}{s^*(\varepsilon + \lambda\|\boldsymbol{\theta}^*\|_\infty)},$$

we can obtain that

$$\|\boldsymbol{\theta}_T - \boldsymbol{\theta}^*\|_2^2 \leq C_4 \frac{s^*}{\bar{\beta}^2}(\varepsilon^2 + \lambda^2\|\boldsymbol{\theta}^*\|_\infty^2),$$

where $\{C_i\}_{i=1}^4$ are universal constants. □

3.9 Conclusions and Future Work

In this prospectus, we developed a differentially private framework for sparse learning using the idea of knowledge transfer. We establish the linear convergence rate and the utility guarantee of our method. Experiments on both synthetic and real-world data demonstrate the superiority of our algorithm. For the future work, it is very interesting to generalize our framework to other structural constrained learning problems such as the low-rank estimation problem. It is also very interesting to study the theoretical lower-bound of the differentially private sparse learning problem to access the optimality of our proposed method.

CHAPTER 4

Efficient Privacy-Preserving Stochastic Nonconvex Optimization

4.1 Introduction

For many important domains such as health care and medical research, the datasets used to train machine learning models contain sensitive personal information. There is a risk that models trained on this data can reveal private information about individual records in that training data (FLJ14; SSS17b; CLE19). This motivates the research on privacy-preserving machine learning, much of which has focused on achieving *differential privacy* (DMN06), a rigorous definition of privacy that provides statistical data privacy for individual records. In the past decade, many differentially private machine learning algorithms for solving the empirical risk minimization (ERM) problem have been proposed (e.g., (CMS11b; KST12; BST14c; ZZM17b; WYX17b; JWE18b; WG19b; WG20)). Almost all of these are for ERM with convex loss functions, but many important machine learning approaches, including deep learning, are formulated as ERM problems with nonconvex loss functions. Furthermore, these learning problems often require large training sets, necessitating the use of stochastic optimization algorithms such as stochastic gradient descent (SGD).

Several recent studies have advanced the application of differential privacy in deep learning (ACG16b; PAE16; MRT18b; BDL19). The studies prove differential privacy is satisfied, but evaluate utility experimentally. Only a few differentially private algorithms for solving nonconvex optimization problems have proven utility bounds (ZZM17b; WYX17b). For ex-

ample, (WYX17b) proposed a differentially private gradient descent (DP-GD) algorithm with both privacy and utility guarantees. However, each iteration of DP-GD requires computing the full gradient, which makes it too expensive for use on large training sets. (ZZM17b) proposed a random round private stochastic gradient descent (RRPSGD) that can achieve the same privacy guarantee as DP-GD with reduced runtime complexity, but with slightly worse utility bounds. In this paper, we propose a differentially private Stochastic Recursive Momentum (DP-SRM) algorithm for nonconvex ERM. At the core of our algorithm is the stochastic recursive momentum technique (CO19) that can consistently reduce the accumulated variance of the gradient estimator. Our approach is more scalable than stochastic variance reduced algorithms (JZ13; RHS16; AH16; LJC17; NLS17; FLL18; ZXG18) since it eliminates the periodical computation of the checkpoint gradient which usually requires a giant batch size.

Contributions. We develop a new differentially private stochastic optimization algorithm for nonconvex ERM and provide a sharp analysis of the privacy guarantee using Rényi Differential Privacy (RDP) (Mir17) (Section 4.6). Our algorithm matches the best-known utility guarantee for nonconvex optimization, with lower computational complexity. To achieve the same utility guarantee, the gradient complexity (i.e., the number of stochastic gradients calculated in total) of our algorithm is $O\big(n^{3/2}\big)$, which outperforms the best previous results (ZZM17b; WYX17b) by a factor of $\Theta(n^{1/2})$. We evaluate our proposed methods on two nonconvex ERM techniques: nonconvex logistic regression and convolutional neural networks. We report on experiments on several benchmark datasets (Section 4.7), finding that our method not only produces models that are the closest to the non-private models in terms of model accuracy but also reduces the computational cost.

Notation. We use curly symbol such as $\mathcal{B}$ to denote the index set. For a set $\mathcal{B}$, we use $|\mathcal{B}|$ to denote its cardinality. For a finite sum function $F = \sum_{i=1}^{n} f_i/n$, we denote $F_{\mathcal{B}}$ by $\sum_{i\in\mathcal{B}} f_i/|\mathcal{B}|$. For a d-dimensional vector $\mathbf{x} \in \mathbb{R}^d$, we use $\|\mathbf{x}\|_2$ to denote its ℓ_2-norm. Given two sequences $\{a_n\}$ and $\{b_n\}$, if there exists a constant $0 < C < \infty$ such that $a_n \leq Cb_n$, we write

$a_n = O(b_n)$. Besides, if there exist constants $0 < C_1, C_2 < \infty$ such that $C_1 b_n \leq a_n \leq C_2 b_n$, we write $a_n = \Theta(b_n)$. We use n, d to represent the number of training examples and the problem dimension, respectively. We also use the standard notation for (ϵ, δ)-DP where ϵ is the privacy budget and δ is the failure probability.

4.2 Related Work

Over the past decade, many differentially private machine learning algorithms for convex ERM have been proposed. There are three main approaches to achieve differential privacy in such settings, including output perturbation (WLK17; ZZM17b), objective perturbation (CMS11b; KST12; INS19), and gradient perturbation (BST14c; WYX17b; JWE18b). However, other than the methods using gradient perturbation, it is very hard to generalize these methods to nonconvex ERM because of the difficulty in computing the sensitivity for nonconvex ERM. Thus, most differentially private algorithms for nonconvex ERM are based on the gradient perturbation, including our work. The problem with gradient perturbation approaches is that their iterative nature quickly consumes any reasonable privacy budget. Hence, the main challenge is to develop algorithms for nonconvex ERM that can provide sufficient utility while maintaining privacy with high computational efficiency.

Several recent works (ACG16b; PAE16; XLW18) studied deep learning with differential privacy. (ACG16b) proposed a method called moments accountant to keep track of the privacy cost of stochastic gradient descent algorithm during the training process, which provides a strong privacy guarantee. (PAE16) established a Private Aggregation of Teacher Ensembles (PATE) framework to improve the privacy guarantee of deep learning for classification tasks. (XLW18) and (YJS19) investigated the differentially private Generative Adversarial Nets (GAN) with different distance metrics. However, none of these works provide utility guarantees for their algorithms.

Table 4.1 summarizes differentially private nonconvex optimization algorithms that pro-

Table 4.1: Comparison of different (ϵ, δ)-DP algorithms for nonconvex optimization. We report the utility bound in terms of $\mathbb{E}\|\nabla F(\boldsymbol{\theta}^p)\|_2$, where $\boldsymbol{\theta}^p$ is the output of the differentially private algorithm, d is the problem dimension, $\mathbb{E}$ is taken over the randomness of the algorithm.

Algorithm	Utility	Gradient Complexity
RRPSGD (ZZM17b)	$O\Big(\frac{(d\log(n/\delta)\log(1/\delta))^{1/4}}{(n\epsilon)^{1/2}}\Big)$	$O\big(n^2\big)$
DP-GD (WYX17b)	$O\Big(\frac{(d\log(1/\delta))^{1/4}}{(n\epsilon)^{1/2}}\Big)$	$O\Big(\frac{n^2\epsilon}{d^{1/2}}\Big)$
DP-SRM (This paper)	$O\Big(\frac{(d\log(1/\delta))^{1/4}}{(n\epsilon)^{1/2}}\Big)$	$O\Big(\frac{(n\epsilon)^{3/2}}{d^{3/4}}\Big)$

vide utility guarantees for nonconvex ERM. The Random Round Private Stochastic Gradient Descent (RRPSGD) method developed by (ZZM17b) is the first differentially private nonconvex optimization algorithm with the utility guarantee. This method performs the perturbed SGD (adding Gaussian noise to the stochastic gradients), for a random number of iterations (GL13). The gradient complexity of RRPSGD is $O(n^2)$, which makes it impractical for most settings. (ZZM17b) showed that RRPSGD is able to find a stationary point in expectation with a diminishing error $O\big((d\log(n/\delta)\log(1/\delta))^{1/4}/(n\epsilon)^{1/2}\big)$. Their analysis of the privacy guarantee is based on the standard privacy-amplification by subsampling result and strong composition theorem (BST14c). Although such an analysis can be easily adapted to the nonconvex setting with stochastic optimization algorithms, it results in a large bound on the variance of the added noise compared with relaxed definitions such as the moments accountant (ACG16b) and Gaussian differential privacy (DRS19).

(WYX17b) proposed the Differentially Private Gradient Descent (DP-GD) algorithm for nonconvex optimization. DP-GD has a comparable gradient complexity $O\big(n^2\epsilon/d^{1/2}\big)$, and an

improved utility guarantee $O\big((d\log(1/\delta))^{1/4}/(n\epsilon)^{1/2}\big)$ compared with that of RRPSGD. The reason DP-GD can achieve this factor of $O\big((\log(n/\delta))^{1/4}\big)$ improvement, is that it uses the full gradient rather than the stochastic gradient. This makes DP-GD computationally very expensive or even intractable for large-scale machine learning problems (n is big). Recently, (WCX19) also proposed a differentially private stochastic algorithm for nonconvex optimization. Their goal is to find the local minima, while we aim to find the stationary point. In addition, their utility guarantee is asymptotic—it provides the desired utility guarantee only if an infinite number of iterations could be run. In contrast, our utility guarantee holds for a finite number of iterations.

4.3 Preliminaries

We consider the empirical risk minimization (ERM) problem: given a training set $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ drawn from some unknown but fixed data distribution with $\mathbf{x}_i \in \mathbb{R}^D, y_i \in \mathcal{Y} \subseteq \mathbb{R}$, we aim to find a solution $\widehat{\boldsymbol{\theta}} \in \mathbb{R}^d$ that minimizes the following empirical risk,

$$F(\boldsymbol{\theta}) := \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{\theta}), \tag{4.3.1}$$

where $F(\boldsymbol{\theta})$ is the empirical risk function (i.e., training loss), $f_i(\boldsymbol{\theta}) = \ell(\boldsymbol{\theta}; \mathbf{x}_i, y_i)$ is the loss function defined on the i-th training example $(\mathbf{x}_i, y_i)$, and $\boldsymbol{\theta} \in \mathbb{R}^d$ is the model parameter we want to learn.

Here, we provide some definitions and lemmas that will be used in our theoretical analysis.

Definition 4.3.1. $\boldsymbol{\theta} \in \mathbb{R}^d$ is an ζ-approximate stationary point if $\|\nabla f(\boldsymbol{\theta})\|_2 \leq \zeta$.

Definition 4.3.2. A function $f : \mathbb{R}^d \to \mathbb{R}$ is G-Lipschitz, if for all $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \mathbb{R}^d$, we have $|f(\boldsymbol{\theta}_1) - f(\boldsymbol{\theta}_2)| \leq G\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2$.

Definition 4.3.3. A function $f : \mathbb{R}^d \to \mathbb{R}$ has L-Lipschitz gradient, if for all $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \mathbb{R}^d$, we have $\|\nabla f(\boldsymbol{\theta}_1) - \nabla f(\boldsymbol{\theta}_2)\|_2 \leq L\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2$.

Differential privacy provides a formal notion of privacy, introduced by (DMN06):

Definition 4.3.4 ((ϵ, δ)-DP (DMN06))**.** A randomized mechanism $\mathcal{M} : \mathcal{S}^n \to \mathcal{R}$ satisfies (ϵ, δ)-differential privacy if for any two adjacent data sets $S, S' \in \mathcal{S}^n$ differing by one element, and any output subset $O \subseteq \mathcal{R}$, it holds that $\mathbb{P}[\mathcal{M}(S) \in O] \leq e^{\epsilon} \cdot \mathbb{P}[\mathcal{M}(S') \in O] + \delta$.

To achieve (ϵ, δ)-DP for a given function $q : \mathcal{S}^n \to \mathcal{R}$, we can use Gaussian mechanism (DR14) $\mathcal{M} = q(S) + \mathbf{u}$, where $\mathbf{u}$ is a standard Gaussian random vector with variance that is proportional to the ℓ_2-sensitivity of the function q, $\Delta(q)$, which is defined as follows.

Definition 4.3.5 (ℓ_2-sensitivity(DR14))**.** For two adjacent datasets $S, S' \in \mathcal{S}^n$ differing by one element, the ℓ_2-sensitivity $\Delta(q)$ of a function $q : \mathcal{S}^n \to \mathcal{R}$ is defined as $\Delta(q) = \sup_{S,S'} \|q(S) - q(S')\|_2$.

Rényi differential privacy. Although the notion of (ϵ, δ)-DP is widely used in the output and objective perturbation methods, it suffers from the loose composition and privacy-amplification by subsampling results, which makes it unsuitable for the stochastic iterative learning algorithms. In this work, we will make use of the notion of Rényi Differential Privacy (RDP) (Mir17) which is particularly useful when the dataset is accessed by a sequence of randomized mechanisms (WBK19).

Definition 4.3.6 (RDP (Mir17))**.** For $\alpha > 1, \rho > 0$, a randomized mechanism $\mathcal{M} : \mathcal{S}^n \to \mathcal{R}$ satisfies (α, ρ)-Rényi differential privacy, i.e., (α, ρ)-RDP, if for all adjacent datasets $S, S' \in \mathcal{S}^n$ differing by one element, we have $D_\alpha\big(\mathcal{M}(S)||\mathcal{M}(S')\big) := \log \mathbb{E}\big[\big(\mathcal{M}(S)/\mathcal{M}(S')\big)^\alpha\big]/(\alpha - 1) \leq \rho$.

According to Definition 4.3.6, RDP measures the ratio of probability distributions $\mathcal{M}(S)$ and $\mathcal{M}(S')$ by α-order Renyi Divergence with $\alpha \in (1, \infty)$. As $\alpha \to \infty$, RDP reduces to ϵ-DP.

To further improve the privacy guarantee when using the Gaussian mechanisms to satisfy RDP, we establish the following privacy-amplification by subsampling result, which is derived based on the result in (WBK19).

Lemma 4.3.7. Given a function $q : \mathcal{S}^n \to \mathcal{R}$, the Gaussian Mechanism $\mathcal{M} = q(S)+\mathbf{u}$, where $\mathbf{u} \sim N(0, \sigma^2\mathbf{I})$, satisfies $(\alpha, \alpha\Delta^2(q)/(2\sigma^2))$-RDP. In addition, if we apply the mechanism $\mathcal{M}$ to a subset of samples using uniform sampling without replacement with sampling rate τ, $\mathcal{M}$ satisfies $(\alpha, 3.5\tau^2\Delta^2(q)\alpha/\sigma^2)$-RDP given $\sigma'^2 = \sigma^2/\Delta^2(q) \geq 0.7$, $\alpha \leq 2\sigma^2 \log(1/\tau\alpha\big(1 + \sigma'^2)\big)/3 + 1$.

Remark 4.3.8 (*Comparison with moment accountant*)**.** Suppose $\Delta(q) = 1$, Lemma 4.3.7 suggests that to achieve $(\alpha, 3.5\tau^2\alpha/\sigma^2)$-RDP of the subsampled Gaussian mechanism, we require $\sigma^2 \geq 0.7$. For the moment accountant based method (ACG16b), it can achieve the asymptotic privacy guarantee of $\big(\alpha, \tau^2\alpha/(1-\tau)\sigma^2 + O(\tau^3\alpha^3/\sigma^3)\big)$-RDP when τ goes to zero and $\sigma^2 \geq 1$, $\alpha \leq \sigma^2 \log(1/\tau\sigma)$. In contrast to moment accountant, our result has a closed-form bound on the privacy guarantee and a relaxed requirement of σ^2.

It is worth noting that there exist some other works (MTZ19; ZW19) also studying the privacy-amplification by subsampling results. However, they consider the Poisson subsampling approach, which is different from our uniform subsampling method.

Based on Lemma 4.3.7, we can establish a strong privacy guarantee of our method in terms of RDP, and then transfer it to (ϵ, δ)-DP using the following lemma.

Lemma 4.3.9 ((Mir17))**.** If a randomized mechanism $\mathcal{M} : \mathcal{S}^n \to \mathcal{R}$ satisfies (α, ρ)-RDP, then $\mathcal{M}$ satisfies $(\rho + \log(1/\delta)/(\alpha - 1), \delta)$-DP for all $\delta \in (0, 1)$.

4.4 Algorithm

Our proposed algorithm for differentially private nonconvex ERM, is illustrated in Algorithm 4.

The main idea is to construct the differentially private gradient estimator $\mathbf{v}_p^t$ iteratively based on the information obtained from the previous updates. We initialize $\mathbf{v}^0$ to be the mini-batch stochastic gradient $\nabla F_{\mathcal{B}_0}(\boldsymbol{\theta}^0)$ and inject Gaussian noise, $\mathbf{u}^0$, with covariance matrix

Algorithm 4 Differentially Private Stochastic Recursive Momentum (DP-SRM)

input $\boldsymbol{\theta}^0, T, G, L, \gamma, \beta, n_0$, privacy parameters ϵ, δ, accuracy for the first-order stationary point ζ

1: Uniformly sample b_0 examples without replacement indexed by $\mathcal{B}_0$

2: Compute $\mathbf{v}^0 = \nabla F_{\mathcal{B}_0}(\boldsymbol{\theta}^0)$, where $\nabla F_{\mathcal{B}_0}(\boldsymbol{\theta}^0) = \sum_{i \in \mathcal{B}_0} \nabla f_i(\boldsymbol{\theta}^0)/b_0$, draw $\mathbf{u}^0 \sim N(0, \sigma_0^2 \mathbf{I}_d)$ with $\sigma_0^2 = 14TG^2\alpha/(\beta n^2 \epsilon)$, $\alpha = \log(1/\delta)/\big((1-\beta)\epsilon\big) + 1$

3: Release the differentially private gradient estimator $\mathbf{v}_p^0 = \mathbf{v}^0 + \mathbf{u}^0$

4: **for** $t = 0, 1, 2, \ldots, T-1$ **do**

5: $\quad \boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t - \eta_t \mathbf{v}_p^t$, where $\eta_t = \min\big\{\zeta/(n_0 L \|\mathbf{v}_p^t\|_2), 1/(2n_0 L)\big\}$

6: $\quad$ Uniformly sample b examples without replacement indexed by $\mathcal{B}_{t+1}$

7: $\quad$ Compute $\mathbf{v}^{t+1} = \nabla F_{\mathcal{B}_{t+1}}(\boldsymbol{\theta}^{t+1}) + (1-\gamma)\big(\mathbf{v}_p^t - \nabla F_{\mathcal{B}_{t+1}}(\boldsymbol{\theta}^t)\big)$, draw $\mathbf{u}^{t+1} \sim N(0, \sigma^2 \mathbf{I}_d)$ with $\sigma^2 = 14T\big((1-\gamma)\zeta/n_0 + \gamma G\big)^2 \alpha/(\beta n^2 \epsilon)$, $\alpha = \log(1/\delta)/\big((1-\beta)\epsilon\big) + 1$

8: $\quad$ Release the differentially private gradient estimator $\mathbf{v}_p^{t+1} = \mathbf{v}^{t+1} + \mathbf{u}^{t+1}$

9: **end for**

output $\widetilde{\boldsymbol{\theta}}$ chosen uniformly at random from $\{\boldsymbol{\theta}^t\}_{t=0}^{T-1}$

$\sigma_0^2\mathbf{I}_d$ (lines 2, 3), to make it differentially private. Then, we recursively update $\mathbf{v}^t$ (line 7) as $\mathbf{v}^t = \nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^t) + (1-\gamma)\big(\mathbf{v}_p^{t-1} - \nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^{t-1})\big)$, where $\nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^t)$, $\nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^{t-1})$ are mini-batch stochastic gradients and $\mathbf{v}_p^{t-1}$ is the private gradient estimator released at the last iteration. The momentum parameter, γ, is used to control the decay rate of the prior information, $\mathbf{v}_p^{t-1} - \nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^{t-1})$. This is called stochastic recursive momentum (CO19), which can lead to fast convergence. After updating $\mathbf{v}^t$, we again inject Gaussian noise $\mathbf{u}^t$ with covariance matrix $\sigma_2\mathbf{I}_d$ (line 8), to provide differential privacy. The variance σ_0^2, σ^2 of the Gaussian random vectors are determined by our RDP-based analysis. We choose an adaptive step size (line 5) to bound the sensitivity of the gradient estimator $\mathbf{v}_p^t$, which is the key to establish the tight privacy and utility guarantees (Section 4.6) of our algorithm.

4.5 Main Results

In this section, we establish formal privacy and utility guarantees for Algorithm 4.

Theorem 4.5.1. Suppose that each component function f_i is G-Lipschitz and has L-Lipschitz gradient. Given the total number of iterations T, the momentum parameter γ and the accuracy for the first-order stationary point ζ, for any $\delta > 0$ and the privacy budget ϵ, Algorithm 4 satisfies (ϵ, δ)-differential privacy with $\sigma_0^2 = 14TG^2\alpha/(\beta n^2\epsilon)$ and $\sigma^2 = 14T\big((1-\gamma)\zeta/n_0 + \gamma G\big)^2\alpha/(\beta n^2\epsilon)$ if we have $\alpha - 1 = \log(1/\delta)/\big((1-\beta)\epsilon\big) \leq 2\sigma'^2\log\big(1/\tau\alpha(1+\sigma'^2)\big)/3$ with $\beta \in (0,1)$ and $\sigma'^2 = \min\{b^2\sigma^2/\big(4((1-\gamma)\zeta/n_0 + \gamma G)^2\big), b_0\sigma_0^2/(4G^2)\} \geq 0.7$, where b_0 and b are batch sizes, and $\tau = \max\{b_0/n, b/n\}$.

Remark 4.5.2. According to Theorem 4.5.1, there exists a constraint on the parameter α, which is due to the privacy-amplification by subsampling result in Lemma 4.3.7, and is similar to the constraint given by the moments accountant (ACG16b) and other RDP-based analyses with subsampling approaches (MTZ19; ZW19). Furthermore, as we mentioned in Remark 4.3.8, our result relaxes the requirement of the variance σ'^2 compared with the moments accountant based analysis.

Following the previous work (BFT19), we can get rid of the constraints in Theorem 4.5.1 by using the larger mini-batch size, as states in the following corollary.

Corollary 4.5.3. Given the total number of iterations T, the momentum parameter γ and the accuracy for the first-order stationary point ζ. Under the same conditions of Theorem 4.5.1 on $f_i, \sigma_0^2, \sigma^2$, for any $\delta > 0$ and the privacy budget ϵ, Algorithm 4 satisfies (ϵ, δ)-differential privacy if we choose $b_0^2 = b^2 = n^2\epsilon/T$, $\beta = 1/2$, and T is larger than $O\big(\log^4(1/\delta)/\epsilon^3\big)$.

Theorem 4.5.1 requires that each component function f_i is G-Lipschitz and has L-Lipschitz gradient which will be used to derive the sensitivity of the underlying query function (i.e., the gradient estimator $\mathbf{v}^t$ in Algorithm 4) and thus determine the variance of the Gaussian noise. The G-Lipschitz condition has been widely assumed in the literature of differential privacy (ACG16b; WYX17b; JWE18b; BFT19), and the L-Lipschitz gradient condition has also been made in several previous works (ZZM17b; FKT20). In practice, we can use the clipping technique (ACG16b) to ensure that at each iteration, $\|\nabla f_i(\boldsymbol{\theta}^t)\|_2 \leq C_1$ and $\|\nabla f_i(\boldsymbol{\theta}^t) - \nabla f_i(\boldsymbol{\theta}^{t-1})\|_2 \leq C_2$, where C_1, C_2 are some predefined constants. As a result, we can guarantee that the sensitivity of $\mathbf{v}^t$ is bounded by $2\big((1-\gamma)C_2 + \gamma C_1\big)/b$ (see (4.6.1)). Then, we can replace G and ζ/n_0 with C_1 and C_2 in Algorithm 4 to establish the same privacy guarantee.

The following theorem shows the utility guarantee and the gradient complexity, which is the total number of the stochastic gradients we need to estimate during the training process, of Algorithm 4.

Theorem 4.5.4. Under the same conditions of Theorem 4.5.1 on $f_i, \sigma^2, \sigma_0^2, \alpha$, if we choose the number of iterations $T = C_1 n\epsilon\sqrt{LD_F}/\big(G\sqrt{d\log(1/\delta)}\big)$, where $D_F = F(\boldsymbol{\theta}^0) - F(\boldsymbol{\theta}^*)$ and $F(\boldsymbol{\theta}^*)$ is a global minimum of F, the accuracy for the first-order stationary point $\zeta = C_2 G^{1/2}\big(LD_F d\log(1/\delta)\big)^{1/4}/(n\epsilon)^{1/2}$, batch sizes $b_0 = C_3G^2/(\zeta LD_F)$, $b = C_4G^2/(n_0\zeta)$, $n_0 = LD_F/\zeta$ and the momentum parameter $\gamma = C_5\zeta/n_0$, then the output $\widetilde{\boldsymbol{\theta}}$ of Algorithm 4 satisfies

the following

$$\mathbb{E}\|\nabla F(\widetilde{\boldsymbol{\theta}})\|_2 \leq C_6 G^{1/2}\big(LD_F d\log(1/\delta)\big)^{1/4}/(n\epsilon)^{1/2},$$

where $\{C_i\}_{i=1}^6$ are absolute constants, and the expectation is taken over all the randomness of the algorithm, i.e., the random Gaussian noise and the subsample gradient. Since we have $T = O\big(n\epsilon/d^{1/2}\big)$, $b_0 = O\big((n\epsilon)^{1/2}/d^{1/4}\big)$ and $b = O(1)$, the total gradient complexity of Algorithm 4 is $O\big((n\epsilon)^{3/2}/d^{3/4} + (n\epsilon)^{1/2}/d^{1/4}\big)$.

Remark 4.5.5 (*Comparison with existing methods*)**.** According to Theorem 4.5.4, our method can achieve the following utility guarantee

$$O\bigg(\frac{G^{1/2}(LD_F d\log(1/\delta))^{1/4}}{(n\epsilon)^{1/2}}\bigg).$$

This result matches the best known result for differentially private nonconvex optimization method (WYX17b). However, there method is based on gradient descent, which is computationally very expensive in large-scale machine learning problems. Furthermore, the gradient complexity of our method is

$$O\bigg(\frac{(n\epsilon)^{3/2}}{d^{3/4}} + \frac{(n\epsilon)^{1/2}}{d^{1/4}}\bigg).$$

This result is smaller than $O(n^2)$ gradient complexity provided by (ZZM17b) and $O\big(n^2\epsilon/d^{1/2}\big)$ gradient complexity provided by (WYX17b).

Theorem 4.5.4 shows that our method only requires the computation of a large batch gradient with batch size $b_0 = O\big((n\epsilon)^{1/2}/d^{1/4}\big)$ at the beginning. Therefore, our method is more scalable than existing differentially private stochastic variance reduced algorithms, such as DP-SVRG (WYX17b) designed for convex optimization, which often require the periodic computation of the checkpoint gradient with a giant batch size (full batch in DP-SVRG).

4.6 Proof Outline of the Main Results

In this section, we present the proof outline of the main results in Section 4.5. Our proof involves new techniques for the privacy and utility guarantees that are of general use for variance reduction-based algorithms. The detailed proof can be found in Section B in the supplemental material.

4.6.1 Privacy Guarantee

According to Algorithm 4, the mechanism at t-th iteration is $\mathcal{M}_t$, which is a composition of t Gaussian mechanisms: $\mathcal{G}_0, \ldots, \mathcal{G}_t$, where $\mathcal{G}_0 = \nabla F_{\mathcal{B}_0}(\boldsymbol{\theta}^0) + \mathbf{u}^0$ and $\mathcal{G}_t = \nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^t) - (1-\gamma)\nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^{t-1}) + \mathbf{u}^t$. Therefore, we want to show that $\mathcal{M}_t$ is differentially private. For the given dataset S, we use S' to denote its neighboring dataset with one different example indexed by i'

There are two main challenges in providing a tight privacy analysis. The first one is to deal with the subsampled mechanisms $\{\mathcal{G}_i\}_{i=0}^{T-1}$. The second one is to control the sensitivity of $\mathcal{G}_t$ when $t > 0$. The first challenge can be addressed by our privacy-amplification by subsampling result (Lemma 4.3.7), which gives us a tight closed-form bound on the privacy guarantee. We can overcome the second challenge by using an adaptive stepsize, enabling us to use a small amount of random noise to achieve differential privacy.

According to Algorithm 4, $\mathcal{G}_t$ is the application of the following Gaussian mechanism $\widetilde{\mathcal{G}}_t$ to a subset of uniformly sampled examples, indexed by $\mathcal{B}_t$

$$\widetilde{\mathcal{G}}_t = \begin{cases} \frac{1}{b}\sum_{i=1}^{n} \nabla f_i(\boldsymbol{\theta}^0) + \mathbf{u}^0, & t = 0 \\ \frac{1}{b}\sum_{i=1}^{n} \left(\nabla f_i(\boldsymbol{\theta}^t) - \phi \nabla f_i(\boldsymbol{\theta}^{t-1})\right) + \mathbf{u}^t, & t > 0, \end{cases}$$

where $\phi = 1 - \gamma$. For $\widetilde{\mathbf{q}}_0 = \sum_{i=1}^{n} \nabla f_i(\boldsymbol{\theta}^0)/b_0$ in $\widetilde{\mathcal{G}}_0$, the sensitivity $\Delta(\widetilde{\mathbf{q}}_0)$ is determined by

$$\|\widetilde{\mathbf{q}}_0(S) - \widetilde{\mathbf{q}}_0(S')\|_2 \leq \frac{1}{b}\|\nabla f_i(\boldsymbol{\theta}^0) - \nabla f_{i'}(\boldsymbol{\theta}^0)\|_2 \leq \frac{2G}{b_0},$$

where the last inequality is due to G-Lipschitz of each component function. For $\widetilde{\mathbf{q}}_t = \sum_{i=1}^n \nabla f_i(\boldsymbol{\theta}^t)/b - (1-\gamma)\sum_{i=1}^n \nabla f_i(\boldsymbol{\theta}^{t-1})/b$ in $\widetilde{\mathcal{G}}_t$ when $t > 0$, the sensitivity $\Delta(\widetilde{\mathbf{q}}_t) = \|\widetilde{\mathbf{q}}_t(S) - \widetilde{\mathbf{q}}_t(S')\|_2$ is determined by

$$\frac{1-\gamma}{b}\|\nabla f_i(\boldsymbol{\theta}^t) - \nabla f_i(\boldsymbol{\theta}^{t-1}) + \nabla f_{i'}(\boldsymbol{\theta}^t) - \nabla f_{i'}(\boldsymbol{\theta}^{t-1})\|_2 + \frac{\gamma}{b}\|\nabla f_i(\boldsymbol{\theta}^t) - \nabla f_{i'}(\boldsymbol{\theta}^t)\|_2. \quad (4.6.1)$$

Therefore, we have

$$\begin{aligned}\|\mathbf{q}_t(S) - \mathbf{q}_t(S')\|_2 &\leq \frac{2L(1-\gamma)}{b}\|\boldsymbol{\theta}^t - \boldsymbol{\theta}^{t-1}\|_2 + \frac{2\gamma G}{b} \\ &= \frac{2L(1-\gamma)}{b}\eta_{t-1}\|\mathbf{v}_p^{t-1}\|_2 + \frac{2\gamma G}{b} \\ &\leq \frac{2(1-\gamma)\zeta}{n_0 b} + \frac{2\gamma G}{b},\end{aligned}$$

where the first inequality is due to L-Lipschitz continuous gradient and G-Lipschitz of each component function. The last inequality comes from the adaptive stepsize we choose as follows $\eta_t = \min\big\{\zeta/(n_0 L\|\mathbf{v}_p^t\|_2), 1/(2n_0 L)\big\}$. Note that the proposed adaptive stepsize η_t is the key to control the sensitivity of $\widetilde{\mathbf{q}}_t$. If we choose a fixed stepsize such as $\eta_t = 1/(2L)$, the sensitivity of $\widetilde{\mathbf{q}}_t$ will be in the order of $O(G^2/b)$, which will lead to a much larger random noise to achieve differential privacy and thus deteriorate the utility of our method.

According to Lemma 4.3.7, if the noise $\mathbf{u}^0$ and $\mathbf{u}^t$ satisfy $\sigma_0^2 = 14T\alpha G^2/(\beta n^2\epsilon)$ and $\sigma^2 = 14T\alpha\big((1-\gamma)\zeta/n_0 + \gamma G\big)^2/(\beta n^2\epsilon)$, the Gaussian mechanism $\widetilde{\mathcal{G}}_t$ satisfies $\big(\alpha, \beta\epsilon n^2/\big(7b_0^2 T\big)\big)$-RDP, and the privacy-amplification by subsampling result shows that $\mathcal{G}_t$ satisfies $(\alpha, \beta\epsilon/T)$-RDP. Therefore, by the composition rule of RDP (Mir17), after T' iterations, Algorithm 4 satisfies $(\alpha, \beta T'\epsilon/T)$-RDP. According to Lemma 4.3.9 and $\alpha = \log(1/\delta)/\big((1-\beta)\epsilon\big) + 1$, we have that after T' iterations, Algorithm 4 satisfies $(T'\epsilon/T, \delta)$-DP.

4.6.2 Utility Guarantee

According to the definition of $\widetilde{\boldsymbol{\theta}}$, we have

$$\mathbb{E}\|\nabla F(\widetilde{\boldsymbol{\theta}})\|_2 = \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla F(\boldsymbol{\theta}^t)\|_2 \leq \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\big\|\mathbf{v}_p^t\big\|_2 + \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\big\|\nabla F(\boldsymbol{\theta}^t) - \mathbf{v}_p^t\big\|_2,$$

where the expectation is taken over all the randomness of the algorithm. The key challenge in establishing a tight utility guarantee is to derive tight upper bounds for $\sum_{t=0}^{T-1}\mathbb{E}\left\|\mathbf{v}_p^t\right\|_2/T$ and $\sum_{t=0}^{T-1}\mathbb{E}\left\|\nabla F(\boldsymbol{\theta}^t)-\mathbf{v}_p^t\right\|_2/T$ when we have adaptive stepsize η_t and the random noise $\mathbf{u}^t$ in $\mathbf{v}_p^t$.

First of all, by taking into account the adaptive stepsize η_t, we can upper bound the term $\sum_{t=0}^{T-1}\mathbb{E}\left\|\mathbf{v}_p^t\right\|_2/T$ as follows

$$\frac{4n_0LD_F}{T\zeta}+\frac{1}{T\zeta}\sum_{t=0}^{T-1}\mathbb{E}\left\|\nabla F(\boldsymbol{\theta}^t)-\mathbf{v}_p^t\right\|_2^2+2\zeta,$$

where $D_F=F(\boldsymbol{\theta}^0)-F(\boldsymbol{\theta}^*)$. Furthermore, we can obtain the upper bound for $\sum_{t=0}^{T-1}\mathbb{E}\left\|\mathbf{v}_p^t-\nabla F(\boldsymbol{\theta}^t)\right\|_2^2/T$ as follows

$$\frac{2(1-\gamma)^2\zeta^2}{n_0^2\gamma b}+\frac{2\gamma G^2}{b}+\frac{G^2}{T\gamma b_0}+\frac{Td\sigma^2+d\sigma_0^2}{T\gamma},$$

where the first term is determined by η_t, and the last term is determined by the random noise $\mathbf{u}^t$ in $\mathbf{v}_p^t$. The last term in this bound is dominated by $d\sigma^2/\gamma$, which validates the necessity of using the adaptive stepsize to control the sensitivity of $\mathbf{v}^t$ and thus enable a small σ^2.

