Towards More Scalable and Robust Machine Learning

Dong Yin

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2019-175

http://www2.eecs.berkeley.edu/Pubs/TechRpts/2019/EECS-2019-175.html

December 17, 2019

Copyright © 2019, by the author(s).
All rights reserved.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

Towards More Scalable and Robust Machine Learning

by

Dong Yin

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering - Electrical Engineering and Computer Sciences

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Kannan Ramchandran, Chair
Professor Peter Bartlett
Professor Zuo-Jun Shen

Fall 2019

The dissertation of Dong Yin, titled Towards More Scalable and Robust Machine Learning, is approved:

Chair	________________________	Date	__________
	________________________	Date	__________
	________________________	Date	__________

University of California, Berkeley

Towards More Scalable and Robust Machine Learning

Copyright 2019
by
Dong Yin

Abstract

Towards More Scalable and Robust Machine Learning

by

Dong Yin

Doctor of Philosophy in Engineering - Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Kannan Ramchandran, Chair

For many data-intensive real-world applications, such as recognizing objects from images, detecting spam emails, and recommending items on retail websites, the most successful current approaches involve learning rich prediction rules from large datasets. There are many challenges in these machine learning tasks. For example, as the size of the datasets and the complexity of these prediction rules increase, there is a significant challenge in designing *scalable* methods that can effectively exploit the availability of distributed computing units. As another example, in many machine learning applications, there can be data corruptions, communication errors, and even adversarial attacks during training and test. Therefore, to build reliable machine learning models, we also have to tackle the challenge of *robustness* in machine learning.

In this dissertation, we study several topics on the scalability and robustness in large-scale learning, with a focus of establishing solid theoretical foundations for these problems, and demonstrate recent progress towards the ambitious goal of building more scalable and robust machine learning models. We start with the speedup saturation problem in distributed stochastic gradient descent (SGD) algorithms with large mini-batches. We introduce the notion of gradient diversity, a metric of the dissimilarity between concurrent gradient updates, and show its key role in the convergence and generalization performance of mini-batch SGD. We then move forward to Byzantine distributed learning, a topic that involves both scalability and robustness in distributed learning. In the Byzantine setting that we consider, a fraction of distributed worker machines can have arbitrary or even adversarial behavior. We design statistically and computationally efficient algorithms to defend against Byzantine failures in distributed optimization with convex and non-convex objectives. Lastly, we discuss the adversarial example phenomenon. We provide theoretical analysis of the adversarially robust generalization properties of machine learning models through the lens of Radamacher complexity.

To my wife and parents.

Contents

List of Figures

List of Tables

Acknowledgments

I would like to begin with my foremost thanks to my advisor, Kannan Ramchandran. I started working with Kannan from the beginning of my PhD study, and we have closely collaborated on all my research projects at UC Berkeley. Through Kannan's guidance, I learned all the aspects of how to do real academic research, including but not limited to identifying good problem setups, pursuing the best possible results, and writing papers that impress the readers. The most important lesson that I learned from Kannan is that as a researcher, asking a good question is perhaps more important than working out the details. In fact, Kannan is a great advisor who is really good at identifying problems that are interesting and important both theoretically and practically. Kannan also has a great vision and broad interest in many research areas. I am extremely grateful that Kannan provides me with the full freedom of choosing research directions of my own interest and encourages me to build diverse expertise in different fields.

I am also very fortunate to work with Peter Bartlett on various topics during my PhD. I started collaborations with Peter back in 2016 when I was taking his course in statistical learning theory. Peter has great vision of the entire field of machine learning, including statistical learning theory, neural networks, and reinforcement learning. Peter is really knowledgeable about how the field of machine learning has been evolving, and has given me very useful advice from a "historical" perspective. From Peter, I learned both solid theoretical analysis techniques and approaches to finding good research directions. In particular, he always encourages me to work on challenging problems, such as proving the optimality of the results by finding matching lower bounds.

I would like to thank Max Shen, for joining my qualifying exam and dissertation committees. I have heard of Max's name in operations research when I was in my undergraduate school, and it is a pleasure to meet him at Berkeley. I sincerely appreciate his helpful suggestions for my dissertation. In addition, I would like to thank Laurent El Ghaoui for joining my qualifying exam committee.

During the long journey of my PhD, I was fortunate to have many amazing collaborators. I would like to give special thanks to Yudong Chen. I have known Yudong when I started my PhD at Berkeley, and at that time he was a postdoc here. During the past five years, we had many insightful discussions and nice collaborations on various topics, including learning mixtures of regressions and robust distributed learning. Yudong has helped me with many aspects of research, from technical details to the structure and writing of papers.

I would also like to thank Ramtin Pedarsani, Xiao Simon Li, and Kangwook Lee for their help with my research projects on sparse graph codes during the first two years of my PhD. I am also very grateful to Dimitris Papailiopoulos, who led me to my first research project on machine learning and distributed optimization. During the last year of my PhD, I also had many insightful discussions with Jiantao Jiao, who has shown me many smart the approaches to tackling various problems.

I would like to thank all the faculties in the BLISS research lab, for providing an open and collaborative research environment. Besides Kannan and Jiantao, the other professors in BLISS lab are: Venkat Anantharam, Thomas Courtade, Anant Sahai, Pravin Varaiya, Martin Wainwright, and Jean Walrand. I am also grateful to all the professors of the courses that I have taken at Berkeley. I really appreciate the knowledge that I learned from the courses: they made me better prepared for my research. In a chronological order, the professors are: Thomas Courtade, Jean Walrand, Laura Waller, Benjamin Recht, Michael I. Jordan, Peter Bartlett, Martin Wainwright, Laurent El Ghaoui, Dawn Song, and Trevor Darrell. I am also thankful to Kannan Ramchandran and Shobhana Stoyanov, who were the instructors of my GSI courses, for teaching me how to teach.

During my PhD, I have done three internships, all at Google. These internships are really good opportunities for me to combine my background in academic research with more applied problems. Some of the results in my internship projects end up being published in academic conferences. I am really thankful to my internship hosts — Andrés Muñoz Medina, Qifan Wang, and Justin Gilmer, for their guidance and helpful advice during the internships.

In addition to the people mentioned above, I would like to thank all my other coauthors and collaborators in my ongoing projects. I won't be able to complete my research projects without their help. Here, I list them in a chronological order: Adam S. Charles, Christopher J. Rozell, Sameer Pawar, Ashwin Pananjady, Max Lam, Sergei Vassilvitskii, Ekin D. Cubuk, Raphael Gontijo Lopes, Jonathon Shlens, Ben Poole, Avishek Ghosh, Justin Hong, Jichan Chung, Xiang Cheng, and Michael I. Jordan.

I would also like to thank my friends in the BLISS, BAIR, Simons Institute for the Theory of Computing, Google, etc. It was a pleasure to meet all of them and the insightful research discussions with some of them really opened my mind. Here, I would like to give thanks to the following people, listed in an alphabetic order: Reza Abbasi-Asl, Amirali Aghazadeh, Zeyuan Allen-Zhu, Efe Aras, Sivaraman Balakrishnan, Kush Bhatia, Niladri Chatterji, Lin Chen, Zihao Chen, Yeshwanth Cherapanamjeri, Bo Dai, Payam Delgosha, Ilias Diakonikolas, Simon S Du, Raaz Dwivedi, Giulia Fanti, Vipul Gupta, Sang Min Han, Kate Harrison, Horace He, Reinhard Heckel, Shuguang Hu, Longbo Huang, Shloak Jain, Chi Jin, Varun Jog, Rashmi K V, Swanand Kadhe, Vijay Kamble, Koulik Khamaru, Alex Kulesza, Jason Lee, Kuan-Yun Lee, Jerry Li, Shiau Hong Lim, Liu Liu, Po-Ling Loh, Phil Long, Tengyu Ma, Alan Malek, Wenlong Mou, Vidya Muthukumar, Orhan Ocal, Frank Ong, Aldo Pacchiano, Soham Phade, Yannik Pitcan, Gireeja Ranade, Ludwig Schmidt, Nihar B Shah, D. Sivakumar, Vasuki Narasimha Swamy, Hongyi Wang, Yuting Wei, Fanny Yang, Yaoqing Yang, Yun Yang, Felix Yu, Yang Yuan, Chiyuan Zhang, Yuchen Zhang, and Banghua Zhu.

I would like to give special thanks to Berkeley DeepDrive (BDD) for generously funding me for many of the semesters during my PhD. I would also like to thank many staffs in the EECS department, especially Shirley Salanio, for promptly answering my questions.

Lastly but most importantly, my foremost gratitude goes to my family. I would like to thank my wife Miao Cheng, for the sweet time that we spent together, and for her encouragement and support during the long journey of my PhD. I would also like to thank my parents and parent-in-laws for their selfless support during these years. Finally, I would like to thank our cat, Miku, for bringing so much fun to our family.

Chapter 1

Introduction

For many data-intensive real-world applications, such as recognizing objects from images, detecting spam emails, and recommending items on retail websites, the most successful current approaches involve learning rich prediction rules from large datasets. There are many challenges in these machine learning tasks. For example, as the size of the datasets and the complexity of these prediction rules increase, there is a significant challenge in designing *scalable* methods that can effectively exploit the availability of distributed computing units. As another example, in many machine learning applications, there can be data corruptions, communication errors, and even adversarial attacks during training and test. Therefore, to build reliable machine learning models, we have to tackle the challenge of *robustness* in machine learning.

1.1 Challenges in Scalable Machine Learning

As the scale of training datasets and model parameters increases, it becomes more and more important to efficiently utilize the availability of distributed computing devices in training machine learning models. In fact, distributed optimization has become the cornerstone of many real-world machine learning applications. Many of the state-of-the-art publicly available (distributed) machine learning frameworks, such as Tensorflow [1] and MXNet [40], offer distributed implementations of popular learning algorithms. In many applications, a batch of data are distributed over multiple machines for parallel processing in order to speed up computation. In other settings, the data sources are naturally distributed, and for privacy and efficiency considerations, the data are not transmitted to a central machine. An example is the recently proposed *Federated Learning* paradigm [135, 108, 107], in which the data are stored and processed locally in end users' cellphones and personal computers. We illustrate Federated Learning in Figure 1.1.

Let us consider the problem of gaining speedup using distributed computing. Ideally, the speedup gain by using parallel computing nodes should be linear, i.e., when we double the number of parallel devices, the time that we need to train a good model should be halved.

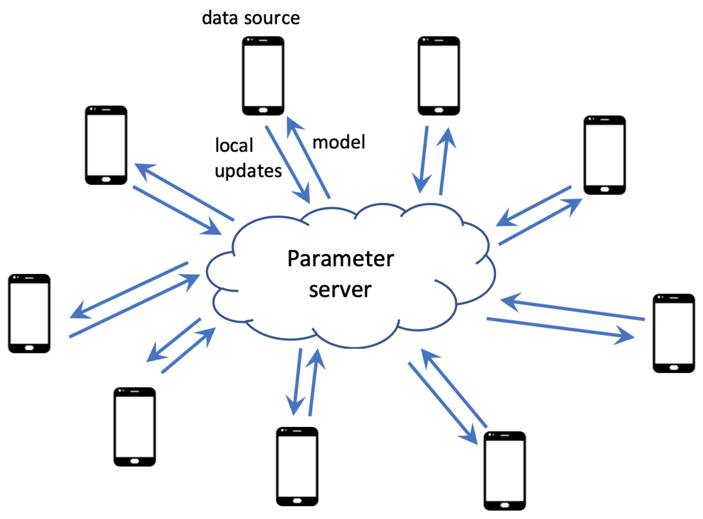


Figure 1.1: Federated Learning. The parameter server stores the model and the end users' devices store the data. In a Federated Learning algorithm, the parameter server sends the model to the personal devices, and the devices send the local updates back to the parameter server.

However, for many machine learning algorithms, this is not true. In fact, although with more distributed computing nodes, the training algorithms are usually faster, it is often observed that, in many distributed implementation of learning algorithms, linear speedup is only possible for up to tens of computing nodes, and several studies have shown that there is a significant gap between ideal and realizable speedups when scaling out to hundreds of compute nodes [52, 153]. This commonly observed phenomenon is referred to as *speedup saturation*.

This phenomenon is not hard to understand when we take the communication overhead in distributed systems into account. For example, when we implement the stochastic gradient descent (SGD) algorithm in a standard system with one master machine (parameter server) and several worker machines (computing nodes), in each iteration, the master machine needs to send the model parameters to every worker machine, wait until all the worker machine finish their computation jobs and send back the results, and then update the model parameters. In this process, the delay in communication channels and stragglers among worker machines can both be the communication overhead, and when the communication cost is comparable with the speedup gain by using more worker machines, the speedup saturation phenomenon happens.

Therefore, the key to designing communication-efficient distributed learning algorithms is reducing the communication cost. There are many approaches in the literature; however, most of them can be classified into two categories: 1) reducing the total number of iterations, and 2) reducing the communication cost in each iteration. Examples of the first category are one-round algorithm [209, 206], approximate Newton method [165], large-batch training [80], etc; and examples of the second category are asynchronous training (reducing the cost of synchronization in each iteration) [156], coded computing (reducing the effect of stragglers) [120], gradient compression (reducing the size of the messages transmitted in each iteration) [5, 189], etc.

However, we usually cannot reduce the communication cost for free, and when we apply these communication-efficient learning algorithms, there are often trade-offs between the speedup gain due to smaller communication cost and the degradation in the performance of the algorithm. For example, if we apply the one-round algorithm, in which each worker machine computes a local solution and the master machine averages them, to solve a learning problem, the final solution will suffer from an unavoidable bias since each worker machine only uses a relatively small amount of data [165]. Another example is that, in asynchronous training and gradient compression schemes, although we can gain speedup by using less accurate information about the gradients, the final solution that we get might become worse. Thus, understanding and mitigating these trade-offs are crucial to the field of distributed learning.

In this dissertation, we will focus on large-batch training—one common way to reduce the number of iterations in SGD algorithms, and study the fundamental trade-offs when using large batch sizes. We will also discuss potential approaches to mitigating these trade-offs.

1.2 Challenges in Robust Machine Learning

Robustness is another significant challenge in machine learning. In recent years, machine learning models, in particular, deep neural networks, have achieved remarkable success in many tasks on standard benchmarks. However, when we implement these models in real-world security-critical applications such as medical diagnosis and autonomous driving, it is crucial to guarantee that these models can work reliably in the presence of data corruption, distributional shift and even adversarial attacks. In practice, the robustness challenge can happen during every stage of the learning algorithm: training, test, and even both. In the following, we elaborate some training-time and test-time robustness problems.

During training, there can be noise in the features and labels [141], adversarially injected poisoning data [171], adversarial manipulation to the training algorithms and systems [23], etc. In particular, training-time robustness problem is highly related to the long-standing topic of robust statistics [89], which considers statistical estimation and inference problems under adversarially corrupted data.

In this dissertation, we discuss a training-time robustness problem in distributed learning, known as the *Byzantine* setting [114]. In a large-scale distributed systems, robustness and security issues have become a major concern. In particular, individual worker machines may exhibit abnormal behavior due to crashes, faulty hardware, stalled computation or unreliable communication channels. Security issues are only exacerbated in the Federated Learning setting, in which the data owners' devices (such as mobile phones and personal computers) are used as worker machines. Such machines are often more unpredictable, and in particular may be susceptible to malicious and coordinated attacks. Here, we note that in the Byzantine distributed learning problem, we consider a stronger attack model than in the traditional robust statistics problems, since the adversary can not only corrupt the data on some of the worker machines, they may also send malicious messages in each iteration. In this dissertation, we will design distributed optimization algorithms that are provably robust to Byzantine failures.

Robustness can also become a crucial concern during test. It has been observed that even for the models that can achieve the state-of-the-art performance in many standard benchmarks or competitions, by adversarially adding some perturbation to the input of the model (images, audio signals), these models can make wrong predictions with high confidence. These adversarial inputs are often called the adversarial examples. Typical methods of generating adversarial examples include adding small perturbations that are imperceptible to humans [177], changing surrounding areas of the main objects in images [76], and even simple rotation and translation [63]. Moreover, it has also been observed that even if the perturbations are not adversarial, some distributional shift between the training data and test data, such as common corruptions to images, can also lead to significant performance degradation [85, 157]. Both adversarial examples and distributional shift raise significant concerns about the reliablity of machine learning models in real-world applications.

Test-time robustness problems, especially the adversarial example phenomenon, are still widely open. The performance of the defense algorithms against adversarial attacks is overall not satisfactory enough [11]. In addition, currently we do not have enough theoretically principled analysis of this phenomenon. In this dissertation, we focus on the adversarially robust generalization problem, and present some rigorous analysis in this field.

1.3 Organization

In this dissertation, we study several topics on the scalability and robustness in large-scale learning, with a focus of establishing solid theoretical foundations for these problems. This dissertation is organized as follows.

In Chapter 2, we study the speedup saturation problem in distributed stochastic gradient descent (SGD) algorithms with large mini-batches. We introduce the notion of *gradient diversity*, a metric of the dissimilarity between concurrent gradient updates, and show its key role in the convergence and generalization performance of mini-batch SGD. We also introduce several diversity-inducing mechanisms, and provide experimental evidence indicating that these mechanisms can enable the use of larger batches without sacrificing the final accuracy, and lead to faster training in distributed learning. This chapter is based on joint work with Ashwin Pananjady, Max Lam, Dimitris Papailiopoulos, Kannan Ramchandran, and Peter Bartlett, and this work was published in Proceedings of the 21st International Conference on Artificial Intelligence and Statistics (AISTATS) in 2018 [199].

In Chapter 3, we move forward to robust distributed learning, a topic that involves both scalability and robustness in machine learning. We consider the *Byzantine* setting, where a fraction of distributed worker machines can have arbitrary or even adversarial behavior. We design statistically and computationally efficient algorithms for Byzantine-robust distributed learning by combining distributed gradient descent with robust estimation subroutines such as median and trimmed mean. We establish the statistical rates of the proposed algorithm with these subroutines and prove their optimality in various regimes. This chapter is based on joint work with Yudong Chen, Kannan Ramchandran, and Peter Bartlett. This work was published in Proceedings of the 35th International Conference on Machine Learning (ICML) in 2018 [197].

In Chapter 4, we continue to study Byzantine-robust distributed learning with focus on the non-convex setting. We design *ByzantinePGD*, an efficient and robust distributed learning algorithm that can provably escape saddle points and converge to second-order stationary points in Byzantine distributed learning. This chapter is based on joint work with Yudong Chen, Kannan Ramchandran, and Peter Bartlett. This work was published in Proceedings of the 36th International Conference on Machine Learning (ICML) in 2019 [198].

In Chapter 5, we move to the test-time adversarial robustness problem. We focus on ℓ_∞ adversarial perturbations and study the adversarially robust generalization problem through the lens of Rademacher complexity. For binary linear classifiers, we establish tight bounds for the adversarial Rademacher complexity, and show that it is never smaller than its natural counterpart, and has an unavoidable dimension dependence, unless the weight vector has bounded ℓ_1 norm. We also present extensions to multi-class linear classifiers and (non-linear) neural networks. This chapter is based on joint work with Kannan Ramchandran, and Peter Bartlett. This work was published in Proceedings of the 36th International Conference on Machine Learning (ICML) in 2019 [200].

In Chapter 6, we make conclusions and discuss future directions.

Chapter 2

Gradient Diversity in Scalable Distributed Learning

It has been experimentally observed that distributed implementations of mini-batch stochastic gradient descent (SGD) algorithms exhibit speedup saturation and decaying generalization ability beyond a particular batch-size. In this work, we present an analysis hinting that high similarity between concurrently processed gradients may be a cause of this performance degradation. We introduce the notion of *gradient diversity* that measures the dissimilarity between concurrent gradient updates, and show its key role in the convergence and generalization performance of mini-batch SGD. We also establish that heuristics similar to DropConnect, Langevin dynamics, and quantization, are provably diversity-inducing mechanisms, and provide experimental evidence indicating that these mechanisms can indeed enable the use of larger batches without sacrificing accuracy and lead to faster training in distributed learning. For example, in one of our experiments, for a convolutional neural network to reach 95% training accuracy on MNIST, using the diversity-inducing mechanism can reduce the training time by 30% in the distributed setting.

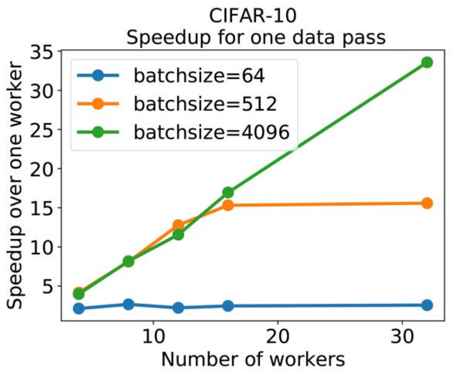
2.1 Introduction

In recent years, deploying algorithms on distributed computing units has become the *de facto* architectural choice for large-scale machine learning. Distributed optimization has gained significant traction with a large body of recent work establishing near-optimal speedup gains on both convex and nonconvex objectives [148, 74, 52, 201, 126, 91, 62, 37], and several state-of-the-art publicly available (distributed) machine learning frameworks, such as Tensorflow [1], MXNet [40], and Caffe2 [44], offer distributed implementations of popular learning algorithms.

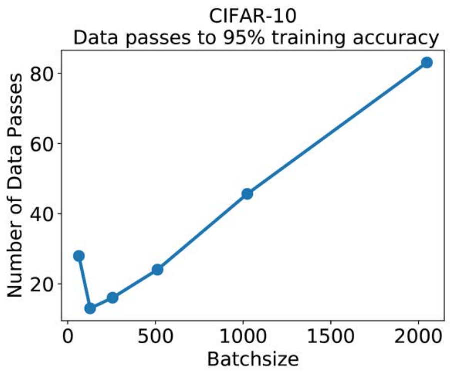
Mini-batch stochastic gradient descent (SGD) is the algorithmic cornerstone for several of these distributed frameworks. During a distributed iteration of mini-batch SGD, a master node stores a global model, and P worker nodes compute gradients for B data points, sampled from a total of n training data (*i.e.,* B/P samples per worker per iteration), with respect to the same global model; the parameter B is commonly referred to as the batch-size. The master, after receiving these B gradients, applies them to the model and sends the updated model back to the workers; this is the equivalent of one round of communication.

Unfortunately, near-optimal scaling for distributed variants of mini-batch SGD is only possible for up to tens of compute nodes. Several studies [52, 153] indicate that there is a significant gap between ideal and realizable speedups when scaling out to hundreds of compute nodes. This commonly observed phenomenon is referred to as *speedup saturation*. A key cause of speedup saturation is the communication overhead of mini-batch SGD.

Ultimately, the batch-size B controls a crucial performance trade-off between communication cost and convergence speed, as observed and analyzed in several studies [178, 184, 80]. When using large batch-sizes, we observe large speedup gains per pass (*i.e.,* per n gradient computations), as shown in Figure 2.1a, due to fewer communication rounds. However, as shown in Figure 2.1b, to achieve a desired level of accuracy for larger batches, we may need a larger number of passes over the dataset, resulting in *overall* slower computation that leads to speedup saturation. Furthermore, recent work shows that large batch sizes lead to models that generalize worse [100], and efforts have been made to improve the generalization ability [86].



(a)



(b)

Figure 2.1: (a) Speedup gains for a single data pass and various batch-sizes, for a cuda-convnet model on CIFAR-10. (b) Number of data passes to reach 95% accuracy for a cuda-convnet model on CIFAR-10, vs batch-size. Step-sizes are tuned to maximize convergence speed. Experiments are conducted on Amazon EC2 instances g2.2xlarge.

The key question that motivates our work is: *How does the batch-size control the scalability and generalization performance of mini-batch SGD*?

Our Contributions

We employ the notion of *gradient diversity* that measures the dissimilarity between concurrent gradient updates. We show that the convergence of mini-batch SGD, on both convex and nonconvex loss functions, including the Polyak-Łojasiewicz functions [151, 128], is identical—up to constant factors—to that of serial SGD (*e.g.,* $B = 1$), if the batch-size is proportional to a bound implied by gradient diversity. We also establish the worst case optimality and tightness of the bound in strongly convex functions.

Although it has been empirically observed that more diversity in the data leads to more parallelism [44], and there has been significant work on the theory of mini-batch algorithms, our results have two major novelties: 1) our batch-size bound is data-dependent, tight, and essentially identical across convex and nonconvex functions, and in some cases leads to guaranteed uniformly larger batch-sizes compared to prior work, and 2) the bound has an operational meaning, and inspired by our theory, we establish that algorithmic heuristics similar to DropConnect [183], Langevin dynamics [188], and quantization [5] are diversity-inducing mechanisms. In our experiments, we find that the proposed mechanisms can indeed enable the use of larger batch-size in distributed learning, and thus reduce training time.

Following our convergence analysis, we study the effect of batch-size on the generalization behavior of mini-batch SGD using the notion of algorithmic stability [25]. Through a similar measure of gradient diversity, we show that as long as the batch-size is below a certain threshold, then mini-batch SGD is as stable as one sample SGD that is analyzed by Hardt, Recht, and Singer [83].

2.2 Related Work

Mini-batch SGD Dekel et al. [54] analyze mini-batch SGD on non-strongly convex functions and propose $B = \mathcal{O}(\sqrt{T})$ as an optimal choice for batch-size. In contrast, our work provides a data-dependent principle for the choice of batch-size, and it holds without the requirement of convexity. Even in the regime where the result in Dekel et al. [54] is valid, depending on the problem, our result may still provide better bounds on the batch-size than $\mathcal{O}(\sqrt{T})$ (*e.g.,* in the sparse conflict setting shown in Section 2.4). Friedlander and Schmidt [69] propose an adaptive batch-size scheme and show that this scheme provides weak linear convergence rate for strongly convex functions. De et al. [51] propose an optimization algorithm for choosing the batch-size, and weighted sampling techniques have also been developed [210, 142, 202].

Diversity and data-dependent bounds In empirical studies, it has been observed that more diversity in the data allows more parallelism [44]. As for the theoretical analysis, data-dependent thresholds for batch-size have been developed for some specific problems such as least squares [92] and SVM [178]. In particular, for least square problems, Jain et al. [92] propose a bound on batch-size similar to our measure of gradient diversity; however, as mentioned in Section 2.1, our result holds for a wider range of problems including nonconvex setups, and can be used to motivate heuristics that result in speedup gains in distributed systems.

Other mini-batching and distributed optimization algorithms Beyond mini-batch SGD, several other mini-batching algorithms have been proposed; we survey a non-exhaustive list. Mini-batch proximal algorithms are studied by Li et al. [125], Wang, Wang, and Srebro [184], and Wang and Srebro [186], and these algorithms require solving a regularized optimization algorithm on a sampled batch as a subroutine. Other algorithms include accelerated methods [46], mini-batch SDCA [164, 179], and the combination of mini-batching and variance reduction such as Acc-Prox-SVRG [147] and mS2GD [106]. Here, we emphasize that although different mini-batching algorithms can be designed for particular problems and may work better in particular regimes, especially in the convex setting, these algorithms are usually more difficult to implement in distributed learning frameworks like Tensorflow or MXNet, and can introduce additional communication costs. A few other algorithms have been recently proposed to reduce the communication cost by inducing sparsity in the gradients, for instance, QSGD [5] and TernGrad [189]. Other algorithms have been proposed under different distributed computation frameworks, examples include one-shot model averaging [212, 209], and the local storage framework [117, 165, 91].

Generalization and stability An important performance measure of a learning algorithm is its generalization ability. In their foundational work, Bousquet and Elisseeff [25] prove the equivalence between algorithmic stability and generalization. This approach is then used to establish generalization bounds for SGD by Hardt, Recht, and Singer [83]. Another approach to prove generalization bounds is to use the operator view of averaged SGD [53]. This method is extended by Jain et al. [92] to the random least-squares regression problems. Variance reduction methods are also used to develop algorithms with good generalization performance [70, 50]. In this chapter, we extend the stability approach to the mini-batch setting, and show that the generalization ability is governed by a quantity that is also function of gradient diversity.

2.3 Problem Setup

We consider the following general supervised learning setup. Suppose that $\mathcal{D}$ is an unknown distribution over a sample space $\mathcal{Z}$, and we have access to a sample $\mathcal{S} = \{\mathbf{z}_1, \ldots, \mathbf{z}_n\}$ of n data points, that are drawn i.i.d. from $\mathcal{D}$. Our goal is to find a model $\mathbf{w}$ from a model space $\mathcal{W} \subseteq \mathbb{R}^d$ with small *population risk* with respect to a loss function f, *i.e.,* we want to minimize $R(\mathbf{w}) = \mathbb{E}_{\mathbf{z}\sim\mathcal{D}}[f(\mathbf{w}; \mathbf{z})]$. Since we do not have access to the population risk, we instead train a model whose aim is to minimize the *empirical risk*

$$R_{\mathcal{S}}(\mathbf{w}) := \frac{1}{n}\sum_{i=1}^{n} f(\mathbf{w};\mathbf{z}_i). \tag{2.1}$$

For any training algorithm that operates on the empirical risk, there are two important aspects to analyze: the convergence speed to a good model with small empirical risk, and the generalization gap $|R_{\mathcal{S}}(\mathbf{w}) - R(\mathbf{w})|$ that quantifies the performance discrepancy of the model between the empirical and population risks. For simplicity, we use the notation $f_i(\mathbf{w}) := f(\mathbf{w};\mathbf{z}_i)$, $F(\mathbf{w}) := R_{\mathcal{S}}(\mathbf{w})$, and define $\mathbf{w}^* \in \arg\min_{\mathbf{w}\in\mathcal{W}} F(\mathbf{w})$. In this work, we focus on families of differentiable losses that satisfy a subset of the following for all parameters $\mathbf{w}, \mathbf{w}' \in \mathcal{W}$:

Definition 2.1 (β-smooth)**.**

$$F(\mathbf{w}) \le F(\mathbf{w}') + \langle \nabla F(\mathbf{w}'), \mathbf{w} - \mathbf{w}'\rangle + \frac{\beta}{2}\|\mathbf{w} - \mathbf{w}'\|_2^2.$$

Definition 2.2 (λ-strongly convex)**.**

$$F(\mathbf{w}) \ge F(\mathbf{w}') + \langle \nabla F(\mathbf{w}'), \mathbf{w} - \mathbf{w}'\rangle + \frac{\lambda}{2}\|\mathbf{w} - \mathbf{w}'\|_2^2.$$

Definition 2.3 (μ-Polyak-Łojasiewicz (PL) [151, 128])**.**

$$\frac{1}{2}\|\nabla F(\mathbf{w})\|_2^2 \ge \mu(F(\mathbf{w}) - F(\mathbf{w}^*)).$$

Mini-batch SGD At each iteration, mini-batch SGD computes B gradients on randomly sampled data at the most current global model. At the $(k+1)$-th distributed iteration, the model is given by

$$\mathbf{w}_{(k+1)B} = \mathbf{w}_{kB} - \gamma \sum_{\ell=kB}^{(k+1)B-1} \nabla f_{s_\ell}(\mathbf{w}_{kB}), \tag{2.2}$$

where each index s_i is drawn uniformly at random from $[n]$, with replacement. Here, we use $\mathbf{w}$ with subscript kB to denote the model we obtain after k distributed iterations, *i.e.,* a total of kB gradient updates. Note that we recover serial SGD when $B = 1$. Our results also apply to varying step-size, but for simplicity we only state our bounds with constant step-size. In related work, there is a normalization of $1/B$ included in the gradient step, here, *without loss of generality* we subsume that in the step-size γ.