Finally, combining these two new bounds and plugging the value of parameters in Theorem 4.5.4, we can obtain that

$$\mathbb{E}\|\nabla F(\widetilde{\boldsymbol{\theta}})\|_2\leq C_1\zeta+C_2\frac{\sqrt{LD_Fd\log(1/\delta)}G}{n\epsilon\zeta}.$$

By solving the smallest ζ, we can obtain $\zeta=(LD_Fd\log(1/\delta))^{1/4}(C_2G)^{1/2}/(C_1n\epsilon)^{1/2}$. Thus we have $\mathbb{E}\|\nabla F(\widetilde{\boldsymbol{\theta}})\|_2\leq C_3\zeta$, where C_1,C_2,C_3 are some constants.

4.7 Experiments

This section presents results from experiments that evaluate our method's performance on different nonconvex ERM problems and different datasets. All experiments are implemented in Pytorch platform version 1.2.0 within Python 3.7.6. on a local machine which comes with Intel Xeon 4214 CPUs and NVIDIA GeForce RTX 2080Ti GPU (11G GPU RAM).

4.7.1 Nonconvex Logistic Regression

We first consider the binary logistic regression problem with a nonconvex regularizer (RSP16)

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n} y_i \log \phi(\mathbf{x}_i^\top \boldsymbol{\theta}) + (1-y_i)\log\big[1-\phi(\mathbf{x}_i^\top \boldsymbol{\theta})\big] + \lambda \sum_{i=1}^{d} \theta_j^2/(1+\theta_j^2),$$

where $\phi(x) = 1/\big(1+\exp(-x)\big)$ is the sigmoid function, θ_j is the j-th coordinate of $\boldsymbol{\theta}$, and $\lambda > 0$ is the regularization parameter. We set $\lambda = 0.001$ in this experiment. Here, we report results for the *a9a* dataset, a commonly-used binary classification benchmark with 32561 training examples, 16281 test examples, and 123 features. We found similar results for the *ijcnn1* dataset (49990 training examples, 91701 test examples, 22 features), which are presented in Section A in the supplemental material.

Baseline methods. We compare our method (DP-SRM) with random round private stochastic gradient descent (RRPSGD) proposed by (ZZM17b) , differentially private gradient descent (DP-GD) proposed by (WYX17b), and differentially private adaptive gradient descent (DP-AGD) proposed by (LK18b).

Gradient clipping and privacy tracking. We use the gradient clipping technique of (ACG16b) to ensure that at t-th iteration of Algorithm 4, $\|\nabla f_i(\theta^t)\|_2$ and $\|\nabla f_i(\theta^t) - \nabla f_i(\theta^{t-1})\|_2$ are upper bounded by some predefined values C_1 and C_2, respectively. This will ensure that the sensitivity of the gradient estimator $\mathbf{v}^t$ is upper bounded by $2\big((1-\gamma)C_1+\gamma C_2\big)$ (see (4.6.1)), and gives us the desired privacy protection. At each iteration, we add the Gaussian noise with variance σ^2, and keep track of the RDP according to Lemma 4.3.7 and transfer it to (ϵ,δ)-DP according to Lemma 4.3.9.

Parameters. For all the algorithms, the step size is tuned around the theoretical values to give the fastest convergence using grid search. For our method, we tune the batch size b by searching the grid $\{50, 100, 200\}$. We set $C_1 = 1, C_2 = 0.01$ and $\gamma = C_2$. We choose $\epsilon \in \{0.2, 0.5\}$ and $\delta = 10^{-5}$.

Results. Due to the randomized nature of all the algorithms, the experimental results are

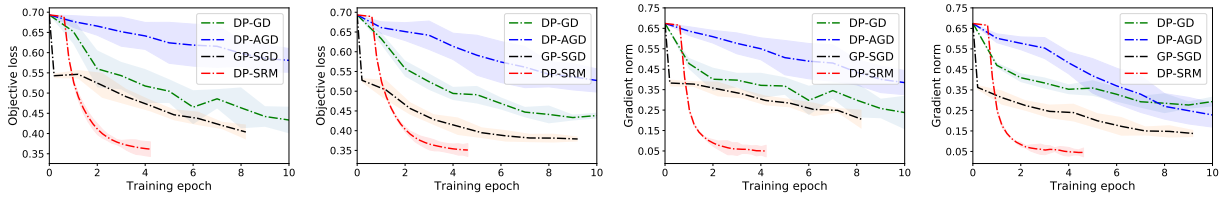
Table 4.2: Comparison of different algorithms on *a9a* dataset when $\epsilon \in \{0.2, 0.5\}$ and $\delta = 10^{-5}$. We use the STORM algorithm (CO19) as the non-private baseline.

Privacy Budget	Non-private Baseline	Method	Test Error	Data Passes	CPU time (s)	Gradient Norm
$\epsilon = 0.2$	0.3346 (0.007)	DP-GD	0.4155 (0.0107)	20	1.245	0.0953 (0.0212)
		DP-AGD	0.3713 (0.0043)	360	96.21	0.0437 (0.0020)
		RRPSGD	0.4019 (0.0033)	8	39.61	0.2175 (0.0116)
		DP-SRM	**0.3579 (0.0009)**	**4**	**0.6007**	**0.0528 (0.0042)**
$\epsilon = 0.5$	0.3346 (0.007)	DP-GD	0.3859 (0.0057)	20	1.261	0.0866 (0.0129)
		DP-AGD	0.3627 (0.0038)	365	95.45	0.0402 (0.0022)
		RRPSGD	0.3861 (0.0028)	10	52.32	0.1454 (0.0126)
		DP-SRM	**0.3506 (0.0011)**	**5**	**0.7383**	**0.0502 (0.0061)**

obtained by averaging the results over 30 runs. Figures 4.1 shows the objective function value and the gradient norm of different algorithms for privacy budgets $\epsilon \in \{0.2, 0.5\}$ on *a9a* datasets. We also report the 95% confidence interval of these results. We can see from the plots that Our DP-SRM algorithm outperforms the other three baseline algorithms in terms of the objective loss, gradient norm, and convergence rate by a large margin. Table 4.2 summarizes the test error of different algorithms as well as the CPU time (in seconds) of the training process. The results also corroborate the advantages of our method in terms of accuracy and efficiency.

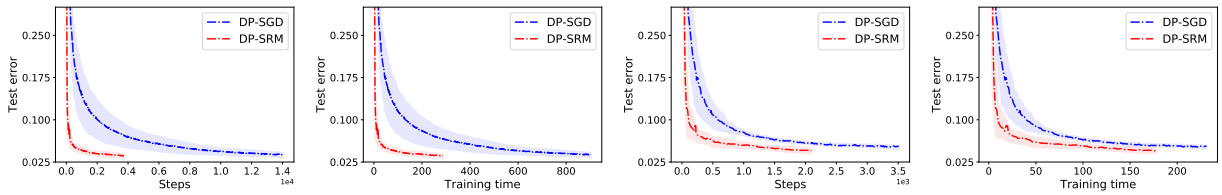
4.7.2 Convolutional Neural Networks

We compare our algorithm with the differentially private stochastic gradient descent (DP-SGD) algorithm proposed by (ACG16b) on training convolutional neural networks for image classification on both MNIST (LBB98) and CIFAR-10 (Kri09) datasets.



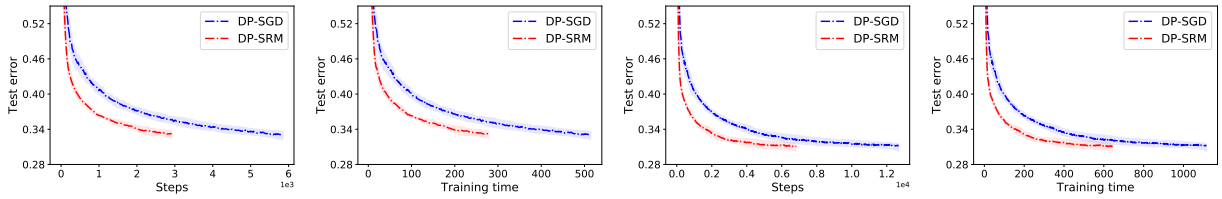
(a) $\epsilon = 0.2$ (b) $\epsilon = 0.5$ (c) $\epsilon = 0.2$ (d) $\epsilon = 0.5$

Figure 4.1: Results for nonconvex logistic regression on *a9a* dataset. (a), (b) illustrate the objective loss versus the number of epochs. (c), (d) present the gradient norm versus the number of epochs.



(a) $\epsilon = 3.0$ (b) $\epsilon = 3.0$ (c) $\epsilon = 1.2$ (d) $\epsilon = 1.2$

Figure 4.2: Results on MNIST dataset. (a), (b) depict the test error under the privacy budget $\epsilon = 3.0$. (c), (d) illustrate the test error under the privacy budget $\epsilon = 1.2$.



(a) $\epsilon = 2.0$ (b) $\epsilon = 2.0$ (c) $\epsilon = 4.0$ (d) $\epsilon = 4.0$

Figure 4.3: Results for CNN6 on CIFAR-10 dataset. (a), (b) depict the test error under the privacy budget $\epsilon = 2.0$. (c), (d) illustrate the test error under the privacy budget $\epsilon = 4.0$

Architecture for MNIST. For MNIST dataset, we consider a 4 layer CNN [1], which can

[1] `https://github.com/facebookresearch/pytorch-dp`.

achieve 99% classification accuracy on the test dataset after training with SGD.

Parameters for MNIST. We choose privacy budgets $\epsilon \in \{1.2, 3.0, 7.0\}$, and set $\delta = 10^{-5}$. To ensure the privacy guarantee (see (4.6.1)), we set the clipping parameter $C_1 = 1.5$ for the term $\|\nabla f_i(\theta^t)\|_2$. For the term $\|\nabla f_i(\theta^t) - \nabla f_i(\theta^{t-1})\|_2$, we choose the clipping parameter C_2 by searching the grid $\{0.01, 0.1, 0.3, 0.5, 0.7, 0.9, 0.99\}$. For both DP-SGD and DP-SRM, we tune the batch size b by searching the grid $\{256, 512, 1024\}$ and the step size by $\{0.01, 0.05, 0.1, 0.25, 0.5\}$. For DP-SRM, we tune the batch size b_0 by $\{b, 2b, 4b\}$. In addition, we set the momentum parameter $\gamma = C_2$.

Results for MNIST. Figures 4.2 illustrates the average test error and the corresponding 95% confidence interval of different methods versus the number of iterations as well as the training time (in seconds) under the privacy budgets $\epsilon = 1.2$ and $\epsilon = 3.0$ over 30 trials. We see similar results under the privacy budget $\epsilon = 7.0$, and thus defer them in Section A in the supplemental material. The CNN trained by the non-private SGD can achieve 1% test error after 20 epochs. Figure 4.2(a) and Figure 4.2(c) show that our proposed method can achieve 3.62% and 4.49% test errors when $\epsilon = 3.0$ and $\epsilon = 1.2$, which are better than DP-SGD with 3.81% and 5.33% test errors. Besides, our method converges faster than DP-SGD. Figure 4.2(a) and Figure 4.2(b) demonstrate that compared with DP-SGD, our method only takes 0.3× iterations and 0.4× training time to achieve comparable performances under the privacy budget $\epsilon = 3.0$.

Architecture for CIFAR-10. We consider two convolutional neural networks for CIFAR-10. The first one is a five layer CNN with two convolutional layers and three fully connected layers, and we call it CNN5 [2]. For CNN5, we train it from the scratch using our DP-SRM method and the DP-SGD method (ACG16b) and compare their performances in terms of the model accuracy, iteration numbers and the training time. For the second one, we consider a similar architecture as in (ACG16b), which has three convolutional layers with 32, 64,

[2] `https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html`.

128 filters in each convolution layer and three fully connected layers, and we denote it by CNN6. For CNN6, we follow the same experiment setting as in (ACG16b): we use CIFAR-100 dataset as a public dataset, and first train a network with the same architecture on this dataset as the pretrained model. Then, we initialize the convolutional layers of CNN6 using the cnvolutional layers of the pretrained model, and only train the fully connected layers of CNN6 on CIFAR-10 dataset using different private methods.

Parameters for CNN6. We choose three different privacy budgets $\epsilon \in \{2.0, 4.0, 8.0\}$ and $\delta = 10^{-5}$. We set the clipping parameter $C_1 = 2$ for the term $\|\nabla f_i(\theta^t)\|_2$. For the term $\|\nabla f_i(\theta^t) - \nabla f_i(\theta^{t-1})\|_2$, we choose the clipping parameter C_2 by searching the grid $\{0.01, 0.05, 0.1, 0.3, 0.5, 0.7, 0.9, 0.95, 0.99\}$. For DP-SGD, we tune the batch size by searching the grid $\{64, 128, 256\}$ and the step size by $\{0.01, 0.02, 0.05, 0.1, 0.15, 0.2\}$. For DP-SRM, we tune the batch size b by searching the grid $\{64, 128, 256\}$, step size by $\{0.01, 0.02, 0.05, 0.1, 0.15, 0.2\}$, and b_0 by $\{b, 2b, 4b\}$. In addition, we set the momentum parameter $\gamma = C_2$.

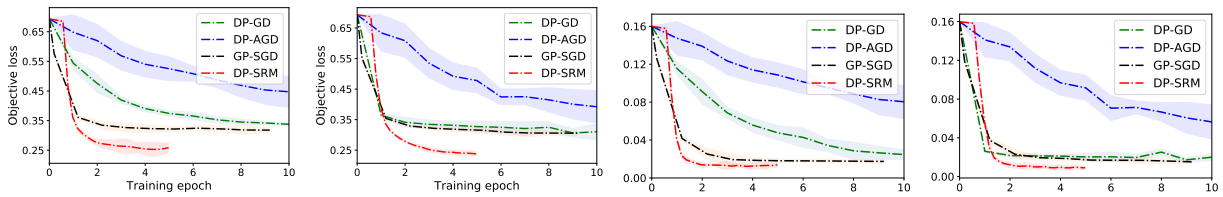
Results for CNN6. Figure 4.3 presents the average test error and the corresponding 95% confidence interval of different methods versus the number of iterations as well as the training time (in seconds) over 30 trials. The CNN6 trained by the non-private SGD will have 18.5% test error after 150 epochs. The results show that our proposed method can achieve 33.2% and 31.0% test errors given $\epsilon = 2.0$ and $\epsilon = 4.0$, which are comparable to the results of DP-SGD with 33.2% and 31.2% under the same privacy budgets. However, we can see from the plots that our method can significantly reduce the iteration numbers and the training time. For example, when $\epsilon = 4.0$, DP-SGD takes 1.3×10^4 iterations and 1115 seconds to achieve 31.2% test error. In sharp contrast, our method only takes 6.8×10^3 iterations and 643 seconds to achieve 31.0% test error. We can observe similar results for CNN5, which are presented in Section A in the supplemental material.

4.8 Additional experiments

In this section, we present additional experiment results on nonconvex logistic regression and convolutional neural networks.

4.8.1 Results on *ijcnn1* dataset

In this subsection, we present the additional experiment of our method on *ijcnn1* dataset. In this dataset, we follow the same settings as before: we set the clipping thresholds $C_1 = 1, C_2 = 0.01$, and set the momentum parameter $\gamma = C_2$. Figures 4.4 illustrates the objective function value and the gradient norm of different algorithms under various privacy budgets $\epsilon \in \{0.2, 0.5\}$. We can see that our proposed algorithm (DP-SRM) outperforms the other three baseline algorithms (RRPSGD, DP-GD, and DP-AGD) in terms of the objective loss, gradient norm, and convergence rate by a large margin. Table 4.3 shows the test error of different algorithms as well as the CPU time (in seconds) of the training process on *ijcnn1* dataset. It demonstrates that our algorithm convergences faster and can achieve a better test error on the test set than other baselines.



(a) $\epsilon = 0.2$ (b) $\epsilon = 0.5$ (c) $\epsilon = 0.2$ (d) $\epsilon = 0.5$

Figure 4.4: Results for nonconvex logistic regression on *ijcnn1* dataset. (a), (b) show the objective loss versus the number of epochs. (c), (d) illustrate the gradient norm versus the number of epochs.

Table 4.3: Comparison of different algorithms on *ijcnn1* dataset under different privacy budgets $\epsilon \in \{0.2, 0.5\}$ and $\delta = 10^{-5}$. Note that the non-private baseline denotes the test error of the non-private STORM algorithm (CO19).

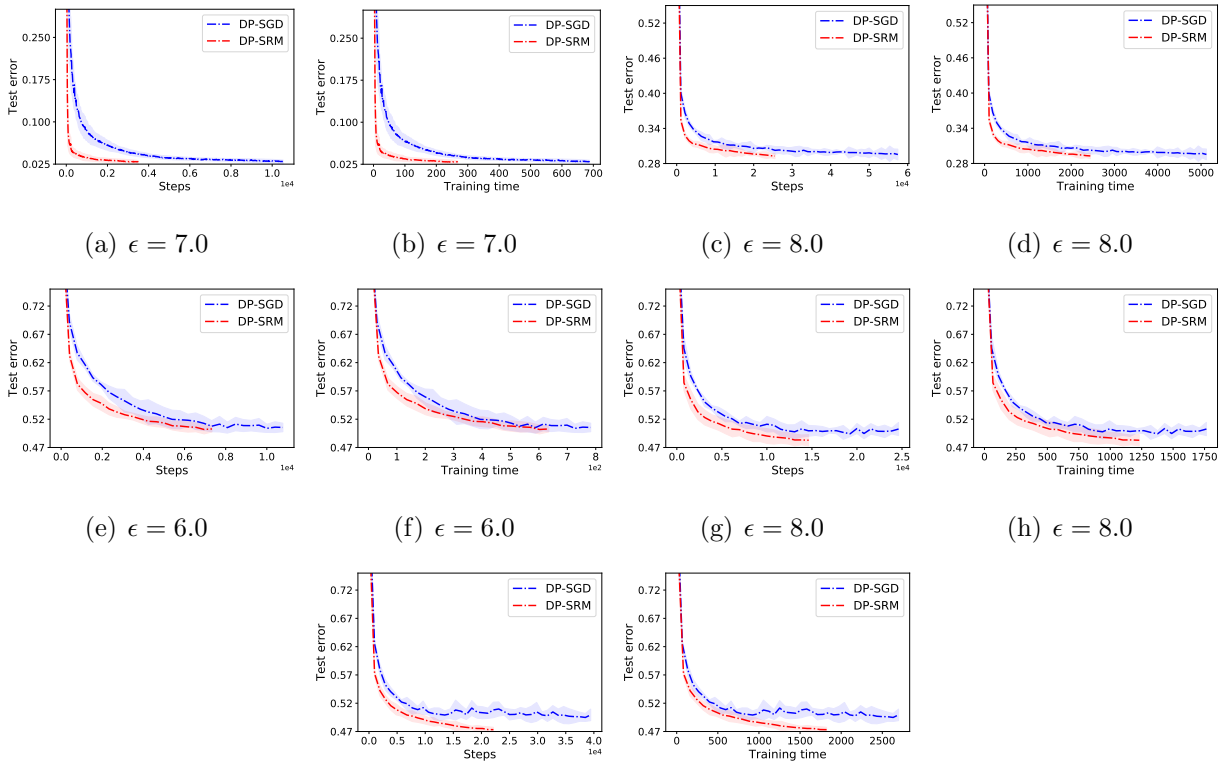
Privacy Budget	Non-private Baseline	Method	Test Error	Data Passes	CPU time	Gradient Norm
$\epsilon = 0.2$	0.2096 (0.002)	DP-GD	0.3160 (0.0120)	20	0.5180	0.0184 (0.0024)
		DP-AGD	0.2645 (0.0044)	346	90.05	0.0133 (0.0018)
		RRPSGD	0.3110 (0.0106)	8	47.64	0.0175 (0.0023)
		DP-SRM	**0.2503 (0.0090)**	**4**	**0.4748**	**0.0117 (0.0008)**
$\epsilon = 0.5$	0.2096 (0.002)	DP-GD	0.2717 (0.0081)	20	0.4990	0.0171 (0.0024)
		DP-AGD	0.2416 (0.0029)	365	94.28	0.0397 (0.0025)
		RRPSGD	0.3033 (0.0110)	10	59.06	0.0160 (0.0018)
		DP-SRM	**0.2341 (0.0042)**	**5**	**0.4368**	**0.0082 (0.0005)**

4.8.2 Additional Results on MNIST and CIFAR-10 datasets

In this subsection, we present additional experiment results on convolutional neural networks. Figures 4.5 shows the average test error (over 30 trials) and the corresponding 95% confidence interval of different methods versus the number of iterations as well as the training time under different privacy budgets on MNIST and CIFAR-10 datasets.

Results on MINST dataset. We can see from Figure 4.5(a) and Figure 4.5(b) that our proposed method can achieve 2.91% test error when $\epsilon = 7.0$, which is comparable to the 2.93% test errors achieved by DP-SGD. Furthermore, the results show that our method is more efficient than DP-SGD in terms of iteration numbers and the training time. More specifically, our method is more than $2\times$ faster than DP-SGD to achieve the desired test error.

Parameters for CNN5. We choose three different privacy budgets $\epsilon \in \{6.0, 8.0, 10.0\}$,



(a) $\epsilon = 7.0$ (b) $\epsilon = 7.0$ (c) $\epsilon = 8.0$ (d) $\epsilon = 8.0$

(e) $\epsilon = 6.0$ (f) $\epsilon = 6.0$ (g) $\epsilon = 8.0$ (h) $\epsilon = 8.0$

(i) $\epsilon = 10.0$ (j) $\epsilon = 10.0$

Figure 4.5: Results for CNN on MNIST and CIFAR-10 datasets. (a), (b) illustrate the results on MNIST dataset. (c), (d) demonstrate the results for CNN6 on CIFAR-10 dataset. (e)-(j) show the results for CNN5 on CIFAR-10 dataset.

and set $\delta = 10^{-5}$. We set the clipping parameter $C_1 = 2$ for the term $\|\nabla f_i(\theta^t)\|_2$. For the term $\|\nabla f_i(\theta^t) - \nabla f_i(\theta^{t-1})\|_2$, we choose the clipping parameter C_2 by searching the grid $\{0.01, 0.1, 0.3, 0.5, 0.7, 0.9, 0.99\}$. For DP-SGD, we tune the batch size by searching the grid $\{32, 64, 128\}$ and the step size by $\{0.01, 0.02, 0.05, 0.1, 0.2\}$. For DP-SRM, we tune the batch size b by searching the grid $\{32, 64, 128\}$, step size by $\{0.01, 0.02, 0.05, 0.1, 0.2\}$, and b_0 by $\{b, 2b, 4b\}$. In addition, we set the momentum parameter $\gamma = C_2$.

Results for CNN5 on CIFAR-10 dataset. Figures 4.5(e)-4.5(j) present the average test error of different methods versus the number of iterations as well as the training time under different privacy budgets for CNN5 on CIFAR-10 dataset. The CNN5 trained by the

non-private SGD will have 39.5% test error after 100 epochs. The results show that that our proposed method has 50.3%, 48.2% and 47.1% test errors when $\epsilon = 6.0$, $\epsilon = 8.0$ and $\epsilon = 10.0$. Nevertheless, DP-SGD has 51.0%, 50.2% and 49.3% test errors under the privacy budgets $\epsilon = 6.0$, $\epsilon = 8.0$ and $\epsilon = 10.0$, which are worse than our method. Furthermore, we can see from the plots that compared with DP-SGD, our method can reduce both the iteration numbers and the training time.

Results for CNN6 on CIFAR-10 dataset. Figure 4.5(c) and Figure 4.5(d) illustrate the average test error of different methods versus the number of iterations and the training time for CNN6 on CIFAR-10 dataset. We can see from the results that that our proposed method can achieve 29.3% test errors given the privacy budget $\epsilon = 8.0$, which are comparable to the results of DP-SGD with 29.4% under the same privacy budget. However, we can see from the plots that our method can significantly reduce the iteration numbers and the training time. When $\epsilon = 8$, DP-SGD takes 5.8×10^4 iterations and 5176 seconds to achiever 29.4% test error. In sharp contrast, our method only takes 2.6×10^4 iterations and 2589 seconds to achieve 29.3% test error.

4.9 Proof of main results

In this section, we present the proofs of our main results.

4.9.1 Proof of Theorem 4.5.1

We will provide the privacy guarantee of Algorithm 4 in this subsection. To this end, we need the following composition rule for RDP.

Lemma 4.9.1 ((Mir17))**.** If k randomized mechanisms $\mathcal{M}_i : \mathcal{S}^n \to \mathcal{R}$ for $i \in [k]$, satisfy (α, ρ_i)-RDP, then their composition $\big(\mathcal{M}_1(S), \ldots, \mathcal{M}_k(S)\big)$ satisfies $(\alpha, \sum_{i=1}^k \rho_i)$-RDP. Moreover, the input of the i-th mechanism can base on the outputs of previous $(i-1)$ mechanisms.

We will first show that our proposed algorithm satisfies RDP using Lemma 4.3.7 and Lemma 4.9.1. Then we will transform it into (ϵ, δ)-DP based on Lemma 4.3.9. For the given dataset S, we use S' to denote its neighboring dataset with one different example indexed by i' in the following discussion. According to Algorithm 4, we use the following $\mathcal{M}_t$ to denote the mechanism at t-th iteration

$$\mathcal{M}_t = \begin{cases} \nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^t) + (1-\gamma)\big(\mathbf{v}_p^{t-1} - \nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^{t-1})\big) + \mathbf{u}^t, & t > 0, \\ \mathbf{v}^0 + \mathbf{u}^0, & t = 0. \end{cases} \tag{4.9.1}$$

Therefore, our goal is to show the privacy guarantees of $\mathcal{M}_t$ for $t = 0, 1, \ldots, T$.

Case 1: If $t = 0$, we have $\mathbf{v}^0 = \nabla F_{\mathcal{B}_0}(\boldsymbol{\theta}^0)$ and $\mathcal{M}_0$ is equivalent to the following Gaussian mechanism

$$\mathcal{G}_0 = \nabla F_{\mathcal{B}_0}(\boldsymbol{\theta}^0) + \mathbf{u}^0,$$

where $\mathbf{u}^0 \sim N(0, \sigma_0^2 \mathbf{I}_d)$. Note that the mechanism $\mathcal{G}_0$ is based on the subsampling, thus we will use the results of privacy-amplification by subsampling, i.e., Lemma 4.3.7, to show that $\mathcal{G}_0$ satisfies RDP given appropriate $\mathbf{u}^0$. To this end, we first consider the following Gaussian mechanism without subsampling

$$\widetilde{\mathcal{G}}_0 = \frac{1}{b_0} \sum_{i=1}^{n} \nabla f_i(\boldsymbol{\theta}^0) + \mathbf{u}^0.$$

Sensitivity. Consider the query on the dataset S as follows $\widetilde{\mathbf{q}}_0(S) = \sum_{i=1}^n \nabla f_i(\boldsymbol{\theta}^0)/b_0$, where $\widetilde{\mathbf{q}}_0(S)$ denotes that the query is based on the dataset S. Thus, we have

$$\widetilde{\mathbf{q}}_0(S) - \widetilde{\mathbf{q}}_0(S') = \frac{1}{b_0}\big(\nabla f_i(\boldsymbol{\theta}^0) - \nabla f_{i'}(\boldsymbol{\theta}^0)\big).$$

Since each component function is G-Lipschitz, we can obtain the ℓ_2-sensitivity of this query as follows

$$\widetilde{\Delta}_0 = \frac{1}{b_0}\|\nabla f_i(\boldsymbol{\theta}^0) - \nabla f_{i'}(\boldsymbol{\theta}^0)\|_2 \leq \frac{2G}{b_0}. \tag{4.9.2}$$

Privacy guarantee of $\mathcal{G}_0$. By Lemma 4.3.7, if the Gaussian noise $\mathbf{u}^0$ in $\widetilde{\mathcal{G}}_0$ has the following variance

$$\sigma_0^2 = \frac{14T\alpha G^2}{\beta n^2 \epsilon}, \tag{4.9.3}$$

the mechanism $\widetilde{\mathcal{G}}_0$ satisfies $\big(\alpha, \beta\epsilon n^2/\big(7b_0^2 T\big)\big)$-RDP. Therefore, according to the privacy-amplification by subsampling result in Lemma 4.3.7, we have that the mechanism $\mathcal{G}_0$ satisfies (α, ρ_0)-RDP, where $\rho_0 = \beta\epsilon/T$. Furthermore, the variance σ_0^2 should satisfy the following condition

$$\frac{\sigma_0^2}{\widetilde{\Delta}_0^2} = \frac{\sigma_0^2 b_0^2}{4G^2} = \frac{7b_0^2 T\alpha}{\beta n^2 \epsilon} \geq 0.7.$$

And the parameter α should satisfy $\alpha \leq 1 + 2(\sigma_0/\widetilde{\Delta}_0)^2 \log\big(1/\tau\alpha(1 + (\sigma_0/\widetilde{\Delta}_0)^2)\big)/3$.

Case 2: If $t > 0$, according to the definition of $\mathcal{M}_t$ in (4.9.1), we consider the following Gaussian mechanism

$$\mathcal{G}_t = \nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^t) - (1-\gamma)\nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^{t-1}) + \mathbf{u}^t.$$

Now, we are going to show that $\mathcal{G}_t$ satisfies RDP given appropriate $\mathbf{u}^t$. Since the mechanism $\mathcal{G}_t$ is based on the subsampling, we will use the similar proof procedure as in **Case 1** to show that $\mathcal{G}_t$ satisfies RDP. Thus we consider the following Gaussian mechanism without subsampling

$$\widetilde{\mathcal{G}}_t = \frac{1}{b}\sum_{i=1}^{n} \nabla f_i(\boldsymbol{\theta}^t) - (1-\gamma)\frac{1}{b}\sum_{i=1}^{n} \nabla f_i(\boldsymbol{\theta}^{t-1}) + \mathbf{u}^t.$$

Sensitivity. We consider the following query without subsampling

$$\widetilde{\mathbf{q}}_t(S) = \frac{1}{b}\sum_{i=1}^{n} \nabla f_i(\boldsymbol{\theta}^t) - (1-\gamma)\frac{1}{b}\sum_{i=1}^{n} \nabla f_i(\boldsymbol{\theta}^{t-1}).$$

Thus we have

$$\widetilde{\mathbf{q}}_t(S) - \widetilde{\mathbf{q}}_t(S') = \frac{1}{b}\big(\nabla f_i(\boldsymbol{\theta}^t) - (1-\gamma)\nabla f_i(\boldsymbol{\theta}^{t-1}) - \nabla f_{i'}(\boldsymbol{\theta}^t) + (1-\gamma)\nabla f_{i'}(\boldsymbol{\theta}^{t-1})\big).$$

As a result, we can obtain the ℓ_2-sensitivity of the query $\widetilde{\mathbf{q}}_t$ as follows

$$\begin{aligned}\widetilde{\Delta}_t &= \frac{1}{b}\big\|(1-\gamma)\big(\nabla f_i(\boldsymbol{\theta}^t) - \nabla f_i(\boldsymbol{\theta}^{t-1}) - \nabla f_{i'}(\boldsymbol{\theta}^t) + \nabla f_{i'}(\boldsymbol{\theta}^{t-1})\big) \\ &\qquad + \gamma\big(\nabla f_i(\boldsymbol{\theta}^t) - \nabla f_{i'}(\boldsymbol{\theta}^t)\big)\big\|_2 \\ &\leq \frac{2L(1-\gamma)}{b}\|\boldsymbol{\theta}^t - \boldsymbol{\theta}^{t-1}\|_2 + \frac{2\gamma G}{b},\end{aligned}$$

where the inequality is due to L-Lipschitz continuous gradient and G-Lipschitz of each component function. Furthermore, according to the update rule of Algorithm 4 and the definition of η_{t-1}, we have

$$\|\boldsymbol{\theta}^t - \boldsymbol{\theta}^{t-1}\|_2 \leq \eta_{t-1}\|\mathbf{v}_p^{t-1}\|_2 \leq \min\left\{\frac{\zeta}{n_0 L\|\mathbf{v}_p^{t-1}\|_2}, \frac{1}{2n_0 L}\right\} \cdot \|\mathbf{v}_p^{t-1}\|_2 \leq \frac{\zeta}{n_0 L},$$

which implies that

$$\widetilde{\Delta}_t \leq \frac{2L(1-\gamma)}{b}\|\boldsymbol{\theta}^t - \boldsymbol{\theta}^{t-1}\|_2 + \frac{2\gamma G}{b} \leq \frac{2\big((1-\gamma)\zeta/n_0 + \gamma G\big)}{b}. \tag{4.9.4}$$

Privacy guarantee of $\mathcal{G}_t$. By Lemma 4.3.7, if we the Gaussian noise $\mathbf{u}^t$ in $\widetilde{\mathcal{G}}_t$ has the variance as follows

$$\sigma_t^2 = \frac{14T\alpha\big((1-\gamma)\zeta/n_0 + \gamma G\big)^2}{\beta n^2 \epsilon}, \tag{4.9.5}$$

the mechanism $\widetilde{\mathcal{G}}_t$ satisfies $\big(\alpha, \beta\epsilon n^2/\big(7b^2T\big)\big)$-RDP. Thus based on the privacy-amplification by subsampling result (Lemma 4.3.7), we can get that the mechanism $\mathcal{G}_t$ satisfies (α, ρ)-RDP, where $\rho = \beta\epsilon/T$. In addition, the variance σ_t^2 should satisfy the following condition

$$\frac{\sigma_t^2}{\widetilde{\Delta}_t^2} = \frac{\sigma_t^2 b^2}{4\big((1-\gamma)\zeta/n_0 + \gamma G\big)^2} = \frac{7b^2T\alpha}{\beta n^2\epsilon} \geq 0.7.$$

And the parameter α should satisfy $\alpha \leq 1 + 2(\sigma_t/\widetilde{\Delta}_t)^2 \log\big(1/\tau\alpha(1 + (\sigma_t/\widetilde{\Delta}_t)^2)\big)/3$. As a result, we show that $\mathcal{G}_t$ satisfies (α, ρ)-RDP.

Privacy guarantee of $\mathcal{M}_t$. By the definition of the mechanism $\mathcal{M}_t$ in (4.9.1), $\mathcal{M}_t$ is a composition of $\mathcal{G}_0, \ldots, \mathcal{G}_t$, i.e., $\mathcal{M}_t = (\mathcal{G}_0, \ldots, \mathcal{G}_t)$. According to the composition property of RDP, i.e., Lemma 4.9.1, we have $\mathcal{M}_t$ satisfies $(\alpha, \rho_0 + (t-1)\rho)$-RDP. Since $\rho_0 = \rho = \beta\epsilon/T$,

we have that after T' iterations of Algorithm 4, it satisfies $(\alpha, \beta T'\epsilon/T)$-RDP. According to Lemma 4.3.9 and $\alpha = \log(1/\delta)/\big((1-\beta)\epsilon\big) + 1$, we have that after T' iterations, Algorithm 4 satisfies $(T'\epsilon/T, \delta)$-DP. As a result, we have that for each $\boldsymbol{\theta}^t$, where $t = 1, \ldots, T$, it satisfies (ϵ, δ)-DP. Finally, by the definition of $\widetilde{\boldsymbol{\theta}}$, we have $\widetilde{\boldsymbol{\theta}}$ satisfies (ϵ, δ)-DP.

4.9.2 Proof of Theorem 4.5.4

In this subsection. we provide the utility guarantee of our method. According to the assumption that each component function has L-Lipschitz continuous gradient, we can obtain that

$$\|\nabla F(\mathbf{x}) - \nabla F(\mathbf{y})\|_2 = \frac{1}{n}\sum_{i=1}^{n}\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\|_2 \leq L\|\mathbf{x}-\mathbf{y}\|_2,$$

which implies that $F(\mathbf{x})$ has L-Lipschitz continuous gradient. Thus we have

$$\begin{aligned}
F(\boldsymbol{\theta}^{t+1}) &\leq F(\boldsymbol{\theta}^t) + \langle \nabla F(\boldsymbol{\theta}^t), \boldsymbol{\theta}^{t+1} - \boldsymbol{\theta}^t\rangle + \frac{L}{2}\|\boldsymbol{\theta}^{t+1} - \boldsymbol{\theta}^t\|_2^2 \\
&= F(\boldsymbol{\theta}^t) - \eta_t\langle \nabla F(\boldsymbol{\theta}^t), \mathbf{v}_p^t\rangle + \frac{\eta_t^2 L}{2}\big\|\mathbf{v}_p^t\big\|_2^2 \\
&= F(\boldsymbol{\theta}^t) + \frac{\eta_t}{2}\big\|\nabla F(\boldsymbol{\theta}^t) - \mathbf{v}_p^t\big\|_2^2 - \frac{\eta_t}{2}\big\|\nabla F(\boldsymbol{\theta}^t)\big\|_2^2 - \eta_t\left(\frac{1}{2} - \frac{\eta_t L}{2}\right)\big\|\mathbf{v}_p^t\big\|_2^2,
\end{aligned}$$

where the last equality is due to the fact that $2\langle \nabla F(\boldsymbol{\theta}^t), \mathbf{v}_p^t\rangle = \big\|\nabla F(\boldsymbol{\theta}^t)\big\|_2^2 + \big\|\mathbf{v}_p^t\big\|_2^2 - \big\|\nabla F(\boldsymbol{\theta}^t) - \mathbf{v}_p^t\big\|_2^2$. Since $\eta_t \leq 1/(2n_0L)$, we can obtain that

$$F(\boldsymbol{\theta}^{t+1}) \leq F(\boldsymbol{\theta}^t) + \frac{1}{4n_0L}\big\|\nabla F(\boldsymbol{\theta}^t) - \mathbf{v}_p^t\big\|_2^2 - \frac{\eta_t}{4}\big\|\mathbf{v}_p^t\big\|_2^2.$$

In addition, we have

$$\frac{\eta_t}{4}\big\|\mathbf{v}_p^t\big\|_2^2 = \frac{\zeta^2}{8n_0L}\min\left\{2\big\|\mathbf{v}_p^t/\zeta\big\|_2, \big\|\mathbf{v}_p^t/\zeta\big\|_2^2\right\} \geq \frac{\zeta\big\|\mathbf{v}_p^t\big\|_2 - 2\zeta^2}{4n_0L}.$$

Thus we have

$$F(\boldsymbol{\theta}^{t+1}) \leq F(\boldsymbol{\theta}^t) + \frac{1}{4n_0L}\big\|\nabla F(\boldsymbol{\theta}^t) - \mathbf{v}_p^t\big\|_2^2 - \frac{\zeta\big\|\mathbf{v}_p^t\big\|_2}{4n_0L} + \frac{\zeta^2}{2n_0L}. \tag{4.9.6}$$

Summing over $t = 0, \ldots, T-1$ and taking expectation in (4.9.6), we can get

$$\begin{aligned}\frac{\zeta}{4n_0L}\sum_{t=0}^{T-1}\mathbb{E}\big\|\mathbf{v}_p^t\big\|_2 &\leq F(\boldsymbol{\theta}^0) - \mathbb{E}F(\boldsymbol{\theta}^T) + \frac{1}{4n_0L}\sum_{t=0}^{T-1}\mathbb{E}\big\|\nabla F(\boldsymbol{\theta}^t) - \mathbf{v}_p^t\big\|_2^2 + \frac{T\zeta^2}{2n_0L} \\ &\leq F(\boldsymbol{\theta}^0) - F(\boldsymbol{\theta}^*) + \frac{1}{4n_0L}\sum_{t=0}^{T-1}\mathbb{E}\big\|\nabla F(\boldsymbol{\theta}^t) - \mathbf{v}_p^t\big\|_2^2 + \frac{T\zeta^2}{2n_0L}. \qquad (4.9.7)\end{aligned}$$

For the term $\mathbb{E}\big\|\nabla F(\boldsymbol{\theta}^t) - \mathbf{v}_p^t\big\|_2^2$, we can bound it as follows: we first consider the conditional expectation

$$\begin{aligned}\mathbb{E}_t\big\|\mathbf{v}_p^t - \nabla F(\boldsymbol{\theta}^t)\big\|_2^2 &= \mathbb{E}_t\big\|(1-\gamma)\big(\mathbf{v}_p^{t-1} - \nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^{t-1})\big) + \nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^t) - \nabla F(\boldsymbol{\theta}^t) + \mathbf{u}^t\big\|_2^2 \\ &= \mathbb{E}_t\big\|(1-\gamma)\big(\mathbf{v}_p^{t-1} - \nabla F(\boldsymbol{\theta}^{t-1})\big) + (1-\gamma)\nabla F(\boldsymbol{\theta}^{t-1}) - (1-\gamma)\nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^{t-1}) \\ &\quad + \nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^t) - \nabla F(\boldsymbol{\theta}^t)\big\|_2^2 + \mathbb{E}_t\|\mathbf{u}^t\|_2^2 \\ &= \mathbb{E}_t\big\|(1-\gamma)\big(\mathbf{v}_p^{t-1} - \nabla F(\boldsymbol{\theta}^{t-1})\big) + (1-\gamma)\big(\nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^t) - \nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^{t-1}) \\ &\quad + \nabla F(\boldsymbol{\theta}^{t-1}) - \nabla F(\boldsymbol{\theta}^t)\big) + \gamma\big(\nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^t) - \nabla F(\boldsymbol{\theta}^t)\big)\big\|_2^2 + \mathbb{E}_t\|\mathbf{u}^t\|_2^2, \\ &\qquad (4.9.8)\end{aligned}$$

where $\mathbb{E}_t$ is taken over the randomness at the t-th iteration given the observations after $(t-1)$-th iteration, the first equation comes from the definition of $\mathbf{v}_p^t$, the second one is due to the independence of the random variables. Therefore, we can obtain that

$$\begin{aligned}\mathbb{E}_t\big\|\mathbf{v}_p^t - \nabla F(\boldsymbol{\theta}^t)\big\|_2^2 &= (1-\gamma)^2\mathbb{E}_t\big\|\mathbf{v}_p^{t-1} - \nabla F(\boldsymbol{\theta}^{t-1})\big\|_2^2 + 2\gamma^2\mathbb{E}_t\big\|\nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^t) - \nabla F(\boldsymbol{\theta}^t)\big\|_2^2 + \mathbb{E}_t\|\mathbf{u}^t\|_2^2 \\ &\quad + 2(1-\gamma)^2\mathbb{E}_t\big\|\nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^t) - \nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^{t-1}) + \nabla F(\boldsymbol{\theta}^{t-1}) - \nabla F(\boldsymbol{\theta}^t)\big\|_2^2, \\ &\qquad (4.9.9)\end{aligned}$$

where the equality is due to the expansion of (4.9.8) and Cauchy-Schwartz inequality. In addition, we have

$$\begin{aligned}
&\mathbb{E}_t\big\|\nabla F(\boldsymbol{\theta}^t) - \nabla F(\boldsymbol{\theta}^{t-1}) - \nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^t) + \nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^{t-1})\big\|_2^2 \\
&\leq \frac{1}{b}\cdot\frac{1}{n}\sum_{i=1}^{n}\big\|\nabla F(\boldsymbol{\theta}^t) - \nabla F(\boldsymbol{\theta}^{t-1}) - \nabla f_i(\boldsymbol{\theta}^t) + \nabla f_i(\boldsymbol{\theta}^{t-1})\big\|_2^2 \\
&\leq \frac{1}{b}\cdot\frac{1}{n}\sum_{i=1}^{n}\big\|\nabla f_i(\boldsymbol{\theta}^t) - \nabla f_i(\boldsymbol{\theta}^{t-1})\big\|_2^2 \\
&\leq \frac{L^2}{b}\|\boldsymbol{\theta}^t - \boldsymbol{\theta}^{t-1}\|_2^2,
\end{aligned}$$

where the first inequality is due to Lemma 4.11.1, the second one comes from the fact that $\mathbb{E}\|\boldsymbol{X} - \mathbb{E}\boldsymbol{X}\|_2^2 \leq \mathbb{E}\|\boldsymbol{X}\|_2^2$ for any random variable $\boldsymbol{X}$, and the last one is due to the gradient Lipschitz property of each component function. According to the update rule, we have

$$\|\boldsymbol{\theta}^t - \boldsymbol{\theta}^{t-1}\|_2 \leq \eta_{t-1}\big\|\mathbf{v}_p^{t-1}\big\|_2 \leq \min\left\{\frac{\zeta}{n_0 L\big\|\mathbf{v}_p^{t-1}\big\|_2}, \frac{1}{2n_0 L}\right\}\cdot\big\|\mathbf{v}_p^{t-1}\big\|_2 \leq \frac{\zeta}{n_0 L},$$

which implies

$$\mathbb{E}_t\big\|\nabla F(\boldsymbol{\theta}^t) - \nabla F(\boldsymbol{\theta}^{t-1}) - \nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^t) + \nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^{t-1})\big\|_2^2 \leq \frac{\zeta^2}{n_0^2 b}. \tag{4.9.10}$$

Thus plugging (4.9.10) into (4.9.9), we can obtain that

$$\begin{aligned}
\mathbb{E}_t\big\|\mathbf{v}_p^t - \nabla F(\boldsymbol{\theta}^t)\big\|_2^2 &\leq (1-\gamma)^2\big\|\mathbf{v}_p^{t-1} - \nabla F(\boldsymbol{\theta}^{t-1})\big\|_2^2 + \frac{2(1-\gamma)^2L^2}{b}\|\boldsymbol{\theta}^t - \boldsymbol{\theta}^{t-1}\|_2^2 \\
&\quad + 2\gamma^2\mathbb{E}_t\big\|\nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^t) - \nabla F(\boldsymbol{\theta}^t)\big\|_2^2 + \mathbb{E}_t\|\mathbf{u}^t\|_2^2 \\
&\leq (1-\gamma)^2\big\|\mathbf{v}_p^{t-1} - \nabla F(\boldsymbol{\theta}^{t-1})\big\|_2^2 + \frac{2(1-\gamma)^2\zeta^2}{n_0^2 b} + \frac{2\gamma^2 G^2}{b} + \mathbb{E}_t\|\mathbf{u}^t\|_2^2,
\end{aligned} \tag{4.9.11}$$

where the second inequality follows the following inequality (using Lemma 4.11.1, $\mathbb{E}\|\boldsymbol{X} - \mathbb{E}\boldsymbol{X}\|_2^2 \leq \mathbb{E}\|\boldsymbol{X}\|_2^2$, and the G-Lipschitz of each component function)

$$\mathbb{E}_t\big\|\nabla F_{\mathcal{B}_t}(\boldsymbol{\theta}^t) - \nabla F(\boldsymbol{\theta}^t)\big\|_2^2 \leq \frac{1}{b}\cdot\frac{1}{n}\sum_{i=1}^{n}\big\|\nabla f_i(\boldsymbol{\theta}^t)\big\|_2^2 \leq \frac{G^2}{b}. \tag{4.9.12}$$

Therefore, taking expectations over all iterations in (4.9.11), we can get

$$\mathbb{E}\big\|\mathbf{v}_p^t - \nabla F(\boldsymbol{\theta}^t)\big\|_2^2 \leq (1-\gamma)^2\mathbb{E}\big\|\mathbf{v}_p^{t-1} - \nabla F(\boldsymbol{\theta}^{t-1})\big\|_2^2 + \frac{2(1-\gamma)^2\zeta^2}{n_0^2 b} + \frac{2\gamma^2 G^2}{b} + d\sigma^2. \tag{4.9.13}$$

Following the proof of Lemma 9 in (YLL20), we have

$$\begin{aligned}\gamma\sum_{t=0}^{T-1}\mathbb{E}\big\|\mathbf{v}_p^t - \nabla F(\boldsymbol{\theta}^t)\big\|_2^2 &\leq \frac{2T(1-\gamma)^2\zeta^2}{n_0^2 b} + \frac{2T\gamma^2 G^2}{b} + Td\sigma^2 + \mathbb{E}\big\|\mathbf{v}_p^0 - \nabla F(\boldsymbol{\theta}^0)\big\|_2^2 \\ &\leq \frac{2T(1-\gamma)^2\zeta^2}{n_0^2 b} + \frac{2T\gamma^2 G^2}{b} + Td\sigma^2 + \frac{G^2}{b_0} + d\sigma_0^2,\end{aligned}$$

where the last line comes from the definition of $\mathbf{v}_p^0 = \nabla F_{\mathcal{B}_0}(\boldsymbol{\theta}^0) + \mathbf{u}^0$ and the inequality $\mathbb{E}\big\|\nabla F_{\mathcal{B}_0}(\boldsymbol{\theta}^0) - \nabla F(\boldsymbol{\theta}^0)\big\|_2^2 \leq G^2/b_0$ (see equation (4.9.12)). Therefore, we can obtain that

$$\sum_{t=0}^{T-1}\mathbb{E}\big\|\mathbf{v}_p^t - \nabla F(\boldsymbol{\theta}^t)\big\|_2^2 \leq \frac{2T(1-\gamma)^2\zeta^2}{n_0^2\gamma b} + \frac{2T\gamma G^2}{b} + \frac{Td\sigma^2 + d\sigma_0^2}{\gamma} + \frac{G^2}{\gamma b_0}. \tag{4.9.14}$$

Combining (4.9.7) and (4.9.14), we can get

$$\begin{aligned}\frac{\zeta}{4n_0 L}\sum_{t=0}^{T-1}\mathbb{E}\big\|\mathbf{v}_p^t\big\|_2 &\leq F(\boldsymbol{\theta}^0) - F(\boldsymbol{\theta}^*) + \frac{1}{4n_0 L}\sum_{t=0}^{T-1}\mathbb{E}\big\|\nabla F(\boldsymbol{\theta}^t) - \mathbf{v}_p^t\big\|_2^2 + \frac{T\zeta^2}{2n_0 L} \\ &\leq F(\boldsymbol{\theta}^0) - F(\boldsymbol{\theta}^*) + \frac{T(1-\gamma)^2\zeta^2}{2n_0^3 L\gamma b} + \frac{T\gamma G^2}{4Ln_0 b} + \frac{Td\sigma^2 + d\sigma_0^2}{4n_0 L\gamma} + \frac{G^2}{4L\gamma n_0 b_0} + \frac{T\zeta^2}{2n_0 L}.\end{aligned}$$

Hence we have

$$\begin{aligned}\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\big\|\mathbf{v}_p^t\big\|_2 &\leq \frac{4n_0 L}{T\zeta}\big(F(\boldsymbol{\theta}^0) - F(\boldsymbol{\theta}^*)\big) + \frac{2\zeta}{n_0^2\gamma b} + \frac{\gamma G^2}{\zeta b} + \frac{d\sigma^2 + d\sigma_0^2/T}{\zeta\gamma} + \frac{G^2}{T\zeta\gamma b_0} + 2\zeta \\ &\leq 6\zeta + \frac{2\zeta}{n_0^2\gamma b} + \frac{\gamma G^2}{\zeta b} + \frac{d\sigma^2 + d\sigma_0^2/T}{\zeta\gamma} + \frac{G^2}{T\zeta\gamma b_0},\end{aligned} \tag{4.9.15}$$

where the first inequality is due to $T = \lfloor 4n_0 L\big(F(\boldsymbol{\theta}^0) - F(\boldsymbol{\theta}^*)\big)/\zeta^2\rfloor + 1$. In addition, according to (4.9.14) and Jensen's inequality, we have

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\big\|\nabla F(\boldsymbol{\theta}^t) - \mathbf{v}_p^t\big\|_2 \leq \frac{\sqrt{2}\zeta}{n_0\sqrt{\gamma b}} + \frac{\sqrt{2\gamma}G}{\sqrt{b}} + \frac{\sqrt{d}\sigma + \sqrt{d}\sigma_0/\sqrt{T}}{\sqrt{\gamma}} + \frac{G}{\sqrt{T\gamma b_0}}. \tag{4.9.16}$$

Thus by the definition of $\widetilde{\boldsymbol{\theta}}$, we have

$$\begin{aligned}\mathbb{E}\|\nabla F(\widetilde{\boldsymbol{\theta}})\|_2 &= \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla F(\boldsymbol{\theta}^t)\|_2 \\ &\leq \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\big\|\mathbf{v}_p^t\big\|_2 + \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\big\|\nabla F(\boldsymbol{\theta}^t) - \mathbf{v}_p^t\big\|_2 \\ &\leq 6\zeta + \frac{2\zeta}{n_0^2\gamma b} + \frac{\gamma G^2}{\zeta b} + \frac{d\sigma^2}{\zeta\gamma} + \frac{d\sigma_0^2}{T\zeta\gamma} + \frac{G^2}{T\zeta\gamma b_0} + \frac{\sqrt{2}\zeta}{n_0\sqrt{\gamma b}} + \frac{\sqrt{2\gamma}G}{\sqrt{b}} + \frac{\sqrt{d}\sigma}{\sqrt{\gamma}} + \frac{\sqrt{d}\sigma_0}{\sqrt{T\gamma}} + \frac{G}{\sqrt{T\gamma b_0}},\end{aligned} \tag{4.9.17}$$

where the second inequality comes from (4.9.15) and (4.9.16). Without loss of generality, we can assume $G > 1$. Let $\gamma^2 = 2\zeta^2/n_0^2$, $b = G^2/(n_0\zeta)$, $b_0 = G^2/(\zeta L D_F)$, where $D_F = F(\boldsymbol{\theta}^0) - F(\boldsymbol{\theta}^*)$ and $F(\boldsymbol{\theta}^*)$ is a global minimum of F, by the definition of T, we can get

$$\mathbb{E}\|\nabla F(\widetilde{\boldsymbol{\theta}})\|_2 \leq 15\zeta + \frac{d\sigma^2}{\zeta\gamma} + \frac{\sqrt{d}\sigma}{\sqrt{\gamma}} + \frac{d\sigma_0^2}{T\zeta\gamma} + \frac{\sqrt{d}\sigma_0}{\sqrt{T\gamma}}. \tag{4.9.18}$$

Furthermore, we have

$$\sigma^2 = \frac{14T\big((1-\gamma)\zeta/n_0 + \gamma G\big)^2\log(1/\delta)}{n^2\epsilon^2}, \quad \sigma_0^2 = \frac{14TG^2\log(1/\delta)}{n^2\epsilon^2}. \tag{4.9.19}$$

Plugging (4.9.19) into (4.9.18), we can obtain

$$\begin{aligned}\mathbb{E}\|\nabla F(\widetilde{\boldsymbol{\theta}})\|_2 &\leq 15\zeta + \frac{C_1 T d\zeta G^2\log(1/\delta)}{n_0^2 n^2\epsilon^2\gamma} + \frac{\sqrt{C_1 T d\log(1/\delta)}\zeta G}{n_0 n\epsilon\sqrt{\gamma}} + \frac{C_2 d G^2\log(1/\delta)}{n^2\epsilon^2\zeta\gamma} \\ &\quad + \frac{\sqrt{C_2 d\log(1/\delta)}G}{n\epsilon\sqrt{\gamma}} \\ &\leq 15\zeta + \frac{C_3 L D_F G^2 d\log(1/\delta)}{n^2\epsilon^2\zeta^2} + \frac{\sqrt{C_4 G^2 L D_F d\log(1/\delta)}}{n\epsilon\sqrt{\zeta}} \\ &\quad + \frac{C_5 n_0 d G^2\log(1/\delta)}{n^2\epsilon^2} + \frac{\sqrt{C_6 n_0 d\log(1/\delta)}G}{n\epsilon\sqrt{\zeta}}.\end{aligned} \tag{4.9.20}$$

Let $\zeta = \sqrt{G}\big(LD_F d\log(1/\delta)\big)^{1/4}/\sqrt{n\epsilon}$, $n_0 = LD_F/\zeta$, we have $T = 4n_0 n\epsilon\sqrt{LD_F}/\big(G\sqrt{d\log(1/\delta)}\big)$, and we can get

$$\mathbb{E}\|\nabla F(\widetilde{\boldsymbol{\theta}})\|_2 \leq C_7\frac{\sqrt{G}(LD_F d\log(1/\delta))^{1/4}}{\sqrt{n\epsilon}},$$

where $\{C_i\}_{i=1}^7$ are absolute constants.

Gradient Complexity. Since we have $b = G^2/(n_0\zeta)$, $b_0 = G^2/(\zeta LD_F)$, the total gradient complexity is

$$2(T-1)b + b_0 \leq \frac{8LDn_0}{\zeta^2} \cdot \frac{G^2}{n_0\zeta} + \frac{G^2}{\zeta LD}.$$

According to the definition of ζ, we have the total gradient complexity is $O\big((n\epsilon)^{3/2}/d^{3/4} + (n\epsilon)^{1/2}/d^{1/4}\big)$.

4.10 Proof of Lemma 4.3.7

Without loss of generality, we assume $\Delta(q) = 1$. According to Theorem 9 in (WBK19), we have

$$\rho'(\alpha) \leq \frac{1}{\alpha - 1} \log\left(1 + \tau^2 \binom{\alpha}{2} \min\left\{4(e^{\rho(2)} - 1), 2e^{\rho(2)}\right\} + \sum_{j=3}^{\alpha} \tau^j \binom{\alpha}{j} 2e^{(j-1)\rho(j)}\right), \quad (4.10.1)$$

where τ is the subsample rate, $\rho(j) = j/(2\sigma^2)$. Next, we will show that the summation term in the right hand side of the above inequality is dominated by the second term under certain conditions. First of all, when σ^2 is large, i.e., $\sigma^2 \geq 0.7$, we have

$$\min\left\{4(e^{\rho(2)} - 1), 2e^{\rho(2)}\right\} \leq 6/\sigma^2,$$

which implies that

$$\tau^2 \binom{\alpha}{2} \min\left\{4(e^{\rho(2)} - 1), 2e^{\rho(2)}\right\} \leq \tau^2 \binom{\alpha}{2} 6/\sigma^2.$$

Next, we consider the summation term in (4.10.1), and we have

$$\begin{aligned}
\sum_{j=3}^{\alpha} \tau^j \binom{\alpha}{j} 2e^{(j-1)\rho(j)} &\leq \tau^2 \binom{\alpha}{2} \left(\sum_{j=3}^{\alpha} \tau^{j-2} \alpha^{j-2} e^{\frac{(\alpha-1)j}{2\sigma^2}} \right) \\
&\leq \tau^2 \binom{\alpha}{2} \frac{\tau\alpha e^{\frac{3(\alpha-1)}{2\sigma^2}}}{1 - \tau\alpha e^{\frac{\alpha-1}{2\sigma^2}}},
\end{aligned}$$

where the first inequality is due to the fact that

$$e^{(j-1)\rho(j)} = e^{\frac{(j-1)j}{2\sigma^2}} \leq e^{\frac{(\alpha-1)j}{2\sigma^2}} \quad \text{and} \quad \binom{\alpha}{j} = \frac{\alpha!}{j!(\alpha-j)!} \leq \frac{\alpha^2\alpha^{j-2}}{3!}.$$

In addition, the last inequality comes from the condition that $\tau\alpha\exp\big((\alpha-1)/(2\sigma^2)\big) < 1$ and the sum of the geometric sequence. Therefore, as long as

$$\alpha - 1 \leq \frac{2}{3}\sigma^2 \log\frac{1}{\tau\alpha(1+\sigma^2)}, \tag{4.10.2}$$

we have

$$\sum_{j=3}^{\alpha} \tau^j \binom{\alpha}{j} 2e^{(j-1)\rho(j)} \leq \tau^2 \binom{\alpha}{2} \frac{1}{\sigma^2}.$$

In addition, we require that $\tau\alpha\exp\big((\alpha-1)/(2\sigma^2)\big) < 1$. By plugging the condition of α into the above requirement, we can obtain that this condition can hold if $\tau < 1$.

As a result, under the conditions that $\sigma^2 \geq 0.7$, $\alpha \leq \log(1/\tau\big(1+\sigma^2\big)\big)$, we can obtain that

$$\rho'(\alpha) \leq \frac{1}{\alpha-1}\log\left(1 + \tau^2\binom{\alpha}{2}\frac{10}{\sigma^2}\right) \leq \frac{1}{\alpha-1}\tau^2\binom{\alpha}{2}\frac{7}{\sigma^2} \leq 3.5\alpha\tau^2/\sigma^2.$$

4.11 Auxiliary Lemmas

Lemma 4.11.1. (LJC17) Consider vectors $\mathbf{a}_i$ satisfying $\sum_{i=1}^n \mathbf{a}_i = 0$. Let $\mathcal{B}$ be a uniform random subset of $\{1, 2, \ldots, n\}$ with size m, we have

$$\mathbb{E}\left\|\frac{1}{m}\sum_{i\in\mathcal{B}}\mathbf{a}_i\right\|_2^2 \leq \frac{\mathbb{1}\{|\mathcal{B}| < n\}}{mn}\sum_{i=1}^n \|\mathbf{a}_i\|_2^2.$$

4.12 Conclusions

We propose an efficient differentially private algorithm for nonconvex ERM. We prove both privacy and utility guarantees for our method. Both theoretical analyses and experiments

demonstrate the advantage of our algorithms compared with the state-of-the-art. It would be very interesting to study our method's performances in super large or even industrial level neural networks. It would also be very interesting to study the optimization lower bound for the differentially private nonconvex stochastic optimization problem.

CHAPTER 5

Dp-lssgd: A Stochastic Optimization Method to Lift the Utility in Privacy-Preserving ERM

5.1 Introduction

Many released machine learning (ML) models are trained on sensitive data that are often crowdsourced or contain private information (YKL11; FYZ17; LGN17). With overparameterization, deep neural nets (DNNs) can memorize the private training data, and it is possible to recover them and break the privacy by attacking the released models (SSS17a). For example, Fredrikson et al. demonstrated that a model-inversion attack can recover training images from a facial recognition system (FJR15). Protecting the private data is one of the most critical tasks in ML.

Differential privacy (DP) (DKM06a) is a theoretically rigorous tool for designing algorithms on aggregated databases with a privacy guarantee. The idea is to add a certain amount of noise to randomize the output of a given algorithm such that the attackers cannot distinguish outputs of any two adjacent input datasets that differ in only one entry.

For repeated applications of additive noise based mechanisms, many tools have been invented to analyze the DP guarantee for the model obtained at the final stage. These include the basic and strong composition theorems and their refinements (DKM06a; DRV10; KOV15), the moments accountant (ACG16a), etc. Beyond the original notion of DP, there are also many other ways to define the privacy, e.g., local DP (DJW14), concentrated/zero-concentrated DP (DR16; BS16a), and Rényi-DP (RDP) (Mir17).

Differentially private stochastic gradient descent (DP-SGD) reduces the utility of the trained models severely compared with SGD. As shown in Figure 5.1, the training and validation losses of the logistic regression on the MNIST dataset increase rapidly when the DP guarantee becomes stronger. The convolutional neural net (CNN) [1] trained by DP-SGD has much lower testing accuracy than the non-private one on the MNIST. We will discuss the detailed experimental settings in Section 5.4. A natural question raised from such performance degradations is:

Can we improve DP-SGD, with negligible extra computational complexity and memory cost, such that it can be used to train general ML models with improved utility?

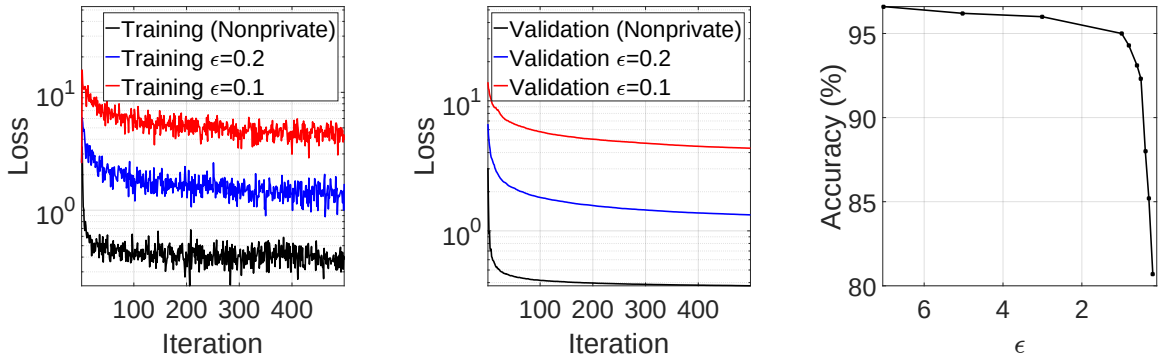


Figure 5.1: Training (left) and validation (middle) losses of the logistic regression on the MNIST trained by DP-SGD with $(\epsilon, \delta = 10^{-5})$-DP guarantee. (right): testing accuracy of a simple CNN on the MNIST trained by DP-SGD with $(\epsilon, \delta = 10^{-5})$-DP guarantee.

We answer the above question affirmatively by proposing differentially private Laplacian smoothing SGD (DP-LSSGD) to improve the utility in privacy-preserving empirical risk minimization (ERM). DP-LSSGD leverages the Laplacian smoothing (OWY18) as a post-processing to smooth the injected Gaussian noise in the differentially private SGD (DP-SGD) to improve the convergence of DP-SGD in training ML models with DP guarantee.

[1] `github.com/tensorflow/privacy/blob/master/tutorials/mnist_dpsgd_tutorial.py`

5.1.1 Our Contributions

The main contributions of our work are highlighted as follows:

- We propose DP-LSSGD and prove its privacy and utility guarantees for convex/nonconvex optimizations. We prove that under the same privacy budget, DP-LSSGD achieves better utility, excluding a small term that is usually dominated by the other terms, than DP-SGD by a factor that is much less than one for convex optimization.
- We perform a large number of experiments logistic regression and CNN to verify the utility improvement by using DP-LSSGD. Numerical results show that DP-LSSGD remarkably reduces training and validation losses and improves the generalization of the trained private models.

In Table 5.1, we compare the privacy and utility guarantees of DP-LSSGD and DP-SGD. For the utility, the notation $\widetilde{O}(\cdot)$ hides the same constant and log factors for each bound. The constants d and n denote the dimension of the model's parameters and the number of training points, respectively. The numbers γ and β are positive constants that are strictly less than one, and D_0, D_σ, G are positive constants, which will be defined in Section 5.3.

Table 5.1: Utility and Differential Privacy Guarantees.

Algorithm	DP	Assumption	Utility	Measurement	Reference
DP-SGD	(ϵ, δ)	convex	$\widetilde{\mathcal{O}}\left(\frac{\sqrt{(D_0+G^2)d}}{(\epsilon n)}\right)$	optimality gap	(BST14a)
DP-SGD	(ϵ, δ)	nonconvex	$\widetilde{\mathcal{O}}\left(\sqrt{d}/(\epsilon n)\right)$	ℓ_2-norm of gradient	(ZZM17a)
DP-LSSGD	(ϵ, δ)	convex	$\widetilde{\mathcal{O}}\left(\frac{\sqrt{\gamma(D_\sigma+G^2)d}}{(\epsilon n)}\right)$	optimality gap	**This Work**
DP-LSSGD	(ϵ, δ)	nonconvex	$\widetilde{\mathcal{O}}\left(\sqrt{\beta d}/(\epsilon n)\right)$ [1]	ℓ_2-norm of gradient	**This Work**

[1] Measured in the norm induced by $\mathbf{A}_\sigma^{-1}$, we will discuss this in detail in Section 5.4.

5.1.2 Related Work

There is a massive volume of research over the past decade on designing algorithms for privacy-preserving ML. Objective perturbation, output perturbation, and gradient perturbation are the three major approaches to perform ERM with a DP guarantee. (CM08; CMS11a) considered both output and objective perturbations for privacy-preserving ERM, and gave theoretical guarantees for both privacy and utility for logistic regression and SVM. (SCS13) numerically studied the effects of learning rate and batch size in DP-ERM. (WLF16) studied stability, learnability and other properties of DP-ERM. (LK18a) proposed an adaptive per-iteration privacy budget in concentrated DP gradient descent. The utility bound of DP-SGD has also been analyzed for both convex and nonconvex smooth objectives (BST14a; ZZM17a). (JWE18a) analyzed the excess empirical risk of DP-ERM in a distributed setting. Besides ERM, many other ML models have been made differentially private. These include: clustering (SCL15; YS15; BDL17), matrix completion (JTT18), online learning (JKT12a), sparse learning (TTZ15; WG19a), and topic modeling (PFC16). (GM17) exploited the ill-conditionedness of inverse problems to design algorithms to release differentially private measurements of the physical system.

(SS15a) proposed distributed selective SGD to train deep neural nets (DNNs) with a DP guarantee in a distributed system, however, the obtained privacy guarantee was very loose. (ACG16a) considered applying DP-SGD to train DNNs in a centralized setting. They clipped the gradient ℓ_2 norm to bound the sensitivity and invented the moment accountant to get better privacy loss estimation. (PAE17) proposed Private Aggregation of Teacher Ensembles/PATE based on the semi-supervised transfer learning to train DNNs, and this framework improves both privacy and utility on top of the work by (ACG16a). Recently (PSM18) introduced new noisy aggregation mechanisms for teacher ensembles that enable a tighter theoretical DP guarantee. The modified PATE is scalable to the large dataset and applicable to more diversified ML tasks.

Laplacian smoothing (LS) can be regarded as a denoising technique that performs post-processing on the Gaussian noise injected stochastic gradient. Denoising has been used in the DP earlier: Post-processing can enforce consistency of contingency table releases (BCD07) and leads to accurate estimation of the degree distribution of private network (HLM09). (NTZ13) showed that post-processing by projecting linear regression solutions, when the ground truth solution is sparse, to a given ℓ_1-ball can remarkably reduce the estimation error. (BMS17) used Expectation-Maximization to denoise a class of graphical models' parameters. (BW18) showed that in the output perturbation based differentially private algorithm design, denoising dramatically improves the accuracy of the Gaussian mechanism in the high-dimensional regime. To the best of our knowledge, we are the first to design a denoising technique on the Gaussian noise injected gradient to improve the utility of the trained private ML models.