We note that some of our analyses require $\mathcal{W}$ to be a bounded convex subset of $\mathbb{R}^d$, where the projected version of SGD can be used, by making Euclidean projections back to $\mathcal{W}$, *i.e.,*

$$\mathbf{w}_{(k+1)B} = \Pi_{\mathcal{W}}(\mathbf{w}_{kB} - \gamma \sum_{\ell=kB}^{(k+1)B-1} \nabla f_{s_\ell}(\mathbf{w}_{kB})).$$

For simplicity, in our main text, we refer to both with/without projection algorithms as "mini-batch SGD", but in Section 2.8 we make the distinction clear when needed.

2.4 Gradient Diversity and Convergence

In this section, we introduce our definition of gradient diversity, and state our convergence results.

Gradient Diversity

Gradient Diversity quantifies the degree to which individual gradients of the loss functions are different from each other. We note that a similar notion was introduced by Jain et al. [92] for least squares problems.

Definition 2.4 (gradient diversity)**.** *We refer to the following ratio as gradient diversity:*

$$\Delta_D(\mathbf{w}) := \frac{\sum_{i=1}^n \|\nabla f_i(\mathbf{w})\|_2^2}{\|\sum_{i=1}^n \nabla f_i(\mathbf{w})\|_2^2} = \frac{\sum_{i=1}^n \|\nabla f_i(\mathbf{w})\|_2^2}{\sum_{i=1}^n \|\nabla f_i(\mathbf{w})\|_2^2 + \sum_{i\neq j}\langle \nabla f_i(\mathbf{w}), \nabla f_j(\mathbf{w})\rangle}.$$

Clearly, $\Delta_D(\mathbf{w})$ is large when the inner products between the gradients taken with respect to different data points are small, and so measures diverse these gradients are. We further define a batch-size bound $B_D(\mathbf{w})$ for each data set $\mathcal{S}$ and each $\mathbf{w} \in \mathcal{W}$:

Definition 2.5 (batch-size bound)**.**

$$B_D(\mathbf{w}) := n\Delta_D(\mathbf{w}).$$

As we see in later parts, the batch-size bound $B_D(\mathbf{w})$ implied by gradient diversity plays a fundamental role in the batch-size selection for mini-batch SGD.

Examples of gradient diversity We provide two examples in which we can compute a uniform lower bound for all $B_D(\mathbf{w})$, $\mathbf{w} \in \mathcal{W}$. Notice that these bounds solely depend on the data set $\mathcal{S}$, and are thus *data dependent.*
Example 1 (generalized linear function) Suppose that any data point $\mathbf{z}$ consists of feature vector $\mathbf{x} \in \mathbb{R}^d$ and some label $y \in \mathbb{R}$, and for sample $\mathcal{S} = \{\mathbf{z}_1, \dots, \mathbf{z}_n\}$, the loss function $f(\mathbf{w}; \mathbf{z}_i)$ can be written as a generalized linear function $f(\mathbf{w}; \mathbf{z}_i) = \ell_i(\mathbf{x}_i^{\mathrm{T}}\mathbf{w})$, where $\ell_i : \mathbb{R} \to \mathbb{R}$ is a differentiable one-dimensional function, and we do not require the convexity of $\ell_i(\cdot)$. Let $\mathbf{X} = [\mathbf{x}_1\ \mathbf{x}_2\ \cdots\ \mathbf{x}_n]^{\mathrm{T}} \in \mathbb{R}^{n\times d}$ be the feature matrix. We have the following results for $B_D(\mathbf{w})$ for generalized linear functions.

Remark 2.1. *For generalized linear functions,* $\forall\ \mathbf{w} \in \mathcal{W}$,

$$B_D(\mathbf{w}) \geq n \min_{i=1,\dots,n} \|\mathbf{x}_i\|_2^2 / \sigma_{\max}^2(\mathbf{X}).$$

We note that it has been shown by Takác et al. [178] that the spectral norm of the data matrix is important for the batch-size choice for SVM problems, and our results have similar implication. In addition, suppose that $n \geq d$, and $\mathbf{x}_i$ has i.i.d. σ-sub-Gaussian entries with

zero mean. Then there exist universal constants $c_1, c_2, c_3 > 0$ such that with probability $1 - c_2 n e^{-c_3 d}$, $B_D(\mathbf{w}) \geq c_1 d$, $\forall\ \mathbf{w} \in \mathcal{W}$. Therefore, as long as we are in the relatively high dimensional regime $d = \Omega(\log(n))$, we have $B_D(\mathbf{w}) \geq \Omega(d)$, $\forall\ \mathbf{w} \in \mathcal{W}$ with high probability.
Example 2 (sparse conflicts) In some applications [97], the gradient of an individual loss function $\nabla f_i(\mathbf{w})$ depends only on a small subset of all the coordinates of $\mathbf{w}$ (called the support), and the supports of the gradients have *sparse conflict*. More specifically, define a graph $G = (V, E)$ with the vertices V representing the n data points, and $(i, j) \in E$ when the supports of $\nabla f_i(\mathbf{w})$ and $\nabla f_j(\mathbf{w})$ have non-empty overlap. Let ρ be the maximum degree of all the vertices in G.

Remark 2.2. *For sparse conflicts, we have $B_D(\mathbf{w}) \geq n/(\rho + 1)$ for all $\mathbf{w} \in \mathcal{W}$. This bound can be large when G is sparse,* i.e., *when ρ is small.*

Convergence Rates

Our convergence results are consequences of the following lemma, which does not require convexity of the losses, and captures the effect of mini-batching on an iterate-by-iterate basis. Here, we define $M^2(\mathbf{w}) := \frac{1}{n}\sum_{i=1}^{n} \|\nabla f_i(\mathbf{w})\|_2^2$ for any $\mathbf{w} \in \mathcal{W}$.

Lemma 2.1. *Let $\mathbf{w}_{kB}$ be a fixed model, and let $\mathbf{w}_{(k+1)B}$ denote the model after a mini-batch iteration with batch-size $B = \delta B_D(\mathbf{w}_{kB}) + 1$. Then*

$$\begin{aligned}\mathbb{E}[\|\mathbf{w}_{(k+1)B} - \mathbf{w}^*\|_2^2 \mid \mathbf{w}_{kB}] \leq & \|\mathbf{w}_{kB} - \mathbf{w}^*\|_2^2 - 2B\gamma \left\langle \nabla F(\mathbf{w}_{kB}), \mathbf{w}_{kB} - \mathbf{w}^* \right\rangle \\ & + (1+\delta) B \gamma^2 M^2(\mathbf{w}_{kB}),\end{aligned}$$

where equality holds when there are no projections.

As one can see, for a single iteration, in expectation, the model trained by serial SGD ($B = 1$), closes the distance to the optimal by exactly

$$2\gamma \left\langle \nabla F(\mathbf{w}_{kB}), \mathbf{w}_{kB} - \mathbf{w}^* \right\rangle - \gamma^2 M^2(\mathbf{w}_{kB}).$$

Our bound says that using the *same* step-size[1] as SGD (*without normalizing* with a factor of B), mini-batch will close that distance to the optimal (or any critical point $\mathbf{w}^*$) by approximately B times more, if $B = \mathcal{O}(B_D(\mathbf{w}_{kB}))$. This matches the best that we could have hoped for: mini-batch SGD with batch-size B should be B times faster per iteration than a single iteration of serial SGD.

We now provide convergence results using gradient diversity. For a mini-batch SGD algorithm, define the set $\mathcal{W}_T \subset \mathcal{W}$ as the collection of all possible model parameters that the algorithm can reach during T/B parallel iterations, *i.e.,*

$$\mathcal{W}_T := \{\mathbf{w} \in \mathcal{W}\ :\ \mathbf{w} = \mathbf{w}_{kB} \text{ for some instance of mini-batch SGD, } k = 0, 1, \ldots, T/B\}.$$

[1]In fact, our choice of step-size is consistent with many state-of-the-art distributed learning frameworks [80], and we would like to point out that this chapter provides theoretical explanation of this choice of step-size.

Our main message can be summarized as follows:

Theorem 2.1 (informal convergence result)**.** *Let $B \leq \delta B_D(\mathbf{w})+1$, $\forall\ \mathbf{w} \in \mathcal{W}_T$. If serial SGD achieves an ϵ-suboptimal*[2] *solution after T gradient updates, then using the same step-size as serial SGD, mini-batch SGD with batch-size B can achieve a $(1+\frac{\delta}{2})\epsilon$-suboptimal solution after the same number of gradient updates (*i.e., *T/B iterations).*

Therefore, our result implies that, as long as the batch-size does not exceed the fundamental bound implied by gradient diversity, using the *same* step-size as the serial algorithm, mini-batch SGD does not suffer from convergence speed saturation.

We provide the precise statements of the results as follows. Define $F^* = \min_{\mathbf{w}\in\mathcal{W}} F(\mathbf{w})$, $D_0 = \|\mathbf{w}_0 - \mathbf{w}^*\|_2^2$. In all the following results, we assume that $B \leq \delta B_D(\mathbf{w}) + 1$, $\forall\ \mathbf{w} \in \mathcal{W}_T$, and $M^2(\mathbf{w}) \leq M^2$, $\forall\ \mathbf{w} \in \mathcal{W}_T$. The step-sizes in the following results are known to be the order-optimal choices for serial SGD with constant step-size [24, 75, 98]. We start with more general function classes, *i.e.,* nonconvex smooth functions and PL functions.

Theorem 2.2 (smooth functions)**.** *Suppose that $F(\mathbf{w})$ is β-smooth, $\mathcal{W} = \mathbb{R}^d$, and use step-size $\gamma = \frac{\epsilon}{\beta M^2}$. Then, after $T \geq \frac{2}{\epsilon^2} M^2\beta(F(\mathbf{w}_0) - F^*)$ gradient updates,*

$$\min_{k=0,\ldots,T/B-1} \mathbb{E}[\|\nabla F(\mathbf{w}_{kB})\|_2^2] \leq (1+\frac{\delta}{2})\epsilon.$$

Theorem 2.3 (PL functions)**.** *Suppose that $F(\mathbf{w})$ is β-smooth, μ-PL, $\mathcal{W} = \mathbb{R}^d$, and use step-size $\gamma = \frac{2\epsilon\mu}{M^2\beta}$, and batch-size $B \leq \frac{1}{2\gamma\mu}$. Then, after $T \geq \frac{M^2\beta}{4\mu^2\epsilon}\log(\frac{2(F(\mathbf{w}_0)-F^*)}{\epsilon})$ gradient updates, we have $\mathbb{E}[F(\mathbf{w}_T) - F^*] \leq (1+\frac{\delta}{2})\epsilon$.*

For convex loss functions, we emphasize that, there have been a lot of studies that establish similar rates, without explicitly using our notion of gradient diversity [69, 92, 178]. We present the results for completeness, and also note that via gradient diversity, we provide a general form of convergence rates that is essentially identical across convex and nonconvex objectives.

Theorem 2.4 (convex functions)**.** *Suppose that $F(\mathbf{w})$ is convex, and use step-size $\gamma = \frac{\epsilon}{M^2}$. Then, after $T \geq \frac{M^2 D_0}{\epsilon^2}$ gradient updates, we have $\mathbb{E}[F(\frac{B}{T}\sum_{k=0}^{\frac{T}{B}-1}\mathbf{w}_{kB}) - F^*] \leq (1+\frac{\delta}{2})\epsilon$.*

Theorem 2.5 (strongly convex functions)**.** *Suppose that $F(\mathbf{w})$ is λ-strongly convex, and use step-size $\gamma = \frac{\epsilon\lambda}{M^2}$ and batch-size $B \leq \frac{1}{2\lambda\gamma}$. Then, after $T \geq \frac{M^2}{2\lambda^2\epsilon}\log(\frac{2D_0}{\epsilon})$ gradient updates, we have $\mathbb{E}[\|\mathbf{w}_T - \mathbf{w}^*\|_2^2] \leq (1+\frac{\delta}{2})\epsilon$.*

[2]Suboptimality is defined differently for different classes of functions.

Worst-case Optimality

Here, we establish that the above bound on the batch-size is worst-case optimal. The following theorem demonstrates this for a convex problem with varying agnostic batch-sizes[3] B_k. Essentially, if we violate the batch bound prescribed above by a factor of δ, then the quality of our model will be penalized by a factor of δ, in terms of accuracy.

Theorem 2.6. *Consider a mini-batch SGD algorithm with K iterations and varying batch-sizes $B_1, B_2, \dots, B_K$, and let $N_k = \sum_{i=1}^{k} B_i$. Then, there exists a λ-strongly convex function $F(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{w})$ with bounded parameter space $\mathcal{W}$, such that, if $B_k \le \frac{1}{2\lambda\gamma}$ and $B_k \ge \delta\mathbb{E}[B_D(\mathbf{w}_{N_{k-1}})] + 1 \; \forall \; k = 1, \dots, K$ (where the expectation is taken over the randomness of the mini-batch SGD algorithm), and the total number of gradient updates $T = N_K \ge \frac{c}{\lambda\gamma}$ for some universal constant $c > 0$, we have: $\mathbb{E}[\|\mathbf{w}_T - \mathbf{w}^*\|_2^2] \ge c'(1+\delta)\frac{\gamma M^2}{\lambda}$, where $c' > 0$ is another universal constant. More concretely, when running mini-batch SGD with step-size $\gamma = \frac{\epsilon\lambda}{M^2}$ and at least $\mathcal{O}(\frac{M^2}{\lambda^2\epsilon})$ gradient updates, we have $\mathbb{E}[\|\mathbf{w}_T - \mathbf{w}^*\|_2^2] \ge c'(1+\delta)\epsilon$.*

Although the above bound is only for strongly convex functions, it reveals that there exist regimes beyond which scaling the batch-size beyond our fundamental bound can lead to only worse performance in terms of the accuracy for a given iteration, or the number of iterations needed for a specific accuracy.

Diversity-inducing Mechanisms

In recent years, several algorithmic heuristics, such as DropConnect [183], stochastic gradient Langevin dynamics (SGLD) [188], and quantization [5], have been shown to be useful for improving large scale optimization in various aspects. For example, they may help improve generalization or escape saddle points [71]. In this section, we demonstrate a *different aspect* of these heuristics. We show that gradient diversity can increase when applying these techniques independently to the data points in a batch, rendering mini-batch SGD more amenable to distributed speedup gains.

We note that these mechanisms have two opposing effects: on one hand, as we show in the sequel they allow the use of larger batch-sizes, and thus can reduce communication cost by reducing the number of iterations; on the other hand, these methods usually introduce additional variance to the stochastic gradients, and may require more iteration to achieve a particular accuracy. Consequently, there is a communication-computation trade-off inherent to these mechanisms. By carefully exploiting this trade-off, our goal would be to see a gain in the *overall* run time. In Section 2.6, we provide experimental evidence to show that this run time gain can indeed be observed in real distributed systems.

We use abbreviation DIM for any diversity-inducing mechanism. When data point i is sampled, instead of making gradient update $\nabla f_i(\mathbf{w})$, the algorithm updates with a random

[3]Here, by saying that the batch-sizes are *agnostic*, we emphasize the fact that the batch-sizes are constants that are picked up without looking at the progress of the algorithm.

surrogate vector $\mathbf{g}_i^{\mathsf{DIM}}(\mathbf{w})$ by introducing some additional randomness, which is acquired i.i.d. across data points and iterations.

We can thus define the corresponding gradient diversity and batch-size bounds

$$\Delta_D^{\mathsf{DIM}}(\mathbf{w}) := \frac{\sum_{i=1}^n \mathbb{E}\|\mathbf{g}_i^{\mathsf{DIM}}(\mathbf{w})\|_2^2}{\mathbb{E}\|\sum_{i=1}^n \mathbf{g}_i^{\mathsf{DIM}}(\mathbf{w})\|_2^2},$$
$$B_D^{\mathsf{DIM}}(\mathbf{w}) := n\Delta_D^{\mathsf{DIM}}(\mathbf{w}),$$

where the expectation is taken over the randomness of the mechanism. In the following parts, we first demonstrate various diversity-inducing mechanisms, and then compare $B_D^{\mathsf{DIM}}(\mathbf{w})$ with $B_D(\mathbf{w})$.

DropConnect We interpret DropConnect as updating a randomly chosen subset of all the coordinates of the model parameter vector[4]. Let $\mathbf{D}_1, \dots, \mathbf{D}_n$ be i.i.d. diagonal matrices with diagonal entries being i.i.d. Bernoulli random variables, and each diagonal entry is 0 with drop probability $p \in (0, 1)$. When data point $\mathbf{z}_i$ is chosen, we make DropConnect update $\mathbf{g}_i^{\mathsf{drop}}(\mathbf{w}) = \mathbf{D}_i \nabla f_i(\mathbf{w})$.

Stochastic gradient Langevin dynamics (SGLD) SGLD takes the gradient updates: $\mathbf{g}_i^{\mathsf{sgld}}(\mathbf{w}) = \nabla f_i(\mathbf{w}) + \xi_i$ where ξ_i, $i = 1, \dots, n$, are independent isotropic Gaussian noise $\mathcal{N}(0, \sigma^2 \mathbf{I})$.

Quantized gradients Define $Q(\mathbf{v})$ as the quantized version of a vector $\mathbf{v}$. More precisely, $[Q(\mathbf{v})]_\ell = \|\mathbf{v}\|_2 \operatorname{sgn}(v_\ell)\eta_\ell(\mathbf{v})$, where $\eta_\ell(\mathbf{v})$'s are independent Bernoulli random variables with $\mathbb{P}\{\eta_\ell = 1\} = |v_\ell|/\|\mathbf{v}\|_2$. We let $\mathbf{g}_i^{quant}(\mathbf{w}) = Q(\nabla f_i(\mathbf{w}))$.

We can show that these mechanisms increases gradient diversity, as long as $B_D(\mathbf{w})$ is not already large. Formally, we have the following result.

Theorem 2.7. *For any $\mathbf{w} \in \mathcal{W}$ such that $B_D(\mathbf{w}) \leq n$, we have $B_D^{DIM}(\mathbf{w}) \geq B_D(\mathbf{w})$, where $DIM \in \{\mathsf{drop}, \mathsf{sgld}, \mathsf{quant}\}$.*

2.5 Differential Gradient Diversity and Stability

In this section, we turn to another important property of mini-batch SGD algorithm, *i.e.,* the generalization ability.

[4]We note that our notion of DropConnect is slightly different from the original paper [183], but is of similar spirit.

Stability and Generalization

Recall that in supervised learning, our goal is to learn a parametric model with small population risk $R(\mathbf{w}) := \mathbb{E}_{\mathbf{z}\sim\mathcal{D}}[f(\mathbf{w};\mathbf{z})]$. In order to do so, we use empirical risk minimization, and hope to obtain a model that has both small empirical risk and small population risk to avoid overfitting. Formally, let A be a possibly randomized algorithm which maps the training data to the parameter space as $\mathbf{w} = A(\mathcal{S})$. In this chapter, we use the model parameter obtained in the final iteration as the output of the mini-batch SGD algorithm, *i.e.,* $A(\mathcal{S}) = \mathbf{w}_T$. We define the *expected generalization error* of the algorithm as $\epsilon_{\text{gen}}(A) := \mathbb{E}_{\mathcal{S},A}[R_{\mathcal{S}}(A(\mathcal{S})) - R(A(\mathcal{S}))]$.

Bousquet and Elisseeff [25] show the equivalence between the generalization error and algorithmic stability. The basic idea of proving generalization bounds using stability is to bound the distance between the model parameters obtained by running an algorithm on two datasets that only differ on one data point. This framework is used by Hardt, Recht, and Singer [83] to show stability guarantees for serial SGD algorithm for Lipschitz and smooth loss functions. Roughly speaking, they show upper bounds $\overline{\gamma}$ on the step-size below which serial SGD is stable. This yields, as a corollary, that mini-batch SGD is stable provided the step-size is upper bounded by $\overline{\gamma}/B$. We remind the reader that since we absorb the $1/B$ factor in the step-size, the only step-size for which the analysis by Hardt, Recht, and Singer [83] would imply stability for SGD is $1/B$ less than what we suggest in the convergence results. In the following parts of this section, we show that the mini-batch algorithm with a similar step-size to SGD is indeed stable, provided the *differential gradient diversity* is large enough.

Differential Gradient Diversity

The stability of mini-batch SGD is governed by the *differential gradient diversity*, defined as follows.

Definition 2.6 (differential gradient diversity and batch-size bound)**.** *For any* $\mathbf{w}, \mathbf{w}' \in \mathcal{W}$, $\mathbf{w} \neq \mathbf{w}'$, *the differential gradient diversity and batch-size bound is given by*

$$\begin{aligned}\overline{\Delta}_D(\mathbf{w},\mathbf{w}') &:= \frac{\sum_{i=1}^n \|\nabla f_i(\mathbf{w}) - \nabla f_i(\mathbf{w}')\|_2^2}{\|\sum_{i=1}^n \nabla f_i(\mathbf{w}) - \nabla f_i(\mathbf{w}')\|_2^2},\\ \overline{B}_D(\mathbf{w},\mathbf{w}') &:= n\overline{\Delta}_D(\mathbf{w},\mathbf{w}').\end{aligned}$$

Although it is a distinct measure, differential gradient diversity shares similar properties with gradient diversity. For example, the lower bounds for $B_D(\mathbf{w})$ in examples 1 and 2 in Section 2.4 also hold for $\overline{B}_D(\mathbf{w},\mathbf{w}')$, and two mechanisms, DropConnect and SGLD also induce differential gradient diversity, as we note Section 2.8.

Stability of Mini-batch SGD

We analyze the stability (generalization) of mini-batch SGD via differential gradient diversity. We assume that, for each $\mathbf{z} \in \mathcal{Z}$, the loss function $f(\mathbf{w}; \mathbf{z})$ is convex, L-Lipschitz and β-smooth in $\mathcal{W}$. We choose not to discuss the generalization error for nonconvex functions, since this may require a significantly small step-size [83].

Our result is stated informally in Theorem 2.8, and holds for both convex and strongly convex functions. Here, $\overline{\gamma}$ is the step-size upper bound required to guarantee stability of serial SGD, and differently from the convergence results, we treat $\overline{B}_D(\mathbf{w}, \mathbf{w}')$ as a random variable defined by the sample $\mathcal{S}$.

Theorem 2.8 (informal stability result)**.** *Suppose that, with high probability, the batch-size $B \lesssim \overline{B}_D(\mathbf{w}, \mathbf{w}')$ for all $\mathbf{w}, \mathbf{w}' \in \mathcal{W}$, $\mathbf{w} \neq \mathbf{w}'$. Then, after the same number of gradient updates, the generalization errors of mini-batch SGD and serial SGD satisfy*

$$\epsilon_{gen}(\textit{minibatch SGD}) \lesssim \epsilon_{gen}(\textit{serial SGD}),$$

and such a guarantee holds for any step-size $\gamma \lesssim \overline{\gamma}$.

Therefore, our main message is that, if with high probability, batch-size B is smaller than $\overline{B}_D(\mathbf{w}, \mathbf{w}')$ for all $\mathbf{w}, \mathbf{w}'$, mini-batch SGD and serial SGD can be both stable in roughly the *same range* of step-sizes, and the generalization error of mini-batch SGD and serial SGD are roughly the *same*. We now provide the precise statements. In the following, we denote by $\mathbb{1}$ the indicator function.

Theorem 2.9 (generalization error of convex functions)**.** *Suppose that for any $\mathbf{z} \in \mathcal{Z}$, $f(\mathbf{w}; \mathbf{z})$ is convex, L-Lipschitz and β-smooth in $\mathcal{W}$. For a fixed step size $\gamma > 0$, let*

$$\eta = \mathbb{P}\{\inf_{\mathbf{w}\neq\mathbf{w}'} \overline{B}_D(\mathbf{w}, \mathbf{w}') < \frac{B-1}{\frac{2}{\gamma\beta} - 1 - \frac{1}{n-1}\mathbb{1}_{B>1}}\},$$

where the probability is over the randomness of $\mathcal{S}$. Then the generalization error of mini-batch SGD satisfies $\epsilon_{gen} \leq 2\gamma L^2 \frac{T}{n}(1-\eta) + 2\gamma L^2 T\eta$.

It is shown by Hardt, Recht, and Singer [83] that $\epsilon_{\text{gen}}(\text{serial SGD}) \leq 2\gamma L^2 \frac{T}{n}$, for convex functions, when $\gamma \leq \frac{2}{\beta}$. Notice that in our result, when $B = 1$, we get $\eta = 0$, and thus recover the generalization bound for serial SGD. Further, suppose one can find $\overline{B}$ such that $\inf_{\mathbf{w}\neq\mathbf{w}'} \overline{B}_D(\mathbf{w}, \mathbf{w}') \geq \overline{B}$ with high probability. Then by choosing $B \leq 1 + \delta\overline{B}$, and $\gamma \leq \frac{2}{\beta(1+\delta+\frac{1}{n-1})}$, we obtain similar generalization error as the serial algorithm without significant change in the step-size range. For strongly convex functions, we have:

Theorem 2.10 (generalization error of strongly convex functions)**.** *Suppose that for any $\mathbf{z} \in \mathcal{Z}$, $f(\mathbf{w}; \mathbf{z})$ is L-Lipschitz, β-smooth, and λ-strongly convex in $\mathcal{W}$, and $B \leq \frac{1}{2\gamma\lambda}$. For a fixed step size $\gamma > 0$, let*

$$\eta = \mathbb{P}\{\inf_{\mathbf{w}\neq\mathbf{w}'} \overline{B}_D(\mathbf{w}, \mathbf{w}') < \frac{B-1}{\frac{2}{\gamma(\beta+\lambda)} - 1 - \frac{1}{n-1}\mathbb{1}_{B>1}}\},$$

where the probability is over the randomness of $\mathcal{S}$. Then the generalization error of mini-batch SGD satisfies $\epsilon_{gen} \leq \frac{4L^2}{\lambda n}(1-\eta) + 2\gamma L^2 T\eta$.

Again, as shown by Hardt, Recht, and Singer [83], we have $\epsilon_{\text{gen}}(\text{serial SGD}) \leq \frac{4L^2}{\lambda n}$ for strongly convex functions, when $\gamma \leq \frac{2}{\beta+\lambda}$. Thus, our remarks for the convex case above also carry over here. We also mention that while in general, the probability parameter η may appear to weaken the bound, there are practical functions for which η has rate decaying in n. For example, for generalized linear functions, we can show that when the feature vectors have i.i.d. sub-Gaussian entries, choosing $B \lesssim d$ yields $\eta \lesssim ne^{-d}$, which has polynomial decay in n when $d = \Omega(\log(n))$. For details, see Section 2.8.

2.6 Experiments

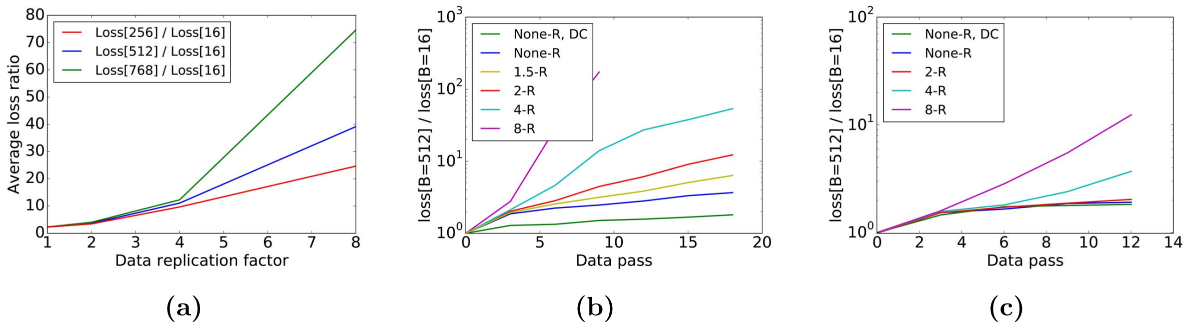


Figure 2.2: Data replication. Here, 2-R, 4-R, etc represent 2-replication, 4-replication, etc, and DC stand for DropConnect. (a) Logistic regression with two classes of CIFAR-10 (b) Cuda convolutional neural network (c) Residual network. For (a), we plot the average loss ratio during all the iterations of the algorithm, and average over 10 experiments; for (b), (c), we plot the loss ratio as a function of the number of passes over the entire dataset, and average over 3 experiments. We observe that with the larger replication factor, the gap of convergence increases.