5.1.3 Notation

We use boldface upper-case letters $\mathbf{A}$, $\mathbf{B}$ to denote matrices and boldface lower-case letters $\boldsymbol{x}$, $\boldsymbol{y}$ to denote vectors. For vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ and positive definite matrix $\mathbf{A}$, we use $\|\boldsymbol{x}\|_2$ and $\|\boldsymbol{x}\|_{\mathbf{A}}$ to denote the ℓ_2-norm and the induced norm by $\mathbf{A}$, respectively; $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ denotes the inner product of $\boldsymbol{x}$ and $\boldsymbol{y}$; and $\lambda_i(\mathbf{A})$ denotes the i-th largest eigenvalue of $\mathbf{A}$. We denote the set of numbers from 1 to n by $[n]$. $\mathcal{N}(\mathbf{0}, \mathbf{I}_{d\times d})$ represents d-dimensional standard Gaussian.

5.1.4 Organization

This paper is organized in the following way: In Section 5.2, we introduce the DP-LSSGD algorithm. In Section 5.3, we analyze the privacy and utility guarantees of DP-LSSGD for both convex and nonconvex optimizations. We numerically verify the efficiency of DP-LSSGD in Section 5.4. We conclude this work and point out some future directions in Section 5.8.

5.2 Problem Setup and Algorithm

5.2.1 Laplacian Smoothing Stochastic Gradient Descent (LSSGD)

In this paper, we consider empirical risk minimization problem as follows. Given a training set $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ drawn from some unknown but fixed distribution, we aim to find an empirical risk minimizer that minimizes the empirical risk as follows,

$$\min_{\mathbf{w}} F(\mathbf{w}) := \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{w}), \quad \mathbf{w} \in \mathbb{R}^d, \tag{5.2.1}$$

where $F(\mathbf{w})$ is the empirical risk (a.k.a., training loss), $f_i(\mathbf{w}) = \ell(\mathbf{w}; \mathbf{x}_i, y_i)$ is the loss function of a given ML model defined on the i-th training example $(\mathbf{x}_i, y_i)$, and $\mathbf{w} \in \mathbb{R}^d$ is the model parameter we want to learn. Empirical risk minimization serves as the mathematical foundation for training many ML models that are mentioned above. The LSSGD (OWY18) for solving (5.2.1) is given by

$$\mathbf{w}^{k+1} = \mathbf{w}^k - \eta \mathbf{A}_\sigma^{-1}\left(\frac{1}{b}\sum_{i_k \in \mathcal{B}_k} \nabla f_{i_k}(\mathbf{w}^k)\right), \tag{5.2.2}$$

where η is the learning rate, ∇f_{i_k} denotes the stochastic gradient of F evaluated from the pair of input-output $\{\boldsymbol{x}_{i_k}, y_{i_k}\}$, and $\mathcal{B}_k$ is a random subset of size b from $[n]$. Let $\mathbf{A}_\sigma = \mathbf{I} - \sigma\mathbf{L}$ for $\sigma \geq 0$ being a constant, where $\mathbf{I} \in \mathbb{R}^{d\times d}$ and $\mathbf{L} \in \mathbb{R}^{d\times d}$ are the identity and the discrete one-dimensional Laplacian matrix with periodic boundary condition, respectively. Therefore,

$$\mathbf{A}_\sigma := \begin{bmatrix} 1+2\sigma & -\sigma & 0 & \ldots & 0 & -\sigma \\ -\sigma & 1+2\sigma & -\sigma & \ldots & 0 & 0 \\ 0 & -\sigma & 1+2\sigma & \ldots & 0 & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ -\sigma & 0 & 0 & \ldots & -\sigma & 1+2\sigma \end{bmatrix} \tag{5.2.3}$$

When $\sigma = 0$, LSSGD reduces to SGD.

Note that $\mathbf{A}_\sigma$ is positive definite with condition number $1 + 4\sigma$ that is independent of $\mathbf{A}_\sigma$'s dimension, and LSSGD guarantees the same convergence rate as SGD in both convex

and nonconvex optimization. Moreover, Laplacian smoothing (LS) can reduce the variance of SGD on-the-fly, and lead to better generalization in training many ML models including DNNs (OWY18). For $\mathbf{v} \in \mathbb{R}^d$, let $\mathbf{u} := \mathbf{A}_\sigma^{-1}\mathbf{v}$, i.e., $\mathbf{v} = \mathbf{A}_\sigma \mathbf{u}$. Note $\mathbf{A}_\sigma$ is a convolution matrix, therefore, $\mathbf{v} = \mathbf{A}_\sigma \mathbf{u} = \mathbf{u} - \sigma \mathbf{d} * \mathbf{u}$, where $\mathbf{d} = [-2, 1, 0, \cdots, 0, 1]^T$ and $*$ is the convolution operator. By the fast Fourier transform (FFT), we have

$$\mathbf{A}_\sigma^{-1}\mathbf{v} = \mathbf{u} = \text{ifft}\left(\text{fft}(\mathbf{v})/(\mathbf{1} - \sigma \cdot \text{fft}(\mathbf{d}))\right),$$

where the division in the right hand side parentheses is performed in a coordinate wise way.

5.2.2 DP-LSSGD

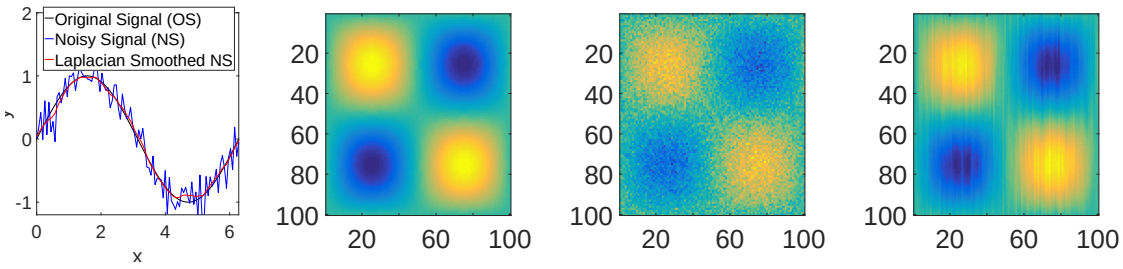
DP ERM aims to learn a DP model, $\mathbf{w}$, for the problem (5.2.1). A common approach is injecting Gaussian noise into the stochastic gradient, and it resulting in the following DP-SGD

$$\mathbf{w}^{k+1} = \mathbf{w}^k - \eta\left(\frac{1}{b}\sum_{i_k \in \mathcal{B}_k} \nabla f_{i_k}(\mathbf{w}^k) + \mathbf{n}\right), \tag{5.2.4}$$

where $\mathbf{n}$ is the injected Gaussian noise for DP guarantee. Note that the LS matrix $\mathbf{A}_\sigma^{-1}$ can remove the noise in $\mathbf{v}$. If we assume $\mathbf{v}$ is the initial signal, then $\mathbf{A}_\sigma^{-1}\mathbf{v}$ can be regarded as performing an approximate diffusion step on the initial noisy signal which removes the noise from $\mathbf{v}$. We will provide a detailed argument for the diffusion process in the appendix. As numerical illustrations, we consider the following two signals:

- 1D: $\mathbf{v}_1 = \{\sin(2i\pi/100) + 0.1\mathcal{N}(0,1) | i = 1, 2, \cdots, 100\}$.
- 2D: $\mathbf{v}_2 = \{\sin(2i\pi/100)\sin(2j\pi/100) + 0.2\mathcal{N}(0, \mathbf{I}_{2\times 2}) | i, j = 1, 2, \cdots, 100\}$.

We reshape $\mathbf{v}_2$ into 1D with row-major ordering and then perform LS. Figure 5.2 shows that LS can remove noise efficiently. This noise removal property enables LSSGD to be more stable to the noise injected stochastic gradient, therefore improves training DP models with gradient perturbations.



(a) (b) (c) (d)

Figure 5.2: Illustration of LS ($\sigma = 10$ for $\mathbf{v}_1$ and $\sigma = 100$ for $\mathbf{v}_2$). (a): 1D signal sampled uniformly from $\sin(x)$ for $x \in [0, 2\pi]$. (b), (c), (d): 2D original, noisy, and Laplacian Smoothed noisy signals sampled uniformly from $\sin(x)\sin(y)$ for $(x, y) \in [0, 2\pi] \times [0, 2\pi]$.

We propose the following DP-LSSGD for solving (5.2.1) with DP guarantee

$$\mathbf{w}^{k+1} = \mathbf{w}^k - \eta \mathbf{A}_\sigma^{-1} \left(\frac{1}{b} \sum_{i_k \in \mathcal{B}_k} \nabla f_{i_k}(\mathbf{w}^k) + \mathbf{n} \right). \tag{5.2.5}$$

In this scheme, we first inject the noise $\mathbf{n}$ to the stochastic gradient $\nabla f_{i_k}(\mathbf{w}^k)$, and then apply the LS operator $\mathbf{A}_\sigma^{-1}$ to denoise the noisy stochastic gradient, $\nabla f_{i_k}(\mathbf{w}^k) + \mathbf{n}$, on-the-fly. We assume that each component function f_i in (5.2.1) is G-Lipschitz. The DP-LSSGD for finite-sum optimization is summarized in Algorithm 5. Compared with LSSGD, the main difference of DP-LSSGD lies in injecting Gaussian noise into the stochastic gradient, before applying the Laplacian smoothing, to guarantee the DP.

5.3 Main Theory

In this section, we present the privacy and utility guarantees for DP-LSSGD. The technical proofs are provided in the appendix.

Definition 5.3.1 ((ϵ, δ)-DP)**.** ((DKM06a)) A randomized mechanism $\mathcal{M} : \mathcal{S}^N \to \mathcal{R}$ satisfies (ϵ, δ)-DP if for any two adjacent datasets $S, S' \in \mathcal{S}^N$ differing by one element, and any output

Algorithm 5 DP-LSSGD

Input: $f_i(\mathbf{w})$ is G-Lipschitz for $i = 1, 2, \cdots, n$.

$\mathbf{w}^0$: initial guess of $\mathbf{w}$, (ϵ, δ): the privacy budget, η: the step size, T: the total number of iterations.

Output: (ϵ, δ)-differentially private classifier $\mathbf{w}_{\text{priv}}$.

for $k = 0, 1, \cdots, T-1$ **do**

$\mathbf{w}^{k+1} = \mathbf{w}^k - \eta \mathbf{A}_\sigma^{-1} \left(\frac{1}{b} \sum_{i_k \in \mathcal{B}_k} \nabla f_{i_k}(\mathbf{w}^k) + \mathbf{n} \right)$, where $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \nu^2 \mathbf{I})$ and ν is defined in Theorem 5.3.2, and $\mathcal{B}_k \subset [n]$.

end for

returnw^T

subset $O \subseteq \mathcal{R}$, it holds that

$$\mathbb{P}[\mathcal{M}(S) \in O] \leq e^\epsilon \cdot \mathbb{P}[\mathcal{M}(S') \in O] + \delta.$$

Theorem 5.3.2 (Privacy Guarantee)**.** Suppose that each component function f_i is G-Lipschitz. Given the total number of iterations T, for any $\delta > 0$ and privacy budget ϵ, DP-LSSGD, with injected Gaussian noise $\mathcal{N}(0, \nu^2)$ for each coordinate, satisfies (ϵ, δ)-DP with $\nu^2 = 20T\alpha G^2/(\mu n^2 \epsilon)$, where $\alpha = \log(1/\delta)/\big((1-\mu)\epsilon\big) + 1$, if there exits $\mu \in (0, 1)$ such that $\alpha \leq \log\big(\mu n^3 \epsilon/(5b^3 T\alpha + \mu b n^2 \epsilon)\big)$ and $5b^2 T\alpha/(\mu n^2 \epsilon) \geq 1.5$.

Remark 5.3.3. It is straightforward to show that the noise in Theorem 5.3.2 is in fact also tight to guarantee the (ϵ, δ)-DP for DP-SGD. . We will omit the dependence of μ in our results in the rest of the paper since μ is a constant.

For convex ERM, DP-LSSGD guarantees the following utility in terms of the gap between the ergodic average of the points along the DP-LSSGD path and the optimal solution $\mathbf{w}^*$.

Theorem 5.3.4 (Utility Guarantee for convex optimization)**.** Suppose F is convex and each component function f_i is G-Lipschitz. Given $\epsilon, \delta > 0$, under the same conditions of Theorem 5.3.2 on ν^2, α, if we choose $\eta_k = 1/\sqrt{T}$ and $T = C_1(D_\sigma + G^2/b)n^2\epsilon^2/\big(dG^2 \log(1/\delta)\big)$,

where $D_\sigma = \|\mathbf{w}^0 - \mathbf{w}^*\|^2_{\mathbf{A}_\sigma}$ and $\mathbf{w}^*$ is the global minimizer of F, the DP-LSSGD output $\widetilde{\mathbf{w}} = \sum_{k=0}^{T-1} \eta_k / \big(\sum_{i=0}^{T-1} \eta_i\big) \mathbf{w}^k$ satisfies the following utility

$$\mathbb{E}\big(F(\widetilde{\mathbf{w}}) - F(\mathbf{w}^*)\big) \leq \frac{C_2 G \sqrt{6\gamma(D_\sigma + G^2/b) d \log(1/\delta)}}{n\epsilon},$$

where $\gamma = 1/d \sum_{i=1}^{d} 1/[1 + 2\sigma - 2\sigma\cos(2\pi i/d)]$, C_1, C_2 are universal constants.

Proposition 5.3.5. In Theorem 5.3.4, $\gamma = \frac{1+\omega^d}{(1-\omega^d)\sqrt{4\sigma+1}}$, where $\omega = \frac{2\sigma+1-\sqrt{4\sigma+1}}{2\sigma} < 1$. That is, γ converge to 0 almost exponentially as the dimension, d, increases.

Remark 5.3.6. In the above utility bound for convex optimization, for different σ ($\sigma = 0$ corresponds to DP-SGD), the only difference lies in the term $\gamma(D_\sigma + G^2)$. The first part γD_σ depends on the gap between initialization $\mathbf{w}^0$ and the optimal solution $\mathbf{w}^*$. The second part γG^2 decrease monotonically as σ increases. σ should be selected to get an optimal trade-off between these two parts. Based on our test on multi-class logistic regression for MNIST classification, $\sigma \neq 0$ always outperforms the case when $\sigma = 0$.

For nonconvex ERM, DP-LSSGD has the following utility bound measured in gradient norm.

Theorem 5.3.7 (Utility Guarantee for nonconvex optimization)**.** Suppose that F is nonconvex and each component function f_i is G-Lipschitz and has L-Lipschitz continuous gradient. Given $\epsilon, \delta > 0$, under the same conditions of Theorem 5.3.2 on ν^2, α, if we choose $\eta = 1/\sqrt{T}$ and $T = C_1(D_F + LG^2/b)n^2\epsilon^2 / \big(dLG^2 \log(1/\delta)\big)$, where $D_F = F(\mathbf{w}^0) - F(\mathbf{w}^*)$ with $\mathbf{w}^*$ being the global minimum of F, then the DP-LSSGD output $\widetilde{\mathbf{w}} = \sum_{k=0}^{T-1} \mathbf{w}^k / T$ satisfies the following utility

$$\mathbb{E}\|\nabla F(\widetilde{\mathbf{w}})\|^2_{\mathbf{A}_\sigma^{-1}} \leq C_2 \frac{G\sqrt{\beta d L (2D_F + LG^2/b)\log(1/\delta)}}{n\epsilon},$$

where $\beta = 1/d \sum_{i=1}^{d} 1/[1 + 2\sigma - 2\sigma\cos(2\pi i/d)]^2$, C_1, C_2 are universal constants.

Proposition 5.3.8. In Theorem 5.3.7, $\beta = \frac{2\omega^{2d+1} - \xi\omega^{2d} + 2\xi d\omega^d - 2\omega + \xi}{\sigma^2\xi^3(1-\omega^d)^2}$, where $\omega = \frac{2\sigma+1-\sqrt{4\sigma+1}}{2\sigma}$ and $\xi = -\frac{\sqrt{1+4\sigma}}{\sigma}$. Therefore, $\beta \in (0, 1)$.

It is worth noting that if we use the ℓ_2-norm instead of the induced norm, we have the following utility guarantee

$$\mathbb{E}\|\nabla F(\widetilde{\mathbf{w}})\|_2^2 \leq \frac{\mathbb{E}\|\nabla F(\widetilde{\mathbf{w}})\|_{\mathbf{A}_\sigma^{-1}}^2}{\lambda_{\min}(\mathbf{A}_\sigma^{-1})} \leq (1+4\sigma)\mathbb{E}\|\nabla F(\widetilde{\mathbf{w}})\|_{\mathbf{A}_\sigma^{-1}}^2 \leq 4\zeta\frac{G\sqrt{6dL(2D_F+LG^2)\log(1/\delta)}}{n\epsilon}$$

where $\zeta = \sqrt{\frac{1}{d}\sum_{i=1}^{d}\frac{(1+4\sigma)^2}{(1+2\sigma-2\sigma\cos(2\pi i/d))^2}} > 1$. In the ℓ_2-norm, DP-LSSGD has a bigger utility upper bound than DP-SGD (set $\sigma = 0$ in ζ). However, this does not mean that DP-LSSGD has worse performance. We provide an example to support this claim in the appendix.

5.4 Experiments

In this section, we verify the efficiency of DP-LSSGD in training multi-class logistic regression and CNNs for MNIST and CIFAR10 classification. We use $\mathbf{v} \leftarrow \mathbf{v}/\max(1, \|\mathbf{v}\|_2/C)$ (ACG16a) to clip the gradient ℓ_2-norms of the CNNs to C. The gradient clipping guarantee the Lipschitz condition for the objective functions. We train all the models below with $(\epsilon, 10^{-5})$-DP guarantee for different ϵ. For Logistic regression we use the privacy budget given by Theorem 5.3.2, and for CNNs we use the privacy budget in the Tensorflow privacy (Aa19). We checked that these two privacy budgets are consistent.

5.4.1 Logistic Regression for MNIST Classification

We ran 50 epochs of DP-LSSGD with learning rate scheduled as $1/t$ with t being the index of the iteration to train the ℓ_2-regularized (regularization constant 10^{-4}) multi-class logistic regression. We split the training data into 50K/10K with batch size 128 for cross-validation. We plot the evolution of training and validation loss over iterations for privacy budgets $(0.2, 10^{-5})$ and $(0.1, 10^{-5})$ in Figure 5.3. We see that the training loss curve of DP-SGD ($\sigma = 0$) is much higher and more oscillatory (log-scale on the y-axis) than that of DP-LSSGD ($\sigma = 1, 3$). Also, the validation loss of the model trained by DP-LSSGD decays faster and has a much smaller loss value than that of the model trained by DP-SGD. Moreover, when

the privacy guarantee gets stronger, the utility improvement by DP-LSSGD becomes more significant.

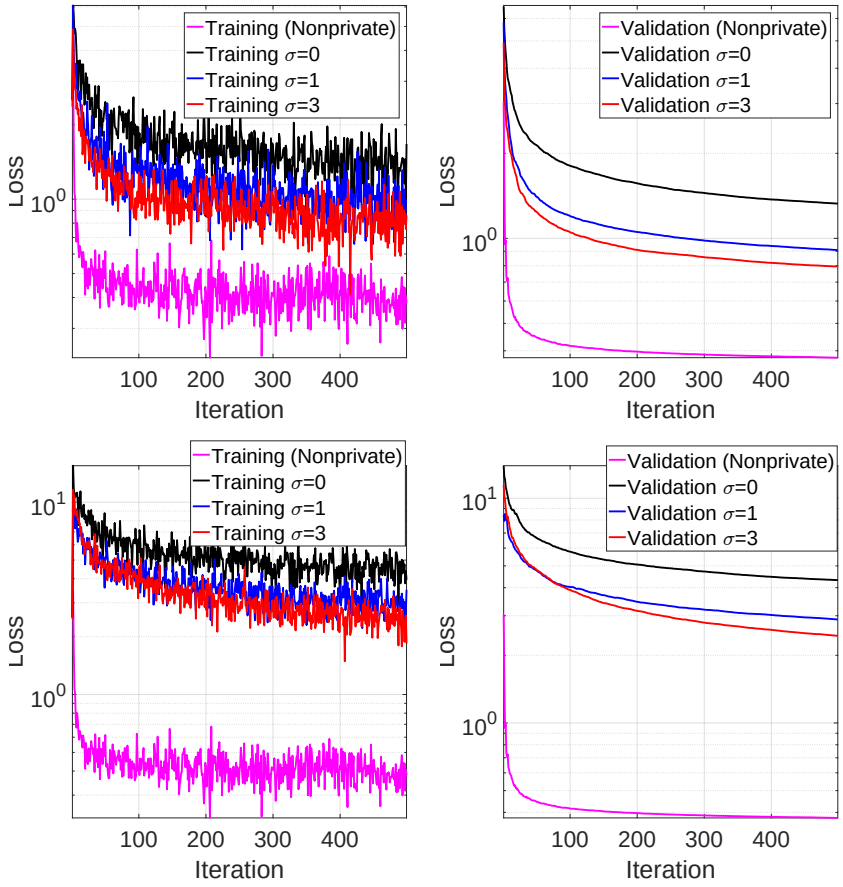


Figure 5.3: Training and validation losses of the multi-class logistic regression by DP-LSSGD. (a) and (b): training and validation curves with $(0.2, 10^{-5})$-DP guarantee; (c) and (d): training and validation curves with $(0.1, 10^{-5})$-DP guarantee. (Average over 5 runs)

Next, consider the testing accuracy of the multi-class logistic regression trained with $(\epsilon, 10^{-5})$-DP guarantee by DP-LSSGD includes $\sigma = 0$, i.e., DP-SGD. We list the test accuracy of logistic regression trained in different settings in Table 5.2. These results reveal that DP-LSSGD with $\sigma = 1, 2, 3$ can improve the accuracy of the trained private model and also reduce the variance, especially when the privacy guarantee is very strong, e.g., $(0.1, 10^{-5})$.

Table 5.2: Testing accuracy of the multi-class logistic regression trained by DP-LSSGD with $(\epsilon, \delta = 10^{-5})$-DP guarantee and different LS parameter σ. Unit: %. (5 runs)

ϵ	0.30	0.25	0.20	0.15	0.10
$\sigma = 0$	81.74 ± 0.96	81.45 ± 1.59	78.92 ± 1.14	77.03 ± 0.69	73.49 ± 1.60
$\sigma = 1$	84.21 ± 0.51	83.27 ± 0.35	81.56 ± 0.79	79.46 ± 1.33	76.29 ± 0.53
$\sigma = 2$	84.23 ± 0.65	**83.65 ± 0.76**	82.15 ± 0.59	80.77 ± 1.26	76.31 ± 0.93
$\sigma = 3$	**85.11 ± 0.45**	82.97 ± 0.48	**82.22 ± 0.28**	**80.81 ± 1.03**	**77.13 ± 0.77**

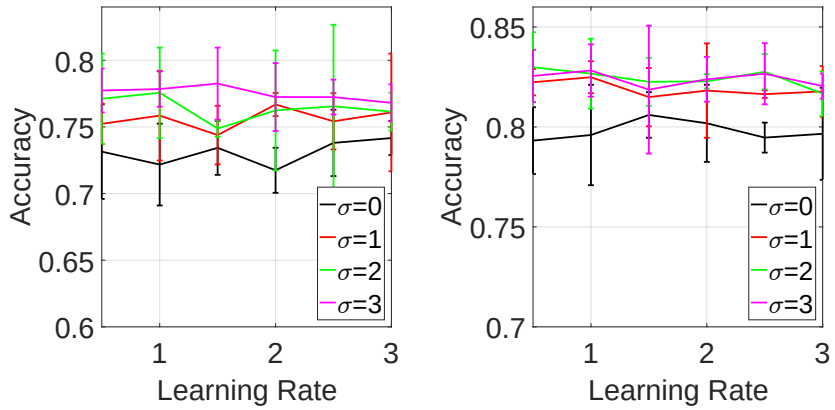


Figure 5.4: Accuracy of the logistic regression on MNIST when different learning rates are used to train the model. Left: $(0.1, 10^{-5})$-DP; Right: $(0.2, 10^{-5})$-DP.

5.4.1.1 The Effects of Step Size

We know that the step size in DP-SGD/DP-LSSGD may affect the accuracy of the trained private models. We try different step size scheduling of the form $\{a/t | a = 0.5, 1.0, 1.5, 2.0, 2.5, 3.0\}$, where t is again the index of iteration, and all the other hyper-parameters are used the same as before. Figure. 5.4 plots the test accuracy of the logistic regression model trained with different learning rate scheduling and different privacy budget. We see that the private logistic regression model trained by DP-LSSGD always outperforms DP-SGD.

5.4.2 CNN for MNIST and CIFAR10 Classification

In this subsection, we consider training a small CNN [2] with DP-guarantee for MNIST classification. We implement DP-LSSGD and DP-LSAdam (KB15) (simply replace the noisy gradient in DP-Adam in the Tensorflow privacy with the Laplacian smoothed surrogate) into the Tensorflow privacy framework (Aa19). We use the default learning rate 0.15 for DP-(LS)SGD and 0.001 for DP-(LS)Adam and decay them by a factor of 10 at the 10K-th iteration, norm clipping (1), batch size (256), and micro-batches (256). We vary the noise multiplier (NM), and larger NM guarantees stronger DP. As shown in Figure 5.5, the privacy budget increases at exactly the same speed (dashed red line) for four optimization algorithms. When the NM is large, i.e., DP-guarantee is strong, DP-SGD performs very well in the initial period. However, after a few epochs, the validation accuracy gets highly oscillatory and decays. DP-LSSGD can mitigate the training instability issue of DP-SGD. DP-Adam outperforms DP-LSSGD, and DP-LSAdam can further improve validation accuracy on top of DP-Adam.

Next, we consider the effects of the LS constant (σ) and the learning rate in training the DP-CNN for MNIST classification. We fixed the NM to be 10, and run 60 epochs of DP-SGD and DP-LSSGD with different σ and different learning rate. We show the comparison of DP-SGD with DP-LSSGD with different σ in the left panel of Figure 5.7, and we see that as σ increases it becomes more stable in training CNNs with DP-guarantee even though initially it becomes slightly slower. In the middle panel of Figure 5.7, we plot the evolution of validation accuracy curves of the DP-CNN trained by DP-SGD and DP-LSSGD with different learning rate, where the solid lines represent results for DP-LSSGD and dashed lines for DP-SGD. DP-LSSGD outperforms DP-SGD in all learning rates tested, and DP-LSSGD is much more stable than DP-SGD when a larger learning rate is used.

Finally, we go back to the accuracy degradation problem raised in Figure 5.1. As shown

[2] `github.com/tensorflow/privacy/blob/master/tutorials/mnist_dpsgd_tutorial.py`

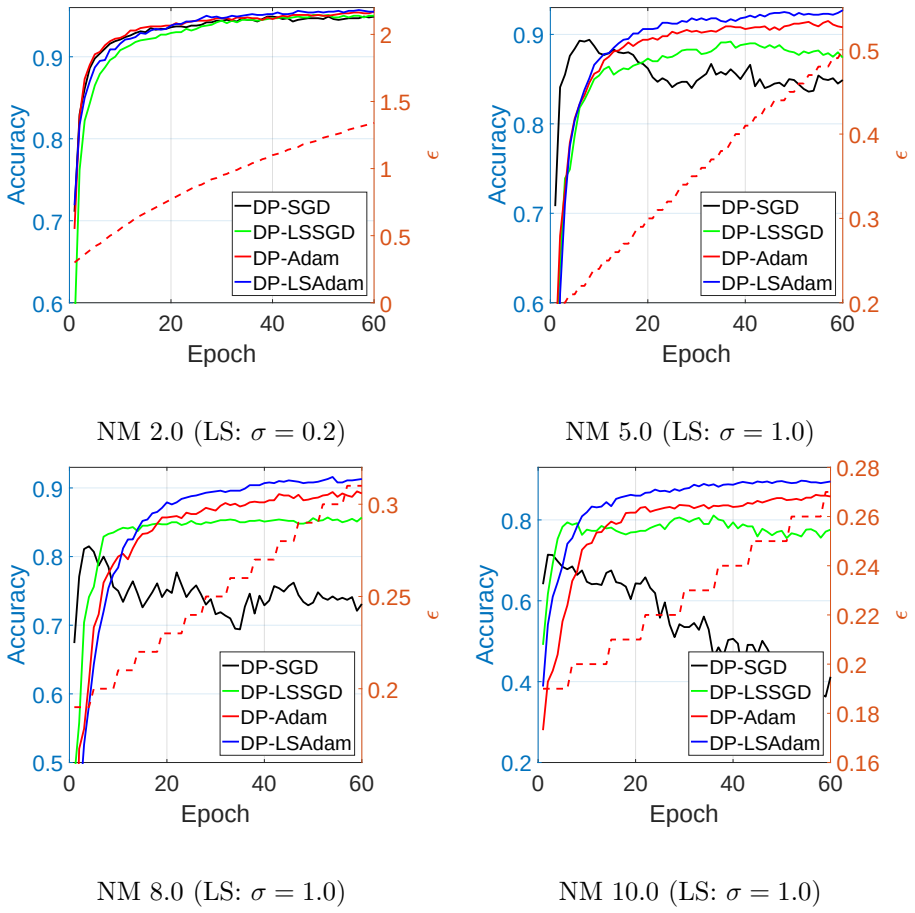


Figure 5.5: Performance comparison (validation accuracy) between different DP optimization algorithms in training CNN for MNIST classification with a fixed $\delta = 10^{-5}$.

in Figure 5.3, LS can efficiently reduce both training and validation losses in training multi-class logistic regression for MNIST classification. Moreover, as shown in the right panel of Figure 5.7, DP-LSSGD can improve the testing accuracy of the CNN used above significantly. In particular, DP-LSSGD improves the testing accuracy of CNN by 3.2% and 5.0% for $(0.4, 10^{-5})$ and $(0.2, 10^{-5})$, respectively, on top of DP-SGD. DP-LSAdam can further boost test accuracy. All the accuracies associated with any given privacy budget in Figure 5.7 (right panel), are the optimal ones searched over the results obtained in the above experiments with different learning rate, number of epochs, and NM.

5.4.3 CNN for CIFAR10 Classification

In this section, we will show that LS can also improve the utility of the DP-CNN trained by DP-SGD and DP-Adam for CIFAR10 classification. We simply replace the CNN architecture used above for MNIST classification with the benchmark architecture in the Tensorflow tutorial [3] for CIFAR10 classification. Also, we use the same set of parameters as that used for training DP-CNN for MNIST classification except we fixed the noise multiplier to be 2.0 and clip the gradient ℓ_2 norm to 3. As shown in Figure 5.6, LS can significantly improve the validation accuracy of the model trained by DP-SGD and DP-Adam, and the DP guarantee for all these algorithms are the same (dashed line in Figure 5.6).

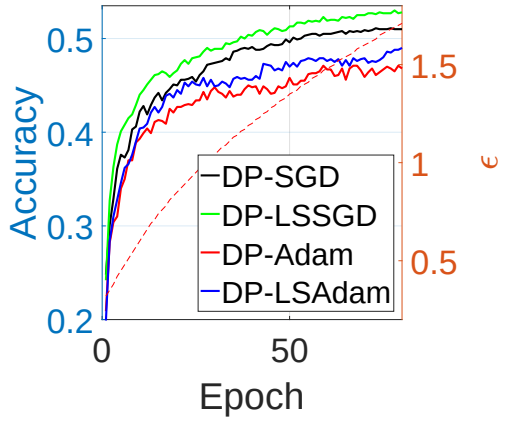


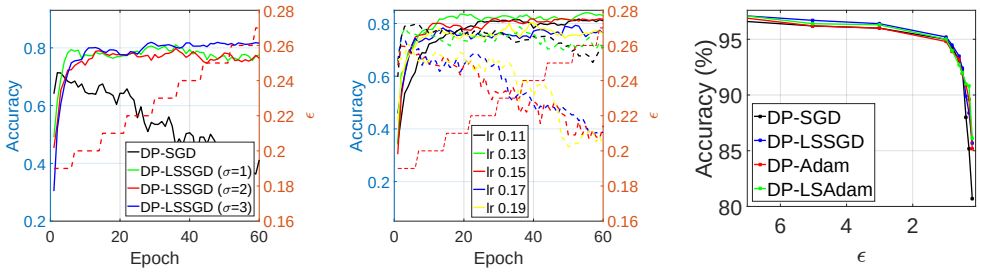
Figure 5.6: Performance comparison between different differentially private optimization algorithms in training CNN for CIFAR10 classification with a fixed $\delta = 10^{-5}$.

5.5 Proof of the Main Theorems

5.5.1 Privacy Guarantee

To prove the privacy guarantee in Theorem 5.3.2, we first introduce the following ℓ_2-sensitivity.

[3] github.com/tensorflow/models/tree/master/tutorials/image/cifar10



Diff LS (σ) Diff Learning Rate ϵ vs. Accuracy

Figure 5.7: Left & middle panels: Contrasting performance (validation acc) of DP-SGD and DP-LSSGD with different σ and different learning rate. Right panel: ϵ vs. Testing accuracy of the private models trained by different DP-optimization algorithms with a fixed $\delta = 10^{-5}$.

Definition 5.5.1 (ℓ_2-Sensitivity)**.** For any given function $f(\cdot)$, the ℓ_2-sensitivity of f is defined by

$$\Delta(f) = \max_{\|S-S'\|_1=1} \|f(S) - f(S')\|_2,$$

where $\|S - S'\|_1 = 1$ means the data sets S and S' differ in only one entry.

We will adapt the concepts and techniques of Rényi DP (RDP) to prove the DP-guarantee of the proposed DP-LSSGD.

Definition 5.5.2 (RDP)**.** For $\alpha > 1$ and $\rho > 0$, a randomized mechanism $\mathcal{M} : \mathcal{S}^n \to \mathcal{R}$ satisfies (α, ρ)-Rényi DP, i.e., (α, ρ)-RDP, if for all adjacent datasets $S, S' \in \mathcal{S}^n$ differing by one element, we have

$$D_\alpha\big(\mathcal{M}(S)||\mathcal{M}(S')\big) := \frac{1}{\alpha - 1} \log \mathbb{E}\left(\frac{\mathcal{M}(S)}{\mathcal{M}(S')}\right)^\alpha \leq \rho,$$

where the expectation is taken over $\mathcal{M}(S')$.

Lemma 5.5.3. (WJE19) Given a function $q : \mathcal{S}^n \to \mathcal{R}$, the Gaussian Mechanism $\mathcal{M} = q(S) + \mathbf{u}$, where $\mathbf{u} \sim N(0, \sigma^2\mathbf{I})$, satisfies $(\alpha, \alpha\Delta^2(q)/(2\sigma^2))$-RDP. In addition, if we apply the mechanism $\mathcal{M}$ to a subset of samples using uniform sampling without replacement, $\mathcal{M}$

satisfies $(\alpha, 5\tau^2\Delta^2(q)\alpha/\sigma^2)$-RDP given $\sigma'^2 = \sigma^2/\Delta^2(q) \geq 1.5$, $\alpha \leq \log(1/\tau\big(1+\sigma'^2\big))$, where τ is the subsample rate.

Lemma 5.5.4. (Mir17) If k randomized mechanisms $\mathcal{M}_i : \mathcal{S}^n \to \mathcal{R}$, for $i \in [k]$, satisfy (α, ρ_i)-RDP, then their composition $\big(\mathcal{M}_1(S), \ldots, \mathcal{M}_k(S)\big)$ satisfies $(\alpha, \sum_{i=1}^k \rho_i)$-RDP. Moreover, the input of the i-th mechanism can be based on outputs of the previous $(i-1)$ mechanisms.

Lemma 5.5.5. If a randomized mechanism $\mathcal{M} : \mathcal{S}^n \to \mathcal{R}$ satisfies (α, ρ)-RDP, then $\mathcal{M}$ satisfies $(\rho + \log(1/\delta)/(\alpha-1), \delta)$-DP for all $\delta \in (0,1)$.

With the definition (Def. 5.5.2) and guarantees of RDP (Lemmas 5.5.3 and 5.5.4), and the connection between RDP and (ϵ, δ)-DP (Lemma 5.5.5), we can prove the following DP-guarantee for DP-LSSGD.

Proof of Theorem 5.3.2. Let us denote the update of DP-SGD and DP-LSSGD at the k-th iteration starting from any given points $\mathbf{w}^k$ and $\widetilde{\mathbf{w}}^k$, respectively, as

$$\mathbf{w}^{k+1} = \mathbf{w}^k - \eta_k \bigg(\frac{1}{b} \sum_{i_k \in \mathcal{B}_k} \nabla f_{i_k}(\mathbf{w}^k) + \mathbf{n} \bigg), \tag{5.5.1}$$

and

$$\widetilde{\mathbf{w}}^{k+1} = \widetilde{\mathbf{w}}^k - \eta_k \mathbf{A}_\sigma^{-1} \bigg(\frac{1}{b} \sum_{i_k \in \mathcal{B}_k} \nabla f_{i_k}(\widetilde{\mathbf{w}}^k) + \mathbf{n} \bigg), \tag{5.5.2}$$

where $\mathcal{B}_k$ is a mini batch that are drawn uniformly from $[n]$, and $|\mathcal{B}_k| = b$ is the mini batch size.

We will show that with the aforementioned Gaussian noise $\mathcal{N}(0, \nu^2)$ for each coordinate of $\mathbf{n}$, the output of DP-SGD, $\widetilde{\mathbf{w}}$, after T iterations is (ϵ, δ)-DP. Let us consider the mechanism $\widehat{\mathcal{M}}_k = \frac{1}{b}\sum_{i_k \in \mathcal{B}_k} \nabla f_{i_k}(\mathbf{w}^k) + \mathbf{n}$, and $\mathcal{M}_k = \frac{n}{b}\nabla F(\mathbf{w}^k) + \mathbf{n}$ with the query $\mathbf{q}_k = \frac{n}{b}\nabla F(\mathbf{w}^k)$. We have the ℓ_2-sensitivity of $\mathbf{q}_k$ as $\Delta(\mathbf{q}_k) = \|\nabla f_{i_k}(\mathbf{w}^k) - \nabla f_{i'_k}(\mathbf{w}^k)\|_2 \leq \frac{2G}{b}$. According to Lemma 5.5.3, if we add noise with variance

$$\nu^2 = \frac{20T\alpha G^2}{n^2 \epsilon \mu},$$

the mechanism $\mathcal{M}_k$ will satisfy $\big(\alpha, (n^2\epsilon\mu/b^2)/\big(10T\big)\big)$-RDP. By post-processing theorem, we immediately have that under the same noise, $\widetilde{\mathcal{M}}_k = \mathbf{A}_\sigma^{-1}(\nabla F(\mathbf{w}^k) + \mathbf{n})$ also satisfies $\big(\alpha, (n^2\epsilon\mu/b^2)/\big(10T\big)\big)$-RDP. According to Lemma 5.5.3, $\widehat{\mathcal{M}}_k$ will satisfy $\big(\alpha, \mu\epsilon/T\big)$-RDP provided that $\nu^2/\Delta(\mathbf{q}_k)^2 \geq 1.5$, because $\tau = b/n$. Let $\alpha = \log(1/\delta)/\big((1-\mu)\epsilon\big) + 1$, we obtain that $\widehat{\mathcal{M}}_k$ satisfies $\big(\log(1/\delta)/\big((1-\mu)\epsilon\big) + 1, \mu\epsilon/T\big)$-RDP as long as we have

$$\frac{\nu^2}{\Delta(\mathbf{q}_k)^2} = \frac{5T\alpha b^2}{n^2\epsilon\mu} \geq 1.5.$$

In addition, we have

$$\frac{1}{\tau\big(1+\nu^2/\Delta(\mathbf{q}_k)^2\big)} = \frac{\mu n^3\epsilon}{5b^3T\alpha + \mu b n^2\epsilon},$$

which implies that $\alpha = \log(1/\delta)/\big((1-\mu)\epsilon\big) + 1 \leq \log\big(\mu n^3\epsilon/(5b^3T\alpha + \mu b n^2\epsilon)\big)$. Therefore, according to Lemma 5.5.4, we have $\mathbf{w}^k$ satisfies $\big(\log(1/\delta)/\big((1-\mu)\epsilon\big) + 1, k\mu\epsilon/T\big)$-RDP. Finally, by Lemma 5.5.5, we have $\mathbf{w}^k$ satisfies $\big(k\mu\epsilon/T + (1-\mu)\epsilon, \delta\big)$-DP. Therefore, the output of DP-SGD, $\widetilde{\mathbf{w}}$, is (ϵ, δ)-DP. □

Remark 5.5.6. In the above proof, we used the following estimate of the ℓ_2 sensitivity

$$\Delta(\mathbf{q}_k) = \|\mathbf{A}_\sigma^{-1}\nabla f_i(\mathbf{w}^k) - \mathbf{A}_\sigma^{-1}\nabla f_{i'}(\mathbf{w}^k)\|_2/n \leq 2G/n.$$

Indeed, let $\mathbf{g} = \nabla f_i(\mathbf{w}^k) - \nabla f_{i'}(\mathbf{w}^k)$ and $\mathbf{d} = \mathbf{A}_\sigma^{-1}\mathbf{g}$, then according to (OWY18) we have

$$\|\mathbf{d}\|_2 + 2\sigma\frac{\|\mathbf{D}_+\mathbf{d}\|_2^2}{d} + \sigma^2\frac{\|\mathbf{L}\mathbf{d}\|_2^2}{d} = \|\mathbf{g}\|_2,$$

where d is the dimension of $\mathbf{d}$, and

$$\mathbf{D}_+ = \begin{bmatrix} -1 & 1 & 0 & \dots & 0 & 0 \\ 0 & -1 & 1 & \dots & 0 & 0 \\ 0 & 0 & -1 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & 0 & 0 & \dots & 0 & -1 \end{bmatrix}.$$

Moreover, if we assume the $\mathbf{g}$ is randomly sampled from a unit ball in a high dimensional space, then a high probability estimation of the compression ratio of the ℓ_2 norm can be derived from Lemma. 5.5.8.

Numerical experiments show that $\|\mathbf{A}_\sigma^{-1}\nabla f_i(\mathbf{w}^k) - \mathbf{A}_\sigma^{-1}\nabla f_{i'}(\mathbf{w}^k)\|_2$ is much less than $\|\nabla f_i(\mathbf{w}^k) - \nabla f_{i'}(\mathbf{w}^k)\|_2$, so for the above noise, it can give much stronger privacy guarantee.

5.5.2 Utility Guarantee – Convex Optimization

To prove the utility guarantee for convex optimization, we first show that the LS operator compresses the ℓ_2 norm of any given Gaussian random vector with a specific ratio in expectation.

Lemma 5.5.7. Let $\boldsymbol{x} \in \mathbb{R}^d$ be the standard Gaussian random vector. Then

$$\mathbb{E}\|\boldsymbol{x}\|^2_{\mathbf{A}_\sigma^{-1}} = \sum_{i=1}^{d} \frac{1}{1 + 2\sigma - 2\sigma\cos(2\pi i/d)},$$

where $\|\boldsymbol{x}\|^2_{\mathbf{A}_\sigma^{-1}} \doteq \langle \boldsymbol{x}, \mathbf{A}_\sigma^{-1}\boldsymbol{x}\rangle$ is the square of the induced norm of $\boldsymbol{x}$ by the matrix $\mathbf{A}_\sigma^{-1}$.

Proof of Lemma 5.5.7. Let the eigenvalue decomposition of $\mathbf{A}_\sigma^{-1}$ be $\mathbf{A}_\sigma^{-1} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$, where $\mathbf{\Lambda}$ is a diagonal matrix with $\Lambda_{ii} = \frac{1}{1+2\sigma-2\sigma\cos(2\pi i/d)}$ We have

$$\begin{aligned}
\mathbb{E}\|\boldsymbol{x}\|^2_{\mathbf{A}_\sigma^{-1}} &= \mathbb{E}[\mathrm{Tr}(\boldsymbol{x}^\top\mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top\boldsymbol{x})] \\
&= \sum_{i=1}^{d} \Lambda_{ii} \\
&= \sum_{i=1}^{d} \frac{1}{1 + 2\sigma - 2\sigma\cos(2\pi i/d)} = \gamma.
\end{aligned}$$

□

Proof of Theorem 5.3.4. Recall that we have the following update rule $\mathbf{w}^{k+1} = \mathbf{w}^k - \eta_k\mathbf{A}_\sigma^{-1}(\nabla f_{i_k}(\mathbf{w}^k) + \mathbf{n})$, where i_k are drawn uniformly from $[n]$, and $\mathbf{n} \sim \mathcal{N}(0, \nu^2\mathbf{I})$. Let $\nabla f_{\mathcal{B}_k} = \sum_{i_k\in\mathcal{B}_k} \nabla f_{i_k}(\mathbf{w}^k)/b$,

observe that

$$\begin{aligned}\|\mathbf{w}^{k+1}-\mathbf{w}^*\|_{\mathbf{A}_\sigma}^2 &= \|\mathbf{w}^k-\eta_k\mathbf{A}_\sigma^{-1}(\nabla f_{\mathcal{B}_k}(\mathbf{w}^k)+\mathbf{n})-\mathbf{w}^*\|_{\mathbf{A}_\sigma}^2\\ &= \|\mathbf{w}^k-\mathbf{w}^*\|_{\mathbf{A}_\sigma}^2+\eta_k^2\big(\big\|\mathbf{A}_\sigma^{-1}\big(\nabla f_{\mathcal{B}_k}(\mathbf{w}^k)-\nabla F(\mathbf{w}^k)+\nabla F(\mathbf{w}^k)\big)\big\|_{\mathbf{A}_\sigma}^2+\|\mathbf{A}_\sigma^{-1}\mathbf{n}\|_{\mathbf{A}_\sigma}^2\\ &\quad+2\langle\mathbf{A}_\sigma^{-1}\nabla f_{\mathcal{B}_k}(\mathbf{w}^k),\mathbf{n}\rangle\big)-2\eta_k\langle\nabla f_{\mathcal{B}_k}(\mathbf{w}^k)+\mathbf{n},\mathbf{w}^k-\mathbf{w}^*\rangle.\end{aligned}$$

Taking expectation with respect to $\mathcal{B}_k$ and $\mathbf{n}$ given $\mathbf{w}^k$, we have

$$\begin{aligned}\mathbb{E}\|\mathbf{w}^{k+1}-\mathbf{w}^*\|_{\mathbf{A}_\sigma}^2 &= \mathbb{E}\|\mathbf{w}^k-\mathbf{w}^*\|_{\mathbf{A}_\sigma}^2-2\eta_k\mathbb{E}\langle\nabla F(\mathbf{w}^k),\mathbf{w}^k-\mathbf{w}^*\rangle+\eta_k^2\mathbb{E}\|\nabla f_{\mathcal{B}_k}(\mathbf{w}^k)-\nabla F(\mathbf{w}^k)\|_{\mathbf{A}_\sigma^{-1}}^2\\ &\quad+\eta_k^2\mathbb{E}\|\nabla F(\mathbf{w}^k)\|_{\mathbf{A}_\sigma^{-1}}^2+\eta_k^2\mathbb{E}\|\mathbf{n}\|_{\mathbf{A}_\sigma^{-1}}^2.\end{aligned}$$

In addition, we have

$$\mathbb{E}\|\nabla f_{\mathcal{B}_k}(\mathbf{w}^k)-\nabla F(\mathbf{w}^k)\|_{\mathbf{A}_\sigma^{-1}}^2\le\mathbb{E}\|\nabla f_{\mathcal{B}_k}(\mathbf{w}^k)-\nabla F(\mathbf{w}^k)\|_2^2\le\frac{G^2}{b},\tag{5.5.3}$$

and

$$\Big(1-\frac{L\eta_k}{2}\Big)\eta_k\|\nabla F(\mathbf{w}^k)\|_2^2\le F(\mathbf{w}^k)-F(\mathbf{w}^*),\tag{5.5.4}$$

which implies that

$$\eta_k^2\mathbb{E}\|\nabla F(\mathbf{w}^k)\|_{\mathbf{A}_\sigma^{-1}}^2\le\eta_k^2\mathbb{E}\|\nabla F(\mathbf{w}^k)\|_2^2\le\Big(\frac{2}{2-L\eta_k}\Big)\eta_k\mathbb{E}\big(F(\mathbf{w}^k)-F(\mathbf{w}^*)\big)\le\frac{4}{3}\eta_k\mathbb{E}\big(F(\mathbf{w}^k)-F(\mathbf{w}^*)\big),$$

where the last inequality is due to the fact that $\eta_t\le 1/(2L)$. Therefore, we have

$$\mathbb{E}\|\mathbf{w}^{k+1}-\mathbf{w}^*\|_{\mathbf{A}_\sigma}^2\le\mathbb{E}\|\mathbf{w}^k-\mathbf{w}^*\|_{\mathbf{A}_\sigma}^2-\frac{2}{3}\eta_k\mathbb{E}\big(F(\mathbf{w}^k)-F(\mathbf{w}^*)\big)+\eta_k^2\big(G^2/b+\gamma d\nu^2\big),$$

where the inequality is due to the convexity of F, and Lemma 5.5.7. It implies that

$$\frac{2}{3}\eta_k\mathbb{E}\big(F(\mathbf{w}^k)-F(\mathbf{w}^*)\big)\le\big(\mathbb{E}\|\mathbf{w}^k-\mathbf{w}^*\|_{\mathbf{A}_\sigma}^2-\mathbb{E}\|\mathbf{w}^{k+1}-\mathbf{w}^*\|_{\mathbf{A}_\sigma}^2\big)+\eta_k^2(G^2/b+\gamma d\nu^2).$$

Now taking the full expectation and summing up over T iterations, we have

$$\sum_{k=0}^{T-1}\frac{2}{3}\eta_k\mathbb{E}\big(F(\mathbf{w}^k)-F(\mathbf{w}^*)\big)\le D_\sigma+\sum_{k=0}^{T-1}\eta_k^2(G^2/b+\gamma d\nu^2),$$

where $D_\sigma = \|\mathbf{w}^0 - \mathbf{w}^*\|_{\mathbf{A}_\sigma}^2$. Let $v_k = \eta_k / \big(\sum_{k=0}^{T-1} \eta_k\big)$, we have

$$\sum_{k=0}^{T-1} v_k \mathbb{E}\big(F(\mathbf{w}^k) - F(\mathbf{w}^*)\big) \leq \frac{D_\sigma + \sum_{k=0}^{T-1} \eta_k^2 (G^2/b + \gamma d \nu^2)}{2\sum_{k=0}^{T-1} \eta_k / 3}.$$

According to the definition of $\widetilde{\mathbf{w}}$ and the convexity of F, we obtain

$$\begin{aligned}
\mathbb{E}\big(F(\widetilde{\mathbf{w}}) - F(\mathbf{w}^*)\big) &\leq \frac{D_\sigma + \sum_{k=0}^{T-1} \eta_k^2 (G^2/b + \gamma d \nu^2)}{2\sum_{k=0}^{T-1} \eta_k / 3} \\
&\leq \frac{D_\sigma + \sum_{k=0}^{T-1} \eta_k^2 G^2/b}{2\sum_{k=0}^{T-1} \eta_k / 3} + \frac{\sum_{k=0}^{T-1} \eta_k^2}{2\sum_{k=0}^{T-1} \eta_k / 3} \cdot \frac{20\gamma d T G^2 \log(1/\delta)}{n^2 \epsilon^2 \mu (1-\mu)}.
\end{aligned}$$

Let $\eta = 1/\sqrt{T}$ and $T = C_1 (D_\sigma + G^2/b) n^2 \epsilon^2 / \big(\gamma d G^2 \log(1/\delta)\big)$, we can obtain that

$$\mathbb{E}\big(F(\widetilde{\mathbf{w}}) - F(\mathbf{w}^*)\big) \leq \frac{C_2 G \sqrt{\gamma (D_\sigma + G^2/b) d \log(1/\delta)}}{n\epsilon},$$

where C_1, C_2 are universal constants. □

5.5.3 Utility Guarantee – Nonconvex Optimization

To prove the utility guarantee for nonconvex optimization, we need the following lemma, which shows that the LS operator compresses the ℓ_2 norms of any given Gaussian random vector with a specific ratio in expectation.

Lemma 5.5.8. Let $\boldsymbol{x} \in \mathbb{R}^d$ be the standard Gaussian random vector. Then

$$\mathbb{E}\|\mathbf{A}_\sigma^{-1} \boldsymbol{x}\|_2^2 = \sum_{i=1}^{d} \frac{1}{(1 + 2\sigma - 2\sigma \cos(2\pi i/d))^2}.$$

Proof of Lemma 5.5.8. Let the eigenvalue decomposition of $\mathbf{A}_\sigma^{-1}$ be $\mathbf{A}_\sigma^{-1} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$, where $\mathbf{\Lambda}$ is a diagonal matrix with $\Lambda_{ii} = \frac{1}{1+2\sigma-2\sigma\cos(2\pi i/n)}$ We have

$$\begin{aligned}
\mathbb{E}\|\mathbf{A}_\sigma^{-1} \boldsymbol{x}\|_2^2 &= \mathbb{E}[\mathrm{Tr}(\boldsymbol{x}^\top \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top \boldsymbol{x})] \\
&= \mathbb{E}[\mathrm{Tr}(\boldsymbol{x}^\top \mathbf{U}\mathbf{\Lambda}^2\mathbf{U}^\top \boldsymbol{x})] \\
&= \sum_{i=1}^{d} \Lambda_{ii}^2 \\
&= \sum_{i=1}^{d} \frac{1}{(1 + 2\sigma - 2\sigma \cos(2\pi i/d))^2} = \beta.
\end{aligned}$$

□

Proof of Theorem 5.3.7. Recall that we have the following update rule $\mathbf{w}^{t+1} = \mathbf{w}^k - \eta_k \mathbf{A}_\sigma^{-1}(\nabla f_{i_k}(\mathbf{w}^k) + \mathbf{n})$, where i_k are drawn uniformly from $[n]$, and $\mathbf{n} \sim \mathcal{N}(0, \nu^2\mathbf{I})$. Let $\nabla f_{\mathcal{B}_k} = \sum_{i_k \in \mathcal{B}_k} \nabla f_{i_k}(\mathbf{w}^k)/b$, since F is L-smooth, we have

$$
\begin{aligned}
F(\mathbf{w}^{k+1}) &\leq F(\mathbf{w}^k) + \langle \nabla F(\mathbf{w}^k), \mathbf{w}^{k+1} - \mathbf{w}^k \rangle + \frac{L}{2}\|\mathbf{w}^{k+1} - \mathbf{w}^k\|_2^2 \\
&= F(\mathbf{w}^k) - \eta_k \langle \nabla F(\mathbf{w}^k), \mathbf{A}_\sigma^{-1}(\nabla f_{\mathcal{B}_k}(\mathbf{w}^k) + \mathbf{n}) \rangle \\
&\quad + \frac{\eta_k^2 L}{2}\Big(\big\| \mathbf{A}_\sigma^{-1}\big(\nabla f_{\mathcal{B}_k}(\mathbf{w}^k) - \nabla F(\mathbf{w}^k) + \nabla F(\mathbf{w}^k)\big) \big\|_2^2 + \|\mathbf{A}_\sigma^{-1}\mathbf{n}\|_2^2 + 2\langle \mathbf{A}_\sigma^{-1}\nabla f_{\mathcal{B}_k}(\mathbf{w}^k), \mathbf{A}_\sigma^{-1}\mathbf{n} \rangle \Big).
\end{aligned}
$$

Taking expectation with respect to $\mathcal{B}_k$ and $\mathbf{n}$ given $\mathbf{w}^k$, we have

$$
\begin{aligned}
\mathbb{E}F(\mathbf{w}^{k+1}) &\leq \mathbb{E}F(\mathbf{w}^k) - \eta_k \mathbb{E}\langle \nabla F(\mathbf{w}^k), \mathbf{A}_\sigma^{-1}\nabla f_{\mathcal{B}_k}(\mathbf{w}^k) \rangle + \frac{\eta_k^2 L}{2}\Big(\mathbb{E}\|\mathbf{A}_\sigma^{-1}\big(\nabla f_{\mathcal{B}_k}(\mathbf{w}^k) - \nabla F(\mathbf{w}^k)\big)\|_2^2 \\
&\quad + \mathbb{E}\|\mathbf{A}_\sigma^{-1}\nabla F(\mathbf{w}^k)\big\|_2^2 + \mathbb{E}\|\mathbf{A}_\sigma^{-1}\mathbf{n}\|_2^2 \Big) \\
&\leq \mathbb{E}F(\mathbf{w}^k) - \eta_k\Big(1 - \frac{\eta_k L}{2}\Big)\mathbb{E}\|\nabla F(\mathbf{w}^k)\|_{\mathbf{A}_\sigma^{-1}}^2 + \frac{\eta_k^2 L}{2}(G^2/b + d\beta\nu^2) \\
&\leq \mathbb{E}F(\mathbf{w}^k) - \frac{\eta_k}{2}\mathbb{E}\|\nabla F(\mathbf{w}^k)\|_{\mathbf{A}_\sigma^{-1}}^2 + \frac{\eta_k^2 L(G^2 + d\beta\nu^2)}{2},
\end{aligned}
$$

where the second inequality uses Lemma 5.5.8, the inequality (5.5.3), and the last inequality is due to $1 - \eta_k L/2 > 1/2$. Now taking the full expectation and summing up over T iterations, we have

$$
\mathbb{E}F(\mathbf{w}^T) \leq F(\mathbf{w}^0) - \sum_{k=1}^{T-1} \frac{\eta_k}{2}\mathbb{E}\|\nabla F(\mathbf{w}^k)\|_{\mathbf{A}_\sigma^{-1}}^2 + \sum_{k=1}^{T-1} \frac{\eta_k^2 L(G^2/b + d\beta\nu^2)}{2}.
$$

If we choose fix step size, i.e., $\eta_k = \eta$, and rearranging the above inequality, and using $F(\mathbf{w}^0) - \mathbb{E}F(\mathbf{w}^T) \leq F(\mathbf{w}^0) - F(\mathbf{w}^*)$, we get

$$
\frac{1}{T}\sum_{k=1}^{T-1} \mathbb{E}\|\nabla F(\mathbf{w}^k)\|_{\mathbf{A}_\sigma^{-1}}^2 \leq \frac{2}{\eta T}\big(F(\mathbf{w}^0) - F(\mathbf{w}^*)\big) + \eta L(G^2/b + d\beta\nu^2),
$$

which implies that

$$
\begin{aligned}
\mathbb{E}\|\nabla F(\widetilde{\mathbf{w}})\|_{\mathbf{A}_\sigma^{-1}}^2 &\leq \frac{2D_F}{\eta T} + \eta L(G^2/b + d\beta\nu^2) \\
&\leq \frac{2D_F}{\eta T} + \eta L\bigg(G^2/b + \frac{20 d\beta T G^2 \log(1/\delta)}{n^2\epsilon^2\mu(1-\mu)}\bigg).
\end{aligned}
$$

Let $\eta = 1/\sqrt{T}$ and $T = C_1(2D_F + LG^2/b)n^2\epsilon^2/\big(dL\beta G^2\log(1/\delta)\big)$, where $D_F = F(\mathbf{w}^0) - F(\mathbf{w}^*)$, we obtain

$$\mathbb{E}\|\nabla F(\widetilde{\mathbf{w}})\|^2_{\mathbf{A}_\sigma^{-1}} \leq C_2\frac{G\sqrt{\beta dL(2D_F + LG^2/b)\log(1/\delta)}}{n\epsilon},$$

where C_1, C_2 are universal constants. □

It is worth noting that if we use the ℓ_2-norm instead of the induced norm, we have the following utility guarantee

$$\mathbb{E}\|\nabla F(\widetilde{\mathbf{w}})\|_2^2 \leq \frac{\mathbb{E}\|\nabla F(\widetilde{\mathbf{w}})\|^2_{\mathbf{A}_\sigma^{-1}}}{\lambda_{\min}(\mathbf{A}_\sigma^{-1})} \leq (1+4\sigma)\mathbb{E}\|\nabla F(\widetilde{\mathbf{w}})\|^2_{\mathbf{A}_\sigma^{-1}} \leq 4\zeta\frac{G\sqrt{6dL(2D_F + LG^2)\log(1/\delta)}}{n\epsilon}$$

where $\zeta = \sqrt{\frac{1}{d}\sum_{i=1}^d \frac{(1+4\sigma)^2}{(1+2\sigma-2\sigma\cos(2\pi i/d))^2}} > 1$. In the ℓ_2-norm, DP-LSSGD has a bigger utility upper bound than DP-SGD (set $\sigma = 0$ in ζ). However, this does not mean that DP-LSSGD has worse performance. To see this point, let us consider the following simple nonconvex function

$$f(x,y) = \begin{cases} \frac{x^2}{4} + y^2, & \text{for } \frac{x^2}{4} + y^2 \leq 1 \\ \sin\left(\frac{\pi}{2}\left(\frac{x^2}{4} + y^2\right)\right), & \text{for } \frac{x^2}{4} + y^2 > 1. \end{cases} \tag{5.5.5}$$

For two points $\mathbf{a}_1 = (2,0)$ and $\mathbf{a}_2 = (1, \sqrt{3}/2)$, the distance to the local minima $\mathbf{a}^* = (0,0)$ are 2 and $\sqrt{7}/2$, while $\|\nabla f(\mathbf{a}_1)\|_2 = 1$ and $\|\nabla f(\mathbf{a}_2)\|_2 = \sqrt{13}/2$. So $\mathbf{a}_2$ is closer to the local minima $\mathbf{a}^*$ than $\mathbf{a}_1$ while its gradient has a larger ℓ_2-norm.

5.6 Calculations of β and γ

5.6.1 Calculation of γ

To prove Proposition 5.3.5, we need the following two lemmas.

Lemma 5.6.1 (Residue Theorem)**.** Let $f(z)$ be a complex function defined on $\mathbb{C}$, then the residue of f around the pole $z = c$ can be computed by the formula

$$\mathrm{Res}(f,c) = \frac{1}{(n-1)!}\lim_{z\to c}\frac{d^{n-1}}{dz^{z-1}}\left((z-c)^n f(z)\right). \tag{5.6.1}$$

where the order of the pole c is n. Moreover,

$$\oint f(z)dz = 2\pi i \sum_{c_i} Res(f, c_i), \tag{5.6.2}$$

where $\{c_i\}$ be the set of pole(s) of $f(z)$ inside $\{z||z| < 1\}$.

The proof of Lemma 5.6.1 can be found in any complex analysis textbook.

Lemma 5.6.2. For $0 \leq \theta \leq 2\pi$, suppose

$$F(\theta) = \frac{1}{1 + 2\sigma(1 - \cos(\theta))},$$

has the discrete-time Fourier transform of series $f[k]$. Then, for integer k,

$$f[k] = \frac{\alpha^{|k|}}{\sqrt{4\sigma + 1}}$$

where

$$\alpha = \frac{2\sigma + 1 - \sqrt{4\sigma + 1}}{2\sigma}$$

Proof. By definition,

$$f[k] = \frac{1}{2\pi} \int_0^{2\pi} F(\theta) e^{ik\theta} \, d\theta = \frac{1}{2\pi} \int_0^{2\pi} \frac{e^{ik\theta}}{1 + 2\sigma(1 - \cos(\theta))} \, d\theta. \tag{5.6.3}$$

We compute (5.6.3) by using Residue theorem. First, note that because $F(\theta)$ is real valued, $f[k] = f[-k]$; therefore, it suffices to compute (5.6.3)) for nonnegative k. Set $z = e^{i\theta}$. Observe that $\cos(\theta) = 0.5(z + 1/z)$ and $dz = izd\theta$. Substituting in (5.6.3) and simplifying yields that

$$f[k] = \frac{-1}{2\pi i \sigma} \oint \frac{z^k}{(z - \alpha_-)(z - \alpha_+)} \, dz, \tag{5.6.4}$$

where the integral is taken around the unit circle, and $\alpha_\pm = \frac{2\sigma+1\pm\sqrt{4\sigma+1}}{2\sigma}$ are the roots of quadratic $-\sigma z^2 + (2\sigma + 1)z - \sigma$. Note that α_- lies within the unit circle; whereas, α_+ lies outside of the unit circle. Therefore, because k is nonnegative, α_- is the only singularity of

the integrand in (5.6.4) within the unit circle. A straightforward application of the Residue Theorem, i.e., Lemma 5.6.1, yields that

$$f[k] = \frac{-\alpha_-^k}{\sigma(\alpha_- - \alpha_+)} = \frac{\alpha^k}{\sqrt{4\sigma+1}}.$$

This completes the proof. □

Proof of Proposition 5.3.5. First observe that we can re-write γ as

$$\frac{1}{d}\sum_{j=0}^{d-1} \frac{1}{1+2\sigma(1-\cos(\frac{2\pi j}{d}))}. \tag{5.6.5}$$

It remains to show that the above summation is equal to $\frac{1+\alpha^d}{(1-\alpha^d)\sqrt{4\sigma+1}}$. This follows by lemmas 5.6.2 and standard sampling results in Fourier analysis (i.e. sampling θ at points $\{2\pi j/d\}_{j=0}^{d-1}$). Nevertheless, we provide the details here for completeness: Observe that that the inverse discrete-time Fourier transform of

$$G(\theta) = \sum_{j=0}^{d-1} \delta\left(\theta - \frac{2\pi j}{d}\right).$$

is given by

$$g[k] = \begin{cases} d/2\pi & \text{if } k \text{ divides } d, \\ 0 & \text{otherwise.} \end{cases}$$

Furthermore, let

$$F(\theta) = \frac{1}{1+2\sigma(1-\cos(\theta))},$$

and use $f[k]$ to denote its inverse discrete-time Fourier transform. Now,

$$\begin{aligned}
\frac{1}{d}\sum_{j=0}^{d-1}\frac{1}{1+2\sigma(1-\cos(\frac{2\pi j}{d}))} &= \frac{1}{d}\int_0^{2\pi} F(\theta)G(\theta) \\
&= \frac{2\pi}{d}\,\mathrm{DTFT}^{-1}[F\cdot G][0] \\
&= \frac{2\pi}{d}(\mathrm{DTFT}^{-1}[F] * \mathrm{DTFT}^{-1}[G])[0] \\
&= \frac{2\pi}{d}\sum_{r=-\infty}^{\infty} f[-r]g[r] \\
&= \frac{2\pi}{d}\sum_{\ell=-\infty}^{\infty} f[-\ell d]\frac{d}{2\pi} \\
&= \sum_{\ell=-\infty}^{\infty} f[-\ell d].
\end{aligned}$$

The proof is completed by substituting the result of lemma 5.6.2 in the above sum and simplifying. □

We list some typical values of γ in Table 5.3.5.

Table 5.3: The values of γ corresponding to some σ and d.

σ	1	2	3	4	5
$d = 1000$	0.447	0.333	0.277	0.243	0.218
$d = 10000$	0.447	0.333	0.277	0.243	0.218
$d = 100000$	0.447	0.333	0.277	0.243	0.218

5.6.2 Calculation of β

The proof of Proposition 5.3.8 is similar as the proof of Proposition 5.3.5. The only difference is that we need to compute

$$f[k] = \frac{1}{2\pi}\int_0^{2\pi} \frac{e^{ik\theta}}{\left(1+2\sigma(1-\cos\theta)\right)^2} d\theta. \tag{5.6.6}$$

By Residue theorem, for $k > 0$ (note that $f[-k] = f[k]$), we have

$$\begin{aligned} f[k] &= \frac{1}{2\pi}\int_0^{2\pi} \frac{e^{ik\theta}}{(1+2\sigma(1-\cos\theta))^2} d\theta \\ &= \frac{1}{2\pi i}\oint \frac{z^{k+1}}{(z+\sigma(2z-z^2-1))^2} dz \\ &= \lim_{z\to\alpha^-} \frac{d}{dz}\left((z-\alpha^-)^2 \frac{z^{k+1}}{(z+\sigma(2z-z^2-1))^2}\right) \\ &= \lim_{z\to\alpha^-} \frac{d}{dz}\left(\frac{z^{k+1}}{\sigma^2(z-\alpha^+)^2}\right) \\ &= \frac{(k+1)\alpha^k}{4\sigma+1} + \frac{2\sigma\alpha^{k+1}}{(4\sigma+1)^{3/2}}, \end{aligned}$$

where $\alpha_- = \frac{2\sigma+1-\sqrt{4\sigma+1}}{2\sigma}$. Therefore, we have

$$\beta = \frac{2\alpha^{2d+1} - \xi\alpha^{2d} + 2\xi d\alpha^d - 2\alpha + \xi}{\sigma^2\xi^3(1-\alpha^d)^2}.$$

We list some typical values of β in Table 5.3.8.

Table 5.4: The values of β corresponding to some σ and d.

σ	1	2	3	4	5
$d = 1000$	0.268	0.185	0.149	0.128	0.114
$d = 10000$	0.268	0.185	0.149	0.128	0.114
$d = 100000$	0.268	0.185	0.149	0.128	0.114

5.7 Laplacian Smoothing and Diffusion Equation

Let $u(x,t)$ be a function defined on the space-time domain $[0,1]\times[0,+\infty)$, suppose it satisfies the following diffusion equation with the Neumann boundary condition

$$\begin{cases} \frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2}, \quad (x,t) \in [0,1]\times[0,+\infty), \\ \frac{\partial u(0,t)}{\partial x} = \frac{\partial u(1,t)}{\partial x} = 0, \ t \in [0,+\infty) \\ u(x,0) = f(x), \quad x \in [0,1] \end{cases} \tag{5.7.1}$$

If we apply the backward Euler in time and central finite difference in space to discretize the governing equation in (5.7.1), we get

$$\mathbf{v}^{\Delta t} - \mathbf{v}^0 = \Delta t \mathbf{L} \mathbf{v}^{\Delta t},$$

where $\mathbf{v}^0$ is the discretization of $f(x)$, and $\mathbf{v}^{\Delta t}$ is the numerical solution of (5.7.1) at time Δt. Therefore, we have

$$\mathbf{v}^{\Delta t} = (I - \Delta t \mathbf{L})^{-1} \mathbf{v}^0,$$

which is the LS with $\sigma = \Delta t$.