We conduct experiments to justify our theoretical results. Our neural network experiments are all implemented in Tensorflow and run on Amazon EC2 instances g2.2xlarge.

Convergence We conduct the experiments on a logistic regression model and two deep neural networks (a cuda convolutional neural network [111] and a deep residual network [84]) with cross-entropy loss running on CIFAR-10 dataset. These results are presented in Figure 2.2. We use data replication to implicitly construct datasets with different gradient diversity. By replication with a factor r (or r-replication), we mean picking a random $1/r$ fraction of the data and replicating it r times. Across all configurations of batch-sizes, we tune our (constant) step-size to maximize convergence, *e.g.,* to minimize training time. The sample size does not change by data replication, but gradient diversity conceivably gets

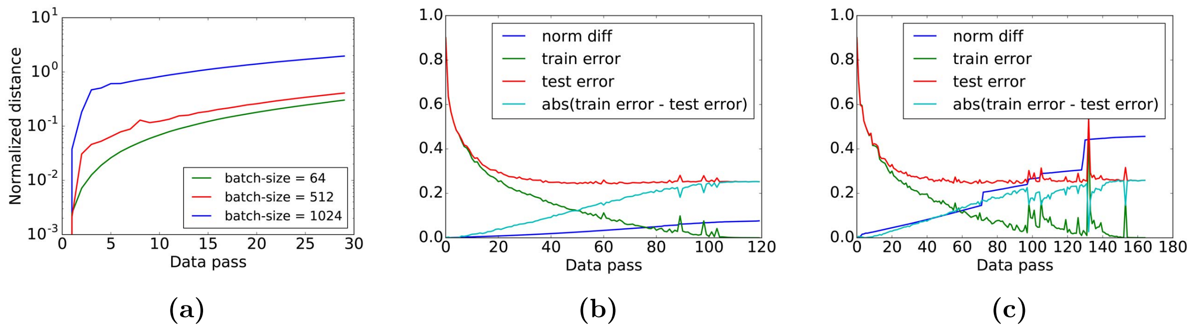


Figure 2.3: Stability. (a) Normalized Euclidean distance vs number of data passes. (b) Generalization behavior of batch-size 512. (c) Generalization behavior of batch-size 1024. Results are averaged over 3 experiments.

smaller while we increase r. We use the ratio of the loss function for large batch-size SGD (*e.g.,* $B = 512$) to the loss for small batch-size SGD (*e.g.,* $B = 16$) to measure the negative effect of large batch sizes on the convergence rate. When this ratio gets larger, the algorithm with the large batch-size is converging slower. We can see from the figures that while we increase r, the large batch-size instances indeed perform worse, and the large batch instance performs the best when we have DropConnect, due to its diversity-inducing effect, as discussed in the previous sections. This experiment thus validates our theoretical findings.

Stability We also conduct experiments to study the effect of large batch-size on the stability of mini-batch SGD. Our experiments essentially use the same technique as in the study for serial SGD by Hardt, Recht, and Singer [83]. Based on the CIFAR-10 dataset, we construct two training datasets which only differ in one data point, and train a cuda convolutional neural network using the same mini-batch SGD algorithm on these two datasets. For different batch-sizes, we test the normalized Euclidean distance $\sqrt{\|\mathbf{w} - \mathbf{w}'\|_2^2/(\|\mathbf{w}\|_2^2 + \|\mathbf{w}'\|_2^2)}$ between the obtained model on the two datasets. As shown in Figure 2.3a, the normalized distance between the two models becomes larger when we increase the batch-size, which implies that we lose stability by having a larger batch-size. We also compare the generalization behavior of mini-batch SGD with $B = 512$ and $B = 1024$, as shown in Figures 2.3b and 2.3c. As we can see, for large batch sizes, the models exhibit higher variance in their generalization behavior, and our observation is in agreement with [100].

Diversity-inducing Mechanisms We finally implement diversity-inducing mechanisms in a distributed setting with 2 workers and test the speedup gains. We use a convolutional neural network on MNIST and implement the DropConnect mechanism with drop probability $p_{\text{drop}} = 0.4, 0.5$. We tune the step-size γ and batch-size B for vanilla mini-batch SGD and the diversity-induced setting, and find the (γ, B) pair that gives the fastest convergence for each setting. Then, we compare the overall run time to reach 90%, 95%, and 99% training accuracy. The results are shown in Table 2.1, where each time measurement is averaged over 5 runs. Comparing wall-clock times, we see DropConnect provides significant improvements. Indeed, the the batch-size gain afforded by DropConnect—the best batch-size for vanilla mini-batch SGD is 256, while with the diversity-inducing mechanism, it becomes 512—is able to dwarf the noise in gradient computation. Reducing communication cost thus has the biggest effect on runtime, more so than introducing additional variance in stochastic gradient computations.

Table 2.1: Speedup Gains via DropConnect

train accuracy (%)		90	95	99
mini-batch	time (sec)	46.97	57.39	361.52
$p_{\text{drop}} = 0.4$	time (sec)	24.88	39.12	313.60
	gain (%)	46.98	31.83	13.25
$p_{\text{drop}} = 0.5$	time (sec)	29.68	43.24	317.79
	gain (%)	36.76	24.66	12.09

2.7 Conclusions

We propose the notion of gradient diversity to measure the dissimilarity between concurrent gradient updates in mini-batch SGD. We show that, for both convex and nonconvex loss functions, the convergence rate of mini-batch SGD is identical—up to constant factors—to that of serial SGD, provided that the batch-size is at most proportional to a bound implied by gradient diversity. We also develop a corresponding lower bound for the convergence rate of strongly convex objectives. Our results show that on problems with high gradient diversity, the distributed implementation of mini-batch SGD is amenable to better speedups. We also establish similar results for generalization using the notion of differential gradient diversity. Some open problems include finding more mechanisms that improve gradient diversity, and in neural network learning, studying how the network structure, such as width, depth, and activation functions, impacts gradient diversity.

2.8 Proofs

Examples of Gradient Diversity

Proof of Remark 2.1: Generalized linear models

Let $\ell'(\cdot)$ be the derivative of $\ell(\cdot)$. Since we have

$$\nabla f_i(\mathbf{w}) = \ell_i'(\mathbf{x}_i^{\mathrm{T}}\mathbf{w})\mathbf{x}_i,$$

by letting $a_i := \ell_i'(\mathbf{x}_i^{\mathrm{T}}\mathbf{w})$ and $\mathbf{a} = [a_1 \ \cdots \ a_n]^{\mathrm{T}}$, we obtain

$$\begin{aligned} B_D(\mathbf{w}) &= \frac{n\sum_{i=1}^n a_i^2 \|\mathbf{x}_i\|_2^2}{\|\sum_{i=1}^n a_i\mathbf{x}_i\|_2^2} = \frac{n\sum_{i=1}^n a_i^2\|\mathbf{x}_i\|_2^2}{\|\mathbf{X}^{\mathrm{T}}\mathbf{a}\|_2^2} \\ &\geq \frac{n\min_{i=1,\dots,n}\|\mathbf{x}_i\|_2^2\sum_{i=1}^n a_i^2}{\sigma_{\max}^2(\mathbf{X})\|\mathbf{a}\|_2^2} \geq \frac{n\min_{i=1,\dots,n}\|\mathbf{x}_i\|_2^2}{\sigma_{\max}^2(\mathbf{X})}, \end{aligned}$$

which completes the proof.

We made a claim after the remark about instantiating it for random design matrices. We provide the proof of that claim below.

Generalized Linear Function with Random Features

We have the following two results.

Proposition 2.1. *Suppose that $n \geq d$, and $\mathbf{x}_i$ has i.i.d. σ-sub-Gaussian entries with zero mean. Then, there exist universal constants $c_1, c_2, c_3 > 0$, such that, with probability at least $1 - c_2ne^{-c_3d}$, we have $B_D(\mathbf{w}) \geq c_1d \ \forall \ \mathbf{w} \in \mathcal{W}$.*

Proposition 2.2. *Suppose that $n \geq d$, and the entries of $\mathbf{x}_i$ are i.i.d. uniformly distributed in $\{-1, 1\}$. Then, there exist universal constants $c_4, c_5, c_6 > 0$, such that, with probability at least $1 - c_5e^{-c_6n}$, we have $B_D(\mathbf{w}) \geq c_4d \ \forall \ \mathbf{w} \in \mathcal{W}$.*

Proof. By the concentration results of the maximum singular value of random matrices, we know that when $n \geq d$, there exist universal constants $C_1, C_2, C_3 > 0$, such that

$$\mathbb{P}\{\sigma_{\max}^2(\mathbf{X}) \leq C_1\sigma^2 n\} \geq 1 - C_2e^{-C_3n}. \tag{2.3}$$

By the concentration results of sub-Gaussian random variables, we know that there exist universal constants $C_4, C_5 > 0$ such that

$$\mathbb{P}\{\|\mathbf{x}_i\|_2^2 \geq C_4\sigma^2 d\} \geq 1 - e^{-C_5d},$$

and then by union bound, we have

$$\mathbb{P}\left\{\min_{i=1,\dots,n}\|\mathbf{x}_i\|_2^2 \geq C_4\sigma^2 d\right\} \geq 1 - ne^{-C_5d}. \tag{2.4}$$

Then, by combining (2.3) and (2.4) and using union bound, we obtain

$$\mathbb{P}\left\{\frac{n\min_{i=1,\dots,n}\|\mathbf{x}_i\|_2^2}{\sigma_{\max}^2(\mathbf{X})} \geq \frac{C_4}{C_1}d\right\} \geq 1 - C_2e^{-C_3n} - ne^{-C_5d},$$

which yields the desired result.

Proposition 2.2 can be proved using the fact that for Rademacher entries, we have $\|\mathbf{x}_i\|_2^2 = d$ with probability one. $\square$

Proof of Remark 2.2: Sparse Conflict

We prove the following result for Example 2 in Section 2.4.

Proposition 2.3. *Let ρ be the maximum degree of all the vertices in G. Then, we have $\forall\ \mathbf{w} \in \mathcal{W}$, $B_D(\mathbf{w}) \geq n/(\rho+1)$.*

Proof. We adopt the convention that when $(i,j) \in E$, we also have $(j,i) \in E$. By definition, we have

$$\begin{aligned} B_D(\mathbf{w}) &= \frac{n\sum_{i=1}^n \|\nabla f_i(\mathbf{w})\|_2^2}{\sum_{i=1}^n \|\nabla f_i(\mathbf{w})\|_2^2 + \sum_{i\neq j}\langle \nabla f_i(\mathbf{w}), \nabla f_j(\mathbf{w})\rangle} \\ &= \frac{n\sum_{i=1}^n \|\nabla f_i(\mathbf{w})\|_2^2}{\sum_{i=1}^n \|\nabla f_i(\mathbf{w})\|_2^2 + \sum_{(i,j)\in E}\langle \nabla f_i(\mathbf{w}), \nabla f_j(\mathbf{w})\rangle} \\ &\geq \frac{n\sum_{i=1}^n \|\nabla f_i(\mathbf{w})\|_2^2}{\sum_{i=1}^n \|\nabla f_i(\mathbf{w})\|_2^2 + \sum_{(i,j)\in E}\frac{1}{2}\|\nabla f_i(\mathbf{w})\|_2^2 + \frac{1}{2}\|\nabla f_j(\mathbf{w})\|_2^2}. \end{aligned}$$

Since ρ is the maximum degree of the vertexes in G, we know that for each $i \in [n]$, the term $\frac{1}{2}\|\nabla f_i(\mathbf{w})\|_2^2$ appears at most 2ρ times in the summation $\sum_{(i,j)\in E}\frac{1}{2}\|\nabla f_i(\mathbf{w})\|_2^2 + \frac{1}{2}\|\nabla f_j(\mathbf{w})\|_2^2$. Therefore, we obtain

$$\sum_{(i,j)\in E}\frac{1}{2}\|\nabla f_i(\mathbf{w})\|_2^2 + \frac{1}{2}\|\nabla f_j(\mathbf{w})\|_2^2 \leq \rho\sum_{i=1}^n \|\nabla f_i(\mathbf{w})\|_2^2,$$

which completes the proof. □

Convergence Rates

In this section, we prove our convergence results for different types of functions. To assist the demonstration of the proofs of convergence rates, for any $\mathbf{w} \in \mathcal{W}$, we define the following two quantities:

$$M^2(\mathbf{w}) := \frac{1}{n}\sum_{i=1}^n \|\nabla f_i(\mathbf{w})\|_2^2 \quad \text{and} \quad G(\mathbf{w}) := \|\nabla F(\mathbf{w})\|_2^2 = \|\frac{1}{n}\sum_{i=1}^n \nabla f_i(\mathbf{w})\|_2^2$$

One can check that the batch-size bound obeys $B_D(\mathbf{w}) = \frac{M^2(\mathbf{w})}{G(\mathbf{w})}$.

Proof of Lemma 2.1

We have

$$\begin{aligned}
\mathbb{E}[\|\mathbf{w}_{(k+1)B} - \mathbf{w}^*\|_2^2 \mid \mathbf{w}_{kB}] =& \mathbb{E}\left[\|\mathbf{w}_{kB} - \mathbf{w}^* - \gamma \sum_{\ell=kB}^{(k+1)B-1} \nabla f_{s_\ell}(\mathbf{w}_{kB})\|_2^2 \mid \mathbf{w}_{kB}\right] \\
=& \|\mathbf{w}_{kB} - \mathbf{w}^*\|_2^2 - 2\gamma \sum_{\ell=kB}^{(k+1)B-1} \mathbb{E}[\langle \mathbf{w}_{kB} - \mathbf{w}^*, \nabla f_{s_\ell}(\mathbf{w}_{kB})\rangle \mid \mathbf{w}_{kB}] \\
&+ \gamma^2 \mathbb{E}\left[\|\sum_{\ell=kB}^{(k+1)B-1} \nabla f_{s_\ell}(\mathbf{w}_{kB})\|_2^2 \mid \mathbf{w}_{kB}\right].
\end{aligned}$$

Since s_ℓ's are sampled i.i.d. uniformly from $[n]$, we know that

$$\begin{aligned}
\mathbb{E}[\|\mathbf{w}_{(k+1)B} - \mathbf{w}^*\|_2^2 \mid \mathbf{w}_{kB}] =& \|\mathbf{w}_{kB} - \mathbf{w}^*\|_2^2 - 2\gamma B\langle \mathbf{w}_{kB} - \mathbf{w}^*, \nabla F(\mathbf{w}_{kB})\rangle \\
&+ \gamma^2(BM^2(\mathbf{w}_{kB}) + B(B-1)G(\mathbf{w}_{kB})) \\
=& \|\mathbf{w}_{kB} - \mathbf{w}^*\|_2^2 - 2\gamma B\langle \mathbf{w}_{kB} - \mathbf{w}^*, \nabla F(\mathbf{w}_{kB})\rangle \\
&+ \gamma^2 B\left(1 + \frac{B-1}{B_D(\mathbf{w}_{kB})}\right) M^2(\mathbf{w}_{kB}) \\
=& \|\mathbf{w}_{kB} - \mathbf{w}^*\|_2^2 - 2\gamma B\langle \mathbf{w}_{kB} - \mathbf{w}^*, \nabla F(\mathbf{w}_{kB})\rangle \\
&+ \gamma^2 B(1+\delta) M^2(\mathbf{w}_{kB}).
\end{aligned} \tag{2.5}$$

We also mention here that this result becomes inequality for the projected mini-batch SGD algorithm, since Euclidean projection onto a convex set is non-expansive. □

Proof of Theorem 2.2

Recall that we have the iteration $\mathbf{w}_{(k+1)B} = \mathbf{w}_{kB} - \gamma \sum_{t=kB}^{(k+1)B-1} \nabla f_{s_t}(\mathbf{w}_{kB})$. Since $F(\mathbf{w})$ has β-Lipschitz gradients, we have

$$F(\mathbf{w}_{(k+1)B}) \le F(\mathbf{w}_{kB}) + \langle \nabla F(\mathbf{w}_{kB}), \mathbf{w}_{(k+1)B} - \mathbf{w}_{kB}\rangle + \frac{\beta}{2}\|\mathbf{w}_{(k+1)B} - \mathbf{w}_{kB}\|_2^2.$$

Then, we obtain

$$\left\langle \nabla F(\mathbf{w}_{kB}), \gamma \sum_{t=kB}^{(k+1)B-1} \nabla f_{s_t}(\mathbf{w}_{kB}) \right\rangle \le F(\mathbf{w}_{kB}) - F(\mathbf{w}_{(k+1)B}) + \frac{\beta}{2}\left\|\gamma \sum_{t=kB}^{(k+1)B-1} \nabla f_{s_t}(\mathbf{w}_{kB})\right\|_2^2.$$

Now we take expectation on both sides. By iterative expectation, we know that for any $t \ge kB$,

$$\mathbb{E}[\langle \nabla F(\mathbf{w}_{kB}), \nabla f_{s_t}(\mathbf{w}_{kB})\rangle] = \mathbb{E}[\|\nabla F(\mathbf{w}_{kB})\|_2^2].$$

We also have

$$\mathbb{E}\left[\left\|\sum_{t=kB}^{(k+1)B-1} \nabla f_{s_t}(\mathbf{w}_{kB})\right\|_2^2\right] = \mathbb{E}[BM^2(\mathbf{w}_{kB}) + B(B-1)G(\mathbf{w}_{kB})] \leq B(1+\delta)M^2.$$

Consequently,

$$\gamma B\mathbb{E}[\|\nabla F(\mathbf{w}_{kB})\|_2^2] \leq \mathbb{E}[F(\mathbf{w}_{kB})] - \mathbb{E}[F(\mathbf{w}_{(k+1)B})] + \frac{\beta}{2}\gamma^2 B(1+\delta)M^2. \tag{2.6}$$

Summing up equation (2.6) for $k = 0, \ldots, T/B - 1$ yields

$$\gamma B \sum_{k=0}^{T/B-1} \mathbb{E}[\|\nabla F(\mathbf{w}_{kB})\|_2^2] \leq F(\mathbf{w}_0) - F^* + \frac{\beta}{2}\gamma^2 T(1+\delta)M^2,$$

which simplifies to

$$\min_{k=0,\ldots,T/B-1} \mathbb{E}[\|\nabla F(\mathbf{w}_{kB})\|_2^2] \leq \frac{F(\mathbf{w}_0) - F^*}{\gamma T} + \frac{\beta}{2}\gamma(1+\delta)M^2.$$

We can then derive the results by replacing γ and T with the particular choices. □

Proof of Theorem 2.3

Substituting $\mathbf{w} = \mathbf{w}_{(k+1)B}$ and $\mathbf{w}' = \mathbf{w}_{kB}$ in the condition for β-smoothness in Definition 2.1, we obtain

$$\begin{aligned} F(\mathbf{w}_{(k+1)B}) \leq & F(\mathbf{w}_{kB}) - \gamma \left\langle \nabla F(\mathbf{w}_{kB}), \sum_{t=kB}^{(k+1)B-1} \nabla f_{s_t}(\mathbf{w}_{kB}) \right\rangle \\ & + \frac{\beta\gamma^2}{2} \left\| \sum_{t=kB}^{(k+1)B-1} \nabla f_{s_t}(\mathbf{w}_{kB}) \right\|_2^2 . \end{aligned}$$

Condition on $\mathbf{w}_{kB}$ and take expectations over the choice of s_t, $t = kB, \ldots, (k+1)B-1$. We obtain

$$\begin{aligned} \mathbb{E}[F(\mathbf{w}_{(k+1)B}) \mid \mathbf{w}_{kB}] \leq & F(\mathbf{w}_{kB}) - \gamma B\|\nabla F(\mathbf{w}_{kB})\|_2^2 \\ & + \frac{\beta\gamma^2}{2}\left(BM^2(\mathbf{w}_{kB}) + B(B-1)G(\mathbf{w}_{kB})\right). \end{aligned} \tag{2.7}$$

Then, we take expectation over all the randomness of the algorithm. Using the PL condition in Definition 2.3 and the fact that $B \leq 1 + \delta B_D(\mathbf{w})$ for all $\mathbf{w} \in \mathcal{W}_T$, we write

$$\mathbb{E}\left[F(\mathbf{w}_{(k+1)B}) - F^*\right] \leq (1 - 2\gamma\mu B)\mathbb{E}\left[F(\mathbf{w}_{kB}) - F^*\right] + (1+\delta)\frac{\beta B\gamma^2 M^2}{2}. \tag{2.8}$$

Then, if $B \leq \frac{1}{2\gamma\mu}$, we have

$$\mathbb{E}\left[F(\mathbf{w}_T) - F^*\right] \leq (1 - 2\gamma\mu B)^{T/B}(F(\mathbf{w}_0) - F^*) + (1+\delta)\frac{\beta\gamma M^2}{4\mu}.$$

Using the fact that $1 - x \leq e^{-x}$ for any $x \geq 0$, and choosing $\gamma = \frac{2\epsilon\mu}{M^2\beta}$, we obtain the desired result. □

Proof of Theorem 2.4

According to Lemma 2.1, for every $k = 0, 1, \ldots, \frac{T}{B} - 1$, we have

$$\mathbb{E}[\|\mathbf{w}_{(k+1)B} - \mathbf{w}^*\|_2^2 \mid \mathbf{w}_{kB}] \leq \|\mathbf{w}_{kB} - \mathbf{w}^*\|_2^2 - 2\gamma B \left\langle \nabla F(\mathbf{w}_{kB}), \mathbf{w}_{kB} - \mathbf{w}^* \right\rangle + (1+\delta)\gamma^2 B M^2.$$

Then, we take expectation over all the randomness of the algorithm. Let $D_{kB} = \mathbb{E}[\|\mathbf{w}_{kB} - \mathbf{w}^*\|_2^2]$. We have

$$\mathbb{E}[\langle \nabla F(\mathbf{w}_{kB}), \mathbf{w}_{kB} - \mathbf{w}^* \rangle] \leq \frac{1}{2\gamma B}(D_{kB} - D_{(k+1)B}) + (1+\delta)\frac{\gamma}{2}M^2. \tag{2.9}$$

We use equation (2.9) to prove the convergence rate. We have by convexity

$$\begin{aligned}
\mathbb{E}\left[F\left(\frac{B}{T}\sum_{k=0}^{\frac{T}{B}-1}\mathbf{w}_{kB}\right) - F(\mathbf{w}^*)\right] &\leq \mathbb{E}\left[\frac{B}{T}\sum_{k=0}^{\frac{T}{B}-1} F(\mathbf{w}_{kB}) - F(\mathbf{w}^*)\right] \\
&= \frac{B}{T}\sum_{t=0}^{\frac{T}{B}-1}\mathbb{E}[F(\mathbf{w}_{kB}) - F(\mathbf{w}^*)] \\
&\leq \frac{B}{T}\sum_{t=0}^{\frac{T}{B}-1}\mathbb{E}[\langle \nabla F(\mathbf{w}_{kB}), \mathbf{w}_{kB} - \mathbf{w}^* \rangle] \\
&\leq \frac{D_0}{2\gamma T} + (1+\delta)\frac{\gamma M^2}{2},
\end{aligned}$$

where the last inequality is obtained by summing inequality (2.9) over $k = 0, 1, \ldots, \frac{T}{B} - 1$. Then, we can derive the results by replacing γ and T with the particular choices. □

Proof of Theorem 2.5

According to Lemma 2.1, we have

$$\mathbb{E}[\|\mathbf{w}_{(k+1)B} - \mathbf{w}^*\|_2^2 \mid \mathbf{w}_{kB}] \leq \|\mathbf{w}_{kB} - \mathbf{w}^*\|_2^2 - 2\gamma B \left\langle \nabla F(\mathbf{w}_{kB}), \mathbf{w}_{kB} - \mathbf{w}^* \right\rangle + (1+\delta)\gamma^2 B M^2(\mathbf{w}_{kB}).$$

By strong convexity of $F(\mathbf{w})$, we have

$$\langle \nabla F(\mathbf{w}_{kB}), \mathbf{w}_{kB} - \mathbf{w}^* \rangle \geq \lambda \|\mathbf{w}_{kB} - \mathbf{w}^*\|_2^2,$$

which yields

$$\mathbb{E}[\|\mathbf{w}_{(k+1)B} - \mathbf{w}^*\|_2^2 \mid \mathbf{w}_{kB}] \leq (1 - 2\gamma\lambda B)\|\mathbf{w}_{kB} - \mathbf{w}^*\|_2^2 + (1+\delta)\gamma^2 B M^2(\mathbf{w}_{kB}). \tag{2.10}$$

Then, by taking expectations over the randomness of the whole algorithm on both sizes of (2.10), we obtain

$$\mathbb{E}[\|\mathbf{w}_{(k+1)B} - \mathbf{w}^*\|_2^2] \leq (1 - 2\gamma\lambda B)\mathbb{E}[\|\mathbf{w}_{kB} - \mathbf{w}^*\|_2^2] + (1+\delta)\gamma^2 B M^2.$$

Then if $B \leq \frac{1}{2\gamma\lambda}$, we obtain

$$\mathbb{E}[\|\mathbf{w}_T - \mathbf{w}^*\|_2^2] \leq (1 - 2\gamma\lambda B)^{T/B}\|\mathbf{w}_0 - \mathbf{w}^*\|_2^2 + (1+\delta)\frac{\gamma M^2}{2\lambda}.$$

Using the fact that $1 - x \leq e^{-x}$ for any $x \geq 0$, we obtain

$$\mathbb{E}[\|\mathbf{w}_T - \mathbf{w}^*\|_2^2] \leq e^{-2\gamma\lambda T} D_0 + (1+\delta)\frac{\gamma M^2}{2\lambda}.$$

We complete the proof by taking $\gamma = \frac{\epsilon\lambda}{M^2}$.

Lower Bound

In this section, we prove the lower bound on convergence for strongly convex functions.

Proof of Theorem 2.6

We set $f_i(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w} - \mathbf{x}_i\|_2^2$, and thus $F(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^n \frac{\lambda}{2}\|\mathbf{w} - \mathbf{x}_i\|_2^2$. We choose $\mathcal{W} = \{\mathbf{w} : \|\mathbf{w}\|_2 \leq 1\}$, and $\mathbf{x}_i$'s such that $\|\mathbf{x}_i\|_2 = 1$ for all $i = 1, \ldots, n$, and $\sum_{i=1}^n \mathbf{x}_i = \mathbf{0}$.

One can check that $\nabla f_i(\mathbf{w}) = \lambda(\mathbf{w} - \mathbf{x}_i)$, $\nabla F(\mathbf{w}) = \lambda\mathbf{w}$, and

$$M^2(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^n \|\nabla f_i(\mathbf{w})\|_2^2 = \frac{1}{n}\sum_{i=1}^n \lambda^2\|\mathbf{w} - \mathbf{x}_i\|_2^2 = \frac{1}{n}\sum_{i=1}^n \lambda^2(\|\mathbf{w}\|_2^2 + \|\mathbf{x}_i\|_2^2).$$

Since $M^2(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^n \lambda^2(\|\mathbf{w}\|_2^2 + \|\mathbf{x}_i\|_2^2) \in [\lambda^2, 2\lambda^2]$ for all $\mathbf{w} \in \mathcal{W}$, we know that we have $M^2(\mathbf{w}) \geq \frac{1}{2}M^2$ for all $\mathbf{w} \in \mathcal{W}$.

Since $\mathcal{W}$ is a bounded set, the projection step has to be taken in order to guarantee that $\mathbf{w}_{N_k} \in \mathcal{W}$. However, one can show that, if the initial guess $\mathbf{w}_0$ is in the convex hull of $\mathbf{x}_1, \ldots, \mathbf{x}_n$ (denoted by $\mathcal{C} \subset \mathcal{W}$), then, without using projection, the obtained model parameter $\mathbf{w}_{N_k}$ always stays inside $\mathcal{C}$. More specifically, we have the following result.