5.8 Conclusions

In this paper, we integrated Laplacian smoothing with DP-SGD for privacy-presrving ERM. The resulting algorithm is simple to implement and the extra computational cost compared with the DP-SGD is almost negligible. We show that DP-LSSGD can improve the utility of the trained private ML models both numerically and theoretically.

CHAPTER 6

Distributed Learning Without Distress: Privacy-preserving ERM

6.1 Introduction

In many applications, such as medical research and financial fraud detection, it is valuable to build machine learning models by training on sensitive data. This raises privacy concerns since adversaries may be able to infer information about the training data from the learned model. Model parameters can reveal sensitive information about individual records including specific features of the records (FLJ14) to the presence of particular records in the data set (SSS17b). In the case of neural networks, the model parameters can also inadvertently store sensitive parts of the training data (CLE19). Differential privacy (DN04; Dwo08) aims to thwart such analysis. It provides statistical privacy for individual records by adding random noise to the model parameters. Many works have shown that differential privacy can be used to enable privacy-preserving machine learning in the centralized setting where a single organization owns all the data (CM09; CMS11b; JKT12b; JT13; WYX17a; ZZM17b).

The problem becomes more acute when the data is owned by different organizations that wish to collaboratively learn from their private data. For instance, multiple hospitals may want to collaboratively train a classifier over their patient medical records without disclosing their own records to other hospitals. The goal of distributed machine learning (also referred to as *federated learning* (MRT18a)) is to enable a group of independent data owners to develop a model from their combined data without exposing that data to others.

Multi-party computation (MPC) protocols allow participants to jointly compute a functionality over their private inputs by employing cryptographic techniques like homomorphic encryption, secret sharing, and oblivious transfer. Lindell and Pinkas (LP00) proposed one of the earliest approaches to use MPC for private data mining, which was followed by several works considering different adversarial models or applications (YZW05; VKC08; LP09; PSS09). A recent focus has been to achieve practical and efficient distributed machine learning using MPC protocols (CRT18; WDC18; MZC18), and in certain settings such methods have been shown to scale to learning tasks with hundreds of millions of records (NWI13b; GSB17). However, unlike approaches using differential privacy on the model, these approaches only protect the training data during the learning process; they provide no protection against inference attacks on the resulting model.

Pathak et al. (PRR10) proposed the first differentially-private machine learning in distributed setting. Their method securely aggregates local models and uses output perturbation to achieve differential privacy. However, the noise scales inversely proportional to the smallest data set size of the m parties. This can be improved by a factor of $\sqrt{m}$ by first training differentially-private local models using the method of Chaudhuri et al. (CMS11b), and then performing naïve aggregation of the local models. In this work, we propose an output perturbation method that improves over Pathak et al.'s method by a factor of m by adding the noise inside an MPC with the scale of noise required roughly inversely proportional to the size of the entire data set. Recent works on distributed noise generation (DKM06b; BRB17; HLK17; SCR17) try to achieve a similar bound by requiring parties to add partial noise locally, and combining these noises to ensure differential privacy. However, these methods require additional noise to tolerate corruptions and collusion. More concretely, with a minimum of k honest parties out of m, their noise bound is worse than ours by a factor of $\sqrt{m/k}$. On the other hand, in our approach the noise is generated inside the MPC such that any honest participant can be assured that sufficient noise is added to protect their own privacy even if all other participants are dishonest and colluding.

While these model aggregation approaches are computationally efficient, they tend to produce less accurate global models compared to the centralized setting, especially when the number of data owners is large (in the extreme, when each party has only one training instance). For such scenarios, distributed iterative learning with gradient perturbation is a better option. Shokri and Shmatikov (SS15b) provide such a solution for deep learning, where the local gradients are perturbed and then revealed for updating the global model. Their privacy budget is *per parameter*, however, and not for the entire training so huge total privacy budgets are required. Abadi et al. (ACG16b) proposed a tighter bound on the privacy budget using moments accountant which is applicable to centralized setting. Wang et al. (WYX17a) used the moments accountant to propose iterative learning with gradient perturbation for the centralized setting. We propose a method for distributed setting using *zero-concentrated differential privacy* (BS16b) which achieves similar tight bound on privacy budget. Moreover, we add noise inside MPC after gradient aggregation, thus reducing the noise by a factor of $\sqrt{m}$ compared to the naïve aggregation of noisy gradients. While Chase et al. (CGL17) also achieve similar bound on noise in distributed learning setting, their method considers only the convex case. We achieve a different (and tighter) utility bound for the strong convexity case. Further, Chase et al. use differential privacy which has different composition properties than the zero-concentrated differential privacy that we consider. We also note that the method proposed by Rajkumar and Agarwal (RA12) has similar objectives, but their protocol requires a trusted third party to execute the SGD algorithm, whereas our method does not depend on any trusted party. In addition, although their method has the same scale of noise as ours, in their method each party samples local noise which is aggregated by the trusted third party. This is not secure in the presence of colluding parties as noted by Bindschaedler et al. (BRB17) and Shi et al. (SCR17). In our method, parties collaboratively generate noise within the MPC. Finally, their method requires noise from two sources: the Gaussian noise η and the Laplace noise ρ. Generation of ρ consumes ϵ privacy budget per iteration, as opposed to using ϵ budget for the entire learning process, and hence

violates the privacy constraints.

In this paper, we introduce differentially-private distributed machine learning protocols using both output perturbation and gradient perturbation where the noise is added within a secure multi-party computation. Our output perturbation method securely aggregates the local models and achieves ϵ-differential privacy by adding Laplace noise to the aggregated model parameters. In our gradient perturbation method, the parties collaboratively run an iterative, gradient-based learning algorithm where they securely aggregate the local gradients at each iteration. This provides (ϵ, δ)-differential privacy by adding Gaussian noise to the aggregated gradients. In both the methods the sampled noise is (roughly) inversely proportional to the size of the entire data set. While the first method is computationally efficient, requiring only single invocation of MPC, its accuracy decreases compared to centralized method when the number of parties is large relative to the total amount of data — this is inherent to any model aggregation based method. The iterative gradient perturbation method, on the other hand, does not suffer from accuracy degradation but requires one MPC protocol execution per iteration. Both methods achieve accuracy close to their non-private counterparts where no noise is added and no data privacy provided.

6.1.1 Contributions

This work makes the following contributions, which address challenges in distributed learning.

Output Perturbation and Gradient Perturbation Methods. We propose two approaches to privately train accurate machine learning models in the distributed setting. While the output perturbation method (Section 6.3.1) is computationally more efficient, the gradient perturbation method (Section 6.3.2) maintains high accuracy regardless of how the data is partitioned.

Reduced Noise Bounds. We give noise bounds for each method that are smaller than

the best previous approaches for output perturbation (PRR10) and gradient perturbation (SS15b), while ensuring differential privacy in the distributed setting (Theorem 6.3.1 for output perturbation and Theorem 6.3.4 for gradient perturbation). For gradient perturbation, we use *zero-concentrated differential privacy* to achieve the lowest known bound on the privacy budget. Moreover, we generate the noise within the MPC protocol. This allows us to add only a single copy of noise, compared to previous works that combine noise from each participant (DKM06b; BIK17; BRB17; HLK17; SCR17). We provide a theoretical analysis of our methods' error bounds which match the state-of-art error bounds in centralized settings.

Experimental Evaluation on Real Data Sets. We implement regularized logistic regression and regularized linear regression models for classification and regression tasks respectively. We report results from experiments performed on the *KDDCup99* and *Adult* data sets for classification and the *KDDCup98* data set for regression. We compare our methods with previous work on distributed learning, varying the number of parties and local data set sizes. Our methods produce models that are closest to the non-private models in terms of model accuracy and generalization error since we add less noise than previous distributed learning methods.

6.2 Background on Differential Privacy and Multi-Party Computation

This section introduces differential privacy (including the zero-concentrated differential privacy notion we use), and secure multi-party computation.

Notation: For any d-dimensional vector $\mathbf{x} = [x_1, ..., x_d]^\top$, we use $\|\mathbf{x}\| = (\sum_{i=1}^{d} |x_i|^2)^{1/2}$ to denote its ℓ_2-norm. Given two sequences $\{a_n\}$ and $\{b_n\}$, we write $a_n = O(b_n)$ if there exists a constant $0 < C < \infty$ such that $a_n \leq Cb_n$, and we use $\widetilde{O}(\cdot)$ to hide the logarithmic factors.

6.2.1 Differential Privacy

Differential privacy was introduced by Dwork (DMN06) and is defined as follows:

Definition 6.2.1 ((ϵ, δ)-Differential Privacy)**.** Given two adjacent data sets $D, D' \in \mathcal{D}^n$ differing by a single element, a randomized mechanism $\mathcal{M} : \mathcal{D}^n \to \mathbb{R}^d$ provides (ϵ, δ)-differential privacy if it produces response in the set S with probability $\mathbb{P}[\mathcal{M}(D) \in S] \leq e^{\epsilon}\mathbb{P}[\mathcal{M}(D') \in S] + \delta$.

The above definition reduces to ϵ-Differential Privacy (ϵ-DP) when $\delta = 0$. We can achieve ϵ-DP and (ϵ, δ)-DP by adding noise sampled from Laplace and Gaussian distributions respectively, where the noise is proportional to the sensitivity of $\mathcal{M}$, given as $\Delta\mathcal{M} = \|\mathcal{M}(D) - \mathcal{M}(D')\|$. Throughout this paper we assume the ℓ_2-sensitivity which considers the upper bound on the ℓ_2-norm of $\mathcal{M}(D) - \mathcal{M}(D')$.

Zero-Concentrated Differential Privacy While, the notion of differential privacy performs well for methods like output perturbation, it is not suitable for gradient perturbation methods which require repeated sampling of noise in the iterative training procedure. Zero-concentrated differential privacy (BS16b) (zCDP) has a tight composition bound and hence is a better option for gradient perturbation.

We first define the privacy loss random variable which is used in the definition of zCDP.

Definition 6.2.2. For two adjacent data sets $D, D' \in \mathcal{D}^n$ differing by one sample, a randomized mechanism $\mathcal{M} : \mathcal{D}^n \to \mathbb{R}^d$, and an outcome $o \in \mathbb{R}^d$, the privacy loss random variable Z is defined as

$$Z = \log \frac{\mathbb{P}[\mathcal{M}(D) = o]}{\mathbb{P}[\mathcal{M}(D') = o]}. \tag{6.2.1}$$

Definition 6.2.3. A randomized mechanism $\mathcal{M} : \mathcal{D}^n \to \mathbb{R}^d$ satisfies ρ-zCDP if for any two adjacent data sets $D, D' \in \mathcal{D}^n$ differing by one sample, it holds that for all $\alpha \in (1, \infty)$,

$$\mathbb{E}\big[e^{(\alpha-1)Z}\big] \leq e^{(\alpha-1)\alpha\rho}. \tag{6.2.2}$$

Note that (6.2.2) implies that $\mathbb{P}[Z > \lambda + \rho] \leq e^{-\lambda^2/(4\rho)}$ for all $\lambda > 0$, which suggests that the privacy loss Z is tightly concentrated around zero mean, and hence it is unlikely to distinguish D from D' given their outputs.

Bun and Steinke (BS16b) give the following lemmas to achieve zCDP with the Gaussian mechanism. Lemma 6.2.4 bounds the amount of Gaussian noise to guarantee ρ-zCDP. Lemma 6.2.5 gives the composition of multiple zCDP mechanisms. Finally, Lemma 6.2.6 specifies the mapping from ρ-zCDP to (ϵ, δ)-DP.

Lemma 6.2.4. Given a function $q : \mathcal{D}^n \to \mathbb{R}^d$, the Gaussian Mechanism $\mathcal{M} = q(D) + \mathbf{u}$, where $\mathbf{u} \sim N(0, \sigma^2 \mathbf{I}_d)$, satisfies $\Delta_2(q)^2/(2\sigma^2)$-zCDP.

Lemma 6.2.5. For two randomized mechanisms $\mathcal{M}_1 : \mathcal{D}^n \to \mathbb{R}^d$, $\mathcal{M}_2 : \mathcal{D}^n \times \mathbb{R}^d \to \mathbb{R}^d$. If $\mathcal{M}_1$ satisfies ρ_1-zCDP and $\mathcal{M}_2$ satisfies ρ_2-zCDP, then $\mathcal{M}_2(D, \mathcal{M}_1(D))$ satisfies $(\rho_1 + \rho_2)$-zCDP.

Lemma 6.2.6. If a randomized mechanism $\mathcal{M} : \mathcal{D}^n \to \mathbb{R}^d$ satisfies ρ-zCDP, then it satisfies $(\rho + 2\sqrt{\rho \log(1/\delta)}, \delta)$-differential privacy for any $\delta > 0$.

6.2.2 Secure Multi-Party Computation

Our threat model considers semi-honest participants who wish to compute a joint functionality without revealing their individual inputs to other participants. In this threat model, while the parties do not tamper with the joint functionality or provide garbage inputs, they are allowed to passively infer about inputs of other parties based on the protocol execution. We use generic multi-party computation protocols to securely aggregate local models and gradients. A multi-party computation (MPC) protocol enables two or more parties to jointly compute a function of their private inputs, without disclosing any information about those inputs other than their size and whatever can be inferred from the revealed output (Yao82). The notion of MPC goes back to a series of talks given by Andrew Yao in the 1980s. The protocol he introduced, now known as Yao's garbled circuits

protocol, can compute any function securely. Numerous other secure multi-party computation protocols have been devised since then (e.g., (GMW87; LP07; DPS12; NNO12)), and many tools have been developed for efficiently implementing MPC computations (e.g., (MNP04; DGK09; BSM10; HEK11; HFK12; RHH14; WMK16; ZE15b)). It is now practical to execute two-party protocols with millions of inputs (GSB17; GFA17), and global-scale, many-party protocols with malicious level security for small inputs (WRK17).

Secure aggregation of local classification models using MPC was shown to be practical by Tian et al. (TJG16). This work used a two-party computation, with a semi-honest threat model and non-colluding servers. A similar approach has been used to scale multi-party regressions (NWI13a; GSB17). We can use these methods to achieve secure aggregation. For scenarios where the risks of collusion are too high, many-party MPC protocols can be used that provide security to a single honest participant even if all other participants are malicious. In this work, we do not focus on improving or evaluating the MPC execution, since the methods we propose can be implemented using well known MPC techniques. Appendix 6.7 provides information on the MPC implementation we use and its cost.

6.3 Multi-Party Machine Learning

In this section we describe our output perturbation and gradient perturbation methods in detail along with theoretic analysis of differential privacy and generalization error bound.

We consider the following empirical risk minimization (ERM) objective:

$$J_D(\theta) = \frac{1}{n}\sum_{i=1}^{n} \ell(\theta, x_i, y_i) + \lambda N(\theta),$$

where $\ell(\theta)$ is a convex loss function that is G-Lipschitz and L-smooth over $\theta \in \mathbb{R}^d$. $N(\cdot)$ is regularization term. We consider $J(\cdot)$ to be λ-strongly convex. Each data instance $(x_i, y_i) \in D$ lies in a unit ball. For a party j, with data set D_j of size n_j, we denote its data instance as $(x_i^{(j)}, y_i^{(j)})$.

6.3.1 Model Aggregation with Output Perturbation

We extend the differential privacy bound of Chaudhuri et al. (CM09) to the multi-party setting, ensuring sufficient noise to preserve the privacy of each participant's data throughout the multi-party computation, including the final output.

Given m parties, each having a data set D_j of size n_j and the corresponding local model estimator $\widehat{\theta}^{(j)}$ obtained by minimizing the local objective function: $J_{D_j}(\theta) = \frac{1}{n_j}\sum_{i=1}^{n_j} \ell(\theta, x_i^{(j)}, y_i^{(j)}) + \lambda N(\theta)$. The perturbed aggregate model estimator is given as $\widehat{\theta}^{\text{priv}} = \frac{1}{m}\sum_{j=1}^{m} \widehat{\theta}^{(j)} + \eta$, where η is the Laplace noise added to the aggregate model estimator to preserve differential privacy. Secure model aggregation can be performed using the framework of Tian et al. (TJG16) as mentioned in Section 6.2.2.

The next theory provides a bound on the noise magnitude needed to achieve differential privacy:

Theorem 6.3.1. Given a perturbed aggregate model estimator $\widehat{\theta}^{\text{priv}} = \frac{1}{m}\sum_{j=1}^{m} \widehat{\theta}^{(j)} + \eta$ where $\widehat{\theta}^{(j)} = \arg\min_\theta \frac{1}{n_j}\sum_{i=1}^{n_j} \ell(\theta, x_i^{(j)}, y_i^{(j)}) + \lambda N(\theta)$ and the data lie in a unit ball and $\ell(\cdot)$ is G-Lipschitz , then $\widehat{\theta}^{\text{priv}}$ is ϵ-differentially private if

$$\eta = \text{Lap}\left(\frac{2G}{m n_{(1)} \lambda \epsilon}\right),$$

where $n_{(1)}$ is the size of the smallest data set among the m parties, λ is the regularization parameter and ϵ is the differential privacy budget.

Proof. Let there be m parties such that one record of party j changes in the neighbouring data sets, then

$$\begin{aligned}\frac{\Pr(\widehat{\theta}|D)}{\Pr(\widehat{\theta}|D')} &= \frac{\Pr\left(\frac{1}{m}\sum_{i\neq j}\widehat{\theta}^{(i)} + \frac{1}{m}\widehat{\theta}^{(j)} + \eta | D\right)}{\Pr\left(\frac{1}{m}\sum_{i\neq j}\widehat{\theta}^{(i)} + \frac{1}{m}\widehat{\theta}'^{(j)} + \eta | D'\right)} = \frac{\exp\left[\frac{m.n_{(1)}\epsilon\lambda}{2G}\frac{\|\widehat{\theta}^{(j)}\|}{m}\right]}{\exp\left[\frac{m.n_{(1)}\epsilon\lambda}{2G}\frac{\|\widehat{\theta}'^{(j)}\|}{m}\right]} \\ &\leq \exp\left[\frac{n_{(1)}\epsilon\lambda}{2G}\|\widehat{\theta}^{(j)} - \widehat{\theta}'^{(j)}\|\right] \leq \exp\left[\frac{n_{(1)}\epsilon\lambda}{2G}\frac{2G}{n_j\lambda}\right] \leq \exp(\epsilon),\end{aligned}$$

where the second inequality follows from Corollary 8 of Chaudhuri et al. (CMS11b). □

We now provide a bound on the excess empirical risk and true risk similar to Pathak et al. (PRR10). Our bounds are tighter than Parthak et al.'s as we require less differential privacy noise.

Theorem 6.3.2. Given a perturbed aggregate model estimator $\widehat{\theta}^{\text{priv}} = \frac{1}{m}\sum_{j=1}^{m}\widehat{\theta}^{(j)} + \eta$ where $\widehat{\theta}^{(j)} = \arg\min_\theta \frac{1}{n_j}\sum_{i=1}^{n_j}\ell(\theta, x_i^{(j)}, y_i^{(j)}) + \lambda N(\theta)$ and an optimal model estimator θ^* trained on the centralized data such that the data lie in a unit ball and $\ell(\cdot)$ is G-Lipschitz and L-smooth, then the bound on excess empirical risk is given as:

$$J(\widehat{\theta}^{\text{priv}}) \leq J(\theta^*) + C_1\frac{G^2(\lambda+L)}{n_{(1)}^2\lambda^2}\left(m^2 + \frac{d^2\log^2(d/\delta)}{m^2\epsilon^2} + \frac{d\log(d/\delta)}{\epsilon}\right),$$

where C_1 is an absolute constant.

The proof of Theorem 6.3.2 follows from Pathak et al. (PRR10). The main difference is that we use the sensitivity bound as $2G/(mn_{(1)}\lambda)$ instead of $2G/(n_{(1)}\lambda)$ and thereby achieve a tighter bound. The full proof is given in Appendix 6.5.1.

Theorem 6.3.3. Given a perturbed aggregate model estimator $\widehat{\theta}^{\text{priv}} = \frac{1}{m}\sum_{j=1}^{m}\widehat{\theta}^{(j)} + \eta$ where $\widehat{\theta}^{(j)} = \arg\min_\theta \frac{1}{n_j}\sum_{i=1}^{n_j}\ell(\theta, x_i^{(j)}, y_i^{(j)}) + \lambda N(\theta)$ and an optimal model estimator θ^* trained on the centralized data such that the data lie in a unit ball and $\ell(\cdot)$ is G-Lipschitz and L-smooth, then the following bound on true excess risk holds with probability at least $1-\gamma$:

$$\mathbb{E}[\widetilde{J}(\widehat{\theta}^{\text{priv}})] - \min_\theta \widetilde{J}(\theta) \leq C_1\frac{G^2(\lambda+L)}{n_{(1)}^2\lambda^2}\left(m^2 + \frac{d^2\log^2(d/\delta)}{m^2\epsilon^2} + \frac{d\log(d/\delta)}{\epsilon}\right) + C_2\frac{G^2\log(1/\gamma)}{\lambda n},$$

where n is the size of the centralized data set. $\widetilde{J}(\theta) = \mathbb{E}_{x,y}[\ell(\theta, x, y)] + \lambda N(\theta)$, C_1, C_2 are absolute constants, and the expectation is taking with respect to the noise η.

See Appendix 6.5.2 for the proof of Theorem 6.3.3. The true excess risk bound in Theorem 6.3.3 implies that the private output of our algorithm converges to the population optimum at the order of $1/n$.

6.3.2 Iterative Learning with Gradient Perturbation

We consider this centralized ERM objective for m parties, each with a data set D_j of size n_j:

$$J_D(\theta) = \min_{\theta} \frac{1}{m} \sum_{j=1}^{m} \frac{1}{n_j} \sum_{i=1}^{n_j} \ell(\theta, x_i^{(j)}, y_i^{(j)}) + \lambda N(\theta).$$

The parties can collaboratively learn a differentially private model via iterative learning by adding noise to the aggregated gradients within the MPC in each iteration with the following noise bound.

Theorem 6.3.4. Given a centralized model estimator θ_T obtained by minimizing $J_D(\theta)$ after T iterations of gradient descent algorithm executed jointly by m parties each having dataset $D^{(j)}$ of size n_j where each data instance $(x_i^{(j)}, y_i^{(j)}) \in D^{(j)}$ lie in a unit ball and $\ell(\theta)$ is G-Lipschitz and L-smooth over $\theta \in \mathcal{C}$. If the learning rate is $1/L$ and the gradients are perturbed with noise $z \in \mathcal{N}(0, \sigma^2 I_d)$, then θ_T is (ϵ, δ)-differentially private if

$$\sigma^2 = \frac{8G^2 T \log(1/\delta)}{m^2 n_{(1)}^2 \epsilon^2}, \tag{6.3.1}$$

where $n_{(1)}$ is the size of the smallest data set among the m parties.

Proof. Given a gradient at step t,

$$M_t = \nabla J(\theta, D) + \mathcal{N}(0, \sigma^2 I_p) = \frac{1}{m} \sum_{j=1}^{m} \frac{1}{n_j} \sum_{i=1}^{n_j} \nabla \ell(\theta, x_i^{(j)}, y_i^{(j)}) + \mathcal{N}(0, \sigma^2 I_p).$$

We assume that only one data instance of one party changes in neighbouring datasets D and D'. Hence the sensitivity bound, $\|\nabla J(\theta, D) - \nabla J(\theta, D')\| \leq \frac{2G}{m n_{(1)}}$.

Thus, using Lemma 6.2.4, M_t is ρ-zCDP with $\rho = \frac{2G^2}{m^2 n_{(1)}^2 \sigma^2}$. By composition from Lemma 6.2.5, we observe that θ_T is $T\rho$-zCDP. Applying Lemma 6.2.6, we obtain $T\rho + 2\sqrt{T\rho \log(1/\delta)} = \epsilon$. Solving the roots of this equation, we obtain

$$\rho \approx \frac{\epsilon^2}{4T \log(1/\delta)} \implies \sigma^2 = \frac{8G^2 T \log(1/\delta)}{m^2 n_{(1)}^2 \epsilon^2}.$$

Thus, θ_T is (ϵ, δ)-differentially private for the above value of σ^2. □

Additionally, we observe that differential privacy is also guaranteed for each intermediate model estimator:

Corollary 6.3.5. Intermediate model estimator θ_t at each iteration $t \in [1, T]$ is $(\sqrt{t/T}\epsilon, \delta)$-differentially private.

Hence, an adversary cannot obtain additional information from the intermediate computations. See Appendix 6.5.3 for the proof of Corollary 6.3.5.

Next, we provide theoretical bounds on the excess empirical risk and the true excess risk of our proposed method.

Theorem 6.3.6. Given a centralized model estimator θ_T obtained by minimizing $J_D(\theta)$ after T iterations of gradient descent algorithm executed jointly by m parties each having dataset $D^{(j)}$ of size n_j where each data instance $(x_i^{(j)}, y_i^{(j)}) \in D^{(j)}$ lie in a unit ball and $\ell(\theta)$ is G-Lipschitz and L-smooth over $\theta \in \mathcal{C}$. If the learning rate is $1/L$ and the gradients are perturbed with noise $z \in \mathcal{N}(0, \sigma^2 I_d)$ with σ^2 defined in (6.3.1), and if we choose the iteration number as

$$T = \widetilde{O}\left(\log \left(\frac{m^2 n_{(1)}^2 \epsilon^2}{dG^2 \log(1/\delta)} \right) \right),$$

then we have a bound on excess empirical risk:

$$\mathbb{E}[J(\theta_T)] - J(\theta^*) \leq C_1 \frac{G^2 L d \log^2(m n_{(1)}) \log(1/\delta)}{m^2 n_{(1)}^2 \lambda^2 \epsilon^2},$$

where the expectation is taking with respect to the noise η, $n_{(1)}$ is the size of the smallest data set among the m parties, C_1 is an absolute constant.

Appendix 6.5.4 provides the proof.

Based on the excess empirical risk, we next derive the true excess risk.

Theorem 6.3.7. Given a centralized model estimator θ_T obtained by minimizing $J_D(\theta)$ after T iterations of a gradient descent algorithm executed jointly by m parties each having dataset $D^{(j)}$ of size n_j where each data instance $(x_i^{(j)}, y_i^{(j)}) \in D^{(j)}$ lie in a unit ball and $\ell(\theta)$ is G-Lipschitz and L-smooth over $\theta \in \mathcal{C}$. If we choose the learning rate, noise level, and iteration number as suggested in Theorem 6.3.6, with probability at least $1-\gamma$, we have the following bound on true excess risk:

$$\mathbb{E}[\widetilde{J}(\theta_T)] - \min_\theta \widetilde{J}(\theta) \leq C_1 \frac{G^2 L d \log^2(mn_{(1)}) \log(1/\delta)}{m^2 n_{(1)}^2 \lambda^2 \epsilon^2} + C_2 \frac{G^2 \log(1/\gamma)}{\lambda n},$$

where n is the size of the centralized data set, $n_{(1)}$ is the size of the smallest data set among the m parties and C_1, C_2 are absolute constants.

Theorem 6.3.7 (proof in Appendix 6.5.5) suggests that the output of our iterative gradient perturbation method converges to the population optimum at the order of $1/n$. Note that our true excess risk bound is comparable to that of Wang et al. (WYX17a) in centralized setting.

6.4 Experiments

We report on experiments for both classification and regression tasks. For classification, we use a regularized logistic regression model over the *KDDCup99* (HB99) data set (additional experiments on the *Adult* (AN07) data set yield similar results, described in Appendix 6.6.3). The *KDDCup99* data set contains around 5,000,000 network instances. The task is to predict whether a network connection is a denial-of-service attack or not. We randomly sample 70,000 records and divide it into training set of 50,000 records and test set of 20,000 records. We pre-processed the data according to the procedure of Chaudhuri et al. (CMS11b), resulting in records with 122 features. For regression, we train a ridge regression model over the *KDDCup98* (PH98) data set, consisting of demographic and other related information of approximately 200,000 American veterans. The task is to predict the donation amount

Table 6.1: Comparison of noise magnitudes for various multi-party differential privacy methods.

Pathak	**Local Out P**	**Local Obj P**	**Local Grad P**	**MPC Out P**	**MPC Grad P**
Analytical Bound ($\mathcal{L}$ - Laplace, $\mathcal{N}$ - Gaussian)					
$\mathcal{L}(\frac{2G}{n_{(1)}\lambda\epsilon})$	$\mathcal{L}(\frac{2G}{\sqrt{m}n_{(1)}\lambda\epsilon})$	$\mathcal{L}(\frac{2G}{n_{(1)}\epsilon})$	$\mathcal{N}(\frac{\sqrt{2T}G}{\sqrt{m}n_{(1)}\epsilon})$	$\mathcal{L}(\frac{2G}{mn_{(1)}\lambda\epsilon})$	$\mathcal{N}(\frac{\sqrt{2T}G}{mn_{(1)}\epsilon})$
Noise Generation Input ($m = 100$, $n_{(1)} = 500$, $\lambda = 0.01$, $\epsilon = 0.5$, $G = 1$ and $T = 100$)					
800×10^{-3}	80.0×10^{-3}	8.00×10^{-3}	5.66×10^{-3}	8.00×10^{-3}	0.57×10^{-3}
Generated Noise (standard deviation over 1000 samples)					
1150×10^{-3}	112×10^{-3}	11.6×10^{-3}	5.63×10^{-3}	12.2×10^{-3}	0.572×10^{-3}

of an individual in dollars. We randomly sample 70,000 records and divide it into a training set of 50,000 records and test set of 20,000 records. We perform the same pre-processing as in the case of previous data sets and additionally perform feature selection using PCA to retain around 100 features. After pre-processing, each record consists of 95 features.

For all the experiments, we set Lipschitz constant $G = 1$, learning rate $\eta = 1$, regularization coefficient $\lambda = 0.001$, privacy budget $\epsilon = 0.5$, failure probability $\delta = 0.001$ and total number of iterations $T = 1,500$ for gradient descent. We compare our methods with the baselines in terms of optimality gap and relative accuracy loss. Optimality gap is the measure of empirical risk bound $J(\theta) - J(\theta^*)$ over the training data, where θ^* is the optimal non-private model in the centralized setting. Relative accuracy loss is the difference in the accuracy (mean square error in case of regression) of θ and θ^* over the test data. We measure the optimality gap and relative accuracy loss of all the models up to 1,500 iterations of gradient descent training and report the results for different partitioning of training data sets. We vary the number of parties m from 100 (where each party has 500 data instances) to 1,000 parties (with each party having 50 data instances) and up to 50,000 parties (each having only one data instance).

Baselines for comparison. For the model aggregation method, we compare to the method of Pathak et al. (PRR10) (denoted as **Pathak**), and the other differential privacy baselines are obtained by applying the output perturbation (denoted as **Local Out P**) and objective perturbation (denoted as **Local Obj P**) method of Chaudhuri et al. (CMS11b) on each local model estimator $\widehat{\theta}^{(j)}$ to obtain a differentially private local model estimator and then the model aggregation is performed to obtain the differentially private aggregate model $\widehat{\theta}^{\text{priv}}$. For the iterative learning method, we consider the baseline of aggregation of locally perturbed gradients similar to that of Shokri and Shmatikov (SS15b) (denoted as **Local Grad P**), except that we improve the noise bound by using zCDP. We also include the method of Rajkumar and Agarwal (RA12) (denoted as **Rajkumar and Agarwal**) in our comparison, though note that their method does not provide the same level of privacy as our method. Our output perturbation based model aggregation method and gradient perturbation based iterative learning method are denoted as **MPC Out P** and **MPC Grad P** respectively. All the above methods consume a total privacy budget of $\epsilon = 0.5$, except **Rajkumar and Agarwal** which consumes $\epsilon = 0.5$ budget each iteration. Table 6.1 summarizes the amount of noise each method needs to preserve differential privacy. As the table shows, our methods add the least amount of noise. Though **Local Obj P** adds noise in the same range as our methods, it uses the noise in a fundamentally different way. While the other methods add the sampled noise (either via output perturbation or via gradient perturbation) to the optimal non-private model that minimizes the required objective function $J(\theta)$, **Local Obj P** adds the sampled noise directly to the objective function $J(\theta)$ and hence optimizes an altogether different objective function $J'(\theta) = J(\theta) + \text{Lap}(\frac{2G}{n_1\epsilon})$, which explains why its optimality gap increases with decreasing value of local data set size $n_{(1)}$.

Results. Figures 6.1 and 6.2 show the results for $m = 1,000$; Appendix 6.6 includes plots for other numbers of parties. For both the classification and regression tasks, our proposed methods perform better than the baselines both in terms of optimality gap and relative accuracy loss. For the classification task (Figure 6.1), **MPC Grad P** achieves optimality

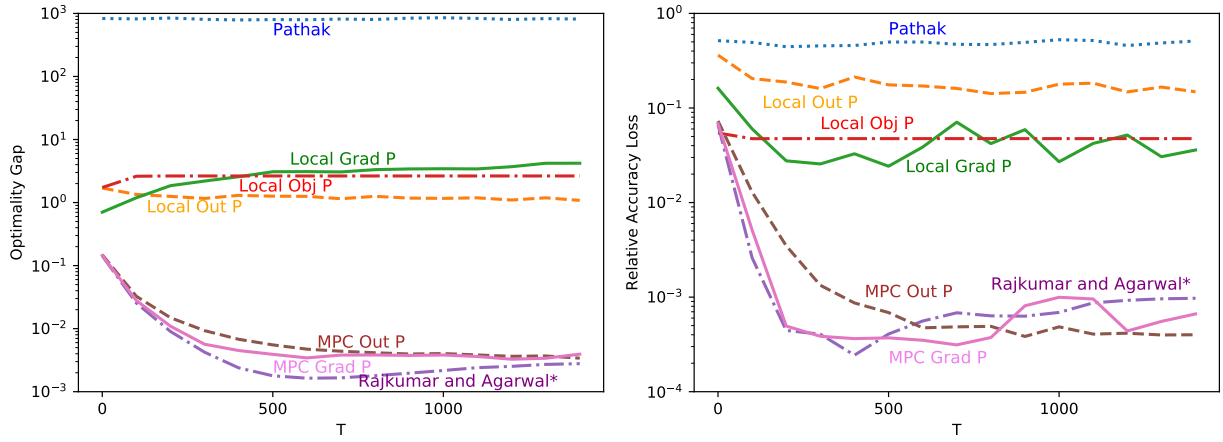


Figure 6.1: Optimality Gap and Relative Accuracy Loss Comparison on *KDDCup99* ($m = 1,000$). (All models have privacy budget $\epsilon = 0.5$, except **Rajkumar and Agarwal** which consumers $\epsilon = 0.5$ privacy budget each iteration.)

gap in the order of 10^{-3} in 500 iterations and relative accuracy loss in the order of 10^{-4} within 200 iterations, and **MPC Out P** also achieves values in the same range. **Rajkumar and Agarwal** adds noise of the same order as our methods and hence achieves performance close to ours, but as noted earlier, their method consumes ϵ budget per iteration. Our methods perform order of magnitudes better than the other baselines.