Proposition 2.4. *Suppose that $B_k \leq \frac{1}{\lambda\gamma}$ for all $k = 1, \ldots, K$, and $\mathbf{w}_0 \in \mathcal{C}$. Then, without using projection, $\mathbf{w}_{N_k} \in \mathcal{C}$ for all k.*

Proof. We prove this result using induction. Suppose that $\mathbf{w}_{N_{k-1}} \in \mathcal{C}$. Then, we have

$$\begin{aligned}\mathbf{w}_{N_k} =& \mathbf{w}_{N_{k-1}} - \gamma \sum_{\ell=N_{k-1}}^{N_k-1} \nabla f_{s_\ell}(\mathbf{w}_{N_{k-1}}) = \mathbf{w}_{N_{k-1}} - \gamma \sum_{\ell=N_{k-1}}^{N_k-1} \lambda(\mathbf{w}_{N_{k-1}} - \mathbf{x}_{s_\ell}) \\ =& (1-\gamma\lambda B_k)\mathbf{w}_{N_{k-1}} + \gamma\lambda B_k \left(\frac{1}{B_k} \sum_{\ell=N_{k-1}}^{N_k-1} \mathbf{x}_{s_\ell} \right).\end{aligned}$$

Since $\mathbf{w}_{N_{k-1}}, \frac{1}{B_k}\sum_{\ell=N_{k-1}}^{N_k-1} \mathbf{x}_{s_\ell} \in \mathcal{C}$, we prove Lemma 2.4. □

From now on we assume $\mathbf{w}_0 \in \mathcal{C}$ and do not consider projection. According to equation (2.5) in the proof of Lemma 2.1, we have[5]

$$\begin{aligned}\mathbb{E}[\|\mathbf{w}_{N_k} - \mathbf{w}^*\|_2^2 \mid \mathbf{w}_{N_{k-1}}] =& \|\mathbf{w}_{N_{k-1}} - \mathbf{w}^*\|_2^2 - 2\gamma B_k \langle \mathbf{w}_{N_{k-1}} - \mathbf{w}^*, \nabla F(\mathbf{w}_{N_{k-1}})\rangle \\ &+ \gamma^2 B_k \left(1 + \frac{B_k - 1}{B_D(\mathbf{w}_{N_{k-1}})}\right) M^2(\mathbf{w}_{N_{k-1}}) \\ \geq& (1-2\gamma\lambda B_k)\|\mathbf{w}_{N_{k-1}} - \mathbf{w}^*\|_2^2 + \frac{1}{2}\gamma^2 M^2 B_k \left(1 + \frac{B_k-1}{B_D(\mathbf{w}_{N_{k-1}})}\right).\end{aligned}$$

Then, we take expectation over the randomness of the whole algorithm and obtain

$$\begin{aligned}&\mathbb{E}[\|\mathbf{w}_{N_k} - \mathbf{w}^*\|_2^2] \\ \geq& (1-2\gamma\lambda B_k)\mathbb{E}[\|\mathbf{w}_{N_{k-1}} - \mathbf{w}^*\|_2^2] + \frac{1}{2}\gamma^2 M^2 B_k \left(1 + (B_k-1)\mathbb{E}\left[\frac{1}{B_D(\mathbf{w}_{N_{k-1}})}\right]\right) \\ \geq& (1-2\gamma\lambda B_k)\mathbb{E}[\|\mathbf{w}_{N_{k-1}} - \mathbf{w}^*\|_2^2] + \frac{1}{2}\gamma^2 M^2 B_k \left(1 + (B_k-1)\frac{1}{\mathbb{E}[B_D(\mathbf{w}_{N_{k-1}})]}\right) \\ \geq& (1-2\gamma\lambda B_k)\mathbb{E}[\|\mathbf{w}_{N_{k-1}} - \mathbf{w}^*\|_2^2] + \frac{1}{2}(1+\delta)\gamma^2 M^2 B_k,\end{aligned}$$

where the second inequality is due to Jensen's inequality, and the third inequality is due to the fact that $B_k \geq 1 + \delta\mathbb{E}[B_D(\mathbf{w}_{N_{k-1}})]$.

Rolling out the above recursion, and denoting $\alpha_k = 2\gamma\lambda B_k \in [0,1]$, we have

$$\begin{aligned}&\mathbb{E}\left[\|\mathbf{w}_{N_K} - \mathbf{w}^*\|_2^2\right] \\ \geq& \|\mathbf{w}_0 - \mathbf{w}^*\|_2^2 \left(\prod_{k=1}^{K}(1-\alpha_k)\right) + \frac{1}{2}(1+\delta)\gamma^2 M^2 \left[B_K + \sum_{k=1}^{K-1}\prod_{i=k+1}^{K}(1-\alpha_i)B_k\right] \\ =& \|\mathbf{w}_0 - \mathbf{w}^*\|_2^2 \left(\prod_{i=1}^{K}(1-\alpha_i)\right) + \frac{1}{4}(1+\delta)\frac{\gamma M^2}{\lambda}\left[\alpha_K + \sum_{k=1}^{K-1}\prod_{i=k+1}^{K}(1-\alpha_i)\alpha_k\right].\end{aligned}$$

[5]We still keep $\mathbf{w}^*$ although $\mathbf{w}^* = \mathbf{0}$.

Now the number of gradient updates is given by $\sum_{k=1}^K B_k = T$, and consequently, $\sum_{k=1}^K \alpha_k = 2\gamma\lambda T$. Since we consider the case when $T \geq \frac{c}{\gamma\lambda}$ for some universal constant $c > 0$ (and SGD only converges in this regime), so we have $\sum_{k=1}^K \alpha_k \geq 2c$.

Substituting the value of step-size γ, we see that in order to complete the proof, it suffices to show that the quantity

$$J(\alpha) = \alpha_K + \sum_{k=1}^{K-1} \prod_{i=k+1}^{K} (1-\alpha_i)\alpha_k$$

is lower bounded as $\Omega(1)$. In order to show this, note that $J(\alpha)$ can be equivalently expressed as the CDF of a geometric distribution with non-uniform probabilities of success α_k. We could further see that

$$J(\alpha) = 1 - \prod_{k=1}^{K}(1-\alpha_k) \geq 1 - \left[\frac{1}{K}\sum_{k=1}^{K}(1-\alpha_k)\right]^K \geq 1-(1-2c/K)^K,$$

and the last term is lower bounded by a constant for all $K \geq 1$. □

A second bound that was implicit in our convergence rates is also sharp, as shown in the following section.

Necessity of $B \leq \mathcal{O}(\frac{1}{\lambda\gamma})$

In this section, we show that, up to a constant factor, the condition $B \leq \frac{1}{2\gamma\lambda}$ in Theorem 2.5 and 2.6, is actually necessary for mini-batch SGD to converge when $F(\mathbf{w})$ is strongly convex. More precisely, we can show that, when $B > \frac{2}{\gamma\lambda}$, mini-batch SGD diverges.

Proposition 2.5. *Suppose that $F(\mathbf{w})$ is λ-strongly convex. Condition on the model parameter $\mathbf{w}_{kB}$ obtained after k iterations. Suppose that $\mathbf{w}_{kB} - \gamma\sum_{i\in\mathcal{I}} \nabla f_i(\mathbf{w}_{kB}) \in \mathcal{W}$ for all $\mathcal{I} \in [n]^B$. Then, if $B > \frac{2}{\gamma\lambda}$, we have*

$$\mathbb{E}\left[\|\mathbf{w}_{(k+1)B} - \mathbf{w}^*\|_2 \mid \mathbf{w}_{kB}\right] > \|\mathbf{w}_{kB} - \mathbf{w}^*\|_2.$$

Proof. We have

$$\begin{aligned}
\mathbb{E}\left[\|\mathbf{w}_{(k+1)B} - \mathbf{w}_{kB}\|_2 \mid \mathbf{w}_{kB}\right] &\geq \left\|\mathbb{E}[\mathbf{w}_{(k+1)B} - \mathbf{w}_{kB} \mid \mathbf{w}_{kB}]\right\|_2 \\
&= \gamma \left\| \sum_{t=kB}^{(k+1)B-1} \mathbb{E}[\nabla f_{s_t}(\mathbf{w}_{kB}) \mid \mathbf{w}_{kB}] \right\|_2 \\
&= \gamma B \|\nabla F(\mathbf{w}_{kB})\|_2 \\
&\geq \gamma B \lambda \|\mathbf{w}_{kB} - \mathbf{w}^*\|_2,
\end{aligned}$$

where the first step follows by Jensen's inequality, and the last by strong convexity.

This allows us to conclude that if $B > \frac{2}{\gamma\lambda}$, $\mathbb{E}\left[\|\mathbf{w}_{(k+1)B} - \mathbf{w}_{kB}\|_2 \mid \mathbf{w}_{kB}\right] > 2\|\mathbf{w}_{kB} - \mathbf{w}^*\|_2$. Then, by triangle inequality,

$$\mathbb{E}\left[\|\mathbf{w}_{(k+1)B} - \mathbf{w}^*\|_2 \mid \mathbf{w}_{kB}\right] \geq \mathbb{E}\left[\|\mathbf{w}_{(k+1)B} - \mathbf{w}_{kB}\|_2 \mid \mathbf{w}_{kB}\right] - \|\mathbf{w}_{kB} - \mathbf{w}^*\|_2 > \|\mathbf{w}_{kB} - \mathbf{w}^*\|_2,$$

and thus mini-batch SGD diverges. □

We now turn to showing that various heuristics for SGD are also diversity-inducing.

Proof of Theorem 2.7

For DropConnect, we have

$$\begin{aligned} B_D^{\mathsf{drop}}(\mathbf{w}) &= n\frac{\sum_{i=1}^n \mathbb{E}[\|\mathbf{D}_i \nabla f_i(\mathbf{w})\|_2^2]}{\mathbb{E}[\|\sum_{i=1}^n \mathbf{D}_i \nabla f_i(\mathbf{w})\|_2^2]} \\ &= \frac{n\sum_{i=1}^n (1-p)\|\nabla f_i(\mathbf{w})\|_2^2}{\sum_{i=1}^n (1-p)\|\nabla f_i(\mathbf{w})\|_2^2 + (1-p)^2 \sum_{j\neq k}\langle \nabla f_j(\mathbf{w}), \nabla f_k(\mathbf{w})\rangle}. \end{aligned} \tag{2.11}$$

Recall that

$$B_D(\mathbf{w}) = \frac{n\sum_{i=1}^n \|\nabla f_i(\mathbf{w})\|_2^2}{\sum_{i=1}^n \|\nabla f_i(\mathbf{w})\|_2^2 + \sum_{j\neq k}\langle \nabla f_j(\mathbf{w}), \nabla f_k(\mathbf{w})\rangle},$$

and we can see that for any $\mathbf{w}$ such that $\sum_{j\neq k}\langle \nabla f_j(\mathbf{w}), \nabla f_k(\mathbf{w})\rangle \geq 0$, we must have $B_D(\mathbf{w}) \leq n$. In this case, we have

$$B_D^{\mathsf{drop}}(\mathbf{w}) \geq \frac{n\sum_{i=1}^n (1-p)\|\nabla f_i(\mathbf{w})\|_2^2}{\sum_{i=1}^n (1-p)\|\nabla f_i(\mathbf{w})\|_2^2 + (1-p)\sum_{j\neq k}\langle \nabla f_j(\mathbf{w}), \nabla f_k(\mathbf{w})\rangle} = B_D(\mathbf{w}).$$

On the other hand, if $\sum_{j\neq k}\langle \nabla f_j(\mathbf{w}), \nabla f_k(\mathbf{w})\rangle < 0$, we must have $B_D(\mathbf{w}) > n$, and one can simply check that we also have $B_D^{textsfdrop}(\mathbf{w}) > n$.

For stochastic gradient Langevin dynamics, we have

$$B_D^{\mathsf{sgld}}(\mathbf{w}) = \frac{n\sum_{i=1}^n \mathbb{E}[\|\nabla f_i(\mathbf{w}) + \xi_i\|_2^2]}{\mathbb{E}[\|\sum_{i=1}^n (\nabla f_i(\mathbf{w}) + \xi_i)\|_2^2]} = \frac{n\sum_{i=1}^n \|\nabla f_i(\mathbf{w})\|_2^2 + n^2 d\sigma^2}{\|\sum_{i=1}^n \nabla f_i(\mathbf{w})\|_2^2 + nd\sigma^2}. \tag{2.12}$$

Therefore, as long as $B_D(\mathbf{w}) = \frac{n\sum_{i=1}^n \|\nabla f_i(\mathbf{w})\|_2^2}{\|\sum_{i=1}^n \nabla f_i(\mathbf{w})\|_2^2} \leq n$, we have $B_D^{\mathsf{sgld}}(\mathbf{w}) \geq B_D(\mathbf{w})$. In addition, if $B_D(\mathbf{w}) > n$, then $B_D^{\mathsf{sgld}}(\mathbf{w}) > n$.

For quantization, one can simply check that for any $i \in [n]$, we have

$$\mathbb{E}[\|Q(\nabla f_i(\mathbf{w}))\|_2^2] = \|\nabla f_i(\mathbf{w})\|_2 \|\nabla f_i(\mathbf{w})\|_1,$$

and for any $j \neq k$, we have

$$\mathbb{E}[\langle Q(\nabla f_j(\mathbf{w})), Q(\nabla f_k(\mathbf{w}))\rangle] = \langle \nabla f_j(\mathbf{w}), \nabla f_k(\mathbf{w})\rangle.$$

Consequently,

$$\begin{aligned} B_D^{\mathsf{quant}}(\mathbf{w}) &= \frac{n\sum_{i=1}^n \mathbb{E}[\|Q(\nabla f_i(\mathbf{w}))\|_2^2]}{\mathbb{E}[\|\sum_{i=1}^n Q(\nabla f_i(\mathbf{w}))\|_2^2]} \\ &= \frac{n\sum_{i=1}^n \|\nabla f_i(\mathbf{w})\|_2 \|\nabla f_i(\mathbf{w})\|_1}{\sum_{i=1}^n \|\nabla f_i(\mathbf{w})\|_2 \|\nabla f_i(\mathbf{w})\|_1 + \sum_{j\neq k}\langle \nabla f_j(\mathbf{w}), \nabla f_k(\mathbf{w})\rangle}. \end{aligned} \tag{2.13}$$

We define

$$\Delta_D^{\mathsf{quant}}(\mathbf{w}) := \frac{\sum_{i=1}^n \|\nabla f_i(\mathbf{w})\|_2 \|\nabla f_i(\mathbf{w})\|_1}{\sum_{i=1}^n \|\nabla f_i(\mathbf{w})\|_2 \|\nabla f_i(\mathbf{w})\|_1 + \sum_{j\neq k}\langle \nabla f_j(\mathbf{w}), \nabla f_k(\mathbf{w})\rangle},$$

and

$$\Delta_D(\mathbf{w}) := \frac{\sum_{i=1}^n \|\nabla f_i(\mathbf{w})\|_2^2}{\sum_{i=1}^n \|\nabla f_i(\mathbf{w})\|_2^2 + \sum_{j\neq k}\langle \nabla f_j(\mathbf{w}), \nabla f_k(\mathbf{w})\rangle},$$

and we have $B_D^{\mathsf{quant}}(\mathbf{w}) = n\Delta_D^{\mathsf{quant}}(\mathbf{w})$ and $B_D = n\Delta_D(\mathbf{w})$. One can now check that due to the fact that $\|\mathbf{v}\|_2\|\mathbf{v}\|_1 \geq \|\mathbf{v}\|_2^2$ for any vector $\mathbf{v}$, when $\Delta_D(\mathbf{w}) \in (0,1)$, we have $\Delta_D^{\mathsf{quant}}(\mathbf{w}) > \Delta_D(\mathbf{w})$, and when $\Delta_D(\mathbf{w}) > 1$, we have $\Delta_D^{\mathsf{quant}}(\mathbf{w}) > 1$. □

Stability

Let us begin by defining some useful notation. We let

$$\overline{M}^2(\mathbf{w}, \mathbf{w}') := \frac{1}{n}\sum_{i=1}^n \|\nabla f_i(\mathbf{w}) - \nabla f_i(\mathbf{w}')\|_2^2 \quad \text{and} \quad \overline{G}(\mathbf{w}, \mathbf{w}') := \|\nabla F(\mathbf{w}) - \nabla F(\mathbf{w}')\|_2^2.$$

One can see that $\overline{B}_D(\mathbf{w}, \mathbf{w}') = \frac{\overline{M}^2(\mathbf{w}, \mathbf{w}')}{\overline{G}(\mathbf{w}, \mathbf{w}')}$. We also define

$$\overline{B}_D = \inf_{\mathbf{w}\neq\mathbf{w}'} \overline{B}_D(\mathbf{w}, \mathbf{w}').$$

Before turning to the proofs, we first provide some background.

Background on Stability and Generalization

Recall that in supervised learning problems, our goal is to learn a parametric model with small population risk $R(\mathbf{w}) := \mathbb{E}_{\mathbf{z}\sim\mathcal{D}}[f(\mathbf{w};\mathbf{z})]$. In order to do so, we use empirical risk minimization, and hope to obtain a model that has both small empirical risk and small population risk to avoid overfitting. Formally, let A be a possibly randomized algorithm which maps the training data to the parameter space as $\mathbf{w} = A(\mathcal{S})$. We define the *expected generalization error* of the algorithm as

$$\epsilon_{\text{gen}}(A) := \left|\mathbb{E}_{\mathcal{S},A}[R_{\mathcal{S}}(A(\mathcal{S})) - R(A(\mathcal{S}))]\right|.$$

Bousquet and Ellisseef show that there is a fundamental connection between the generalization error and algorithmic stability. An algorithm is said to be stable if it produces similar models given similar training data. We summarize their result as follows.

Proposition 2.6. *Let $\mathcal{S} = (\mathbf{z}_1, \ldots, \mathbf{z}_n)$ and $\mathcal{S}' = (\mathbf{z}'_1, \ldots, \mathbf{z}'_n)$ be two independent random samples from $\mathcal{D}$, and let $\mathcal{S}^{(i)} = (\mathbf{z}_1, \ldots, \mathbf{z}_{i-1}, \mathbf{z}'_i, \mathbf{z}_{i+1}, \ldots, \mathbf{z}_n)$ be the sample that is identical to $\mathcal{S}$ except in the i-th data point where we replace $\mathbf{z}_i$ with $\mathbf{z}'_i$. Then, we have*

$$\mathbb{E}_{\mathcal{S},A}[R_{\mathcal{S}}(A(\mathcal{S})) - R(A(\mathcal{S}))] = \mathbb{E}_{\mathcal{S},\mathcal{S}',A}\left[\frac{1}{n}\sum_{i=1}^{n} f(A(\mathcal{S}^{(i)}); \mathbf{z}'_i) - \frac{1}{n}\sum_{i=1}^{n} f(A(\mathcal{S}); \mathbf{z}'_i)\right].$$

With the notation in Proposition 2.6, we define the following quantity that characterizes the algorithmic *stability* of the learning algorithm given the data points:

$$\epsilon_{\text{stab}}(\mathcal{S}, \mathcal{S}') = \mathbb{E}_A\left[\frac{1}{n}\sum_{i=1}^{n} f(A(\mathcal{S}^{(i)}); \mathbf{z}'_i) - \frac{1}{n}\sum_{i=1}^{n} f(A(\mathcal{S}); \mathbf{z}'_i)\right], \tag{2.14}$$

where we condition on the data sets $\mathcal{S}$ and $\mathcal{S}'$ and take expectation over the randomness of the learning algorithm (mini-batch SGD). Recall from Theorem 2.6 that

$$\epsilon_{\text{gen}}(A) = |\mathbb{E}_{\mathcal{S},\mathcal{S}'}[\epsilon_{\text{stab}}(\mathcal{S}, \mathcal{S}')]| \le \mathbb{E}_{\mathcal{S},\mathcal{S}'}[|\epsilon_{\text{stab}}(\mathcal{S}, \mathcal{S}')|]. \tag{2.15}$$

We bound $\epsilon_{\text{gen}}(A)$ by first showing a bound on $\epsilon_{\text{stab}}(\mathcal{S}, \mathcal{S}')$ that depends on the sample $(\mathcal{S}, \mathcal{S}')$, then using equation (2.15) to obtain, as a corollary, results for generalization error.

For convex and strongly convex functions, we have the following two results on stability.

Proposition 2.7 (stability of convex functions)**.** *Fix sample $(\mathcal{S}, \mathcal{S}')$. Suppose that for any $\mathbf{z} \in \mathcal{Z}$, $f(\mathbf{w}; \mathbf{z})$ is convex, L-Lipschitz and β-smooth in $\mathcal{W}$. Provided the step-size and batch-size satisfy*

$$\gamma \le \frac{2}{\beta\left(1 + \frac{1}{n-1}\mathbb{1}_{B>1} + \frac{B-1}{B_D(\mathbf{w},\mathbf{w}')}\right)}, \tag{2.16}$$

for all $\mathbf{w} \neq \mathbf{w}'$, we have $|\epsilon_{stab}(\mathcal{S}, \mathcal{S}')| \le 2\gamma L^2 \frac{T}{n}$.

Proposition 2.8 (stability of strongly convex functions)**.** *Fix the sample $(\mathcal{S}, \mathcal{S}')$. Suppose that for any $\mathbf{z} \in \mathcal{Z}$, $f(\mathbf{w}; \mathbf{z})$ is L-Lipschitz, β-smooth, and λ-strongly convex in $\mathcal{W}$, and that $B \le \frac{1}{2\gamma\lambda}$. Provided the step-size and batch-size satisfy*

$$\gamma \le \frac{2}{(\beta + \lambda)\left(1 + \frac{1}{n-1}\mathbb{1}_{B>1} + \frac{B-1}{B_D(\mathbf{w},\mathbf{w}')}\right)}, \tag{2.17}$$

for all $\mathbf{w} \neq \mathbf{w}'$, we have $|\epsilon_{stab}(\mathcal{S}, \mathcal{S}')| \le \frac{4L^2}{\lambda n}$.

We also note that Theorem 2.9 and Theorem 2.10 in this chapter can be derived as Corollaries of Proposition 2.7 and 2.8, respectively. In following sections, we provide the details.

Proofs of Proposition 2.7 and Theorem 2.9

We first recall the problem setting. Suppose that there are two sample sets $\mathcal{S}$ and $\mathcal{S}^{(I)}$ which differ at one data point located at a random position I, which is uniformly distributed in $[n]$. We run the same (projected) parallel mini-batch SGD on both data sets, and after the k-th parallel iteration, we obtain $\mathbf{w}_{kB}$ and $\widetilde{\mathbf{w}}_{kB}$, respectively. After a total number of T gradient updates, *i.e.,* T/B parallel iterations, we obtain $\mathbf{w}_T$ and $\widetilde{\mathbf{w}}_T$. Let s_t, $t = 0, 1, \ldots, T-1$ be the sequence of indices of samples used by the algorithm. In our setting, s_t are i.i.d. uniformly distributed in $\{1, 2, \ldots, n\}$. Let $\mathbf{z}_{s_t} \in \mathcal{S}$ and $\widetilde{\mathbf{z}}_{s_t} \in \mathcal{S}^{(I)}$, $t = 0, \ldots, T-1$ be the data point used in the algorithms running on the two data sets, respectively. Then, we know that with probability $1 - \frac{1}{n}$, $\mathbf{z}_{s_t} = \widetilde{\mathbf{z}}_{s_t}$, and with probability $\frac{1}{n}$, $\mathbf{z}_{s_t} \neq \widetilde{\mathbf{z}}_{s_t}$. We simplify the notations of the risk function associated with $\mathbf{z}_{s_t}$ and $\widetilde{\mathbf{z}}_{s_t}$ by $f_{s_t}(\mathbf{w}) := f(\mathbf{w}; \mathbf{z}_{s_t})$, and $\widetilde{f}_{s_t}(\mathbf{w}) := f(\mathbf{w}; \widetilde{\mathbf{z}}_{s_t})$, respectively.

We now prove Proposition 2.7. Throughout this proof, we only consider the case where $B > 1$ and omit the indicator function $\mathbb{1}_{B>1}$. We condition on the data sets and the event that the choice of γ is "good", as shown in (2.16). Specifically, we condition on the samples $\mathcal{S}$ and $\mathcal{S}'$, and the event Γ:

$$\Gamma = \left\{ \gamma \leq \frac{2}{\beta(1 + \frac{1}{n-1} + \frac{B-1}{\overline{B}_D})} \right\} = \left\{ \overline{B}_D \geq \frac{B-1}{\frac{2}{\gamma\beta} - 1 - \frac{1}{n-1}} \right\}. \tag{2.18}$$

Recall the definition of η:

$$\eta = \mathbb{P}\left\{ \inf_{\mathbf{w} \neq \mathbf{w}'} \overline{B}_D(\mathbf{w}, \mathbf{w}') < \frac{B-1}{\frac{2}{\gamma\beta} - 1 - \frac{1}{n-1}\mathbb{1}_{B>1}} \right\}. \tag{2.19}$$

We know that $\eta = \mathbb{P}\{\bar{\Gamma}\}$, and so our goal is to bound $|\epsilon_{\text{stab}}(\mathcal{S}, \mathcal{S}')|$ conditioned on the event Γ. Since we assume that $f(\mathbf{w}; \mathbf{z})$ is L-Lipschitz on $\mathcal{W}$, we have

$$|\epsilon_{\text{stab}}(\mathcal{S}, \mathcal{S}')| \leq L\mathbb{E}_{I,A|\Gamma}\left[\|A(\mathcal{S}^{(I)}) - A(\mathcal{S})\|_2\right] = L\mathbb{E}_{I,A|\Gamma}\left[\|\mathbf{w}_T - \widetilde{\mathbf{w}}_T\|_2\right], \tag{2.20}$$

and thus it suffices to bound $\mathbb{E}_{I,A|\Gamma}\left[\|\mathbf{w}_T - \widetilde{\mathbf{w}}_T\|_2\right]$.

Consider the samples used in the $(k+1)$-th parallel iteration in the two algorithm instances, *i.e.,* $\{\mathbf{z}_{s_t}\}_{t=kB}^{(k+1)B-1}$, and $\{\widetilde{\mathbf{z}}_{s_t}\}_{t=kB}^{(k+1)B-1}$. Let H_{k+1} be the the number of samples that differ between the two minibatches in iteration $k+1$. According to our sampling scheme, $H_{k+1} \sim \text{bin}(B, \frac{1}{n})$. We condition on the event that $H_{k+1} = h$. Without loss of generality, we assume that $\mathbf{z}_{s_t} = \widetilde{\mathbf{z}}_{s_t}$ for all $t = kB, \ldots, (k+1)B - h - 1$, and $\mathbf{z}_{s_t} \neq \widetilde{\mathbf{z}}_{s_t}$ for all $t = (k+1)B - h, \ldots, (k+1)B - 1$. Consider the first $B - h$ terms. For the unconstrained optimization, we have

$$\begin{aligned} &\|\mathbf{w}_{(k+1)B-h} - \widetilde{\mathbf{w}}_{(k+1)B-h}\|_2^2 \\ =&\|(\mathbf{w}_{kB} - \gamma \sum_{t=kB}^{(k+1)B-h-1} \nabla f_{s_t}(\mathbf{w}_{kB})) - (\widetilde{\mathbf{w}}_{kB} - \gamma \sum_{t=kB}^{(k+1)B-h-1} \nabla \widetilde{f}_{s_t}(\widetilde{\mathbf{w}}_{kB}))\|_2^2. \end{aligned} \tag{2.21}$$

For the algorithm with projection, the B gradient update steps are the same as the unconstrained algorithm, and projection step is conducted once all the gradient updates are finished. Therefore, equation (2.21) also holds for projected algorithm.

Since $f_{s_t}(\mathbf{w}) = \widetilde{f}_{s_t}(\mathbf{w})$ for all $t = kB, \dots, (k+1)B - h - 1$, we further have

$$\begin{aligned}
&\|\mathbf{w}_{(k+1)B-h} - \widetilde{\mathbf{w}}_{(k+1)B-h}\|_2^2 \\
=&\|\mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}\|_2^2 - 2\langle \mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}, \gamma \sum_{t=kB}^{(k+1)B-h-1} \nabla f_{s_t}(\mathbf{w}_{kB}) - \nabla f_{s_t}(\widetilde{\mathbf{w}}_{kB})\rangle \\
&+ \gamma^2 \| \sum_{t=kB}^{(k+1)B-h-1} \nabla f_{s_t}(\mathbf{w}_{kB}) - \nabla f_{s_t}(\widetilde{\mathbf{w}}_{kB})\|_2^2 \\
=&\|\mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}\|_2^2 - 2\langle \mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}, \gamma \sum_{t=kB}^{(k+1)B-h-1} \nabla f_{s_t}(\mathbf{w}_{kB}) - \nabla f_{s_t}(\widetilde{\mathbf{w}}_{kB})\rangle \\
&+ \gamma^2 \sum_{t=kB}^{(k+1)B-h-1} \|\nabla f_{s_t}(\mathbf{w}_{kB}) - \nabla f_{s_t}(\widetilde{\mathbf{w}}_{kB})\|_2^2 \\
&+ 2\gamma^2 \sum_{i=kB}^{(k+1)B-h-1} \sum_{j=i+1}^{(k+1)B-h-1} \langle \nabla f_{s_i}(\mathbf{w}_{kB}) - \nabla f_{s_i}(\widetilde{\mathbf{w}}_{kB}), \nabla f_{s_j}(\mathbf{w}_{kB}) - \nabla f_{s_j}(\widetilde{\mathbf{w}}_{kB})\rangle.
\end{aligned} \tag{2.22}$$

We denote the sequence of indices selected by the mini-batch SGD algorithm up to the t-th sampled data point by A_t, i.e., $A_t = \{s_0, \dots, s_{t-1}\}$. In the following steps, we condition on A_{kB} and the event that $H_{k+1} = h$, and take expectation over the randomness of the SGD algorithm in the $(k+1)$-th parallel iteration and the random choice of I.

We consider each term in equation (2.22). For the term $\|\nabla f_{s_t}(\mathbf{w}_{kB}) - \nabla f_{s_t}(\widetilde{\mathbf{w}}_{kB})\|_2^2$, conditioned on the event that $\mathbf{z}_{s_t} = \widetilde{\mathbf{z}}_{s_t}$, we know that s_t is uniformly distributed in $[n] \setminus \{I\}$. Since I is uniformly distributed in $[n]$, we know that the marginal distribution of s_t is uniform on the set $[n]$. We have

$$\mathbb{E}_{I,A|H_{k+1},A_{kB},\Gamma}[\|\nabla f_{s_t}(\mathbf{w}_{kB}) - \nabla f_{s_t}(\widetilde{\mathbf{w}}_{kB})\|_2^2] = \overline{M}^2(\mathbf{w}_{kB}, \widetilde{\mathbf{w}}_{kB}).$$

Then, we find the conditional expectation of $\langle \nabla f_{s_i}(\mathbf{w}_{kB}) - \nabla f_{s_i}(\widetilde{\mathbf{w}}_{kB}), \nabla f_{s_j}(\mathbf{w}_{kB}) - \nabla f_{s_j}(\widetilde{\mathbf{w}}_{kB})\rangle$. The following lemma does precisely this.

Proposition 2.9. *For any i, j such that $kB \le i, j \le (k+1)B - h - 1$ and $i \ne j$, we have*

$$\begin{aligned}
&\mathbb{E}_{I,A|H_{k+1},A_{kB},\Gamma}[\langle \nabla f_{s_i}(\mathbf{w}_{kB}) - \nabla f_{s_i}(\widetilde{\mathbf{w}}_{kB}), \nabla f_{s_j}(\mathbf{w}_{kB}) - \nabla f_{s_j}(\widetilde{\mathbf{w}}_{kB})\rangle] \\
=&\frac{1}{(n-1)^2}\overline{M}^2(\mathbf{w}_{kB}, \widetilde{\mathbf{w}}_{kB}) + \frac{n(n-2)}{(n-1)^2}\overline{G}(\mathbf{w}_{kB}, \widetilde{\mathbf{w}}_{kB}).
\end{aligned} \tag{2.23}$$

We prove Proposition 2.9 in Section 2.8. Combining this lemma with the result of equation (2.22), we have

$$\begin{aligned}
&\mathbb{E}_{I,A|H_{k+1},A_{kB},\Gamma}[\|\mathbf{w}_{(k+1)B-h} - \widetilde{\mathbf{w}}_{(k+1)B-h}\|_2^2] \\
=&\|\mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}\|_2^2 - 2\mathbb{E}_{I,A|H_{k+1},A_{kB},\Gamma}[\langle \mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}, \gamma \sum_{t=kB}^{(k+1)B-h-1} \nabla f_{s_t}(\mathbf{w}_{kB}) - \nabla f_{s_t}(\widetilde{\mathbf{w}}_{kB})\rangle] \\
&+ \gamma^2 (B-h)\overline{M}^2(\mathbf{w}_{kB}, \widetilde{\mathbf{w}}_{kB}) \\
&+ \gamma^2 (B-h)(B-h-1)\left[\frac{1}{(n-1)^2}\overline{M}^2(\mathbf{w}_{kB}, \widetilde{\mathbf{w}}_{kB}) + \frac{n(n-2)}{(n-1)^2}\overline{G}(\mathbf{w}_{kB}, \widetilde{\mathbf{w}}_{kB})\right] \\
\le&\|\mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}\|_2^2 - 2\mathbb{E}_{I,A|H_{k+1},A_{kB},\Gamma}[\langle \mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}, \gamma \sum_{t=kB}^{(k+1)B-h-1} \nabla f_{s_t}(\mathbf{w}_{kB}) - \nabla f_{s_t}(\widetilde{\mathbf{w}}_{kB})\rangle] \\
&+ \gamma^2 (B-h)\overline{M}^2(\mathbf{w}_{kB}, \widetilde{\mathbf{w}}_{kB}) \\
&+ \gamma^2 (B-h)\left[\frac{1}{n-1}\overline{M}^2(\mathbf{w}_{kB}, \widetilde{\mathbf{w}}_{kB}) + (B-1)\frac{\overline{M}^2(\mathbf{w}_{kB}, \widetilde{\mathbf{w}}_{kB})}{\overline{B}_D(\mathbf{w}_{kB}, \widetilde{\mathbf{w}}_{kB})}\right] \\
\le&\|\mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}\|_2^2 - 2\gamma \sum_{t=kB}^{(k+1)B-h-1} \mathbb{E}_{I,A|H_{k+1},A_{kB},\Gamma}[\langle \mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}, \nabla f_{s_t}(\mathbf{w}_{kB}) - \nabla f_{s_t}(\widetilde{\mathbf{w}}_{kB})\rangle] \\
&+ \gamma^2 (B-h)(1 + \frac{1}{n-1} + \frac{B-1}{\overline{B}_D})\overline{M}^2(\mathbf{w}_{kB}, \widetilde{\mathbf{w}}_{kB}).
\end{aligned} \tag{2.24}$$

By the co-coercive property of convex and smooth functions, we know that

$$\langle \mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}, \nabla f_{s_t}(\mathbf{w}_{kB}) - \nabla f_{s_t}(\widetilde{\mathbf{w}}_{kB})\rangle \ge \frac{1}{\beta}\|\nabla f_{s_t}(\mathbf{w}_{kB}) - \nabla f_{s_t}(\widetilde{\mathbf{w}}_{kB})\|_2^2.$$

We thus obtain

$$\begin{aligned}
&\mathbb{E}_{I,A|H_{k+1},A_{kB},\Gamma}[\|\mathbf{w}_{(k+1)B-h} - \widetilde{\mathbf{w}}_{(k+1)B-h}\|_2^2] \\
\le&\|\mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}\|_2^2 - (2\frac{\gamma}{\beta} - \gamma^2(1 + \frac{1}{n-1} + \frac{B-1}{\overline{B}_D}))(B-h)\overline{M}^2(\mathbf{w}_{kB}, \widetilde{\mathbf{w}}_{kB}).
\end{aligned} \tag{2.25}$$

Since we condition on the event Γ, we have that γ obeys the relation in equation (2.18). Consequently,

$$\mathbb{E}_{I,A|H_{k+1},A_{kB},\Gamma}[\|\mathbf{w}_{(k+1)B-h} - \widetilde{\mathbf{w}}_{(k+1)B-h}\|_2^2] \le \|\mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}\|_2^2.$$

Then by Jensen's inequality, we have

$$\mathbb{E}_{I,A|H_{k+1},A_{kB},\Gamma}[\|\mathbf{w}_{(k+1)B-h} - \widetilde{\mathbf{w}}_{(k+1)B-h}\|_2] \le \|\mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}\|_2. \tag{2.26}$$

For the last h terms, since the loss functions are all L-Lipschitz, we obtain

$$\|\mathbf{w}_{(k+1)B} - \widetilde{\mathbf{w}}_{(k+1)B}\|_2 \leq \|\mathbf{w}_{(k+1)B-h} - \widetilde{\mathbf{w}}_{(k+1)B-h}\|_2 + 2\gamma L h. \tag{2.27}$$

Then, combining with equation (2.26), we have

$$\mathbb{E}_{I,A|H_{k+1},A_{kB},\Gamma}[\|\mathbf{w}_{(k+1)B} - \widetilde{\mathbf{w}}_{(k+1)B}\|_2] \leq \|\mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}\|_2 + 2\gamma L h. \tag{2.28}$$

Recall that H_{k+1} is a binomial random variable. Taking expectation over H_{k+1} yields

$$\mathbb{E}_{I,A|A_{kB},\Gamma}[\|\mathbf{w}_{(k+1)B} - \widetilde{\mathbf{w}}_{(k+1)B}\|_2] \leq \|\mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}\|_2 + 2\gamma L \frac{B}{n}.$$

Then we take expectation over the randomness of the first k parallel iterations and obtain

$$\mathbb{E}_{I,A|\Gamma}[\|\mathbf{w}_{(k+1)B} - \widetilde{\mathbf{w}}_{(k+1)B}\|_2] \leq \mathbb{E}_{I,A|\Gamma}[\|\mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}\|_2] + 2\gamma L \frac{B}{n}. \tag{2.29}$$

Summing up (2.29) for $k = 0, 1, \dots, \frac{T}{B} - 1$ and taking expectation over the data sets, we have

$$\mathbb{E}_{I,A|\Gamma}[\|\mathbf{w}_T - \widetilde{\mathbf{w}}_T\|_2] \leq 2\gamma L \frac{T}{n}. \tag{2.30}$$

Combining equations (2.20) and (2.30), we complete the proof of Proposition 2.7, *i.e.,* when the event Γ occurs, we have

$$|\epsilon_{\text{stab}}(\mathcal{S}, \mathcal{S}')| \leq L\mathbb{E}_{I,A|\Gamma}\left[\|\mathbf{w}_T - \widetilde{\mathbf{w}}_T\|_2\right] \leq 2\gamma L^2 \frac{T}{n}. \tag{2.31}$$

To prove Theorem 2.9, we notice that when Γ does not occur, we simply have

$$|\epsilon_{\text{stab}}(\mathcal{S}, \mathcal{S}')| \leq L\mathbb{E}_{I,A|\bar{\Gamma}}[\|\mathbf{w}_T - \widetilde{\mathbf{w}}_T\|_2] \leq 2\gamma L^2 T. \tag{2.32}$$

Using equations (2.31) and (2.32) along with the definition of η, we obtain

$$\begin{aligned}\epsilon_{\text{gen}} &\leq \mathbb{E}_{\mathcal{S},\mathcal{S}'|\Gamma}\left[|\epsilon_{\text{stab}}(\mathcal{S}, \mathcal{S}')|\right]\mathbb{P}\{\Gamma\} + \mathbb{E}_{\mathcal{S},\mathcal{S}'|\bar{\Gamma}}\left[|\epsilon_{\text{stab}}(\mathcal{S}, \mathcal{S}')|\right]\mathbb{P}\{\bar{\Gamma}\} \\ &\leq 2\gamma L^2 \frac{T}{n}(1-\eta) + 2\gamma L^2 T\eta,\end{aligned}$$

which completes the proof. □

Proof of Proposition 2.8 and Theorem 2.10

The proof of Proposition 2.8 follows an argument similar to the proof of Proposition 2.7. We define the analogous event Γ to signal that the step size is "good", according to equation (2.17); note that this is slightly different from convex risk functions:

$$\Gamma = \left\{\gamma \leq \frac{2}{(\beta+\lambda)(1 + \frac{1}{n-1} + \frac{B-1}{\overline{B}_D})}\right\} = \left\{\overline{B}_D \geq \frac{B-1}{\frac{2}{\gamma(\beta+\lambda)} - 1 - \frac{1}{n-1}}\right\}. \tag{2.33}$$

Recall the definition of η:

$$\eta = \mathbb{P}\left\{ \inf_{\mathbf{w}\neq\mathbf{w}'} \overline{B}_D(\mathbf{w},\mathbf{w}') < \frac{B-1}{\frac{2}{\gamma(\beta+\lambda)} - 1 - \frac{1}{n-1}\mathbb{1}_{B>1}} \right\}, \tag{2.34}$$

we know that $\eta = \mathbb{P}\{\bar{\Gamma}\}$. To prove Proposition 2.8, our goal is still to bound

$$\mathbb{E}_{I,A|\Gamma}\left[\|\mathbf{w}_T - \widetilde{\mathbf{w}}_T\|_2\right].$$

Since the result in (2.24) still holds for strongly convex functions, we have

$$\begin{aligned} &\mathbb{E}_{I,A|H_{k+1},A_{kB},\Gamma}[\|\mathbf{w}_{(k+1)B-h} - \widetilde{\mathbf{w}}_{(k+1)B-h}\|_2^2] \\ \le &\|\mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}\|_2^2 - 2\gamma \sum_{t=kB}^{(k+1)B-h-1} \mathbb{E}_{I,A|H_{k+1},A_{kB},\Gamma}[\langle \mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}, \nabla f_{s_t}(\mathbf{w}_{kB}) - \nabla f_{s_t}(\widetilde{\mathbf{w}}_{kB})\rangle] \\ &+ \gamma^2(B-h)(1+\frac{1}{n-1}+\frac{B-1}{\overline{B}_D})\overline{M}^2(\mathbf{w}_{kB},\widetilde{\mathbf{w}}_{kB}), \end{aligned} \tag{2.35}$$

where H_{k+1} is defined in the same way as in the proof of Proposition 2.7. For strongly convex functions, we have the following co-coercive property:

$$\begin{aligned} &\langle \mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}, \nabla f_{s_t}(\mathbf{w}_{kB}) - \nabla f_{s_t}(\widetilde{\mathbf{w}}_{kB})\rangle \\ \ge &\frac{\beta\lambda}{\beta+\lambda}\|\mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}\|_2^2 + \frac{1}{\beta+\lambda}\|\nabla f_{s_t}(\mathbf{w}_{kB}) - \nabla f_{s_t}(\widetilde{\mathbf{w}}_{kB})\|_2^2, \end{aligned}$$

which gives us

$$\begin{aligned} &\mathbb{E}_{I,A|H_{k+1},A_{kB},\Gamma}[\|\mathbf{w}_{(k+1)B-h} - \widetilde{\mathbf{w}}_{(k+1)B-h}\|_2^2] \\ \le &\left(1 - 2\gamma(B-h)\frac{\beta\lambda}{\beta+\lambda}\right)\|\mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}\|_2^2 \\ &- \gamma(B-h)\left[\frac{2}{\beta+\lambda} - \gamma(1+\frac{1}{n-1}+\frac{B-1}{\overline{B}_D})\right]\overline{M}^2(\mathbf{w}_{kB},\widetilde{\mathbf{w}}_{kB}). \end{aligned} \tag{2.36}$$

Since we only consider the regime where $B \le \frac{1}{2\gamma\lambda}$, one can check that $1 - 2\gamma(B-h)\frac{\beta\lambda}{\beta+\lambda} > 0$ for any $h = 0, \dots, B$. Conditioned on the data sets and the event Γ, we have

$$\mathbb{E}_{I,A|H_{k+1},A_{kB},\Gamma}[\|\mathbf{w}_{(k+1)B-h} - \widetilde{\mathbf{w}}_{(k+1)B-h}\|_2^2] \le \left(1 - 2\gamma B\frac{\beta\lambda}{\beta+\lambda}\right)\|\mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}\|_2^2. \tag{2.37}$$

With Jensen's inequality and the fact that $\sqrt{1-x} \le 1 - \frac{x}{2}$ for any $x \in [0,1]$, we have

$$\mathbb{E}_{I,A|H_{k+1},A_{kB},\Gamma}[\|\mathbf{w}_{(k+1)B-h} - \widetilde{\mathbf{w}}_{(k+1)B-h}\|_2] \le \left(1 - \gamma B\frac{\beta\lambda}{\beta+\lambda}\right)\|\mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}\|_2. \tag{2.38}$$

For the last h terms, we have

$$\mathbb{E}_{I,A|H_{k+1},A_{kB},\Gamma}[\|\mathbf{w}_{(k+1)B} - \widetilde{\mathbf{w}}_{(k+1)B}\|_2] \leq \|\mathbf{w}_{(k+1)B-h} - \widetilde{\mathbf{w}}_{(k+1)B-h}\|_2 + 2\gamma L h. \tag{2.39}$$

Combined with equation (2.38), we obtain

$$\mathbb{E}_{I,A|H_{k+1},A_{kB},\Gamma}[\|\mathbf{w}_{(k+1)B} - \widetilde{\mathbf{w}}_{(k+1)B}\|_2] \leq \left(1 - \gamma B \frac{\beta\lambda}{\beta+\lambda}\right) \|\mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}\|_2 + 2\gamma L h,$$

and by taking expectation over h we have

$$\mathbb{E}_{I,A|A_{kB},\Gamma}[\|\mathbf{w}_{(k+1)B} - \widetilde{\mathbf{w}}_{(k+1)B}\|_2] \leq \left(1 - \gamma B \frac{\beta\lambda}{\beta+\lambda}\right) \|\mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}\|_2 + 2\gamma L \frac{B}{n}.$$

Taking expectation over A_{kB} yields

$$\mathbb{E}_{I,A|\Gamma}[\|\mathbf{w}_{(k+1)B} - \widetilde{\mathbf{w}}_{(k+1)B}\|_2] \leq \left(1 - \gamma B \frac{\beta\lambda}{\beta+\lambda}\right) \mathbb{E}_{I,A|\Gamma}[\|\mathbf{w}_{kB} - \widetilde{\mathbf{w}}_{kB}\|_2] + 2\gamma L \frac{B}{n}. \tag{2.40}$$

Iterating equation (2.40) yields

$$\mathbb{E}_{I,A|\Gamma}[\|\mathbf{w}_T - \widetilde{\mathbf{w}}_T\|_2] \leq \frac{4L}{\lambda n}. \tag{2.41}$$

Combining equations (2.20) and (2.41), we prove Proposition 2.8, *i.e.,* when Γ occurs,

$$|\epsilon_{\text{stab}}(\mathcal{S}, \mathcal{S}')| \leq L \mathbb{E}_{I,A|\Gamma}\left[\|\mathbf{w}_T - \widetilde{\mathbf{w}}_T\|_2\right] \leq \frac{4L^2}{\lambda n}. \tag{2.42}$$

To prove Theorem 2.10, we notice the fact that, when Γ does not occur, we simply have

$$|\epsilon_{\text{stab}}(\mathcal{S}, \mathcal{S}')| \leq L \mathbb{E}_{I,A|\bar{\Gamma}}[\|\mathbf{w}_T - \widetilde{\mathbf{w}}_T\|_2] \leq 2\gamma L^2 T. \tag{2.43}$$

Combining equations (2.42) and (2.43) with the definition of η then yields

$$\begin{aligned}\epsilon_{\text{gen}} &\leq \mathbb{E}_{\mathcal{S},\mathcal{S}'|\Gamma}\left[|\epsilon_{\text{stab}}(\mathcal{S}, \mathcal{S}')|\right] \mathbb{P}\{\Gamma\} + \mathbb{E}_{\mathcal{S},\mathcal{S}'|\bar{\Gamma}}\left[|\epsilon_{\text{stab}}(\mathcal{S}, \mathcal{S}')|\right] \mathbb{P}\{\bar{\Gamma}\} \\ &\leq \frac{4L^2}{\lambda n}(1-\eta) + 2\gamma L^2 T \eta,\end{aligned}$$

which completes the proof. □

Proof of Proposition 2.9

One can interpret $\overline{M}^2(\mathbf{w}, \mathbf{w}')$ and $\overline{G}(\mathbf{w}, \mathbf{w}')$ as follows. Let $\mathcal{P}_1$ be a distribution on $[n] \times [n]$ with PMF

$$p_1(u, v) = \frac{1}{n} \mathbb{1}_{u=v}, \tag{2.44}$$

and $\mathcal{P}_2$ be the uniform distribution on $[n] \times [n]$, i.e.,

$$p_2(u, v) = \frac{1}{n^2} \tag{2.45}$$

for all $(u, v) \in [n] \times [n]$. Then, we know that

$$\overline{M}^2(\mathbf{w}, \mathbf{w}') = \mathbb{E}_{(i,j)\sim\mathcal{P}_1}[\langle \nabla f_i(\mathbf{w}) - \nabla f_i(\mathbf{w}'), \nabla f_j(\mathbf{w}) - \nabla f_j(\mathbf{w}')\rangle],$$

and

$$\overline{G}(\mathbf{w}, \mathbf{w}') = \mathbb{E}_{(i,j)\sim\mathcal{P}_2}[\langle \nabla f_i(\mathbf{w}) - \nabla f_i(\mathbf{w}'), \nabla f_j(\mathbf{w}) - \nabla f_j(\mathbf{w}')\rangle].$$

Then we find the joint distribution $\mathcal{P}_3$ of (s_i, s_j) where $kB \le i, j \le (k+1)B - h - 1$ and $i \neq j$. Since $\mathbf{z}_{s_t} = \widetilde{\mathbf{z}}_{s_t}$, we know that $s_t \neq I$ for all $t = kB, \ldots, (k+1)B - h - 1$. Then conditioned on I, (s_i, s_j) is uniformly distributed in $([n] \setminus \{I\}) \times ([n] \setminus \{I\})$. For any $u \in [n]$, we have

$$\begin{aligned} p_3(u, u) = \mathbb{P}\{s_i = u, s_j = u\} &= \frac{1}{n}\sum_{\ell=1}^{n} \mathbb{P}\{s_i = u, s_j = u \mid I = \ell\} \\ &= \frac{1}{n}\sum_{\ell=u} \mathbb{P}\{s_i = u, s_j = u \mid I = \ell\} = \frac{1}{n(n-1)}. \end{aligned}$$

For any $(u, v) \in [n] \times [n]$ such that $u \neq v$, we have

$$\begin{aligned} p_3(u, v) = \mathbb{P}\{s_i = u, s_j = v\} &= \frac{1}{n}\sum_{\ell=1}^{n} \mathbb{P}\{s_i = u, s_j = v \mid I = \ell\} \\ &= \frac{1}{n}\sum_{\ell\neq u,v} \mathbb{P}\{s_i = u, s_j = v \mid I = \ell\} \\ &= \frac{n-2}{n(n-1)^2}. \end{aligned}$$

Then, we know that

$$p_3(u, v) = \frac{1}{(n-1)^2} p_1(u, v) + \frac{n(n-2)}{(n-1)^2} p_2(u, v).$$

Therefore, for any i, j such that $kB \le i, j \le (k+1)B - h - 1$ and $i \neq j$, we have

$$\begin{aligned} &\mathbb{E}_{I,A|H_{k+1},A_{kB},\Gamma}[\langle \nabla f_{s_i}(\mathbf{w}_{kB}) - \nabla f_{s_i}(\widetilde{\mathbf{w}}_{kB}), \nabla f_{s_j}(\mathbf{w}_{kB}) - \nabla f_{s_j}(\widetilde{\mathbf{w}}_{kB})\rangle] \\ =&\mathbb{E}_{(s_i,s_j)\sim\mathcal{P}_3}[\langle \nabla f_{s_i}(\mathbf{w}_{kB}) - \nabla f_{s_i}(\widetilde{\mathbf{w}}_{kB}), \nabla f_{s_j}(\mathbf{w}_{kB}) - \nabla f_{s_j}(\widetilde{\mathbf{w}}_{kB})\rangle] \\ =&\frac{1}{(n-1)^2}\overline{M}^2(\mathbf{w}_{kB}, \widetilde{\mathbf{w}}_{kB}) + \frac{n(n-2)}{(n-1)^2}\overline{G}(\mathbf{w}_{kB}, \widetilde{\mathbf{w}}_{kB}). \end{aligned}$$

□

Examples of Differential Gradient Diversity and Diversity-inducing Mechanisms

We now prove auxiliary results for differential gradient diversity that are stated in this chapter.

Generalized Linear Functions We can show that for generalized linear functions, the lower bound in Theorem 2.1 still holds, *i.e.,* , for any $\mathbf{w}, \mathbf{w}' \in \mathcal{W}$, $\mathbf{w} \neq \mathbf{w}'$, we have

$$\overline{B}_D(\mathbf{w}, \mathbf{w}') \geq \frac{\min_{i=1,\ldots,n} \|\mathbf{x}_i\|_2^2}{\sigma_{\max}^2(\mathbf{X})}.$$

To see this, one can simply replace $\nabla f_i(\mathbf{w})$ with $\nabla f_i(\mathbf{w}) - \nabla f_i(\mathbf{w}')$ in the proof of Theorem 2.7, and define $a_i = \ell_i'(\mathbf{x}_i^{\mathrm{T}}\mathbf{w}) - \ell_i'(\mathbf{x}_i^{\mathrm{T}}\mathbf{w}')$. The same arguments in the proof of Theorem 2.7 still go through. Consequently, for i.i.d. σ-sub-Gaussian features, we have $\overline{B}_D(\mathbf{w}, \mathbf{w}') \geq c_1 d$ $\forall\ \mathbf{w}, \mathbf{w}' \in \mathcal{W}$ with probability at least $1 - c_2 n e^{-c_3 d}$; and for Rademacher entries, we have $\overline{B}_D(\mathbf{w}, \mathbf{w}') \geq c_4 d\ \forall\ \mathbf{w}, \mathbf{w}' \in \mathcal{W}$ with probability greater than $1 - c_5 e^{-c_6 n}$.

Sparse Conflicts The result for gradient diversity still holds for $\overline{B}_D(\mathbf{w}, \mathbf{w}')$, *i.e.,* for all $\mathbf{w}, \mathbf{w}' \in \mathcal{W}$, $\overline{B}_D(\mathbf{w}, \mathbf{w}') \geq n/(\rho + 1)$, where ρ is the maximum degree of all the vertices in the conflict graph G. To see this, one should notice that the support of $\nabla f_i(\mathbf{w})$ only depends on the data point, instead of the model parameter, and thus, in general, $\nabla f_i(\mathbf{w})$ and $\nabla f_i(\mathbf{w}) - \nabla f_i(\mathbf{w}')$ have the same support. Then, one can simply replace $\nabla f_i(\mathbf{w})$ with $\nabla f_i(\mathbf{w}) - \nabla f_i(\mathbf{w}')$ in the proof of Theorem 2.7 and the same arguments still go through.

DropConnect When we analyze the stability of mini-batch SGD, we apply the *same* algorithm to two different samples $\mathcal{S}$ and $\mathcal{S}^{(I)}$ that only differ at one data point. Since the algorithm is the same, the random matrices $\mathbf{D}_1, \ldots, \mathbf{D}_n$ are also the same in the two instances. Therefore, one can replace $\nabla f_i(\mathbf{w})$ with $\nabla f_i(\mathbf{w}) - \nabla f_i(\mathbf{w}')$, and the same arguments still work. Then, we know that when $\overline{B}_D(\mathbf{w}, \mathbf{w}') \leq n$, we have $\overline{B}_D^{\mathsf{drop}}(\mathbf{w}, \mathbf{w}') \geq \overline{B}_D(\mathbf{w}, \mathbf{w}')$, and when $\overline{B}_D(\mathbf{w}, \mathbf{w}') > n$, we have $\overline{B}_D^{\mathsf{drop}}(\mathbf{w}, \mathbf{w}') > n$.

Stochastic Gradient Langevin Dynamics For SGLD, we can make similar arguments as in dropout, since the additive noise vectors $\xi_1, \ldots, \xi_n$ are the same for the two instances. One can then show that when $\overline{B}_D(\mathbf{w}, \mathbf{w}') \leq n$, we have $\overline{B}_D^{\mathsf{sgld}}(\mathbf{w}, \mathbf{w}') \geq \overline{B}_D(\mathbf{w}, \mathbf{w}')$, and when $\overline{B}_D(\mathbf{w}, \mathbf{w}') > n$, we have $\overline{B}_D^{\mathsf{sgld}}(\mathbf{w}, \mathbf{w}') > n$. $\square$

Chapter 3

Statistical Rates in Byzantine-Robust Distributed Learning

In large-scale distributed learning, security issues have become increasingly important. Particularly in a decentralized environment, some computing units may behave abnormally, or even exhibit Byzantine failures—arbitrary and potentially adversarial behavior. In this chapter, we develop distributed learning algorithms that are provably robust against such failures, with a focus on achieving optimal statistical performance. A main result of this work is a sharp analysis of two robust distributed gradient descent algorithms based on median and trimmed mean operations, respectively. We prove statistical error rates for three kinds of population loss functions: strongly convex, non-strongly convex, and smooth non-convex. In particular, these algorithms are shown to achieve order-optimal statistical error rates for strongly convex losses. To achieve better communication efficiency, we further propose a median-based distributed algorithm that is provably robust, and uses only one communication round. For strongly convex quadratic loss, we show that this algorithm achieves the same optimal error rate as the robust distributed gradient descent algorithms.

3.1 Introduction

Many tasks in computer vision, natural language processing and recommendation systems require learning complex prediction rules from large datasets. As the scale of the datasets in these learning tasks continues to grow, it is crucial to utilize the power of distributed computing and storage. In such large-scale distributed systems, robustness and security issues have become a major concern. In particular, individual computing units—known as worker machines—may exhibit abnormal behavior due to crashes, faulty hardware, stalled computation or unreliable communication channels. Security issues are only exacerbated in the so-called *Federated Learning* setting, a modern distributed learning paradigm that is more decentralized, and that uses the data owners' devices (such as mobile phones and personal computers) as worker machines [135, 109]. Such machines are often more unpredictable, and in particular may be susceptible to malicious and coordinated attacks.

Due to the inherent unpredictability of this abnormal (sometimes adversarial) behavior, it is typically modeled as *Byzantine failure* [114], meaning that some worker machines may behave completely arbitrarily and can send any message to the master machine that maintains and updates an estimate of the parameter vector to be learned. Byzantine failures can incur major degradation in learning performance. It is well-known that standard learning algorithms based on naive aggregation of the workers' messages can be arbitrarily skewed by a single Byzantine-faulty machine. Even when the messages from Byzantine machines take only moderate values—and hence are difficult to detect—and when the number of such machines is small, the performance loss can still be significant. We demonstrate such an example in our experiments in Section 3.7.

In this chapter, we aim to develop distributed statistical learning algorithms that are provably robust against Byzantine failures. While this objective is considered in a few recent works [68, 23, 41], a fundamental problem remains poorly understood, namely the *optimal statistical performance* of a robust learning algorithm. A learning scheme in which the master machine always outputs zero regardless of the workers' messages is certainly not affected by Byzantine failures, but it will not return anything statistically useful either. On the other hand, many standard distributed algorithms that achieve good statistical performance in the absence of Byzantine failures, become completely unreliable otherwise. Therefore, a main goal of this work is to understand the following questions: what is the best achievable statistical performance while being Byzantine-robust, and what algorithms achieve this performance?

To formalize this question, we consider a standard statistical setting of empirical risk minimization (ERM). Here nm data points are sampled independently from some distribution and distributed evenly among m machines, αm of which are Byzantine. The goal is to learn a parametric model by minimizing some loss function defined by the data. In this statistical setting, one expects that the error in learning the parameter, measured in an appropriate metric, should decrease when the amount of data nm becomes larger and the fraction of Byzantine machines α becomes smaller. In fact, we can show that, at least for strongly convex problems, no algorithm can achieve an error lower than

$$\widetilde{\Omega}\left(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}}\right) = \widetilde{\Omega}\left(\frac{1}{\sqrt{n}}\Big(\alpha + \frac{1}{\sqrt{m}}\Big)\right),$$

regardless of communication costs;[1] see Observation 3.1 in Section 3.6. Intuitively, the above error rate is the optimal rate that one should target, as $\frac{1}{\sqrt{n}}$ is the effective standard deviation for each machine with n data points, α is the bias effect of Byzantine machines, and $\frac{1}{\sqrt{m}}$ is the averaging effect of m normal machines. When there are no or few Byzantine machines, we see the usual scaling $\frac{1}{\sqrt{mn}}$ with the total number of data points; when some machines are Byzantine, their influence remains bounded, and moreover is proportional to α. If an

[1]Throughout the chapter, unless otherwise stated, $\Omega(\cdot)$ and $\mathcal{O}(\cdot)$ hide universal multiplicative constants; $\widetilde{\Omega}(\cdot)$ and $\widetilde{\mathcal{O}}(\cdot)$ further hide terms that are independent of α, n, m or logarithmic in n, m.

algorithm is guaranteed to attain this bound, we are assured that we do not sacrifice the quality of learning when trying to guard against Byzantine failures—we pay a price that is unavoidable, but otherwise we achieve the best possible statistical accuracy in the presence of Byzantine failures.