For the regression task (Figure 6.2), **MPC Grad P** gradually converges to an optimality gap in the order of 10^{-3} and relative accuracy loss in the order of 10^{-2}. **MPC Out P** incurs loss due to data partitioning (which is unavoidable even for non-private aggregation methods) but still outperforms the baselines of model aggregation by orders of magnitude.

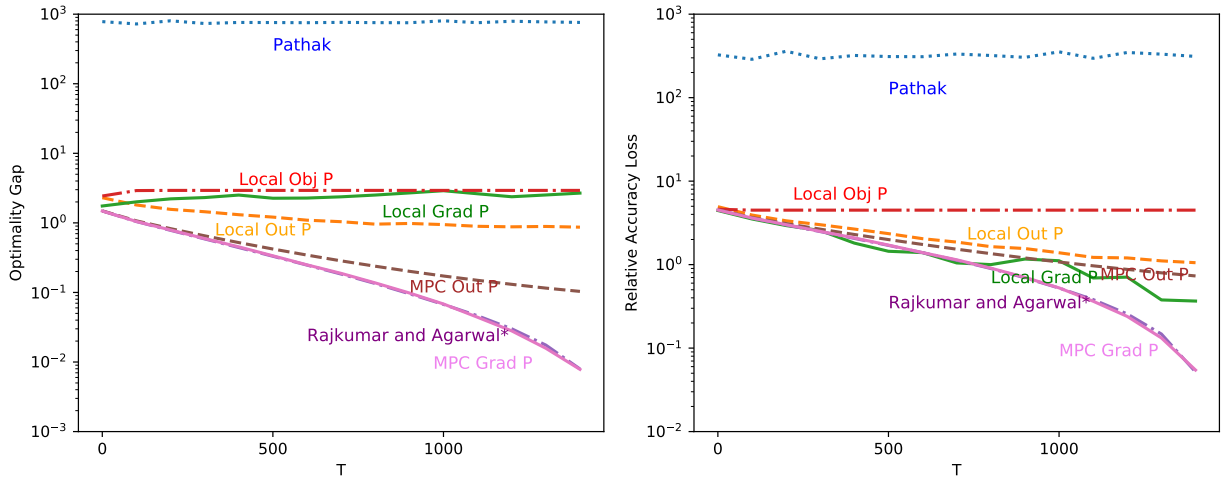


Figure 6.2: Optimality Gap and Relative Accuracy Loss Comparison on *KDDCup98* ($m = 1,000$). (As in Figure 6.1, all models have privacy budget $\epsilon = 0.5$, except **Rajkumar and Agarwal**.)

6.5 Proofs of the Main Theorems

6.5.1 Proof of Theorem 6.3.2

Before we prove Theorem 6.3.2, we provide a bound on the Laplace random vector given in below Lemma (as proved in Chaudhuri and Monteleoni (CM09)).

Lemma 6.5.1. Given a d-dimensional random variable $\eta \sim \text{Lap}(\beta)$ with $P(\eta) = \frac{1}{2\beta} e^{-\frac{\|\eta\|_1}{\beta}}$, with probability at least $1 - \delta$, the ℓ_2-norm of the random variable is bounded as $\|\eta\| \leq d\beta \log\left(\frac{d}{\delta}\right)$.

For any differentiable and convex objective function, Chaudhuri and Monteleoni (CM09) propose the following lemma to bound the sensitivity of model estimator:

Lemma 6.5.2 (Lemma 1 of Chaudhuri and Monteleoni (CM09))**.** Let $\theta_1 = \arg\min_\theta G(\theta)$ and $\theta_2 = \arg\min_\theta G(\theta) + g(\theta)$ such that $G(\theta)$ and $g(\theta)$ are both differentiable and convex. Then $\|\theta_1 - \theta_2\| \leq \frac{g_1}{G_2}$ where $G_2 = \min_v \min_\theta v^\top \nabla^2 G(\theta) v$ for any unit vector $v \in \mathbb{R}^d$ and $g_1 = \max_\theta \|\nabla g(\theta)\|$.

We also give the following theorem to bound the excess risk between distributed non-private model estimator and the optimal model estimator in the centralized setting. This theorem is used to prove the Theorem 6.3.2.

Theorem 6.5.3. Given an aggregate model estimator, $\widehat{\theta} = \frac{1}{m}\sum_{j=1}^{m}\widehat{\theta}^{(j)}$ where we have $\widehat{\theta}^{(j)} = \arg\min_\theta \frac{1}{n_j}\sum_{i=1}^{n_j}\ell(\theta, x_i^{(j)}, y_i^{(j)}) + \lambda N(\theta)$ and an optimal model estimator θ^* trained on the centralized data such that the data lie in a unit ball and $\ell(\cdot)$ is G-Lipschitz , we have

$$\|\widehat{\theta} - \theta^*\| \leq \frac{G(m-1)}{n_{(1)}\lambda}.$$

Proof. For a party P_j, the local model estimator is given as:

$$\widehat{\theta}^{(j)} = \arg\min_\theta \frac{1}{n_j}\sum_{i=1}^{n_j}\ell(\theta, x_i^{(j)}, y_i^{(j)}) + \lambda N(\theta) = \arg\min_\theta G(\theta).$$

The centralized model estimator is hence given as:

$$\theta^* = \arg\min_\theta \frac{1}{n_j}\sum_{i=1}^{n_j}\ell(\theta, x_i^{(j)}, y_i^{(j)}) + \sum_{l\neq j}\frac{1}{n_l}\sum_{i=1}^{n_l}\ell(\theta, x_i^{(l)}, y_i^{(l)}) + \lambda N(\theta) = \arg\min_\theta G(\theta) + g(\theta).$$

Thus we have the following values of g_1 and G_2:

$$g_1 = \max_\theta \|\nabla g(\theta)\| = \max_\theta \sum_{l\neq j}\frac{1}{n_l}\|\nabla\ell(\theta, x_i^{(l)}, y_i^{(l)})\| \leq G\sum_{l\neq j}\frac{1}{n_l}$$

$$G_2 = \min_v \min_\theta \|v^\top\nabla^2 G(\theta)v\| = \min_v \min_\theta \|v^\top\big(\frac{1}{n_j}\|\nabla^2\ell(\theta, x_i^{(j)}, y_i^{(j)})\| + \lambda.1\big)v\| \geq \lambda$$

Using Lemma 6.5.2, we have $\|\widehat{\theta}^{(j)} - \theta^*\| = \frac{G}{\lambda}\sum_{l\neq j}\frac{1}{n_l}$. Applying triangle inequality, we get:

$$\|\widehat{\theta} - \theta^*\| \leq \frac{1}{m}\sum_j\|\widehat{\theta}^{(j)} - \theta^*\| = \frac{G}{m\lambda}\sum_j\sum_{l\neq j}\frac{1}{n_l} = \frac{G(m-1)}{m\lambda}\sum_j\frac{1}{n_j} \leq \frac{G(m-1)}{n_{(1)}\lambda}.$$

□

Now we are ready to prove Theorem 6.3.2 using Lemma 6.5.1 and Theorem 6.5.3.

Proof of Theorem 6.3.2. Using Taylor Expansion, we have

$$J(\widehat{\theta}^{\text{priv}}) = J(\theta^*) + (\widehat{\theta}^{\text{priv}} - \theta^*)\nabla J(\theta^*) + \frac{1}{2}(\widehat{\theta}^{\text{priv}} - \theta^*)\nabla^2 J(\theta)(\widehat{\theta}^{\text{priv}} - \theta^*),$$

where $\theta = \alpha\widehat{\theta}^{\text{priv}} + (1-\alpha)\theta^*$ for some $\alpha \in [0,1]$. By definition, $\nabla J(\theta^*) = 0$, thus we get

$$J(\widehat{\theta}^{\text{priv}}) - J(\theta^*) \leq \frac{1}{2}\|\widehat{\theta}^{\text{priv}} - \theta^*\|^2 . \|\nabla^2 J(\theta)\|.$$

Since $\ell(\cdot)$ is L-smooth, we have $\|\nabla^2 J(\theta)\| \leq \lambda + L$, therefore

$$\begin{aligned}
J(\widehat{\theta}^{\text{priv}}) \leq & J(\theta^*) + \frac{\lambda + L}{2}\|\widehat{\theta} - \theta^* + \eta\|^2 \\
\leq & J(\theta^*) + \frac{\lambda + L}{2}\big[\|\widehat{\theta} - \theta^*\|^2 + \|\eta\|^2 + 2(\widehat{\theta} - \theta^*)^\top \eta\big] \\
\leq & J(\theta^*) + \frac{\lambda + L}{2}\big[\|\widehat{\theta} - \theta^*\|^2 + \|\eta\|^2 + 2\|\widehat{\theta} - \theta^*\| \cdot \|\eta\|\big].
\end{aligned}$$

By Theorem 6.5.3 and Lemma 6.5.1, we obtain

$$\begin{aligned}
J(\widehat{\theta}^{\text{priv}}) \leq & J(\theta^*) + \frac{G^2(m-1)^2(\lambda + L)}{2n_{(1)}^2\lambda^2} + \frac{2G^2 d^2(\lambda + L)}{m^2 n_{(1)}^2 \lambda^2 \epsilon^2}\log^2\left(\frac{d}{\delta}\right) \\
& + \frac{2G^2 d(m-1)(\lambda + L)}{m n_{(1)}^2 \epsilon \lambda^2}\log\left(\frac{d}{\delta}\right) \\
\leq & J(\theta^*) + C_1 \frac{G^2(\lambda + L)}{n_{(1)}^2 \lambda^2}\left(m^2 + \frac{d^2 \log^2(d/\delta)}{m^2 \epsilon^2} + \frac{d\log(d/\delta)}{\epsilon}\right),
\end{aligned}$$

where C_1 is an absolute constant. □

6.5.2 Proof of Theorem 6.3.3

Proof. The proof follows from Sridharan et al. (SSS09) where the authors give the following relation between true risk and empirical risk bounds:

$$\widetilde{J}(\widehat{\theta}^{\text{priv}}) - \min_{\theta}\widetilde{J}(\theta) \leq 2\big[J(\widehat{\theta}^{\text{priv}}) - J(\theta^*)\big] + \frac{16G^2}{\lambda n}\left[32 + \log\left(\frac{1}{\delta}\right)\right].$$

Substituting the empirical risk bound from Theorem 6.3.2 will complete the proof. □

6.5.3 Proof of Corollary 6.3.5

Proof. By composition property of Lemma 6.2.5, each θ_t is $(t\rho)$-zCDP.

From Lemma 6.2.6, the privacy budget ϵ_t for iteration t is given as $\epsilon_t = t\rho + 2\sqrt{t\rho\log(1/\delta)}$

Total privacy budget is $\epsilon = T\rho + 2\sqrt{T\rho\log(1/\delta)}$

$$\implies \frac{t\epsilon}{T} = t\rho + 2\sqrt{t\rho\log(1/\delta)}\left(\sqrt{\frac{t}{T}}\right)$$
$$= t\rho\left(1 - \sqrt{\frac{t}{T}}\right) + \sqrt{\frac{t}{T}}\epsilon_t$$
$$\implies \epsilon_t = \sqrt{\frac{t}{T}}\epsilon + \sqrt{Tt}\left(\sqrt{\frac{t}{T}} - 1\right)\rho$$

In the proof of Theorem 6.3.4, we showed that: $\rho \approx \frac{\epsilon^2}{4T\log(1/\delta)}$

Substituting the value of ρ, we get the relation between ϵ_t and ϵ:

$$\epsilon_t = \sqrt{\frac{t}{T}}\epsilon + \sqrt{\frac{t}{T}}\left(\sqrt{\frac{t}{T}} - 1\right)\frac{\epsilon^2}{4\log(1/\delta)} \leq \sqrt{\frac{t}{T}}\epsilon$$

Hence, each intermediate model estimator θ_t is $(\sqrt{t/T}\epsilon, \delta)$-differentially private. □

6.5.4 Proof of Theorem 6.3.6

Proof. From L-smoothness assumption:

$$\mathbb{E}[J(\theta_{t+1}) - J(\theta_t)] \leq \mathbb{E}[\langle \nabla J(\theta_t), \theta_{t+1} - \theta_t\rangle + \frac{1}{2L}\|\nabla J(\theta_t) + z_t\|^2]$$
$$= \mathbb{E}[-\frac{1}{L}\langle \nabla J(\theta_t), \nabla J(\theta_t) + z_t\rangle + \frac{1}{2L}\|\nabla J(\theta_t) + z_t\|^2]$$
$$= -\frac{1}{2L}\|\nabla J(\theta_t)\|^2 + \frac{1}{2L}\mathbb{E}_{z_t}\|z_t\|^2 \leq -\frac{\lambda}{L}(J(\theta_t) - J^*) + \frac{d\sigma^2}{2L}$$

The last inequality comes from the strong convexity assumption. The above equation is conditioned on the randomness of θ_t, and can be written as:

$$\mathbb{E}[J(\theta_{t+1})] - J(\theta^*) \leq (1 - \frac{\lambda}{L})(J(\theta_t) - J(\theta^*)) + \frac{d\sigma^2}{2L}$$

Summing over $t = 0, \ldots, T$ iterations, and taking expectation:

$$\mathbb{E}[J(\theta_T)] - J(\theta^*) \leq (1 - \frac{\lambda}{L})^T(J(\theta_0) - J(\theta^*)) + \frac{d\sigma^2}{2\lambda}$$

When $T = O\Big(\log\big(\frac{m^2 n_{(1)}^2 \epsilon^2}{dG^2\log(1/\delta)}\big)\Big)$,

$$\mathbb{E}[J(\theta_T)] - J(\theta^*) \leq C_1\frac{G^2 d\log^2(mn_{(1)})\log(1/\delta)}{m^2 n_{(1)}^2\epsilon^2}$$

where C_1 is an absolute constant and the big-O notation hides other $\log, L, \lambda$ terms. □

6.5.5 Proof of Theorem 6.3.7

Proof. We want to bound $\mathbb{E}[\widetilde{J}(\theta_T)] - \min_\theta \widetilde{J}(\theta)$, where $\widetilde{J}(\theta) = \mathbb{E}_D[J_D(\theta)]$. We denote $\widetilde{J}(\widetilde{\theta}) = \min_\theta \widetilde{J}(\theta)$. According to Theorem 1 in (SSS09), we have the following holds with probability at least $1 - \gamma$

$$\widetilde{J}(\theta^T) - \widetilde{J}(\widetilde{\theta}) \leq 2(J(\theta^T) - J(\theta^*)) + C_2 \frac{G^2 \log(1/\gamma)}{\lambda m n_{(1)}},$$

where C_2 is an absolute constant. Therefore, we have the following holds with probability at least $1 - \gamma$

$$\begin{aligned}\mathbb{E}[\widetilde{J}(\theta_T)] - \min_\theta \widetilde{J}(\theta) &\leq 2(\mathbb{E}[J(\theta_T)] - J(\theta^*)) + C_2 \frac{G^2 \log(1/\gamma)}{\lambda m n_{(1)}} \\ &\leq C_1 \frac{G^2 d \log^2(m n_{(1)}) \log(1/\delta)}{m^2 n_{(1)}^2 \epsilon^2} + C_2 \frac{G^2 \log(1/\gamma)}{\lambda m n_{(1)}},\end{aligned}$$

where C_1 is an absolute constant. □

6.6 More Experimental Results

6.6.1 Experiments on *KddCup99* dataset

With the increasing number of parties m (and accordingly decreasing the local data set size $n_{(1)}$), performance of all the methods decrease except that of **MPC Grad P** (see Figures 6.3 and 6.4). While the performance of baselines deteriorate mainly due to the large amount of noise they add, the performance of **MPC Out P**, on the other hand, decreases with decreasing local data set size due to the loss in information from data partitioning (which is the case with any model aggregation based method, including the non-private ones). For $m = 50,000$, the large amount of noise destroys the utility of **Pathak** (Figure 6.4).

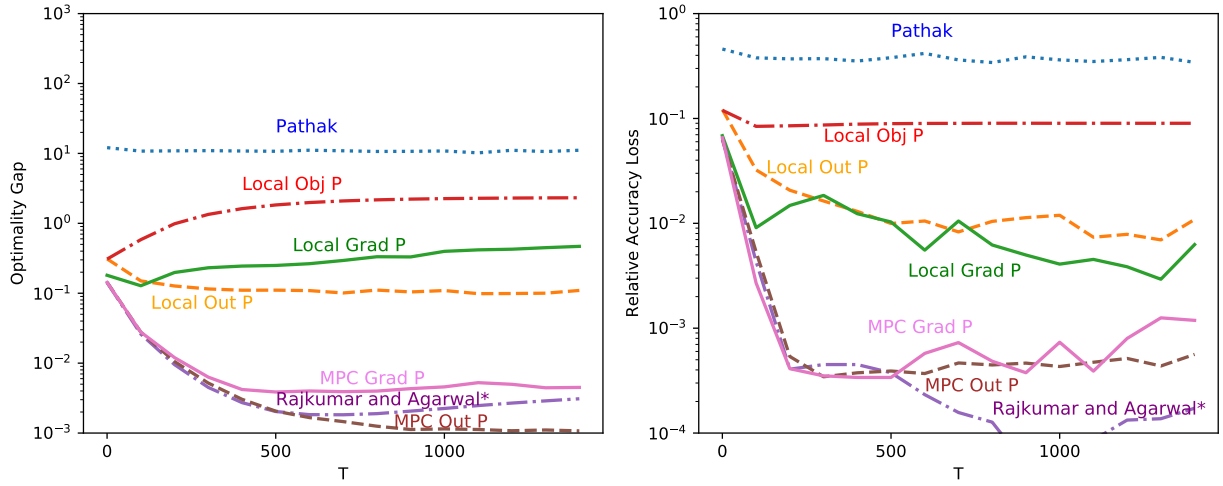


Figure 6.3: Optimality Gap and Relative Accuracy Loss Comparison on *KddCup99* ($m = 100$)

*All models except **Rajkumar and Agarwal** have privacy budget $\epsilon = 0.5$.

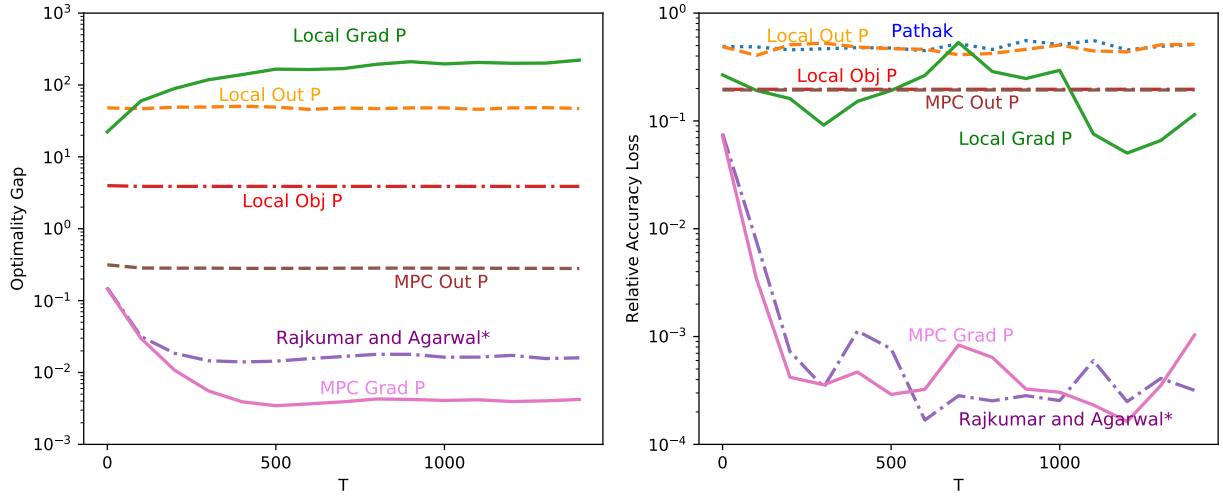


Figure 6.4: Optimality Gap and Relative Accuracy Loss Comparison on *KddCup99* ($m = 50,000$)

*All models except **Rajkumar and Agarwal** have privacy budget $\epsilon = 0.5$.

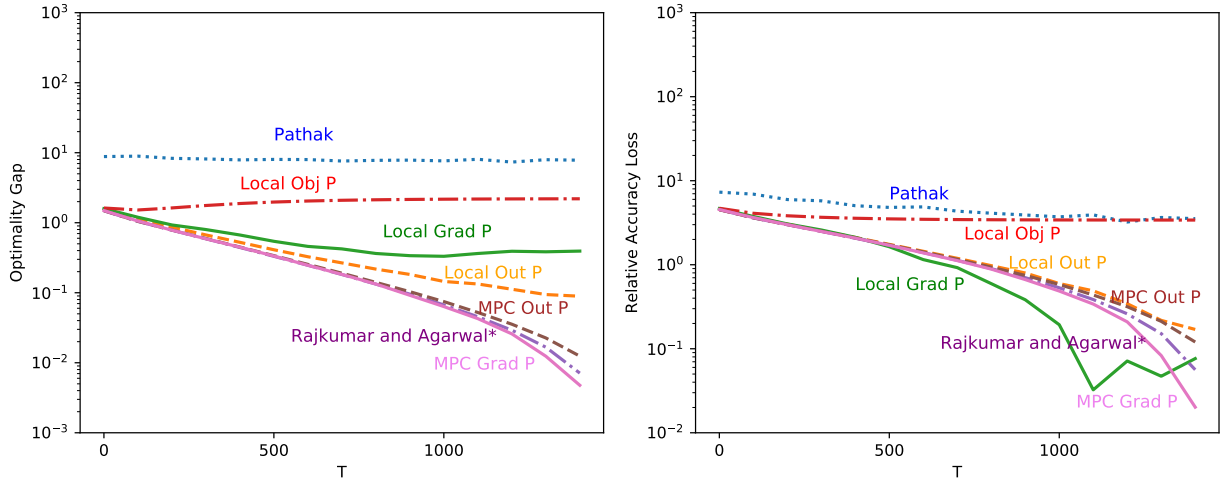


Figure 6.5: Optimality Gap and Relative Accuracy Loss Comparison on *KddCup98* ($m = 100$)

*All models except **Rajkumar and Agarwal** have privacy budget $\epsilon = 0.5$.

6.6.2 Experiments on *KddCup98* dataset

Overall, with the increasing number of parties m (and accordingly decreasing the local data set size $n_{(1)}$), performance of all the methods decrease except that of **MPC Grad P** (see Figures 6.5 and 6.6). Though the performance of **MPC Out P** decreases with decreasing the local data set size, it still outperforms the baselines of model aggregation. We note that for $m = 50,000$, **Local Obj P** performs worse than the **Local Out P** which is due to the deviation in the objective function of **Local Obj P** as mentioned earlier. The utility of **Pathak** is severely affected due to the large amount of noise added, which is why the plot is out of the range for $m = 50,000$ (Figure 6.6).

6.6.3 Experiments on *Adult* dataset

The *Adult* (AN07) data set consists of demographic information of approximately 47,000 individuals, and the task is to predict whether the annual income of an individual is above

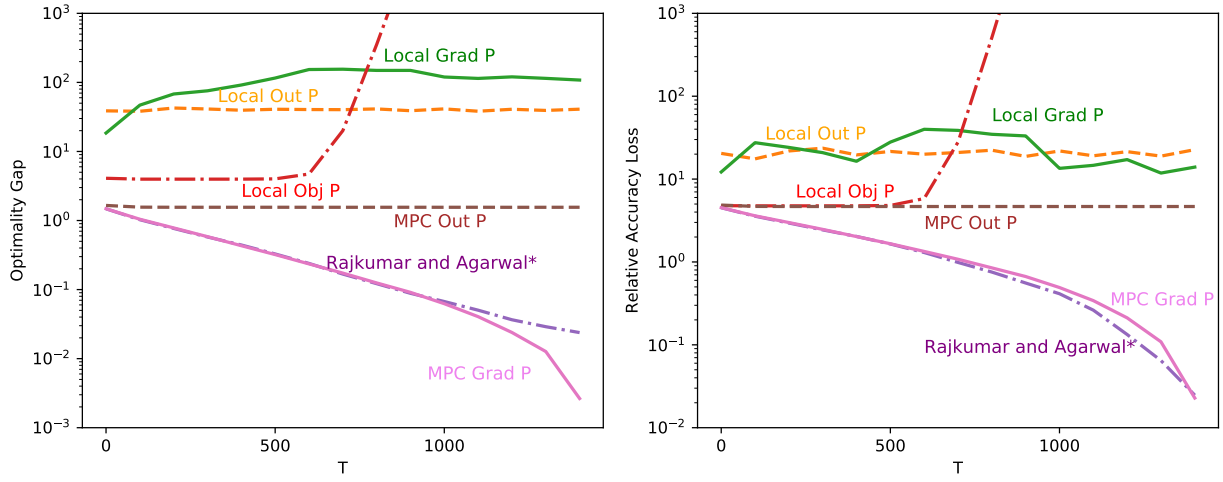


Figure 6.6: Optimality Gap and Relative Accuracy Loss Comparison on *KddCup98* ($m = 50,000$)

*All models except **Rajkumar and Agarwal** have privacy budget $\epsilon = 0.5$.

or below $50,000. After removing records with missing values, we end up with 45,222 records of which 30,000 records formed the training set and the remaining records formed the test set. After pre-processing, each record consisted of 104 features.

We vary the number of parties m from 100 (where each party has 300 data instances) to 1,000 parties (with each party having 30 data instances) and all the way to 30,000 parties (each having only one data instance).

Our proposed methods **MPC Out P** and **MPC Grad P** outperform the baselines both in terms of optimality gap and relative accuracy loss (Figures 6.7, 6.8 and 6.9). **MPC Grad P** achieves optimality gap in the order of 10^{-2} and relative accuracy loss in the order of 10^{-4} for all settings, while the performance of **MPC Out P** deteriorates with decreasing local data set size (due to the information loss from data partitioning). Nevertheless, **MPC Out P** still outperforms all the baselines by orders of magnitude.

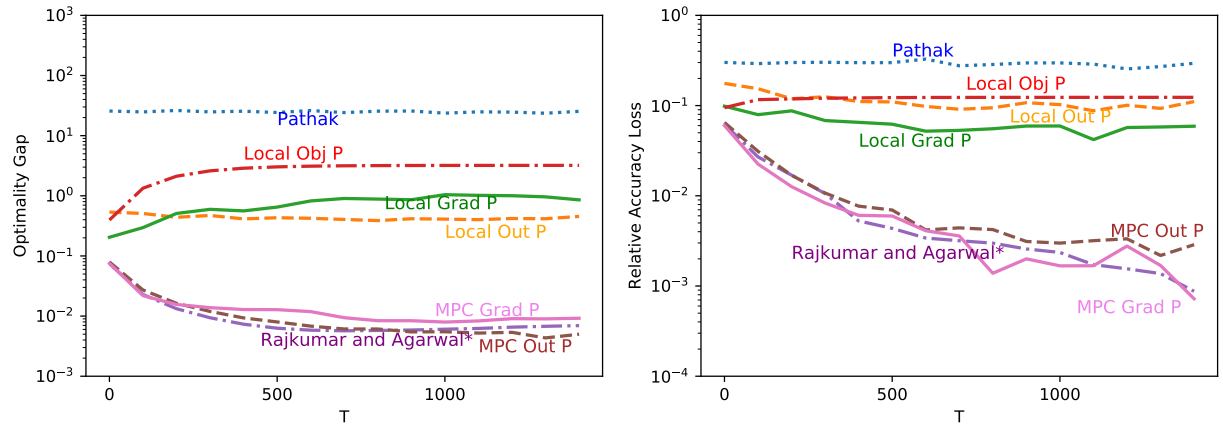


Figure 6.7: Optimality Gap and Relative Accuracy Loss Comparison on *Adult* ($m = 100$)

*All models except **Rajkumar and Agarwal** have privacy budget $\epsilon = 0.5$.

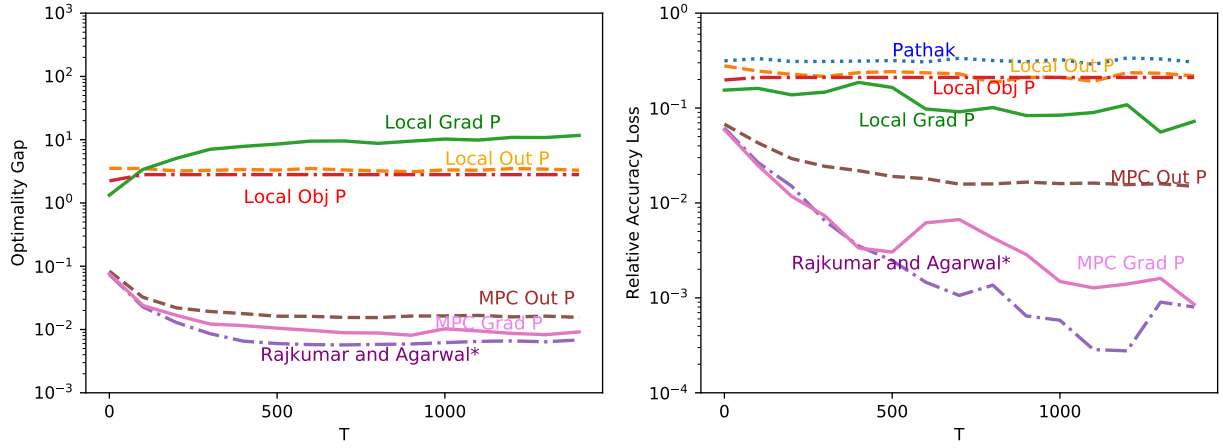


Figure 6.8: Optimality Gap and Relative Accuracy Loss Comparison on *Adult* ($m = 1,000$)

*All models except **Rajkumar and Agarwal** have privacy budget $\epsilon = 0.5$.

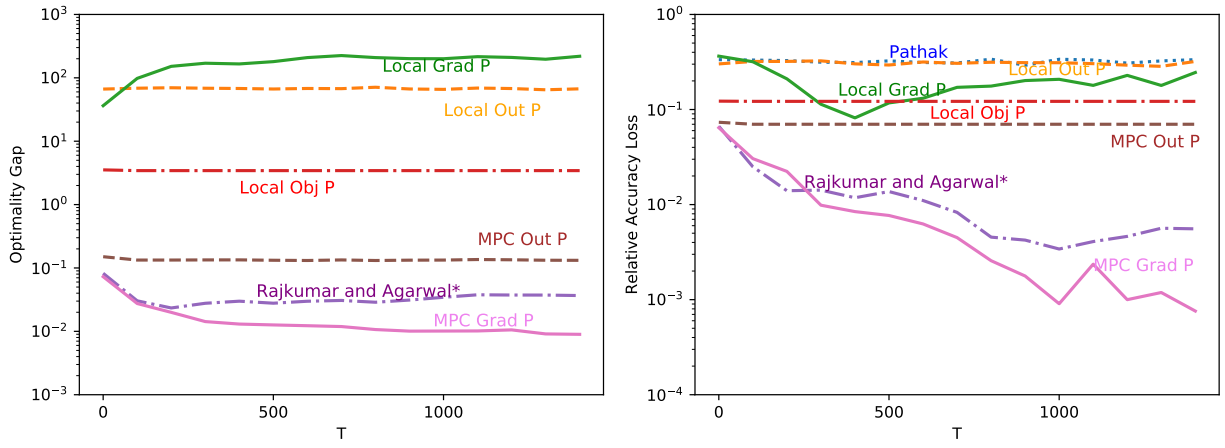


Figure 6.9: Optimality Gap and Relative Accuracy Loss Comparison on *Adult* ($m = 30,000$) *All models except **Rajkumar and Agarwal** have privacy budget $\epsilon = 0.5$.

6.7 Implementation of Secure Aggregation

For our prototype implementation of secure aggregation, we implemented the two-party non-collusive framework of Tian et al. (TJG16). We used the Obliv-C (ZE15b) framework for performing the MPC, and measured the runtime and bandwidth cost of doing secure aggregation along with noise generation inside the MPC framework (See Table 6.2). We conducted experiments using both semi-honest Yao's garbled circuits protocols, and in the active-secure dual execution model (HKE12). This provides security against fully malicious adversaries, but leaks up to one arbitrary bit of information about private inputs with each protocol execution.

Across all the data sets, the garbled circuit for performing the secure aggregation along with noise generation took around 700 to 900 MB of bandwidth, consisting of around 37 million to 48 million gates for Yao's protocol. The primary cost of the protocol is transmitting the ciphertexts of the garbled gates, and the circuit complexity is dominated by the noise generation. The bandwidth of oblivious transfer (OT) ranged from 32 to 41 MB. The cost is double for dual execution protocol. Note that the cost does not depend on the size of the

Table 6.2: Secure Aggregation in MPC

	Adult	KDDCup99	KDDCup98
Number of Features	104	122	95
Gate Count	40,733,500	47,781,200	37,209,600
GC Bandwidth (MB)	776.93	911.35	709.72
OT Bandwidth (MB)	35.17	41.19	32.28
Yao Runtime (sec)	22.28	26.45	20.31
DualEx Runtime (sec)	56.46	65.21	50.67

data set; it only depends on the number of parties, the number of features in the data set, the needed precision, and the method used to add the required differential privacy noise. As the table suggests, the number of gates grows linearly with the number of features, with roughly 391,600 gates per feature.

Overall secure aggregation took less than 2 seconds using Yao's protocol, and the remaining computation time was taken by the noise generation. For sampling the Laplace noise, each party first inputs its share of random number u_i, which is XORed inside the MPC to generate the randomness $u = \bigoplus u_i$. Next, the Laplace noise of scale b is generated as: $\text{Lap}(b) = \pm \log([u]).b$, where [.] shrinks the input to (0,1] range and $b = \frac{2}{m.n_{(1)}\lambda\epsilon}$. We used the big integer library (Doe17) to perform the log operation within the MPC. Gaussian noise can be generated in the same way using the Box-Muller method (BM58). There are many opportunities to reduce this cost without reducing security by generating the required noise differently, which we plan to explore in future work.

CHAPTER 7

Conclusion and Future Work

Privacy concerns emerge as one of the major problems when deploying machine learning methods in real-world applications. The main focus of the thesis is to address these concerns by developing more efficient and effective privacy-preserving machine learning algorithms. To this end, we propose a series of differentially private sparse learning methods in Chapter 2 and Chapter 3. In Chapter 2, we design a differentially private iterative gradient hard thresholding algorithm to train the private sparse model in a more efficient way. In Chapter 3, we establish a knowledge transfer framework, which focuses on improving the utility guarantee of the private sparse model.

In Chapter 4 and Chapter 5, we study the differentially private nonconvex optimization. In Chapter 4, we develop a private stochastic optimization algorithm for training deep neural network which can significantly reduce the computational complexities. Chapter 5 focuses on the utility of solving private empirical risk minimization.

In Chapter 6, we provide a new privacy-preserving framework for distributed empirical risk minimization. We prove that the proposed framework can achieve the same utility guarantee as the centralized one without sacrificing the privacy.