Another important consideration for us is *communication efficiency*. As communication between machines is costly, one cannot simply send all data to the master machine. This constraint precludes direct application of standard robust learning algorithms (such as M-estimators [90]), which assume access to all data. Instead, a desirable algorithm should involve a small number of communication rounds as well as a small amount of data communicated per round. We consider a setting where in each round a worker or master machine can only communicate a vector of size $\mathcal{O}(d)$, where d is the dimension of the parameter to be learned. In this case, the total communication cost is proportional to the number of communication rounds.

To summarize, we aim to develop distributed learning algorithms that simultaneously achieve two objectives:

- **Statistical optimality:** attain an $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}})$ rate.
- **Communication efficiency:** $\mathcal{O}(d)$ communication per round, with as few rounds as possible.

To the best of our knowledge, *no existing algorithm achieves these two goals simultaneously.* In particular, previous robust algorithms either have unclear or sub-optimal statistical guarantees, or incur a high communication cost and hence are not applicable in a distributed setting—we discuss related work in more detail in Section 3.2.

Our Contributions

We propose two robust distributed gradient descent (GD) algorithms. The first one is based on coordinate-wise median, and the other is based on coordinate-wise trimmed mean. We establish their statistical error rates for strongly convex, non-strongly convex, and non-convex *population* loss functions. For strongly convex losses, we show that these algorithms achieve order-optimal statistical rates under mild conditions. We further propose a median-based robust algorithm that only requires one communication round, and show that it also achieves the optimal rate for strongly convex quadratic losses. The statistical error rates of these three algorithms are summarized as follows.

- **Median-based GD:** $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}} + \frac{1}{n})$, order-optimal for strongly convex loss if $n \gtrsim m$.
- **Trimmed-mean-based GD:** $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}})$, order-optimal for strongly convex loss.
- **Median-based one-round algorithm:** $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}} + \frac{1}{n})$, order-optimal for strongly convex quadratic loss if $n \gtrsim m$.

A major technical challenge in our statistical setting here is as follows: the nm data points are sampled once and fixed, and each worker machine has access to a fixed set of data throughout the learning process. This creates complicated probabilistic dependency across the iterations of the algorithms. Worse yet, the Byzantine machines, which have complete knowledge of the data and the learning algorithm used, may create further unspecified probabilistic dependency. We overcome this difficulty by proving certain *uniform* bounds via careful covering arguments. Furthermore, for the analysis of median-based algorithms, we cannot simply adapt standard techniques (such as those in Minsker et al. [138]), which can only show that the output of the master machine is as accurate as that of *one* normal machine, leading to a sub-optimal $\mathcal{O}(\frac{1}{\sqrt{n}})$ rate even without Byzantine failures ($\alpha = 0$). Instead, we make use of a more delicate argument based on normal approximation and Berry-Esseen-type inequalities, which allows us to achieve the better $\mathcal{O}(\frac{1}{mn})$ rates when α is small while being robust for a nonzero α.

Above we have omitted the dependence on the parameter dimension d.; see our main theorems for the precise results. In some settings the rates in these results may not have the optimal dependence on d. Understanding the fundamental limits of robust distributed learning in high dimensions, as well as developing algorithms with optimal dimension dependence, is an interesting and important future direction.

Notation

We denote vectors by boldface lowercase letters such as $\mathbf{w}$, and the elements in the vector are denoted by italics letters with subscripts, such as w_k. Matrices are denoted by boldface uppercase letters such as $\mathbf{H}$. For any positive integer N, we denote the set $\{1, 2, \dots, N\}$ by $[N]$. For vectors, we denote the ℓ_2 norm and ℓ_∞ norm by $\|\cdot\|_2$ and $\|\cdot\|_\infty$, respectively. For matrices, we denote the operator norm and the Frobenius norm by $\|\cdot\|_2$ and $\|\cdot\|_F$, respectively. We denote by $\Phi(\cdot)$ the cumulative distribution function (CDF) of the standard Gaussian distribution. For any differentiable function $f : \mathbb{R}^d \to \mathbb{R}$, we denote its partial derivative with respect to the k-th argument by $\partial_k f$.

3.2 Related Work

Outlier-robust estimation in non-distributed settings is a classical topic in statistics [90]. Particularly relevant to us is the so-called *median-of-means* method, in which one partitions the data into m subsets, computes an estimate from each subset, and finally takes the median of these m estimates. This idea is studied in Nemirovskii, Yudin, and Dawson [143], Jerrum, Valiant, and Vazirani [93], Alon, Matias, and Szegedy [8], Lerasle and Oliveira [122], and Minsker et al. [138], and has been applied to bandit and least square regression problems [27, 129, 103] as well as problems involving heavy-tailed distributions [87, 130]. In a very recent work, Minsker and Strawn [139] provide a new analysis of median-of-means using a normal approximation. We borrow some techniques from this paper, but need to address a significant harder problem: 1) we deal with the Byzantine setting with arbitrary/adversarial outliers, which is not considered in their paper; 2) we study iterative algorithms for general multi-dimensional problems with convex and non-convex losses, while they mainly focus on one-shot algorithms for mean-estimation-type problems.

The median-of-means method is used in the context of Byzantine-robust distributed learning in two recent papers. In particular, the work of Feng, Xu, and Mannor [68] considers a simple one-shot application of median-of-means, and only proves a sub-optimal $\widetilde{\mathcal{O}}(\frac{1}{\sqrt{n}})$ error rate as mentioned. The work of Chen, Su, and Xu [41] considers only strongly convex losses, and seeks to circumvent the above issue by grouping the worker machines into mini-batches; however, their rate $\widetilde{\mathcal{O}}(\frac{\sqrt{\alpha}}{\sqrt{n}} + \frac{1}{\sqrt{nm}})$ still falls short of being optimal, and in particular their algorithm fails even when there is only one Byzantine machine in each mini-batch.

Other methods have been proposed for Byzantine-robust distributed learning and optimization; e.g., Su and Vaidya [172, 173]. These works consider optimizing fixed functions and do not provide guarantees on statistical error rates. Most relevant is the work by Blanchard et al. [23], who propose to aggregate the gradients from worker machines using a robust procedure. Their optimization setting—which is at the level of stochastic gradient descent and assumes unlimited, independent access to a strong stochastic gradient oracle—is fundamentally different from ours; in particular, they do not provide a characterization of the statistical errors given a fixed number of data points.

Communication efficiency has been studied extensively in non-Byzantine distributed settings [136]. An important class of algorithms are based on one-round aggregation methods [209, 206, 159]. More sophisticated algorithms have been proposed in order to achieve better accuracy than the one-round approach while maintaining lower communication costs; examples include DANE [165], Disco [208], distributed SVRG [117] and their variants [158, 185]. Developing Byzantine-robust versions of these algorithms is an interesting future direction.

For outlier-robust estimation in non-distributed settings, much progress has been made recently in terms of improved performance in high-dimensional problems [58, 113, 22] as well as developing list-decodable and semi-verified learning schemes when a majority of the data points are adversarial [35]. These results are not directly applicable to our distributed setting with general loss functions, but it is nevertheless an interesting future problem to investigate their potential extension for our problem.

3.3 Problem Setup

In this section, we formally set up our problem and introduce a few concepts key to our the algorithm design and analysis. Suppose that training data points are sampled from some unknown distribution $\mathcal{D}$ on the sample space $\mathcal{Z}$. Let $f(\mathbf{w}; \mathbf{z})$ be a loss function of a parameter vector $\mathbf{w} \in \mathcal{W} \subseteq \mathbb{R}^d$ associated with the data point $\mathbf{z}$, where $\mathcal{W}$ is the parameter space, and $F(\mathbf{w}) := \mathbb{E}_{\mathbf{z}\sim\mathcal{D}}[f(\mathbf{w}; \mathbf{z})]$ is the corresponding population loss function. Our goal is to learn a model defined by the parameter that minimizes the population loss:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}\in\mathcal{W}} F(\mathbf{w}). \tag{3.1}$$

The parameter space $\mathcal{W}$ is assumed to be convex and compact with diameter D, i.e., $\|\mathbf{w} - \mathbf{w}'\|_2 \le D, \forall \mathbf{w}, \mathbf{w}' \in \mathcal{W}$. We consider a distributed computation model with one master machine and m worker machines. Each worker machine stores n data points, each of which is sampled independently from $\mathcal{D}$. Denote by $\mathbf{z}^{i,j}$ the j-th data on the i-th worker machine, and $F_i(\mathbf{w}) := \frac{1}{n}\sum_{j=1}^{n} f(\mathbf{w}; \mathbf{z}^{i,j})$ the empirical risk function for the i-th worker. We assume that an α fraction of the m worker machines are Byzantine, and the remaining $1-\alpha$ fraction are normal. With the notation $[N] := \{1, 2, \ldots, N\}$, we index the set of worker machines by $[m]$, and denote the set of Byzantine machines by $\mathcal{B} \subset [m]$ (thus $|\mathcal{B}| = \alpha m$). The master machine communicates with the worker machines using some predefined protocol. The Byzantine machines need not obey this protocol and can send arbitrary messages to the master; in particular, they may have complete knowledge of the system and learning algorithms, and can collude with each other.

We introduce the coordinate-wise median and trimmed mean operations, which serve as building blocks for our algorithm.

Definition 3.1 (Coordinate-wise median). *For vectors $\mathbf{x}^i \in \mathbb{R}^d$, $i \in [m]$, the coordinate-wise median $\widehat{\mathbf{g}} := \mathsf{med}\{\mathbf{x}^i : i \in [m]\}$ is a vector with its k-th coordinate being $g_k = \mathsf{med}\{x^i_k : i \in [m]\}$ for each $k \in [d]$, where med is the usual (one-dimensional) median.*

Definition 3.2 (Coordinate-wise trimmed mean). *For $\beta \in [0, \frac{1}{2})$ and vectors $\mathbf{x}^i \in \mathbb{R}^d$, $i \in [m]$, the coordinate-wise β-trimmed mean $\widehat{\mathbf{g}} := \mathsf{trmean}_\beta\{\mathbf{x}^i : i \in [m]\}$ is a vector with its k-th coordinate being $g_k = \frac{1}{(1-2\beta)m}\sum_{x\in U_k} x$ for each $k \in [d]$. Here U_k is a subset of $\{x^1_k, \ldots, x^m_k\}$ obtained by removing the largest and smallest β fraction of its elements.*

For the analysis, we need several standard definitions concerning random variables and vectors.

Definition 3.3 (Variance of random vectors). *For a random vector $\mathbf{x}$, define its variance as $\mathrm{Var}(\mathbf{x}) := \mathbb{E}[\|\mathbf{x} - \mathbb{E}[\mathbf{x}]\|_2^2]$.*

Definition 3.4 (Absolute skewness). *For a one-dimensional random variable X, define its absolute skewness[2] as $\gamma(X) := \frac{\mathbb{E}[|X-\mathbb{E}[X]|^3]}{\mathrm{Var}(X)^{3/2}}$. For a d-dimensional random vector $\mathbf{x}$, we define its absolute skewness as the vector of the absolute skewness of each coordinate of $\mathbf{x}$, i.e., $\gamma(\mathbf{x}) := [\gamma(x_1)\ \gamma(x_2)\ \cdots\ \gamma(x_d)]^\top$.*

Definition 3.5 (Sub-exponential random variables). *A random variable X with $\mathbb{E}[X] = \mu$ is called v-sub-exponential if $\mathbb{E}[e^{\lambda(X-\mu)}] \le e^{\frac{1}{2}v^2\lambda^2}$, $\forall\ |\lambda| < \frac{1}{v}$.*

[2]Note the difference with the usual skewness $\frac{\mathbb{E}[(X-\mathbb{E}[X])^3]}{\mathrm{Var}(X)^{3/2}}$.

Finally, we need several standard concepts from convex analysis regarding a differentiable function $h(\cdot) : \mathbb{R}^d \to \mathbb{R}$.

Definition 3.6 (Lipschitz). *h is L-Lipschitz if* $|h(\mathbf{w}) - h(\mathbf{w}')| \le L\|\mathbf{w} - \mathbf{w}'\|_2, \forall\ \mathbf{w}, \mathbf{w}'$.

Definition 3.7 (Smoothness). *h is L'-smooth if* $\|\nabla h(\mathbf{w}) - \nabla h(\mathbf{w}')\|_2 \le L'\|\mathbf{w} - \mathbf{w}'\|_2, \forall\ \mathbf{w}, \mathbf{w}'$.

Definition 3.8 (Strong convexity). *h is λ-strongly convex if* $h(\mathbf{w}') \ge h(\mathbf{w}) + \langle \nabla h(\mathbf{w}), \mathbf{w}' - \mathbf{w}\rangle + \frac{\lambda}{2}\|\mathbf{w}' - \mathbf{w}\|_2^2, \forall\ \mathbf{w}, \mathbf{w}'$.

3.4 Robust Distributed Gradient Descent

We describe two robust distributed gradient descent algorithms, one based on coordinate-wise median and the other on trimmed mean. These two algorithms are formally given in Algorithm 1 as Option I and Option II, respectively, where the symbol $*$ represents an arbitrary vector. This algorithm is also illustrated in Figure 3.1.

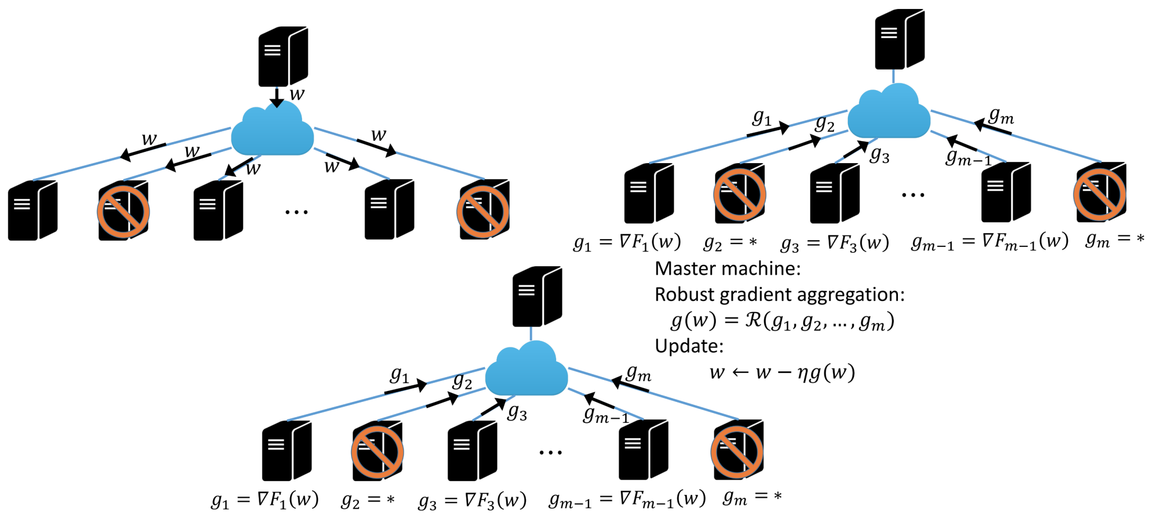

w
w
w
w
w
w
$\dots$
g_1
g_2
g_3
g_{m-1}
g_m
$g_1 = \nabla F_1(w)$ $g_2 = *$ $g_3 = \nabla F_3(w)$ $g_{m-1} = \nabla F_{m-1}(w)$ $g_m = *$
Master machine:
Robust gradient aggregation:
$g(w) = \mathcal{R}(g_1, g_2, \dots, g_m)$
Update:
$w \leftarrow w - \eta g(w)$
g_1
g_2
g_3
g_{m-1}
g_m
$g_1 = \nabla F_1(w)$ $g_2 = *$ $g_3 = \nabla F_3(w)$ $g_{m-1} = \nabla F_{m-1}(w)$ $g_m = *$

Figure 3.1: Byzantine-robust distributed learning algorithm. (i) The master machine sends the model parameters to all the worker machines. (ii) The worker machines send either the gradient (normal machines) or an adversarial vector (Byzantine machines) to the master machine. (iii) The master machine conducts a robust aggregation.

In each parallel iteration of the algorithms, the master machine broadcasts the current model parameter to all worker machines. The normal worker machines compute the gradients of their local loss functions and then send the gradients back to the master machine. The Byzantine machines may send any messages of their choices. The master machine then performs a gradient descent update on the model parameter with step-size η, using either the coordinate-wise median or trimmed mean of the received gradients. The Euclidean projection $\Pi_{\mathcal{W}}(\cdot)$ ensures that the model parameter stays in the parameter space $\mathcal{W}$.

Algorithm 1 Robust Distributed Gradient Descent

```
Require: Initialize parameter vector $\mathbf{w}^0 \in \mathcal{W}$, algorithm parameters $\beta$ (for Option II), $\eta$ and $T$.
  for $t = 0, 1, 2, \dots, T-1$ do
    Master machine: send $\mathbf{w}^t$ to all the worker machines.
    for all $i \in [m]$ do in parallel
      Worker machine i: compute local gradient
          $\widehat{\mathbf{g}}^i(\mathbf{w}^t) \leftarrow \begin{cases} \nabla F_i(\mathbf{w}^t) & \text{normal worker machines,} \\ * & \text{Byzantine machines,} \end{cases}$
      send $\widehat{\mathbf{g}}^i(\mathbf{w}^t)$ to master machine.
    end for
    Master machine: compute aggregate gradient
          $\widehat{\mathbf{g}}(\mathbf{w}^t) \leftarrow \begin{cases} \mathsf{med}\{\widehat{\mathbf{g}}^i(\mathbf{w}^t) : i \in [m]\} & \text{Option I} \\ \mathsf{trmean}_\beta\{\widehat{\mathbf{g}}^i(\mathbf{w}^t) : i \in [m]\} & \text{Option II} \end{cases}$
    update model parameter $\mathbf{w}^{t+1} \leftarrow \Pi_{\mathcal{W}}(\mathbf{w}^t - \eta\widehat{\mathbf{g}}(\mathbf{w}^t))$.
  end for
```

Below we provide statistical guarantees on the error rates of these algorithms, and compare their performance. Throughout we assume that each loss function $f(\mathbf{w};\mathbf{z})$ and the population loss function $F(\mathbf{w})$ are smooth:

Assumption 3.1 (Smoothness of f and F)**.** *For any $\mathbf{z} \in \mathcal{Z}$, the partial derivative of $f(\cdot;\mathbf{z})$ with respect to the k-th coordinate of its first argument, denoted by $\partial_k f(\cdot;\mathbf{z})$, is L_k-Lipschitz for each $k \in [d]$, and the function $f(\cdot;\mathbf{z})$ is L-smooth. Let $\widehat{L} := \sqrt{\sum_{k=1}^d L_k^2}$. Also assume that the population loss function $F(\cdot)$ is L_F-smooth.*

It is easy to see hat $L_F \le L \le \widehat{L}$. When the dimension of $\mathbf{w}$ is high, the quantity $\widehat{L}$ may be large. However, we will soon see that $\widehat{L}$ only appears in the logarithmic factors in our bounds and thus does not have a significant impact.

Guarantees for Median-based Gradient Descent

We first consider our median-based algorithm, namely Algorithm 1 with Option I. We impose the assumptions that the gradient of the loss function f has bounded variance, and each coordinate of the gradient has coordinate-wise bounded absolute skewness:

Assumption 3.2 (Bounded variance of gradient)**.** *For any* $\mathbf{w} \in \mathcal{W}$*,* $\mathrm{Var}(\nabla f(\mathbf{w};\mathbf{z})) \le V^2$*.*

Assumption 3.3 (Bounded skewness of gradient)**.** *For any* $\mathbf{w} \in \mathcal{W}$*,* $\|\gamma(\nabla f(\mathbf{w};\mathbf{z}))\|_\infty \le S$*.*

These assumptions are satisfied in many learning problems with small values of V^2 and S. Below we provide a concrete example in terms of a linear regression problem.

Proposition 3.1. *Suppose that each data point* $\mathbf{z} = (\mathbf{x}, y) \in \mathbb{R}^d \times \mathbb{R}$ *is generated by* $y = \mathbf{x}^\top \mathbf{w}^* + \xi$ *with some* $\mathbf{w}^* \in \mathcal{W}$*. Assume that the elements of* $\mathbf{x}$ *are independent and uniformly distributed in* $\{-1, 1\}$*, and that the noise* $\xi \sim \mathcal{N}(0, \sigma^2)$ *is independent of* $\mathbf{x}$*. With the quadratic loss function* $f(\mathbf{w}; \mathbf{x}, y) = \frac{1}{2}(y - \mathbf{x}^T\mathbf{w})^2$*, we have* $\mathrm{Var}(\nabla f(\mathbf{w}; \mathbf{x}, y)) = (d-1)\|\mathbf{w} - \mathbf{w}^*\|_2^2 + d\sigma^2$*, and* $\|\gamma(\nabla f(\mathbf{w}; \mathbf{x}, y))\|_\infty \le 480$*.*

In this example, the upper bound V^2 on $\mathrm{Var}(\nabla f(\mathbf{w}; \mathbf{x}, y))$ depends on dimension d and the diameter of the parameter space. If the diameter is a constant, we have $V = \mathcal{O}(\sqrt{d})$. Moreover, the gradient skewness is bounded by a universal constant S regardless of the size of the parameter space. In Section 3.9, we provide another example showing that when the features in $\mathbf{x}$ are i.i.d. Gaussian distributed, the coordinate-wise skewness can be upper bounded by 429.

We now state our main technical results on the median-based algorithm, namely statistical error guarantees for strongly convex, non-strongly convex, and smooth non-convex population loss functions F. In the first two cases with a convex F, we assume that $\mathbf{w}^*$, the minimizer of $F(\cdot)$ in $\mathcal{W}$, is also the minimizer of $F(\cdot)$ in $\mathbb{R}^d$, i.e., $\nabla F(\mathbf{w}^*) = 0$.

Strongly Convex Losses: We first consider the case where the population loss function $F(\cdot)$ is strongly convex. Note that we do not require strong convexity of the individual loss functions $f(\cdot; \mathbf{z})$.

Theorem 3.1. *Consider Option I in Algorithm 1. Suppose that Assumptions 3.1, 3.2, and 3.3 hold,* $F(\cdot)$ *is* λ_F*-strongly convex, and the fraction* α *of Byzantine machines satisfies*

$$\alpha + \sqrt{\frac{d \log(1 + nm\widehat{L}D)}{m(1-\alpha)}} + 0.4748\frac{S}{\sqrt{n}} \le \frac{1}{2} - \epsilon \tag{3.2}$$

for some $\epsilon > 0$*. Choose step-size* $\eta = 1/L_F$*. Then, with probability at least* $1 - \frac{4d}{(1+nm\widehat{L}D)^d}$*, after* T *parallel iterations, we have*

$$\|\mathbf{w}^T - \mathbf{w}^*\|_2 \le (1 - \frac{\lambda_F}{L_F + \lambda_F})^T \|\mathbf{w}^0 - \mathbf{w}^*\|_2 + \frac{2}{\lambda_F}\Delta,$$

where

$$\Delta := \mathcal{O}\bigg(C_\epsilon V\Big(\frac{\alpha}{\sqrt{n}} + \sqrt{\frac{d\log(nm\widehat{L}D)}{nm}} + \frac{S}{n}\Big)\bigg), \tag{3.3}$$

and C_ϵ is defined as

$$C_\epsilon := \sqrt{2\pi}\exp\Big(\frac{1}{2}(\Phi^{-1}(1-\epsilon))^2\Big), \tag{3.4}$$

with $\Phi^{-1}(\cdot)$ being the inverse of the cumulative distribution function of the standard Gaussian distribution $\Phi(\cdot)$.

In (3.3), we hide universal constants and a higher order term that scales as $\frac{1}{nm}$, and the factor C_ϵ is a function of ϵ; as a concrete example, $C_\epsilon \approx 4$ when $\epsilon = \frac{1}{6}$. Theorem 3.1 together with the inequality $\log(1-x) \le -x$, guarantees that after running $T \ge \frac{L_F+\lambda_F}{\lambda_F}\log(\frac{\lambda_F}{2\Delta}\|\mathbf{w}^0 - \mathbf{w}^*\|_2)$ parallel iterations, with high probability we can obtain a solution $\widehat{\mathbf{w}} = \mathbf{w}^T$ with error $\|\widehat{\mathbf{w}} - \mathbf{w}^*\|_2 \le \frac{4}{\lambda_F}\Delta$.

Here we achieve an error rate (defined as the distance between $\widehat{\mathbf{w}}$ and the optimal solution $\mathbf{w}^*$) of the form $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}} + \frac{1}{n})$. In Section 3.6, we provide a lower bound showing that the error rate of any algorithm is $\widetilde{\Omega}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}})$. Therefore the first two terms in the upper bound cannot be improved. The third term $\frac{1}{n}$ is due to the dependence of the median on the skewness of the gradients. When each worker machine has a sufficient amount of data, more specifically $n \gtrsim m$, we achieve an order-optimal error rate up to logarithmic factors.

Non-strongly Convex Losses: We next consider the case where the population risk function $F(\cdot)$ is convex, but not necessarily strongly convex. In this case, we need a mild technical assumption on the size of the parameter space $\mathcal{W}$.

Assumption 3.4 (Size of $\mathcal{W}$). *The parameter space $\mathcal{W}$ contains the following ℓ_2 ball centered at $\mathbf{w}^*$: $\{\mathbf{w} \in \mathbb{R}^d : \|\mathbf{w} - \mathbf{w}^*\|_2 \le 2\|\mathbf{w}^0 - \mathbf{w}^*\|_2\}$.*

We then have the following result on the convergence rate in terms of the value of the population risk function.

Theorem 3.2. *Consider Option I in Algorithm 1. Suppose that Assumptions 3.1, 3.2, 3.3 and 3.4 hold, and that the population loss $F(\cdot)$ is convex, and α satisfies (3.2) for some $\epsilon > 0$. Define Δ as in (3.3), and choose step-size $\eta = 1/L_F$. Then, with probability at least $1 - \frac{4d}{(1+nm\widehat{L}D)^d}$, after $T = \frac{L_F}{\Delta}\|\mathbf{w}^0 - \mathbf{w}^*\|_2$ parallel iterations, we have*

$$F(\mathbf{w}^T) - F(\mathbf{w}^*) \le 16\|\mathbf{w}^0 - \mathbf{w}^*\|_2\Delta\Big(1 + \frac{1}{2L_F}\Delta\Big).$$

We observe that the error rate, defined as the excess risk $F(\mathbf{w}^T) - F(\mathbf{w}^*)$, again has the form $\widetilde{\mathcal{O}}\big(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}} + \frac{1}{n}\big)$.

Non-convex Losses: When $F(\cdot)$ is non-convex but smooth, we need a somewhat different technical assumption on the size of $\mathcal{W}$.

Assumption 3.5 (Size of $\mathcal{W}$)**.** *Suppose that* $\forall\ \mathbf{w} \in \mathcal{W}$, $\|\nabla F(\mathbf{w})\|_2 \le M$. *We assume that* $\mathcal{W}$ *contains the* ℓ_2 *ball* $\{\mathbf{w} \in \mathbb{R}^d : \|\mathbf{w} - \mathbf{w}^0\|_2 \le \frac{2}{\Delta^2}(M + \Delta)(F(\mathbf{w}^0) - F(\mathbf{w}^*))\}$*, where* Δ *is defined as in* (3.3)*.*

We have the following guarantees on the rate of convergence to a critical point of the population loss $F(\cdot)$.

Theorem 3.3. *Consider Option I in Algorithm 1. Suppose that Assumptions 3.1 3.2, 3.3 and 3.5 hold, and* α *satisfies* (3.2) *for some* $\epsilon > 0$*. Define* Δ *as in* (3.3)*, and choose step-size* $\eta = 1/L_F$*. With probability at least* $1 - \frac{4d}{(1+nm\widehat{L}D)^d}$*, after* $T = \frac{2L_F}{\Delta^2}(F(\mathbf{w}^0) - F(\mathbf{w}^*))$ *parallel iterations, we have*

$$\min_{t=0,1,\ldots,T} \|\nabla F(\mathbf{w}^t)\|_2 \le \sqrt{2}\Delta.$$

We again obtain an $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}} + \frac{1}{n})$ error rate in terms of the gap to a critical point of $F(\mathbf{w})$.

Guarantees for Trimmed-mean-based Gradient Descent

We next analyze the robust distributed gradient descent algorithm based on coordinate-wise trimmed mean, namely Option II in Algorithm 1. Here we need stronger assumptions on the tail behavior of the partial derivatives of the loss functions—in particular, sub-exponentiality.

Assumption 3.6 (Sub-exponential gradients)**.** *We assume that for all* $k \in [d]$ *and* $\mathbf{w} \in \mathcal{W}$*, the partial derivative of* $f(\mathbf{w}; \mathbf{z})$ *with respect to the* k*-th coordinate of* $\mathbf{w}$, $\partial_k f(\mathbf{w}; \mathbf{z})$*, is* v*-sub-exponential.*

The sub-exponential property implies that all the moments of the derivatives are bounded. This is a stronger assumption than the bounded absolute skewness (hence bounded third moments) required by the median-based GD algorithm.

We use the same example as in Proposition 3.1 and show that the derivatives of the loss are indeed sub-exponential.

Proposition 3.2. *Consider the regression problem in Proposition 3.1. For all* $k \in [d]$ *and* $\mathbf{w} \in \mathcal{W}$*, the partial derivative* $\partial_k f(\mathbf{w}; \mathbf{z})$ *is* $\sqrt{\sigma^2 + \|\mathbf{w} - \mathbf{w}^*\|_2^2}$*-sub-exponential.*

We now proceed to establish the statistical guarantees of the trimmed-mean-based algorithm, for different loss function classes. When the population loss $F(\cdot)$ is convex, we again assume that the minimizer of $F(\cdot)$ in $\mathcal{W}$ is also its minimizer in $\mathbb{R}^d$. The next three theorems are analogues of Theorems 3.1–3.3 for the median-based GD algorithm.