While we have made progress towards developing more efficient privacy-preserving machine learning algorithms with strong privacy and utility guarantees, there are still many unsolved fundamental questions. My future research plans focus on further pushing the frontier of privacy-preserving machine learning by understanding the fundamental limits of privacy-preserving methods, designing new, theoretically principled privacy-preserving ma-

chine learning methods for distributed/federated problems, and building privacy-preserving systems for real-world problems.

Understanding the Privacy Risks in Practice: Although differential privacy provides strong bounds on the worst-case privacy loss, it does not elucidate what privacy attacks could be realized in practice. Therefore, there remains a large gap between achievable privacy guarantees using privacy-preserving machine learning methods and what can be inferred using known attacks in practice. Therefore, it is very important to understand the privacy risks of existing privacy-preserving machine learning methods in practice. To achive this goal, our idea is to use the membership inference attack as the testbed for evaluating the privacy leakage of privacy-preserving machine learning methods. More specifically, the potential solution may consist of two parts: Firstly, we develop a new privacy leakage metric that captures the inference risk under realistic assumptions such as skewed priors (wherein members only make up a small fraction of the candidate pool). Secondly, we design more powerful inference attacks to empirically evaluate the privacy leakage of models trained with privacy-preserving machine learning methods. We believe that the potential solution can shed light on how vulnerable our model could be in practice even trained with privacy-preserving machine learning methods. We also aim to establish more powerful attacks, including attribute inference in the future, to help us better understand the privacy risks in practice.

Privacy-Preserving Federated Learning: Although we developed the privacy-preserving federated learning (FL) method using secure multi-party computation to achieve state-of-the-art privacy and utility guarantees in Chapter 6, there still exist many unsolved research questions in this field. For example, communication cost is often the bottleneck in FL. Therefore, it is imperative for us to design new privacy-preserving federated learning methods that can significantly reduce communication costs while maintaining strong privacy and utility guarantees. Besides, private data are often distributed non-i.i.d. across different users in FL. Thus, privacy-preserving federated learning methods should deal with the heterogeneous data without deteriorating both the privacy and utility guarantees. More importantly, FL

often relies on a trusted central server that can aggregate and analyze local information. What if the central server is vicious, and how can we protect each user's privacy without sacrificing the collective model's utility in such setting? I aim to address these research questions in privacy-preserving federated learning. By solving these research questions, I believe the new privacy-preserving federated learning methods we build will profoundly impact both literature and real-world applications.

Lower Bounds for Differentially Private Optimization: The most widely used optimization method for solving machine learning problems with privacy guarantees is the differentially private gradient-based method, such as DP-SGD. However, there is no lower bound result for differentially private optimization methods, and thus it is unclear whether existing differentially private optimization methods are optimal in terms of computational complexity. To better understand the fundamental limits of differentially private optimization methods, it is very important for us to establish lower bounds on the complexity of solving the privacy-preserving machine learning problem with only accessing up to first-order stochastic oracles. To this end, we will extend the existing proof technique from the non-private optimization problems to differentially private optimization problems, and our lower bounds will reveal how the privacy mechanism affects the optimization process with existing optimization methods such as stochastic gradient descent, and how can we design more efficient differentially private algorithms for solving privacy-preserving machine learning problems.

Bibliography

[Aa19] G. Andrew and et al. "TensorFlow Privacy." `https://github.com/tensorflow/privacy`, 2019.

[ACG16a] M. Abadi, A. Chu, I. Goodfellow, H. McMahan, I. Mironov, K. Talwar, and L. Zhang. "Deep Learning with Differential Privacy." In *23rd ACM Conference on Computer and Communications Security (CCS 2016)*, 2016.

[ACG16b] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. "Deep learning with differential privacy." In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 308–318. ACM, 2016.

[AH16] Zeyuan Allen-Zhu and Elad Hazan. "Variance reduction for faster non-convex optimization." In *International Conference on Machine Learning*, 2016.

[AN07] A. Asuncion and D. J. Newman. "UCI Machine Learning Repository.", 2007.

[ANW10] Alekh Agarwal, Sahand Negahban, and Martin J Wainwright. "Fast global convergence rates of gradient methods for high-dimensional statistical recovery." In *Advances in Neural Information Processing Systems*, pp. 37–45, 2010.

[BCD07] B. Barak, K. Chaudhuri, C. Dwork, S. Kale, F. McSherry, and K. Talwar. "Privacy, accuracy, and consistency too: a holistic solution to contingency table release." In *Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pp. 273–282. ACM, 2007.

[BCN06] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. "Model compression." In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 535–541. ACM, 2006.

[BDL17] M. Balcan, T. Dick, Y. Liang, W. Mou, and H. Zhang. "Differentially Private Clustering in High-Dimensional Euclidean Spaces." In *34th International Conference on Machine Learning (ICML 2017)*, 2017.

[BDL19] Zhiqi Bu, Jinshuo Dong, Qi Long, and Weijie J Su. "Deep Learning with Gaussian Differential Privacy." *arXiv:1911.11607*, 2019.

[BFT19] Raef Bassily, Vitaly Feldman, Kunal Talwar, and Abhradeep Guha Thakurta. "Private stochastic convex optimization with optimal rates." In *NeurIPS*, 2019.

[BIK17] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. "Practical Secure Aggregation for Privacy-Preserving Machine Learning." In *ACM Conference on Computer and Communications Security*, 2017.

[BM58] George E. P. Box and Mervin E. Muller. "A Note on the Generation of Random Normal Deviates." *The Annals of Mathematical Statistics*, 1958.

[BMS17] G. Bernstein, R. McKenna, T. Sun, D. Sheldon, M. Hay, and G. Miklau. "Differentially private learning of undirected graphical models using collective graphical models." In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 478–487. JMLR. org, 2017.

[BRB17] Vincent Bindschaedler, Shantanu Rane, Alejandro E Brito, Vanishree Rao, and Ersin Uzun. "Achieving differential privacy in secure multiparty data aggregation protocols on star networks." In *ACM Conference on Data and Application Security and Privacy*, 2017.

[BRT09] Peter J Bickel, Ya'acov Ritov, Alexandre B Tsybakov, et al. "Simultaneous analysis of Lasso and Dantzig selector." *The Annals of Statistics*, **37**(4):1705–1732, 2009.

[BS16a] M. Bun and T. Steinke. "Concentrated Differential Privacy: Simplifications, Extensions, and Lower Bounds." *ArXiv:1605.02065*, 2016.

[BS16b] Mark Bun and Thomas Steinke. "Concentrated differential privacy: Simplifications, extensions, and lower bounds." In *Theory of Cryptography Conference*, pp. 635–658. Springer, 2016.

[BSM10] Martin Burkhart, Mario Strasser, Dilip Many, and Xenofontas Dimitropoulos. "SEPIA: Privacy-preserving Aggregation of Multi-Domain Network Events and Statistics." In *USENIX Security Symposium*, 2010.

[BST14a] R. Bassily, A. Smith, and A. Thakurta. "Private Empirical Risk Minimization: Efficient Algorithms and Tight Error Bounds." In *55th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2014)*, 2014.

[BST14b] Raef Bassily, Adam Smith, and Abhradeep Thakurta. "Differentially private empirical risk minimization: Efficient algorithms and tight error bounds." *arXiv preprint arXiv:1405.7085*, 2014.

[BST14c] Raef Bassily, Adam Smith, and Abhradeep Thakurta. "Private empirical risk minimization: Efficient algorithms and tight error bounds." In *Symposium on Foundations of Computer Science*, 2014.

[BTT18] Raef Bassily, Om Thakkar, and Abhradeep Thakurta. "Model-Agnostic Private Learning via Stability." *arXiv preprint arXiv:1803.05101*, 2018.

[BW18] Borja Balle and Yu-Xiang Wang. "Improving the Gaussian Mechanism for Differential Privacy: Analytical Calibration and Optimal Denoising." *arXiv:1805.06530*, 2018.

[CG16] Jinghui Chen and Quanquan Gu. "Accelerated Stochastic Block Coordinate

Gradient Descent for Sparsity Constrained Nonconvex Optimization." In *UAI*, 2016.

[CGL17] Melissa Chase, Ran Gilad-Bachrach, Kim Laine, Kristin Lauter, and Peter Rindal. "Private collaborative neural network learning." Technical report, Cryptology ePrint Archive, Report 2017/762, 2017, 2017.

[CLE19] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. "The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks." In *USENIX Security Symposium*, 2019.

[CM08] K. Chaudhuri and C. Monteleoni. "Privacy-Preserving Logistic Regression." In *Advances in Neural Information Processing Systems (NIPS 2008)*, 2008.

[CM09] Kamalika Chaudhuri and Claire Monteleoni. "Privacy-preserving logistic regression." In *Advances in Neural Information Processing Systems*, pp. 289–296, 2009.

[CMS11a] K. Chaudhuri, C. Monteleoni, and A. Sarwate. "Differentially Private Empirical Risk Minimization." *Journal of Machine Learning Research*, **12**, 2011.

[CMS11b] Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. "Differentially private empirical risk minimization." *Journal of Machine Learning Research*, **12**(Mar):1069–1109, 2011.

[CO19] Ashok Cutkosky and Francesco Orabona. "Momentum-based variance reduction in non-convex SGD." In *NeurIPS*, 2019.

[CRT18] Yi-Ruei Chen, Amir Rezapour, and Wen-Guey Tzeng. "Privacy-Preserving Ridge Regression on Distributed Data." *Information Sciences*, 2018.

[DGK09] Ivan Damgård, Martin Geisler, Mikkel Krøigaard, and Jesper Buus Nielsen. "Asynchronous Multiparty Computation: Theory and Implementation." In *International Workshop on Public Key Cryptography*, 2009.

[DJW14] J. Duchi, M. Jordan, and M. Wainwright. "Privacy Aware Learning." *Journal of the Association for Computing Machinery*, **61(6)**, 2014.

[DKM06a] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. "Our data, ourselves: Privacy via distributed noise generation." In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 486–503. Springer, 2006.

[DKM06b] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. "Our Data, Ourselves: Privacy via Distributed Noise Generation." In *Annual International Conference on the Theory and Applications of Cryptographic Techniques (EuroCrypt)*, 2006.

[DKY17] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. "Collecting telemetry data privately." In *Advances in Neural Information Processing Systems*, pp. 3571–3580, 2017.

[DMN06] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. "Calibrating noise to sensitivity in private data analysis." In *Theory of Cryptography Conference*, pp. 265–284. Springer, 2006.

[DN04] Cynthia Dwork and Kobbi Nissim. "Privacy-Preserving Datamining on Vertically Partitioned Databases." In *Advances in Cryptology—CRYPTO*, 2004.

[Doe17] Jack Doerner. "Absentminded Crypto Kit." `https://bitbucket.org/jackdoerner/absentminded-crypto-kit`, 2017.

[DPS12] Ivan Damgård, Valerio Pastro, Nigel Smart, and Sarah Zakarias. "Multiparty Computation from Somewhat Homomorphic Encryption." In *Advances in Cryptology—CRYPTO*. 2012.

[DR14] Cynthia Dwork and Aaron Roth. *The Algorithmic Foundations of Differential Privacy*. Now Publishers, Inc., 2014.

[DR16] C. Dwork and G. Rothblum. "Concentrated Diffentially Privacy." *ArXiv:1603.01887*, 2016.

[DRS19] Jinshuo Dong, Aaron Roth, and Weijie J Su. "Gaussian Differential Privacy." *arXiv:1905.02383*, 2019.

[DRV10] C. Dwork, G. Rothblum, and S. Vadhan. "Boosting and Differential Privacy." In *51th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2010)*, 2010.

[Dwo08] Cynthia Dwork. "Differential Privacy: A Survey of Results." In *International Conference on Theory and Applications of Models of Computation*, 2008.

[EPK14] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. "Rappor: Randomized aggregatable privacy-preserving ordinal response." In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, pp. 1054–1067. ACM, 2014.

[FJR15] M. Fredrikson, S. Jha, and T. Ristenpart. "Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures." In *22nd ACM SIGSAC Conference on Computer and Communications Security (CCS 2015)*, 2015.

[FKT20] Vitaly Feldman, Tomer Koren, and Kunal Talwar. "Private Stochastic Convex Optimization: Optimal Rates in Linear Time." *arXiv preprint arXiv:2005.04763*, 2020.

[FLJ14] Matthew Fredrikson, Eric Lantz, Somesh Jha, Simon Lin, David Page, and Thomas Ristenpart. "Privacy in Pharmacogenetics: An end-to-end case study of personalized Warfarin dosing." In *USENIX Security Symposium*, 2014.

[FLL18] Cong Fang, Chris Junchi Li, Zhouchen Lin, and Tong Zhang. "Spider: Near-optimal non-convex optimization via stochastic path-integrated differential estimator." In *NeurIPS*, 2018.

[FR13] Simon Foucart and Holger Rauhut. *A mathematical introduction to compressive sensing*, volume 1. Birkhäuser Basel, 2013.

[FW56] Marguerite Frank and Philip Wolfe. "An algorithm for quadratic programming." *Naval research logistics quarterly*, **3**(1-2):95–110, 1956.

[FYZ17] W. Feng, Z. Yan, H. Zhang, K. Zeng, Y. Xiao, and Y. T. Hou. "A Survey on Security, Privacy and Trust in Mobile Crowdsourcing." *IEEE Internet of Things Journal*, 2017.

[Gee08] Sara A Van de Geer et al. "High-dimensional generalized linear models and the lasso." *The Annals of Statistics*, **36**(2):614–645, 2008.

[GFA17] Trinabh Gupta, Henrique Fingler, Lorenzo Alvisi, and Michael Walfish. "Pretzel: Email Encryption and Provider-Supplied Functions are Compatible." In *Conference of the ACM Special Interest Group on Data Communication (SIGCOMM)*, 2017.

[GL13] Saeed Ghadimi and Guanghui Lan. "Stochastic first-and zeroth-order methods for nonconvex stochastic programming." *SIAM Journal on Optimization*, **23**(4):2341–2368, 2013.

[GM17] A. Gilbert and A. McMillan. "Local Differential Privacy for Physical Sensor Data and Sparse Recovery." *arXiv:1706.05916*, 2017.

[GMW87] Shafi Goldwasser, Silvio M. Micali, and Avi Wigderson. "How to Play Any Mental Game, or a Completeness Theorem for Protocols with an Honest Majority." In *19th ACM Symposium on Theory of Computing*, 1987.

[GSB17] Adrià Gascón, Phillipp Schoppmann, Borja Balle, Mariana Raykova, Jack Doerner, Samee Zahur, and David Evans. "Privacy-preserving distributed linear regression on high-dimensional data." *Proceedings on Privacy Enhancing Technologies*, 2017.

[HB99] S. Hettich and S. D Bay. "UCI Machine Learning Repository.", 1999.

[HCB16] Jihun Hamm, Yingjun Cao, and Mikhail Belkin. "Learning privately from multiparty data." In *International Conference on Machine Learning*, pp. 555–563, 2016.

[HEK11] Yan Huang, David Evans, Jonathan Katz, and Lior Malka. "Faster Secure Two-Party Computation Using Garbled Circuits." In *20th USENIX Security Symposium*, 2011.

[HFK12] Andreas Holzer, Martin Franz, Stefan Katzenbeisser, and Helmut Veith. "Secure Two-Party Computations in ANSI C." In *ACM Conference on Computer and Communications Security*, 2012.

[HKE12] Yan Huang, Jonathan Katz, and David Evans. "Quid-Pro-Quo-tocols: Strengthening Semi-Honest Protocols with Dual Execution." In *IEEE Symposium on Security and Privacy*, 2012.

[HLK17] Mikko Heikkilä, Eemil Lagerspetz, Samuel Kaski, Kana Shimizu, Sasu Tarkoma, and Antti Honkela. "Differentially private Bayesian learning on distributed data." In *NeurIPS*, 2017.

[HLM09] M. Hay, C. Li, G. Miklau, and D. Jensen. "Accurate estimation of the degree distribution of private networks." In *2009 Ninth IEEE International Conference on Data Mining*, pp. 169–178. IEEE, 2009.

[HVD15] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. "Distilling the knowledge in a neural network." *arXiv preprint arXiv:1503.02531*, 2015.

[INS19] Roger Iyengar, Joseph P Near, Dawn Song, Om Thakkar, Abhradeep Thakurta, and Lun Wang. "Towards Practical Differentially Private Convex Optimization." In *IEEE Symposium on Security and Privacy*, 2019.

[JKT12a] P. Jain, P. Kothari, and A. Thakurta. "Differentially Private Online Learning." In *25th Conference on Learning Theory (COLT 2012)*, 2012.

[JKT12b] Prateek Jain, Pravesh Kothari, and Abhradeep Thakurta. "Differentially Private Online Learning." In *25^{th} Annual Conference on Learning Theory*, 2012.

[JT13] Prateek Jain and Abhradeep Thakurta. "Differentially Private Learning with Kernels." In *International Conference on Machine Learning*, 2013.

[JT14] Prateek Jain and Abhradeep Guha Thakurta. "(Near) dimension independent risk bounds for differentially private learning." In *International Conference on Machine Learning*, pp. 476–484, 2014.

[JTK14] Prateek Jain, Ambuj Tewari, and Purushottam Kar. "On iterative hard thresholding methods for high-dimensional m-estimation." In *Advances in Neural Information Processing Systems*, pp. 685–693, 2014.

[JTT18] P. Jain, O. Thakkar, and A. Thakurta. "Differentially Private Matrix Completion." In *35th International Conference on Machine Learning (ICML 2018)*, 2018.

[JWE18a] B. Jayaraman, L. Wang, D. Evans, and Q. Gu. "Distributed Learning without Distress: Privacy-Preserving Empirical Risk Minimization." In *Advances in Neural Information Processing Systems (NIPS 2018)*, 2018.

[JWE18b] Bargav Jayaraman, Lingxiao Wang, David Evans, and Quanquan Gu. "Distributed Learning without Distress: Privacy-Preserving Empirical Risk Minimization." In *Advances in Neural Information Processing Systems*, pp. 6346–6357, 2018.

[JZ13] Rie Johnson and Tong Zhang. "Accelerating stochastic gradient descent using predictive variance reduction." In *NeurIPS*, 2013.

[KB15] Diederik P Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization." In *International Conference on Learning Representations (ICLR)*, 2015.

[KLR09] Shimon Kogan, Dimitry Levin, Bryan R Routledge, Jacob S Sagi, and Noah A Smith. "Predicting risk from financial reports with regression." In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 272–280. Association for Computational Linguistics, 2009.

[KOV15] P. Kairouz, S. Oh, and P. Viswanath. "The Composition Theorem for Differential Privacy." In *32nd International Conference on Machine Learning (ICML 2015)*, 2015.

[Kri09] Alex Krizhevsky. "Learning Multiple Layers of Features from Tiny Images." Technical report, Citeseer, 2009.

[KST12] Daniel Kifer, Adam Smith, and Abhradeep Thakurta. "Private convex empirical risk minimization and high-dimensional regression." In *Conference on Learning Theory*, pp. 25–1, 2012.

[LAL16] Xingguo Li, Raman Arora, Han Liu, Jarvis Haupt, and Tuo Zhao. "Nonconvex sparse learning via stochastic optimization with progressive variance reduction." *arXiv preprint arXiv:1605.02711*, 2016.

[LBB98] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. "Gradient-based learning applied to document recognition." *Proceedings of the IEEE*, **86**(11):2278–2324, 1998.

[LGN17] Y. Liu, K. Gadepalli, M. Norouzi, G. Dahl, T. Kohlberger, A. Boyko, S. Venugopalan, A. Timofeev, P. Nelson, G. Corrado, and et al. "Detecting Cancer Metastases on Gigapixel Pathology Images." *arXiv:1703.02442*, 2017.

[LJC17] Lihua Lei, Cheng Ju, Jianbo Chen, and Michael I Jordan. "Non-convex finite-sum optimization via scsg methods." In *NeurIPS*, 2017.

[LK18a] J. Lee and D. Kifer. "Concentrated Differentially Private Gradient Descent with Adaptive per-iteration Privacy Budget." *ArXiv:1808.09501*, 2018.

[LK18b] Jaewoo Lee and Daniel Kifer. "Concentrated differentially private gradient descent with adaptive per-iteration privacy budget." In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018.

[LP00] Yehuda Lindell and Benny Pinkas. "Privacy Preserving Data Mining." In *Advances in Cryptology—CRYPTO*, 2000.

[LP07] Yehuda Lindell and Benny Pinkas. "An Efficient Protocol for Secure Two-Party Computation in the Presence of Malicious Adversaries." In *Advances in Cryptology—EUROCRYPT*. 2007.

[LP09] Yehuda Lindell and Benny Pinkas. "Secure Multiparty Computation for Privacy-Preserving Data Mining." *Journal of Privacy and Confidentiality*, 2009.

[LW13] Po-Ling Loh and Martin J Wainwright. "Regularized M-estimators with non-convexity: Statistical and algorithmic theory for local optima." In *Advances in Neural Information Processing Systems*, pp. 476–484, 2013.

[LYR04] David D Lewis, Yiming Yang, Tony G Rose, and Fan Li. "Rcv1: A new benchmark collection for text categorization research." *Journal of machine learning research*, **5**(Apr):361–397, 2004.

[Mir17] Ilya Mironov. "Rényi Differential Privacy." In *IEEE Computer Security Foundations Symposium*, 2017.

[MNP04] Dahlia Malkhi, Noam Nisan, Benny Pinkas, and Yaron Sella. "Fairplay-Secure Two-Party Computation System." In *USENIX Security Symposium*, 2004.

[MRT18a] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. "Learning Differentially Private Language Models Without Losing Accuracy." In *6^{th} International Conference on Learning Representations*, 2018.

[MRT18b] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. "Learning Differentially Private Recurrent Language Models." In *International Conference on Learning Representations*, 2018.

[MTZ19] Ilya Mironov, Kunal Talwar, and Li Zhang. "Rényi Differential Privacy of the Sampled Gaussian Mechanism." *arXiv preprint arXiv:1908.10530*, 2019.

[MZC18] Xu Ma, Fangguo Zhang, Xiaofeng Chen, and Jian Shen. "Privacy preserving multi-party computation delegation for deep learning in cloud computing." *Information Sciences*, **459**:103–116, 2018.

[NLS17] Lam M Nguyen, Jie Liu, Katya Scheinberg, and Martin Takáč. "SARAH: A novel method for machine learning problems using stochastic recursive gradient." In *34^{th} International Conference on Machine Learning*, 2017.

[NNO12] Jesper Buus Nielsen, Peter Sebastian Nordholt, Claudio Orlandi, and Sai Sheshank Burra. "A New Approach to Practical Active-Secure Two-Party Computation." In *Advances in Cryptology—CRYPTO*. 2012.

[NTZ13] A. Nikolov, K. Talwar, and L. Zhang. "The geometry of differential privacy: the sparse and approximate cases." In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pp. 351–360. ACM, 2013.

[NWI13a] V. Nikolaenko, U. Weinsberg, S. Ioannidis, M. Joye, D. Boneh, and N. Taft. "Privacy-Preserving Ridge Regression on Hundreds of Millions of Records." In *IEEE Symposium on Security and Privacy*, 2013.

[NWI13b] Valeria Nikolaenko, Udi Weinsberg, Stratis Ioannidis, Marc Joye, Dan Boneh, and Nina Taft. "Privacy-preserving Ridge Regression on hundreds of millions of records." In *IEEE Symposium on Security and Privacy*, 2013.

[NYW09] Sahand Negahban, Bin Yu, Martin J Wainwright, and Pradeep K Ravikumar. "A unified framework for high-dimensional analysis of M-estimators with decomposable regularizers." In *Advances in Neural Information Processing Systems*, pp. 1348–1356, 2009.

[OWY18] S. Osher, B. Wang, P. Yin, X. Luo, M. Pham, and A. Lin. "Laplacian Smoothing Gradient Descent." *ArXiv:1806.06317*, 2018.

[PAE16] Nicolas Papernot, Martín Abadi, Ulfar Erlingsson, Ian Goodfellow, and Kunal Talwar. "Semi-supervised knowledge transfer for deep learning from private training data." In *International Conference on Learning Representations*, 2016.

[PAE17] N. Papernot, M. Abadi, U. Erlingsson, I. Goodfellow, and K. Talwar. "Semisupervised Knowledge Transfer for Deep Learning from Private Training Data." In *5th International Conference on Learning Representation (ICLR 2017)*, 2017.

[PFC16] M. Park, J. Foulds, K. Chaudhuri, and M. Welling. "Private Topic Modeling." *arXiv:1609:04120*, 2016.

[PH98] Ismail Parsa and Ken Howes. "UCI Machine Learning Repository.", 1998.

[PRR10] Manas Pathak, Shantanu Rane, and Bhiksha Raj. "Multiparty differential privacy via aggregation of locally trained classifiers." In *NeurIPS*, 2010.

[PSM18] N. Papernot, S. Song, I. Mironov, A. Raghunathan, K. Talwar, and U. Erlingsson. "Scalable Private Learning with PATE." In *International Conference on Learning Representations (ICLR 2018)*, 2018.

[PSS09] Benny Pinkas, Thomas Schneider, Nigel P Smart, and Stephen C Williams. "Secure Two-Party Computation Is Practical." In *International Conference on the Theory and Application of Cryptology and Information Security*, 2009.

[RA12] Arun Rajkumar and Shivani Agarwal. "A Differentially Private Stochastic Gradient Descent Algorithm for Multiparty Classification." In *Artificial Intelligence and Statistics*, 2012.

[Ren61] Alfréd Rényi. "On measures of entropy and information." Technical report, HUNGARIAN ACADEMY OF SCIENCES Budapest Hungary, 1961.

[RHH14] Aseem Rastogi, Matthew A Hammer, and Michael Hicks. "Wysteria: A Programming Language for Generic, Mixed-Mode Multiparty Computations." In *35th IEEE Symposium on Security and Privacy*, 2014.

[RHS16] Sashank J Reddi, Ahmed Hefny, Suvrit Sra, Barnabas Poczos, and Alex Smola. "Stochastic variance reduction for nonconvex optimization." In *International Conference on Machine Learning*, 2016.

[RSP16] Sashank J Reddi, Suvrit Sra, Barnabás Póczos, and Alex Smola. "Fast incremental method for smooth nonconvex optimization." In *IEEE Conference on Decision and Control*, 2016.

[RWY11] Garvesh Raskutti, Martin J Wainwright, and Bin Yu. "Minimax rates of esti-

mation for high-dimensional linear regression over ℓ_q-balls." *IEEE transactions on information theory*, **57**(10):6976–6994, 2011.

[RZ12] Mark Rudelson and Shuheng Zhou. "Reconstruction from anisotropic random measurements." In *Conference on Learning Theory*, pp. 10–1, 2012.

[SCL15] D. Su, J. Cao, N. Li, E. Bertino, and H. Jin. "Differentially Private k-Means Clustering." *arXiv:1504.05998*, 2015.

[SCR17] Elaine Shi, T.-H. Hubert Chan, Eleanor Rieffel, and Dawn Song. "Distributed Private Data Analysis: Lower Bounds and Practical Constructions." *ACM Transactions on Algorithms*, 2017.

[SCS13] S. Song, K. Chaudhuri, and A. Sarwate. "Stochastic Gradient Descent with Differentially Private Updates." In *GlobalSIP Conference*, 2013.

[SS15a] R. Shokri and V. Shmatikov. "Privacy-Preserving Deep Learning." In *22nd ACM SIGSAC Conference on Computer and Communications Security (CCS 2015)*, 2015.

[SS15b] Reza Shokri and Vitaly Shmatikov. "Privacy-Preserving Deep Learning." In *ACM Conference on Computer and Communications Security*, 2015.

[SSS09] Karthik Sridharan, Shai Shalev-shwartz, and Nathan Srebro. "Fast Rates for Regularized Objectives." In *Advances in Neural Information Processing Systems*. 2009.

[SSS17a] R. Shokri, M. Stronati, C. Song, and V. Shmatikov. "Membership Inference Attacks Against Machine Learning Models." *Proceedings of the 2017 IEEE Symposium on Security and Privacy*, 2017.

[SSS17b] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. "Membership inference attacks against machine learning models." In *IEEE Symposium on Security and Privacy*, 2017.

[Tib96] Robert Tibshirani. "Regression shrinkage and selection via the lasso." *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.

[TJG16] Lu Tian, Bargav Jayaraman, Quanquan Gu, and David Evans. "Aggregating Private Sparse Learning Models Using Multi-Party Computation." In *NIPS Workshop on Private Multi-Party Machine Learning*, 2016.

[TS13] Abhradeep Guha Thakurta and Adam Smith. "Differentially private feature selection via stability arguments, and the robustness of the lasso." In *Conference on Learning Theory*, pp. 819–850, 2013.

[TTZ15] Kunal Talwar, Abhradeep Guha Thakurta, and Li Zhang. "Nearly optimal private lasso." In *Advances in Neural Information Processing Systems*, pp. 3025–3033, 2015.

[Ver10] R. Vershynin. "Introduction to the non-asymptotic analysis of random matrices." *arXiv preprint arXiv:1011.3027*, 2010.

[VKC08] Jaideep Vaidya, Murat Kantarcıoğlu, and Chris Clifton. "Privacy-Preserving Naïve Bayes Classification." *The VLDB Journal*, 2008.

[WBK19] Yu-Xiang Wang, Borja Balle, and Shiva Prasad Kasiviswanathan. "Subsampled Renyi Differential Privacy and Analytical Moments Accountant." In *International Conference on Artificial Intelligence and Statistics*, 2019.

[WCX19] Di Wang, Changyou Chen, and Jinhui Xu. "Differentially Private Empirical Risk Minimization with Non-convex Loss Functions." In *International Conference on Machine Learning*, 2019.

[WDC18] Qian Wang, Minxin Du, Xiuying Chen, Yanjiao Chen, Pan Zhou, Xiaofeng Chen, and Xinyi Huang. "Privacy-preserving collaborative model learning: The case of word vector training." *IEEE Transactions on Knowledge and Data Engineering*, **30**(12):2381–2393, 2018.

[WG19a] Lingxiao Wang and Quanquan Gu. "Differentially Private Iterative Gradient Hard Thresholding for Sparse Learning." In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 2019.

[WG19b] Lingxiao Wang and Quanquan Gu. "Differentially Private Iterative Gradient Hard Thresholding for Sparse Learning." In *International Joint Conference on Artificial Intelligence*, 2019.

[WG20] Lingxiao Wang and Quanquan Gu. "A Knowledge Transfer Framework for Differentially Private Sparse Learning." *AAAI*, 2020.

[WGX18] Di Wang, Marco Gaboardi, and Jinhui Xu. "Empirical risk minimization in non-interactive local differential privacy revisited." In *Advances in Neural Information Processing Systems*, pp. 973–982, 2018.

[WJE19] Lingxiao Wang, Bargav Jayaraman, David Evans, and Quanquan Gu. "Efficient Privacy-Preserving Nonconvex Optimization." *arXiv preprint arXiv:1910.13659*, 2019.

[WLF16] Y. Wang, J. Lei, and S. Fienberg. "Learning with Differential Privacy: Stability, Learnability and the Sufficiency and Necessity of ERM Principle." *ArXiv:1502.06309*, 2016.

[WLK17] Xi Wu, Fengan Li, Arun Kumar, Kamalika Chaudhuri, Somesh Jha, and Jeffrey Naughton. "Bolt-on differential privacy for scalable stochastic gradient descent-based analytics." In *ACM International Conference on Management of Data*, 2017.

[WMK16] Xiao Wang, Alex J. Malozemoff, and Jonathan Katz. "EMP-toolkit: Efficient MultiParty computation toolkit." `https://github.com/emp-toolkit`, 2016.

[WRK17] Xiao Wang, Samuel Ranellucci, and Jonathan Katz. "Global-Scale Secure Multiparty Computation." In *ACM Conference on Computer and Communications Security*, 2017.

[WYX17a] Di Wang, Minwei Ye, and Jinhui Xu. "Differentially Private Empirical Risk Minimization Revisited: Faster and More General." In *Advances in Neural Information Processing Systems*, 2017.

[WYX17b] Di Wang, Minwei Ye, and Jinhui Xu. "Differentially Private Empirical Risk Minimization Revisited: Faster and More General." In *Advances in Neural Information Processing Systems*, pp. 2719–2728, 2017.

[XCM12] Huan Xu, Constantine Caramanis, and Shie Mannor. "Sparse algorithms are not stable: A no-free-lunch theorem." *IEEE transactions on pattern analysis and machine intelligence*, **34**(1):187–193, 2012.

[XLW18] Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. "Differentially Private Generative Adversarial Network." *arXiv preprint arXiv:1802.06739*, 2018.

[Yao82] Andrew C Yao. "Protocols for Secure Computations." In *Symposium on Foundations of Computer Science*, 1982.

[YJS19] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. "PATE-GAN: Generating Synthetic Data with Differential Privacy Guarantees." In *International Conference on Learning Representations*, 2019.

[YKL11] M. Yuen, I. King, and K. Leung. "A Survey of Crowdsourcing Systems." In

Proceedings of the IEEE international conference on social computing (Socialcom 2011), 2011.

[YLL20] Huizhuo Yuan, Xiangru Lian, Ji Liu, and Yuren Zhou. "Stochastic Recursive Momentum for Policy Gradient Methods." *arXiv preprint arXiv:2003.04302*, 2020.

[YLZ14] Xiaotong Yuan, Ping Li, and Tong Zhang. "Gradient hard thresholding pursuit for sparsity-constrained optimization." In *International Conference on Machine Learning*, pp. 127–135, 2014.

[YS15] Y. Wang Y. Wang and A. Singh. "Differentially Private Subspace Clustering." In *Advances in Neural Information Processing Systems (NIPS 2015)*, 2015.

[YZW05] Zhiqiang Yang, Sheng Zhong, and Rebecca N. Wright. "Privacy-Preserving Classification of Customer Data without Loss of Accuracy." In *SIAM International Conference on Data Mining*, 2005.

[ZE15a] Samee Zahur and David Evans. "Obliv-C: A Language for Extensible Data-Oblivious Computation." *IACR Cryptol. ePrint Arch.*, **2015**:1153, 2015.

[ZE15b] Samee Zahur and David Evans. "Obliv-C: A Language for Extensible Data-Oblivious Computation." Cryptology ePrint Archive, Report 2015/1153, 2015.

[Zha10] Tong Zhang. "Analysis of multi-stage convex relaxation for sparse regularization." *Journal of Machine Learning Research*, **11**(Mar):1081–1107, 2010.

[Zha11] Tong Zhang. "Adaptive forward-backward greedy algorithm for learning sparse representations." *IEEE transactions on information theory*, **57**(7):4689–4708, 2011.

[ZW19] Yuqing Zhu and Yu-Xiang Wang. "Poission Subsampled Rényi Differential Privacy." In *International Conference on Machine Learning*, 2019.

[ZXG18] Dongruo Zhou, Pan Xu, and Quanquan Gu. "Stochastic nested variance reduction for nonconvex optimization." In *NeurIPS*, 2018.

[ZZM17a] J. Zhang, K. Zheng, W. Mou, and L. Wang. "Efficient Private ERM for Smooth Objectives." In *The Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI 2017)*, 2017.

[ZZM17b] Jiaqi Zhang, Kai Zheng, Wenlong Mou, and Liwei Wang. "Efficient private ERM for smooth objectives." *arXiv preprint arXiv:1703.09947*, 2017.