Strongly Convex Losses: We have the following result.

Theorem 3.4. *Consider Option II in Algorithm 1. Suppose that Assumptions 3.1 and 3.6 hold, $F(\cdot)$ is λ_F-strongly convex, and $\alpha \le \beta \le \frac{1}{2} - \epsilon$ for some $\epsilon > 0$. Choose step-size $\eta = 1/L_F$. Then, with probability at least $1 - \frac{4d}{(1+nm\widehat{L}D)^d}$, after T parallel iterations, we have*

$$\|\mathbf{w}^T - \mathbf{w}^*\|_2 \le \Big(1 - \frac{\lambda_F}{L_F + \lambda_F}\Big)^T \|\mathbf{w}^0 - \mathbf{w}^*\|_2 + \frac{2}{\lambda_F}\Delta',$$

where

$$\Delta' := \mathcal{O}\Big(\frac{vd}{\epsilon}\big(\frac{\beta}{\sqrt{n}} + \frac{1}{\sqrt{nm}}\big)\sqrt{\log(nm\widehat{L}D)}\Big). \tag{3.5}$$

In (3.5), we hide universal constants and higher order terms that scale as $\frac{\beta}{n}$ or $\frac{1}{nm}$. By running $T \ge \frac{L_F+\lambda_F}{\lambda_F}\log(\frac{\lambda_F}{2\Delta'}\|\mathbf{w}^0 - \mathbf{w}^*\|_2)$ parallel iterations, we can obtain a solution $\widehat{\mathbf{w}} = \mathbf{w}^T$ satisfying $\|\widehat{\mathbf{w}} - \mathbf{w}^*\|_2 \le \widetilde{\mathcal{O}}(\frac{\beta}{\sqrt{n}} + \frac{1}{\sqrt{nm}})$. Note that one needs to choose the parameter for trimmed mean to satisfy $\beta \ge \alpha$. If we set $\beta = c\alpha$ for some universal constant $c \ge 1$, we can achieve an order-optimal error rate $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}})$.

Non-strongly Convex Losses: Again imposing Assumption 3.4 on the size of $\mathcal{W}$, we have the following guarantee.

Theorem 3.5. *Consider Option II in Algorithm 1. Suppose that Assumptions 3.1, 3.4 and 3.6 hold, $F(\cdot)$ is convex, and $\alpha \le \beta \le \frac{1}{2} - \epsilon$ for some $\epsilon > 0$. Choose step-size $\eta = 1/L_F$, and define Δ' as in (3.5). Then, with probability at least $1 - \frac{4d}{(1+nm\widehat{L}D)^d}$, after $T = \frac{L_F}{\Delta'}\|\mathbf{w}^0 - \mathbf{w}^*\|_2$ parallel iterations, we have*

$$F(\mathbf{w}^T) - F(\mathbf{w}^*) \le 16\|\mathbf{w}^0 - \mathbf{w}^*\|_2\Delta'\Big(1 + \frac{1}{2L_F}\Delta'\Big).$$

The proof of Theorem 3.5 is similar to that of Theorem 3.2, and we refer readers to Remark 3.1 in Section 3.9. Again, by choosing $\beta = c\alpha$ ($c \ge 1$), we obtain the $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}})$ error rate in the function value of $F(\mathbf{w})$.

Non-convex Losses: In this case, imposing a version of Assumption 3.5 on the size of $\mathcal{W}$, we have the following.

Theorem 3.6. *Consider Option II in Algorithm 1, and define Δ' as in (3.5). Suppose that Assumptions 3.1 and 3.6 hold, Assumption 3.5 holds with Δ replaced by Δ', and $\alpha \le \beta \le \frac{1}{2} - \epsilon$ for some $\epsilon > 0$. Choose step-size $\eta = 1/L_F$. Then, with probability at least $1 - \frac{4d}{(1+nm\widehat{L}D)^d}$, after $T = \frac{2L_F}{\Delta'^2}(F(\mathbf{w}^0) - F(\mathbf{w}^*))$ parallel iterations, we have*

$$\min_{t=0,1,\ldots,T} \|\nabla F(\mathbf{w}^t)\|_2 \le \sqrt{2}\Delta'.$$

The proof of Theorem 3.6 is similar to that of Theorem 3.3; see Remark 3.1 in Section 3.9. By choosing $\beta = c\alpha$ with $c \ge 1$, we again achieve the statistical rate $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}})$.

Comparisons

We compare the performance guarantees of the above two robust distribute GD algorithms. The trimmed-mean-based algorithm achieves the statistical error rate $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}})$, which is order-optimal for strongly convex loss. In comparison, the rate of the median-based algorithm is $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}} + \frac{1}{n})$, which has an additional $\frac{1}{n}$ term and is only optimal when $n \gtrsim m$. In particular, the trimmed-mean-based algorithm has better rates when each worker machine has small local sample size—the rates are meaningful even in the extreme case $n = \mathcal{O}(1)$. On the other hand, the median-based algorithm requires milder tail/moment assumptions on the loss derivatives (bounded skewness) than its trimmed-mean counterpart (sub-exponentiality). Finally, the trimmed-mean operation requires an additional parameter β, which can be any upper bound on the fraction α of Byzantine machines in order to guarantee robustness. Using an overly large β may lead to a looser bound and sub-optimal performance. In contrast, median-based GD does not require knowledge of α. We summarize these observations in Table 3.1. We see that the two algorithms are complementary to each other, and our experiment results corroborate this point.

	median GD	trimmed mean GD
Statistical error rate	$\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}} + \frac{1}{n})$	$\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}})$
Distribution of $\partial_k f(\mathbf{w};\mathbf{z})$	Bounded skewness	Sub-exponential
α known?	No	Yes

Table 3.1: Comparison between the two robust distributed gradient descent algorithms.

3.5 Robust One-round Algorithm

As mentioned, in our distributed computing framework, the communication cost is proportional to the number of parallel iterations. The above two GD algorithms both require a number iterations depending on the desired accuracy. Can we further reduce the communication cost while keeping the algorithm Byzantine-robust and statistically optimal?

A natural candidate is the so-called one-round algorithm. Previous work has considered a standard one-round scheme where each local machine computes the empirical risk minimizer (ERM) using its local data and the master machine receives all workers' ERMs and computes their *average* [209]. Clearly, a single Byzantine machine can arbitrary skew the output of this algorithm. We instead consider a Byzantine-robust one-round algorithm. As detailed in Algorithm 2, we employ the coordinate-wise median operation to aggregate all the ERMs.

Our main result is a characterization of the error rate of Algorithm 2 in the presence of Byzantine failures. We are only able to establish such a guarantee when the loss functions are quadratic and $\mathcal{W} = \mathbb{R}^d$. However, one can implement this algorithm in problems with other loss functions.

Algorithm 2 Robust One-round Algorithm

```
for all $i \in [m]$ do in parallel
    Worker machine $i$: compute:
        $\widehat{\mathbf{w}}^i \leftarrow \begin{cases} \arg\min_{\mathbf{w}\in\mathcal{W}} F_i(\mathbf{w}) & \text{normal worker machines} \\ * & \text{Byzantine machines} \end{cases}$
    send $\widehat{\mathbf{w}}^i$ to master machine.
end for
Master machine: compute $\widehat{\mathbf{w}} \leftarrow \mathsf{med}\{\widehat{\mathbf{w}}^i : i \in [m]\}$.
```

Definition 3.9 (Quadratic loss function)**.** *The loss function $f(\mathbf{w};\mathbf{z})$ is quadratic if it can be written as*

$$f(\mathbf{w};\mathbf{z}) = \frac{1}{2}\mathbf{w}^T\mathbf{H}\mathbf{w} + \mathbf{p}^T\mathbf{w} + c,$$

where $\mathbf{z} = (\mathbf{H}, \mathbf{p}, c)$, $\mathbf{H}$, and $\mathbf{p}$, and c are drawn from the distributions $\mathcal{D}_H$, $\mathcal{D}_p$, and $\mathcal{D}_c$, respectively.

Denote by $\mathbf{H}_F$, $\mathbf{p}_F$, and c_F the expectations of $\mathbf{H}$, $\mathbf{p}$, and c, respectively. Thus the population risk function takes the form $F(\mathbf{w}) = \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{H}_F\mathbf{w} + \mathbf{p}_F^{\mathrm{T}}\mathbf{w} + c_F$.

We need a technical assumption which guarantees that each normal worker machine has unique ERM.

Assumption 3.7 (Strong convexity of F_i)**.** *With probability 1, the empirical risk minimization function $F_i(\cdot)$ on each normal machine is strongly convex.*

Note that this assumption is imposed on $F_i(\mathbf{w})$, rather than on the individual loss $f(\mathbf{w};\mathbf{z})$ associated with a single data point. This assumption is satisfied, for example, when all $f(\cdot;\mathbf{z})$'s are strongly convex, or in the linear regression problems with the features $\mathbf{x}$ drawn from some continuous distribution (e.g. isotropic Gaussian) and $n \geq d$. We have the following guarantee for the robust one-round algorithm.

Theorem 3.7. *Suppose that $\forall\ \mathbf{z} \in \mathcal{Z}$, the loss function $f(\cdot;\mathbf{z})$ is convex and quadratic, $F(\cdot)$ is λ_F-strongly convex, and Assumption 3.7 holds. Assume that α satisfies*

$$\alpha + \sqrt{\frac{\log(nmd)}{2m(1-\alpha)}} + \frac{\widetilde{C}}{\sqrt{n}} \leq \frac{1}{2} - \epsilon$$

for some $\epsilon > 0$, where $\widetilde{C}$ is a quantity that depends on $\mathcal{D}_H$, $\mathcal{D}_p$, λ_F and is monotonically decreasing in n. Then, with probability at least $1 - \frac{4}{nm}$, the output $\widehat{\mathbf{w}}$ of the robust one-round algorithm satisfies

$$\|\widehat{\mathbf{w}} - \mathbf{w}^*\|_2 \leq \frac{C_\epsilon}{\sqrt{n}}\widetilde{\sigma}\big(\alpha + \sqrt{\frac{\log(nmd)}{2m(1-\alpha)}} + \frac{\widetilde{C}}{\sqrt{n}}\big),$$

where C_ϵ is defined as in (3.4) *and*

$$\widetilde{\sigma}^2 := \mathbb{E}\big[\|\mathbf{H}_F^{-1}\big((\mathbf{H}-\mathbf{H}_F)\mathbf{H}_F^{-1}\mathbf{p}_F - (\mathbf{p}-\mathbf{p}_F)\big)\|_2^2\big],$$

with $\mathbf{H}$ *and* $\mathbf{p}$ *drawn from* $\mathcal{D}_H$ *and* $\mathcal{D}_p$*, respectively.*

We prove Theorem 3.7 and provide an explicit expression of $\widetilde{C}$ in Section 3.9. In terms of the dependence on α, n, and m, the robust one-round algorithm achieves the same error rate as the robust gradient descent algorithm based on coordinate-wise median, i.e., $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}} + \frac{1}{n})$, for quadratic problems. Again, this rate is optimal when $n \gtrsim m$. Therefore, at least for quadratic loss functions, the robust one-round algorithm has similar theoretical performance as the robust gradient descent algorithm with significantly less communication cost. Our experiments show that the one-round algorithm has good empirical performance for other losses as well.

3.6 Lower Bound

In this section, we provide a lower bound on the error rate for strongly convex losses, which implies that the $\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}}$ term is unimprovable. This lower bound is derived using a mean estimation problem, and is an extension of the lower bounds in the robust mean estimation literature such as Chen, Gao, and Ren [39] and Lai, Rao, and Vempala [113].

We consider the problem of estimating the mean $\boldsymbol{\mu}$ of some random variable $\mathbf{z} \sim \mathcal{Z}$, which is equivalent to solving the following minimization problem:

$$\boldsymbol{\mu} = \arg\min_{\mathbf{w}\in\mathcal{W}} \mathbb{E}_{\mathbf{z}\sim\mathcal{Z}}[\|\mathbf{w}-\mathbf{z}\|_2^2], \tag{3.6}$$

Note that this is a special case of the general learning problem (3.1). We consider the same distributed setting as in Section 3.4, with a minor technical difference regarding the Byzantine machines. We assume that each of the m worker machines is Byzantine with probability α, independently of each other. The parameter α is therefore the *expected* fraction of Byzantine machines. This setting makes the analysis slightly easier, and we believe the result can be extended to the original setting.

In this setting we have the following lower bound.

Observation 3.1. *Consider the distributed mean estimation problem in* (3.6) *with Byzantine failure probability* α*, and suppose that* $\mathcal{Z}$ *is Gaussian distribution with mean* $\boldsymbol{\mu}$ *and covariance matrix* $\sigma^2\mathbf{I}$ ($\sigma = \mathcal{O}(1)$)*. Then, any algorithm that computes an estimation* $\widehat{\boldsymbol{\mu}}$ *of the mean from the data has a constant probability of error* $\|\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_2 = \Omega(\frac{\alpha}{\sqrt{n}} + \sqrt{\frac{d}{nm}})$.

According to this observation, we see that the $\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}}$ dependence cannot be avoided, which in turn implies the order-optimality of the results in Theorem 3.1 (when $n \gtrsim m$) and Theorem 3.4.

3.7 Experiments

We conduct experiments to show the effectiveness of the median and trimmed mean operations. Our experiments are implemented with Tensorflow [1] on Microsoft Azure system. We use the MNIST [115] dataset and randomly partition the 60,000 training data into m subsamples with equal sizes. We use these subsamples to represent the data on m machines.

In the first experiment, we compare the performance of distributed gradient descent algorithms in the following four settings: 1) $\alpha = 0$ (no Byzantine machines), using vanilla distributed gradient descent (aggregating the gradients by taking the mean), 2) $\alpha > 0$, using vanilla distributed gradient descent, 3) $\alpha > 0$, using median-based algorithm, and 4) $\alpha > 0$, using trimmed-mean-based algorithm. We generate the Byzantine machines in the following way: we replace every training label y on these machines with $9 - y$, e.g., 0 is replaced with 9, 1 is replaced with 8, etc, and the Byzantine machines simply compute gradients based on these data. We also note that when generating the Byzantine machines, we do not simply add extreme values in the features or gradients; instead, the Byzantine machines send messages to the master machine with moderate values.

We train a multi-class logistic regression model and a convolutional neural network model using distributed gradient descent, and for each model, we compare the test accuracies in the aforementioned four settings. For the convolutional neural network model, we use the stochastic version of the distributed gradient descent algorithm; more specifically, in every iteration, each worker machine computes the gradient using 10% of its local data. We periodically check the test errors, and the convergence performances are shown in Figure 3.2. The final test accuracies are presented in Tables 3.2 and 3.3.

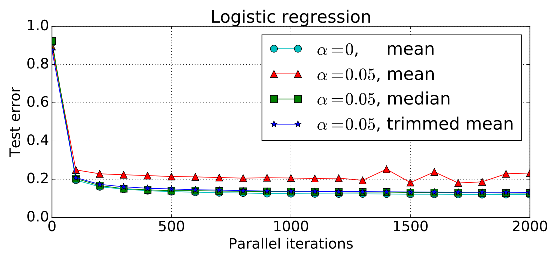
α	0	0.05		
Algorithm	mean	mean	median	trimmed mean
Test accuracy (%)	88.0	76.8	87.2	86.9

Table 3.2: Test accuracy on the logistic regression model using gradient descent. We set $m = 40$, and for trimmed mean, we choose $\beta = 0.05$.

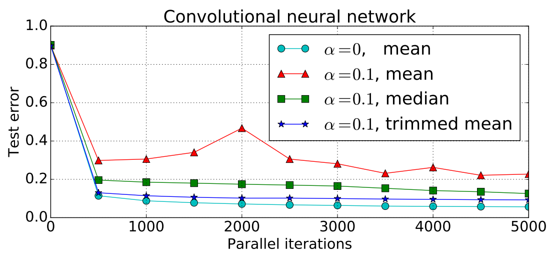
α	0	0.1		
Algorithm	mean	mean	median	trimmed mean
Test accuracy (%)	94.3	77.3	87.4	90.7

Table 3.3: Test accuracy on the convolutional neural network model using gradient descent. We set $m = 10$, and for trimmed mean, we choose $\beta = 0.1$.

As we can see, in the adversarial settings, the vanilla distributed gradient descent algorithm suffers from severe performance loss, and using the median and trimmed mean opera-



(a)



(b)

Figure 3.2: Test error vs the number of parallel iterations.

tions, we observe significant improvement in test accuracy. This shows these two operations can indeed defend against Byzantine failures.

In the second experiment, we compare the performance of distributed one-round algorithms in the following three settings: 1) $\alpha = 0$, mean aggregation, 2) $\alpha > 0$, mean aggregation, and 3) $\alpha > 0$, median aggregation. In this experiment, the training labels on the Byzantine machines are i.i.d. uniformly sampled from $\{0, \dots, 9\}$, and these machines train models using the faulty data. We choose the multi-class logistic regression model, and the test accuracies are presented in Table 3.4.

α	0	0.1	
Algorithm	mean	mean	median
Test accuracy (%)	91.8	83.7	89.0

Table 3.4: Test accuracy on the logistic regression model using one-round algorithm. We set $m = 10$.

As we can see, for the one-round algorithm, although the theoretical guarantee is only proved for quadratic loss, in practice, the median-based one-round algorithm still improves the test accuracy in problems with other loss functions, such as the logistic loss here.

3.8 Conclusions

In this chapter, we study Byzantine-robust distributed statistical learning algorithms with a focus on statistical optimality. We analyze two robust distributed gradient descent algorithms — one is based on coordinate-wise median and the other is based on coordinate-wise trimmed mean. We show that the trimmed-mean-based algorithm can achieve order-optimal $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}})$ error rate, whereas the median-based algorithm can achieve $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}} + \frac{1}{n})$

under weaker assumptions. We further study learning algorithms that have better communication efficiency. We propose a simple one-round algorithm that aggregates local solutions using coordinate-wise median. We show that for strongly convex quadratic problems, this algorithm can achieve $\widetilde{\mathcal{O}}(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}} + \frac{1}{n})$ error rate, similar to the median-based gradient descent algorithm. Our experiments validates the effectiveness of the median and trimmed mean operations in the adversarial setting.

3.9 Proofs

Variance, Skewness, and Sub-exponential Property

Proof of Proposition 3.1

We use the simplified notation $f(\mathbf{w}) := f(\mathbf{w}; \mathbf{x}, y)$. One can directly compute the gradients:

$$\nabla f(\mathbf{w}) = \mathbf{x}(\mathbf{x}^{\mathrm{T}}\mathbf{w} - y) = \mathbf{x}\mathbf{x}^{\mathrm{T}}(\mathbf{w} - \mathbf{w}^*) - \xi\mathbf{x},$$

and thus

$$\nabla F(\mathbf{w}) = \mathbb{E}[\nabla f(\mathbf{w})] = \mathbf{w} - \mathbf{w}^*.$$

Define $\Delta(\mathbf{w}) := \nabla f(\mathbf{w}) - \nabla F(\mathbf{w})$ with its k-th element being $\Delta_k(\mathbf{w})$. We now compute the variance and absolute skewness of $\Delta_k(\mathbf{w})$.

We can see that

$$\Delta_k(\mathbf{w}) = \sum_{\substack{1\le i\le d\\ i\neq k}} x_k x_i (w_i - w_i^*) + (x_k^2 - 1)(w_k - w_k^*) - \xi x_k. \tag{3.7}$$

Thus,

$$\mathbb{E}[\Delta_k^2(\mathbf{w})] = \mathbb{E}[\sum_{\substack{1\le i\le d\\ i\neq k}} x_k^2 x_i^2 (w_i - w_i^*)^2 + \xi^2 x_k^2] = \|\mathbf{w} - \mathbf{w}^*\|_2^2 - (w_k - w_k^*)^2 + \sigma^2, \tag{3.8}$$

which yields

$$\mathrm{Var}(\nabla f(\mathbf{w})) = \mathbb{E}[\|\nabla f(\mathbf{w}) - \nabla F(\mathbf{w})\|_2^2] = (d-1)\|\mathbf{w} - \mathbf{w}^*\|_2^2 + d\sigma^2.$$

Then we proceed to bound $\gamma(\Delta_k(\mathbf{w}))$. By Jensen's inequality, we know that

$$\gamma(\Delta_k(\mathbf{w})) = \frac{\mathbb{E}[|\Delta_k(\mathbf{w})|^3]}{\mathrm{Var}(\Delta_k(\mathbf{w}))^{3/2}} \le \sqrt{\frac{\mathbb{E}[\Delta_k^6(\mathbf{w})]}{\mathrm{Var}(\Delta_k(\mathbf{w}))^3}} \tag{3.9}$$

We first find a lower bound for $\mathrm{Var}(\Delta_k(\mathbf{w}))^3$. According to (3.8), we know that

$$\mathrm{Var}(\Delta_k(\mathbf{w}))^3 = \big(\sum_{\substack{1\le i\le d\\ i\neq k}} (w_i - w_i^*)^2 + \sigma^2\big)^3 \ge \big(\sum_{\substack{1\le i\le d\\ i\neq k}} (w_i - w_i^*)^2\big)^3 + \sigma^6.$$

Define the following three quantities.

$$W_1 = \sum_{\substack{1\le i\le d\\ i\neq k}} (w_i - w_i^*)^6 \tag{3.10}$$

$$W_2 = \sum_{\substack{1\le i,j\le d\\ i,j\neq k\\ i\neq j}} (w_i - w_i^*)^4 (w_j - w_j^*)^2 \tag{3.11}$$

$$W_3 = \sum_{\substack{1\le i,j,\ell\le d\\ i,j,\ell\neq k\\ i\neq j, i\neq \ell, j\neq \ell}} (w_i - w_i^*)^2 (w_j - w_j^*)^2 (w_\ell - w_\ell^*)^2 \tag{3.12}$$

By simple algebra, one can check that

$$\big(\sum_{\substack{1\le i\le d\\ i\neq k}} (w_i - w_i^*)^2\big)^3 = W_1 + 3W_2 + W_3, \tag{3.13}$$

and thus

$$\mathrm{Var}(\Delta_k(\mathbf{w}))^3 \geq W_1 + 3W_2 + W_3 + \sigma^6. \tag{3.14}$$

Then, we find an upper bound on $\mathbb{E}[\Delta_k^6(\mathbf{w})]$. According to (3.7), and Hölder's inequality, we know that

$$\begin{aligned}\mathbb{E}[\Delta_k^6(\mathbf{w})] &= \mathbb{E}[(\sum_{\substack{1\le i\le d\\ i\neq k}} x_k x_i (w_i - w_i^*) - \xi x_k)^6] \leq 32\big(\mathbb{E}[(\sum_{\substack{1\le i\le d\\ i\neq k}} x_k x_i (w_i - w_i^*))^6] + \mathbb{E}[\xi^6 x_k^6]\big) \\ &= 32\big(\mathbb{E}[(\sum_{\substack{1\le i\le d\\ i\neq k}} x_i (w_i - w_i^*))^6] + 15\sigma^6\big), \end{aligned}\tag{3.15}$$

where in the last inequality we use the moments of Gaussian random variables. Then, we compute the first term in (3.15). By algebra, one can obtain

$$\begin{aligned}\mathbb{E}[(\sum_{\substack{1\le i\le d\\ i\neq k}} x_i (w_i - w_i^*))^6] =& \mathbb{E}[\sum_{\substack{1\le i\le d\\ i\neq k}} x_i^6 (w_i - w_i^*)^6] + 15\mathbb{E}[\sum_{\substack{1\le i,j\le d\\ i,j\neq k\\ i\neq j}} x_i^4 x_j^2 (w_i - w_i^*)^4 (w_j - w_j^*)^2] \\ &+ 15\mathbb{E}[\sum_{\substack{1\le i,j,\ell\le d\\ i,j,\ell\neq k\\ i\neq j, i\neq \ell, j\neq \ell}} x_i^2 x_j^2 x_\ell^2 (w_i - w_i^*)^2 (w_j - w_j^*)^2 (w_\ell - w_\ell^*)^2] \\ =& W_1 + 15W_2 + 15W_3. \end{aligned}\tag{3.16}$$

Combining (3.15) and (3.16), we get

$$\mathbb{E}[\Delta_k^6(\mathbf{w})] \leq 32(W_1 + 15W_2 + 15W_3 + 15\sigma^6). \tag{3.17}$$

Combining (3.14) and (3.17), we get

$$\gamma(\Delta_k(\mathbf{w})) \leq \sqrt{\frac{\mathbb{E}[\Delta_k^6(\mathbf{w})]}{\mathrm{Var}(\Delta_k(\mathbf{w}))^3}} \leq \sqrt{\frac{32(W_1 + 15W_2 + 15W_3 + 15\sigma^6)}{W_1 + 3W_2 + W_3 + \sigma^6}} \leq 480.$$

Example of Regression with Gaussian Features

Proposition 3.3. *Suppose that each data point consists of a feature $\mathbf{x} \in \mathbb{R}^d$ and a label $y \in \mathbb{R}$, and the label is generated by*

$$y = \mathbf{x}^T\mathbf{w}^* + \xi$$

with some $\mathbf{w}^* \in \mathcal{W}$. Assume that the elements of $\mathbf{x}$ are i.i.d. samples of standard Gaussian distribution, and that the noise ξ is independent of $\mathbf{x}$ and drawn from Gaussian distribution $\mathcal{N}(0, \sigma^2)$. Define the quadratic loss function $f(\mathbf{w}; \mathbf{x}, y) = \frac{1}{2}(y - \mathbf{x}^T\mathbf{w})^2$. Then, we have

$$\mathrm{Var}(\nabla f(\mathbf{w}; \mathbf{x}, y)) = (d+1)\|\mathbf{w} - \mathbf{w}^*\|_2^2 + d\sigma^2,$$

and

$$\|\gamma(\nabla f(\mathbf{w}; \mathbf{x}, y))\|_\infty \leq 429.$$

Proof. We use the same simplified notation as in Section 3.9. One can also see that (3.7) still holds for in the Gaussian setting. Thus,

$$\begin{aligned}
\mathbb{E}[\Delta_k^2(\mathbf{w})] &= \mathbb{E}[\sum_{\substack{1\leq i\leq d\\ i\neq k}} x_k^2 x_i^2 (w_i - w_i^*)^2 + (x_k^2 - 1)^2 (w_k - w_k^*)^2 + \xi^2 x_k^2] \\
&= \sum_{\substack{1\leq i\leq d\\ i\neq k}} (w_i - w_i^*)^2 + 2(w_k - w_k^*)^2 + \sigma^2 \qquad (3.18)\\
&= \|\mathbf{w} - \mathbf{w}^*\|_2^2 + (w_k - w_k^*)^2 + \sigma^2, \qquad (3.19)
\end{aligned}$$

which yields

$$\mathrm{Var}(\nabla f(\mathbf{w})) = \mathbb{E}[\|\nabla f(\mathbf{w}) - \nabla F(\mathbf{w})\|_2^2] = (d+1)\|\mathbf{w} - \mathbf{w}^*\|_2^2 + d\sigma^2.$$

Then we proceed to bound $\gamma(\Delta_k(\mathbf{w}))$. By Jensen's inequality, we know that

$$\gamma(\Delta_k(\mathbf{w})) = \frac{\mathbb{E}[|\Delta_k(\mathbf{w})|^3]}{\mathrm{Var}(\Delta_k(\mathbf{w}))^{3/2}} \leq \sqrt{\frac{\mathbb{E}[\Delta_k^6(\mathbf{w})]}{\mathrm{Var}(\Delta_k(\mathbf{w}))^3}} \qquad (3.20)$$

We first find a lower bound for $\mathrm{Var}(\Delta_k(\mathbf{w}))^3$. According to (3.18), we know that

$$\begin{aligned}
\mathrm{Var}(\Delta_k(\mathbf{w}))^3 &= \Big(\sum_{\substack{1\leq i\leq d\\ i\neq k}} (w_i - w_i^*)^2 + 2(w_k - w_k^*)^2 + \sigma^2\Big)^3 \\
&\geq \Big(\sum_{\substack{1\leq i\leq d\\ i\neq k}} (w_i - w_i^*)^2\Big)^3 + 8(w_k - w_k^*)^6 + \sigma^6.
\end{aligned}$$

Define the W_1, W_2, and W_3 as in (3.10), (3.11), and (3.12). We can also see that (3.13) still holds, and thus

$$\mathrm{Var}(\Delta_k(\mathbf{w}))^3 \geq W_1 + 3W_2 + W_3 + 8(w_k - w_k^*)^6 + \sigma^6. \tag{3.21}$$

Then, we find an upper bound on $\mathbb{E}[\Delta_k^6(\mathbf{w})]$. According to (3.7), and Hölder's inequality, we know that

$$\begin{aligned}\mathbb{E}[\Delta_k^6(\mathbf{w})] &= \mathbb{E}[(\sum_{\substack{1\leq i\leq d\\ i\neq k}} x_k x_i (w_i - w_i^*) + (x_k^2 - 1)(w_k - w_k^*) - \xi x_k)^6] \\ &\leq 243\big(\mathbb{E}[(\sum_{\substack{1\leq i\leq d\\ i\neq k}} x_k x_i (w_i - w_i^*))^6] + \mathbb{E}[(x_k^2-1)^6 (w_k - w_k^*)^6] + \mathbb{E}[\xi^6 x_k^6]\big) \\ &= 243\big(15\mathbb{E}[(\sum_{\substack{1\leq i\leq d\\ i\neq k}} x_i (w_i - w_i^*))^6] + 6040(w_k - w_k^*)^6 + 225\sigma^6\big), \end{aligned} \tag{3.22}$$

where in the last inequality we use the moments of Gaussian random variables. Then, we compute the first term in (3.22). By algebra, one can obtain

$$\begin{aligned}\mathbb{E}[(\sum_{\substack{1\leq i\leq d\\ i\neq k}} x_i (w_i - w_i^*))^6] =&\mathbb{E}[\sum_{\substack{1\leq i\leq d\\ i\neq k}} x_i^6 (w_i - w_i^*)^6] + 15\mathbb{E}[\sum_{\substack{1\leq i,j\leq d\\ i,j\neq k\\ i\neq j}} x_i^4 x_j^2 (w_i - w_i^*)^4 (w_j - w_j^*)^2] \\ &+ 15\mathbb{E}[\sum_{\substack{1\leq i,j,\ell\leq d\\ i,j,\ell\neq k\\ i\neq j, i\neq \ell, j\neq \ell}} x_i^2 x_j^2 x_\ell^2 (w_i - w_i^*)^2 (w_j - w_j^*)^2 (w_\ell - w_\ell^*)^2] \\ =&15W_1 + 45W_2 + 15W_3. \end{aligned} \tag{3.23}$$

Combining (3.22) and (3.23), we get

$$\mathbb{E}[\Delta_k^6(\mathbf{w})] \leq 243(225W_1 + 675W_2 + 225W_3 + 6040(w_k - w_k^*)^6 + 225\sigma^6). \tag{3.24}$$

Combining (3.21) and (3.24), we get

$$\begin{aligned}\gamma(\Delta_k(\mathbf{w})) &\leq \sqrt{\frac{\mathbb{E}[\Delta_k^6(\mathbf{w})]}{\mathrm{Var}(\Delta_k(\mathbf{w}))^3}} \\ &\leq \sqrt{\frac{243(225W_1 + 675W_2 + 225W_3 + 6040(w_k - w_k^*)^6 + 225\sigma^6)}{W_1 + 3W_2 + W_3 + 8(w_k - w_k^*)^6 + \sigma^6}} \leq 429.\end{aligned}$$

□

Proof of Proposition 3.2

We have

$$\begin{aligned}\partial_k f(\mathbf{w};\mathbf{z}) - F(\mathbf{w}) = \Delta_k(\mathbf{w}) &= \sum_{\substack{1\le i\le d\\ i\ne k}} x_k x_i(w_i - w_i^*) + (x_k^2-1)(w_k-w_k^*) - \xi x_k \\ &= x_k(-\xi + \sum_{\substack{1\le i\le d\\ i\ne k}} x_i(w_i-w_i^*)) := x_k\Delta_k'(\mathbf{w})\end{aligned}$$

Since $\Delta_k'(\mathbf{w})$ has symmetric distribution and x_k is uniformly distributed in $\{-1,1\}$, we know that the distributions of $\Delta_k(\mathbf{w})$ and $\Delta_k'(\mathbf{w})$. We then prove a stronger result on $\Delta_k'(\mathbf{w})$. We first recall the definition of v-sub-Gaussian random variables. A random variable X with mean $\mu = \mathbb{E}[X]$ is v-sub-Gaussian if for all $\lambda\in\mathbb{R}$, $\mathbb{E}[e^{\lambda(X-\mu)}] \le e^{v^2\lambda^2/2}$. We can see that v-sub-Gaussian random variables are also v-sub-exponential. One can also check that x_i's are i.i.d. 1-sub-Gaussian random variables, and then $\Delta_k'(\mathbf{w})$ is v-sub-exponential with

$$v = \big(\sigma^2 + \sum_{\substack{1\le i\le d\\ i\ne k}} (w_i-w_i^*)^2\big)^{1/2} \le \sqrt{\sigma^2 + \|\mathbf{w}-\mathbf{w}^*\|_2^2}.$$

Proof of Theorem 3.1

The proof of Theorem 3.1 consists of two parts: 1) the analysis of coordinate-wise median estimator of the population gradients, and 2) the convergence analysis of the robustified gradient descent algorithm.

Recall that at iteration t, the master machine sends $\mathbf{w}^t$ to all the worker machines. For any normal worker machine, say machine $i\in[m]\setminus\mathcal{B}$, the gradient of the local empirical loss function $\widehat{\mathbf{g}}^i(\mathbf{w}^t) = \nabla F_i(\mathbf{w}^t)$ is computed and returned to the center machine, while the Byzantine machines, say machine $i\in\mathcal{B}$, the returned message $\widehat{\mathbf{g}}^i(\mathbf{w}^t)$ can be arbitrary or even adversarial. The master machine then compute the coordinate-wise median, i.e.,

$$\widehat{\mathbf{g}}(\mathbf{w}^t) = \mathsf{med}\{\widehat{\mathbf{g}}^i(\mathbf{w}^t) : i\in[m]\}.$$

The following theorem provides a uniform bound on the distance between $\widehat{\mathbf{g}}(\mathbf{w}^t)$ and $\nabla F(\mathbf{w}^t)$.

Theorem 3.8. *Define*

$$\widehat{\mathbf{g}}^i(\mathbf{w}) = \begin{cases}\nabla F_i(\mathbf{w}) & i\in[m]\setminus\mathcal{B},\\ * & i\in\mathcal{B}.\end{cases} \tag{3.25}$$

and the coordinate-wise median of $\widehat{\mathbf{g}}^i(\mathbf{w})$*:*

$$\widehat{\mathbf{g}}(\mathbf{w}) = \mathit{med}\{\widehat{\mathbf{g}}^i(\mathbf{w}) : i\in[m]\}. \tag{3.26}$$

Suppose that Assumptions 3.1, 3.2, and 3.3 hold, and inequality (3.2) *is satisfied with some* $\epsilon > 0$*. Then, we have with probability at least* $1 - \frac{4d}{(1+nm\widehat{L}D)^d}$,

$$\|\widehat{\mathbf{g}}(\mathbf{w}) - \nabla F(\mathbf{w})\|_2 \le 2\sqrt{2}\frac{1}{nm} + \sqrt{2}\frac{C_\epsilon}{\sqrt{n}}V\left(\alpha + \sqrt{\frac{d\log(1+nm\widehat{L}D)}{m(1-\alpha)}} + 0.4748\frac{S}{\sqrt{n}}\right), \quad (3.27)$$

for all $\mathbf{w} \in \mathcal{W}$*, where* C_ϵ *is defined as in* (3.4).

Then, we proceed to analyze the convergence of the robust distributed gradient descent algorithm. We condition on the event that the bound in (3.27) is satisfied for all $\mathbf{w} \in \mathcal{W}$. Then, in the t-th iteration, we define

$$\widehat{\mathbf{w}}^{t+1} = \mathbf{w}^t - \eta\widehat{\mathbf{g}}(\mathbf{w}^t).$$

Thus, we have $\mathbf{w}^{t+1} = \Pi_{\mathcal{W}}(\widehat{\mathbf{w}}^{t+1})$. By the property of Euclidean projection, we know that

$$\|\mathbf{w}^{t+1} - \mathbf{w}^*\|_2 \le \|\widehat{\mathbf{w}}^{t+1} - \mathbf{w}^*\|_2.$$

We further have

$$\begin{aligned}\|\mathbf{w}^{t+1} - \mathbf{w}^*\|_2 &\le \|\mathbf{w}^t - \eta\widehat{\mathbf{g}}(\mathbf{w}^t) - \mathbf{w}^*\|_2 \\ &\le \|\mathbf{w}^t - \eta\nabla F(\mathbf{w}^t) - \mathbf{w}^*\|_2 + \eta\|\widehat{\mathbf{g}}(\mathbf{w}^t) - \nabla F(\mathbf{w}^t)\|_2.\end{aligned} \quad (3.28)$$

Meanwhile, we have

$$\|\mathbf{w}^t - \eta\nabla F(\mathbf{w}^t) - \mathbf{w}^*\|_2^2 = \|\mathbf{w}^t - \mathbf{w}^*\|_2^2 - 2\eta\langle\mathbf{w}^t - \mathbf{w}^*, \nabla F(\mathbf{w}^t)\rangle + \eta^2\|\nabla F(\mathbf{w}^t)\|_2^2. \quad (3.29)$$

Since $F(\mathbf{w})$ is λ_F-strongly convex, by the co-coercivity of strongly convex functions (see Lemma 3.11 in [26] for more details), we obtain

$$\langle\mathbf{w}^t - \mathbf{w}^*, \nabla F(\mathbf{w}^t)\rangle \ge \frac{L_F\lambda_F}{L_F+\lambda_F}\|\mathbf{w}^t - \mathbf{w}^*\|_2^2 + \frac{1}{L_F+\lambda_F}\|\nabla F(\mathbf{w}^t)\|_2^2.$$

Let $\eta = \frac{1}{L_F}$. Then we get

$$\begin{aligned}\|\mathbf{w}^t - \eta\nabla F(\mathbf{w}^t) - \mathbf{w}^*\|_2^2 \le &(1 - \frac{2\lambda_F}{L_F+\lambda_F})\|\mathbf{w}^t - \mathbf{w}^*\|_2^2 - \frac{2}{L_F(L_F+\lambda_F)}\|\nabla F(\mathbf{w}^t)\|_2^2 \\ &+ \frac{1}{L_F^2}\|\nabla F(\mathbf{w}^t)\|_2^2 \\ \le &(1 - \frac{2\lambda_F}{L_F+\lambda_F})\|\mathbf{w}^t - \mathbf{w}^*\|_2^2,\end{aligned}$$

where in the second inequality we use the fact that $\lambda_F \le L_F$. Using the fact $\sqrt{1-x} \le 1 - \frac{x}{2}$, we get

$$\|\mathbf{w}^t - \eta\nabla F(\mathbf{w}^t) - \mathbf{w}^*\|_2 \le (1 - \frac{\lambda_F}{L_F+\lambda_F})\|\mathbf{w}^t - \mathbf{w}^*\|_2. \quad (3.30)$$

Combining (3.28) and (3.30), we get

$$\|\mathbf{w}^{t+1} - \mathbf{w}^*\|_2 \leq (1 - \frac{\lambda_F}{L_F + \lambda_F})\|\mathbf{w}^t - \mathbf{w}^*\|_2 + \frac{1}{L_F}\Delta, \tag{3.31}$$

where

$$\Delta = 2\sqrt{2}\frac{1}{nm} + \sqrt{2}\frac{C_\epsilon}{\sqrt{n}}V(\alpha + \sqrt{\frac{d\log(1+nm\widehat{L}D)}{m(1-\alpha)}} + 0.4748\frac{S}{\sqrt{n}}).$$

Then we can complete the proof by iterating (3.31).

Proof of Theorem 3.8

The proof of Theorem 3.8 relies on careful analysis of the median of means estimator in the presence of adversarial data and a covering net argument.

We first consider a general problem of robust estimation of a one dimensional random variable. Suppose that there are m worker machines, and q of them are Byzantine machines, which store n adversarial data (recall that $\alpha := q/m$). Each of the other $m(1-\alpha)$ normal worker machines stores n i.i.d. samples of some one dimensional random variable $x \sim \mathcal{D}$. Denote the j-th sample in the i-th worker machine by $x^{i,j}$. Let $\mu := \mathbb{E}[x]$, $\sigma^2 := \mathrm{Var}(x)$, and $\gamma(x)$ be the absolute skewness of x. In addition, define $\bar{x}^i$ as the average of samples in the i-th machine, i.e., $\bar{x}^i = \frac{1}{n}\sum_{j=1}^n x^{i,j}$. For any $z \in \mathbb{R}$, define $\widetilde{p}(z) := \frac{1}{m(1-\alpha)}\sum_{i\in[m]\setminus\mathcal{B}} \mathbb{1}(\bar{x}^i \leq z)$ as the empirical distribution function of the sample averages on the *normal* worker machines. We have the following result on $\widetilde{p}(z)$.

Lemma 3.1. *Suppose that for a fixed $t > 0$, we have*

$$\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + 0.4748\frac{\gamma(x)}{\sqrt{n}} \leq \frac{1}{2} - \epsilon, \tag{3.32}$$

for some $\epsilon > 0$. Then, with probability at least $1 - 4e^{-2t}$, we have

$$\widetilde{p}\left(\mu + C_\epsilon\frac{\sigma}{\sqrt{n}}(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + 0.4748\frac{\gamma(x)}{\sqrt{n}})\right) \geq \frac{1}{2} + \alpha, \tag{3.33}$$

and

$$\widetilde{p}\left(\mu - C_\epsilon\frac{\sigma}{\sqrt{n}}(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + 0.4748\frac{\gamma(x)}{\sqrt{n}})\right) \leq \frac{1}{2} - \alpha, \tag{3.34}$$

where C_ϵ is defined as in (3.4).

We further define the distribution function of all the m machines as

$$\widehat{p}(z) := \frac{1}{m}\sum_{i\in[m]} \mathbb{1}(\bar{x}^i \leq z).$$

We have the following direct corollary on $\widehat{p}(z)$ and the median of means estimator $\mathsf{med}\{\bar{x}^i : i \in [m]\}$.

Corollary 3.1. *Suppose that condition* (3.32) *is satisfied. Then, with probability at least* $1 - 4e^{-2t}$, *we have,*

$$\widehat{p}\left(\mu + C_\epsilon \frac{\sigma}{\sqrt{n}}(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + 0.4748\frac{\gamma(x)}{\sqrt{n}})\right) \geq \frac{1}{2}, \tag{3.35}$$

and

$$\widehat{p}\left(\mu - C_\epsilon \frac{\sigma}{\sqrt{n}}(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + 0.4748\frac{\gamma(x)}{\sqrt{n}})\right) \leq \frac{1}{2}. \tag{3.36}$$

Thus, we have with probability at least $1 - 4e^{-2t}$,

$$|\mathsf{med}\{\bar{x}^i : i \in [m]\} - \mu| \leq C_\epsilon \frac{\sigma}{\sqrt{n}}(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + 0.4748\frac{\gamma(x)}{\sqrt{n}}). \tag{3.37}$$

Proof. One can easily check that for any $z \in \mathbb{R}$, we have $|\widehat{p}(z) - \widetilde{p}(z)| \leq \alpha$, which yields the results (3.35) and (3.36). The result (3.37) can be derived using the fact that $\widehat{p}(\mathsf{med}\{\bar{x}^i : i \in [m]\}) = 1/2$. $\square$

Lemma 3.1 and Corollary 3.1 can be translated to the estimators of the gradients. Define $\widehat{\mathbf{g}}^i(\mathbf{w})$ and $\widehat{\mathbf{g}}(\mathbf{w})$ as in (3.25) and (3.26), and let $g_k^i(\mathbf{w})$ and $g_k(\mathbf{w})$ be the k-th coordinate of $\widehat{\mathbf{g}}^i(\mathbf{w})$ and $\widehat{\mathbf{g}}(\mathbf{w})$, respectively. In addition, for any $\mathbf{w} \in \mathcal{W}$, $k \in [d]$, and $z \in \mathbb{R}$, we define the empirical distribution function of the k-th coordinate of the gradients on the normal machines:

$$\widetilde{p}(z; \mathbf{w}, k) = \frac{1}{m(1-\alpha)} \sum_{i \in [m] \setminus \mathcal{B}} \mathbb{1}(g_k^i(\mathbf{w}) \leq z), \tag{3.38}$$

and on all the m machines

$$\widehat{p}(z; \mathbf{w}, k) = \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}(g_k^i(\mathbf{w}) \leq z). \tag{3.39}$$

We use the symbol ∂_k to denote the partial derivative of any function with respect to its k-th argument. We also use the simplified notation $\sigma_k^2(\mathbf{w}) := \mathrm{Var}(\partial_k f(\mathbf{w}; \mathbf{z}))$, and $\gamma_k(\mathbf{w}) := \gamma(\partial_k f(\mathbf{w}; \mathbf{z}))$. Then, according to Lemma 3.1, when (3.32) is satisfied, for any fixed $\mathbf{w} \in \mathcal{W}$ and $k \in [d]$, we have with probability at least $1 - 4e^{-2t}$,

$$\widetilde{p}\left(\partial_k F(\mathbf{w}) + C_\epsilon \frac{\sigma_k(\mathbf{w})}{\sqrt{n}}(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + 0.4748\frac{\gamma_k(\mathbf{w})}{\sqrt{n}}); \mathbf{w}, k\right) \geq \frac{1}{2} + \alpha, \tag{3.40}$$

and

$$\widetilde{p}\left(\partial_k F(\mathbf{w}) - C_\epsilon \frac{\sigma_k(\mathbf{w})}{\sqrt{n}}(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + 0.4748\frac{\gamma_k(\mathbf{w})}{\sqrt{n}}); \mathbf{w}, k\right) \le \frac{1}{2} - \alpha. \tag{3.41}$$

Further, according to Corollary 3.1, we know that with probability $1 - 4e^{-2t}$,

$$|g_k(\mathbf{w}) - \partial_k F(\mathbf{w})| \le C_\epsilon \frac{\sigma_k(\mathbf{w})}{\sqrt{n}}(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + 0.4748\frac{\gamma_k(\mathbf{w})}{\sqrt{n}}). \tag{3.42}$$

Here, the inequality (3.42) gives a bound on the accuracy of the median of means estimator for the gradient at any fixed $\mathbf{w}$ and any coordinate $k \in [d]$. To extend this result to all $\mathbf{w} \in \mathcal{W}$ and all the d coordinates, we need to use union bound and a covering net argument.

Let $\mathcal{W}_\delta = \{\mathbf{w}^1, \mathbf{w}^2, \dots, \mathbf{w}^{N_\delta}\}$ be a finite subset of $\mathcal{W}$ such that for any $\mathbf{w} \in \mathcal{W}$, there exists $\mathbf{w}^\ell \in \mathcal{W}_\delta$ such that $\|\mathbf{w}^\ell - \mathbf{w}\|_2 \le \delta$. According to the standard covering net results [182], we know that $N_\delta \le (1 + \frac{D}{\delta})^d$. By union bound, we know that with probability at least $1 - 4dN_\delta e^{-2t}$, the bounds in (3.40) and (3.41) hold for all $\mathbf{w} = \mathbf{w}^\ell \in \mathcal{W}_\delta$, and $k \in [d]$. By gathering all the k coordinates and using Assumption 3.3, we know that this implies for all $\mathbf{w}^\ell \in \mathcal{W}_\delta$,

$$\|\widehat{\mathbf{g}}(\mathbf{w}^\ell) - \nabla F(\mathbf{w}^\ell)\|_2 \le \frac{C_\epsilon}{\sqrt{n}} V \left(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + 0.4748\frac{S}{\sqrt{n}}\right). \tag{3.43}$$

Then, consider an arbitrary $\mathbf{w} \in \mathcal{W}$. Suppose that $\|\mathbf{w}^\ell - \mathbf{w}\|_2 \le \delta$. Since by Assumption 3.1, we assume that for each $k \in [d]$, the partial derivative $\partial_k f(\mathbf{w}; \mathbf{z})$ is L_k-Lipschitz for all $\mathbf{z}$, we know that for every normal machine $i \in [m] \setminus \mathcal{B}$,

$$\left|g_k^i(\mathbf{w}) - g_k^i(\mathbf{w}^\ell)\right| \le L_k \delta.$$

Then, according to the definition of $\widetilde{p}(z; \mathbf{w}, k)$ in (3.38), we know that for any $z \in \mathbb{R}$, $\widetilde{p}(z + L_k\delta; \mathbf{w}, k) \ge \widetilde{p}(z; \mathbf{w}^\ell, k)$ and $\widetilde{p}(z - L_k\delta; \mathbf{w}, k) \le \widetilde{p}(z; \mathbf{w}^\ell, k)$. Then, the bounds in (3.40) and (3.41) yield

$$\widetilde{p}\left(\partial_k F(\mathbf{w}^\ell) + L_k\delta + C_\epsilon \frac{\sigma_k(\mathbf{w}^\ell)}{\sqrt{n}}(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + 0.4748\frac{\gamma_k(\mathbf{w}^\ell)}{\sqrt{n}}); \mathbf{w}, k\right) \ge \frac{1}{2} + \alpha, \tag{3.44}$$

and

$$\widetilde{p}\left(\partial_k F(\mathbf{w}^\ell) - L_k\delta - C_\epsilon \frac{\sigma_k(\mathbf{w}^\ell)}{\sqrt{n}}(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + 0.4748\frac{\gamma_k(\mathbf{w}^\ell)}{\sqrt{n}}); \mathbf{w}, k\right) \le \frac{1}{2} - \alpha. \tag{3.45}$$

Using the fact that $|\partial_k F(\mathbf{w}^\ell) - \partial_k F(\mathbf{w})| \le L_k\delta$, and Corollary 3.1, we have

$$|g_k(\mathbf{w}) - \partial_k F(\mathbf{w})| \le 2L_k\delta + C_\epsilon \frac{\sigma_k(\mathbf{w}^\ell)}{\sqrt{n}}(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + 0.4748\frac{\gamma_k(\mathbf{w}^\ell)}{\sqrt{n}}).$$

Again, by gathering all the k coordinates we get

$$\|\widehat{\mathbf{g}}(\mathbf{w}) - \nabla F(\mathbf{w})\|_2^2 \le 8\delta^2 \sum_{k=1}^{d} L_k^2 + 2\frac{C_\epsilon^2}{n}\sum_{k=1}^{d} \sigma_k^2(\mathbf{w}^\ell)(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + 0.4748\frac{\gamma_k(\mathbf{w}^\ell)}{\sqrt{n}})^2,$$

where we use the fact that $(a+b)^2 \le 2(a^2+b^2)$. Then, by Assumption 3.2 and 3.3, we further obtain

$$\|\widehat{\mathbf{g}}(\mathbf{w}) - \nabla F(\mathbf{w})\|_2 \le 2\sqrt{2}\delta\widehat{L} + \sqrt{2}\frac{C_\epsilon}{\sqrt{n}}V\left(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + 0.4748\frac{S}{\sqrt{n}}\right), \tag{3.46}$$

where we use the fact that $\sqrt{a+b} \le \sqrt{a} + \sqrt{b}$. Combining (3.43) and (3.46), we conclude that for any $\delta > 0$, with probability at least $1 - 4dN_\delta e^{-2t}$, (3.46) holds for all $\mathbf{w} \in \mathcal{W}$. We simply choose $\delta = \frac{1}{nm\widehat{L}}$, and $t = d\log(1 + nm\widehat{L}D)$. Then, we know that with probability at least $1 - \frac{4d}{(1+nm\widehat{L}D)^d}$, we have

$$\|\widehat{\mathbf{g}}(\mathbf{w}) - \nabla F(\mathbf{w})\|_2 \le 2\sqrt{2}\frac{1}{nm} + \sqrt{2}\frac{C_\epsilon}{\sqrt{n}}V\left(\alpha + \sqrt{\frac{d\log(1+nm\widehat{L}D)}{m(1-\alpha)}} + 0.4748\frac{S}{\sqrt{n}}\right)$$

for all $\mathbf{w} \in \mathcal{W}$.

Proof of Lemma 3.1

We recall the Berry-Esseen Theorem [19, 64, 166] and the bounded difference inequality, which are useful in this proof.

Theorem 3.9 (Berry-Esseen Theorem)**.** *Assume that $Y_1, \dots, Y_n$ are i.i.d. copies of a random variable Y with mean μ, variance σ^2, and such that $\mathbb{E}[|Y-\mu|^3] < \infty$. Then,*

$$\sup_{s\in\mathbb{R}} \left| \mathbb{P}\left\{\sqrt{n}\frac{\bar{Y}-\mu}{\sigma} \le s\right\} - \Phi(s) \right| \le 0.4748\frac{\mathbb{E}[|Y-\mu|^3]}{\sigma^3\sqrt{n}},$$

where $\bar{Y} = \frac{1}{n}\sum_{i=1}^n Y_i$ and $\Phi(s)$ is the cumulative distribution function of the standard normal random variable.

Theorem 3.10 (Bounded Difference Inequality)**.** *Let $X_1, \dots, X_n$ be i.i.d. random variables, and assume that $Z = g(X_1, \dots, X_n)$, where g satisfies that for all $j \in [n]$ and all $x_1, x_2, \dots, x_j, x_j', \dots, x_n$,*

$$|g(x_1, \dots, x_{j-1}, x_j, x_{j+1}, \dots, x_n) - g(x_1, \dots, x_{j-1}, x_j', x_{j+1}, \dots, x_n)| \le c_j.$$

Then for any $t \ge 0$,

$$\mathbb{P}\{Z - \mathbb{E}[Z] \ge t\} \le \exp\left(-\frac{2t^2}{\sum_{j=1}^n c_j^2}\right)$$

and

$$\mathbb{P}\left\{Z - \mathbb{E}[Z] \le -t\right\} \le \exp\left(-\frac{2t^2}{\sum_{j=1}^n c_j^2}\right).$$

Let $\sigma_n := \frac{\sigma}{\sqrt{n}}$ and $c_n := 0.4748\frac{\mathbb{E}[|x-\mu|^3]}{\sigma^3\sqrt{n}} = 0.4748\frac{\gamma(x)}{\sqrt{n}}$. Define $W_i := \frac{\bar{x}^i - \mu}{\sigma_n}$ for all $i \in [m]$, and $\Phi_n(\cdot)$ be the distribution function of W_i for any $i \in [m] \setminus \mathcal{B}$. We also define the empirical distribution function of $\{W_i : i \in [m] \setminus \mathcal{B}\}$ as $\widetilde{\Phi}_n(\cdot)$, i.e., $\widetilde{\Phi}_n(z) = \frac{1}{m(1-\alpha)} \sum_{i \in [m]\setminus\mathcal{B}} \mathbb{1}(W_i \le z)$. Thus, we have

$$\widetilde{\Phi}_n(z) = \widetilde{p}(\sigma_n z + \mu). \tag{3.47}$$

We then focus on $\widetilde{\Phi}_n(z)$. We know that for any $z \in \mathbb{R}$, $\mathbb{E}[\widetilde{\Phi}_n(z)] = \Phi_n(z)$. Then, since the bounded difference inequality is satisfied with $c_j = \frac{1}{m(1-\alpha)}$, we have for any $t > 0$,

$$\left|\widetilde{\Phi}_n(z) - \Phi_n(z)\right| \le \sqrt{\frac{t}{m(1-\alpha)}}, \tag{3.48}$$

on the draw of W_i, $i \in [m] \setminus \mathcal{B}$ with probability at least $1 - 2e^{-2t}$. Let $z_1 \ge z_2$ be such that $\Phi_n(z_1) \ge \frac{1}{2} + \alpha + \sqrt{\frac{t}{m(1-\alpha)}}$, and $\Phi_n(z_2) \le \frac{1}{2} - \alpha - \sqrt{\frac{t}{m(1-\alpha)}}$. Then, by union bound, we know that with probability at least $1 - 4e^{-2t}$, $\widetilde{\Phi}_n(z_1) \ge 1/2 + \alpha$ and $\widetilde{\Phi}_n(z_2) \le 1/2 - \alpha$. The next step is to choose z_1 and z_2. According to Theorem 3.9, we know that

$$\Phi_n(z_1) \ge \Phi(z_1) - c_n,$$

and thus, it suffices to find z_1 such that

$$\Phi(z_1) = \frac{1}{2} + \alpha + \sqrt{\frac{t}{m(1-\alpha)}} + c_n.$$

By mean value theorem, we know that there exists $\xi \in [0, z_1]$ such that

$$\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + c_n = z_1 \Phi'(\xi) = \frac{z_1}{\sqrt{2\pi}} e^{-\frac{\xi^2}{2}} \ge \frac{z_1}{\sqrt{2\pi}} e^{-\frac{z_1^2}{2}}$$

Suppose that for some fix constant $\epsilon \in (0, 1/2)$, we have

$$\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + c_n \le \frac{1}{2} - \epsilon.$$

Then, we know that $z_1 \le \Phi^{-1}(1-\epsilon)$, and thus we have

$$\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + c_n \ge \frac{z_1}{\sqrt{2\pi}} \exp(-\frac{1}{2}(\Phi^{-1}(1-\epsilon))^2),$$

which yields

$$z_1 \leq \sqrt{2\pi}\exp(\frac{1}{2}(\Phi^{-1}(1-\epsilon))^2)\left(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + c_n\right).$$

Similarly

$$z_2 \geq -\sqrt{2\pi}\exp(\frac{1}{2}(\Phi^{-1}(1-\epsilon))^2)\left(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + c_n\right).$$

For simplicity, let $C_\epsilon := \sqrt{2\pi}\exp(\frac{1}{2}(\Phi^{-1}(1-\epsilon))^2)$. We conclude that with probability $1-4e^{-2t}$, we have

$$\widetilde{p}(\mu + C_\epsilon\sigma_n(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + c_n)) \geq \frac{1}{2} + \alpha,$$

and

$$\widetilde{p}(\mu - C_\epsilon\sigma_n(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + c_n)) \leq \frac{1}{2} - \alpha.$$

Proof of Theorem 3.2

Since Theorem 3.8 holds without assuming the convexity of $F(\mathbf{w})$, when $F(\mathbf{w})$ is non-strongly convex, the event that (3.27) holds for all $\mathbf{w} \in \mathcal{W}$ still happens with probability at least $1 - \frac{4d}{(1+nm\widehat{L}D)^d}$. We condition on this event. We first show that when Assumption 3.4 is satisfied and we choose $\eta = \frac{1}{L_F}$, the iterates $\mathbf{w}^t$ stays in $\mathcal{W}$ without using projection. Namely, define

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta\widehat{\mathbf{g}}(\mathbf{w}^t),$$

for $T = 0, 1, \ldots, T-1$, then $\mathbf{w}^t \in \mathcal{W}$ for all $t = 0, 1, \ldots, T$. To see this, we have

$$\|\mathbf{w}^{t+1} - \mathbf{w}^*\|_2 \leq \|\mathbf{w}^t - \eta\nabla F(\mathbf{w}^t) - \mathbf{w}^*\|_2 + \eta\|\widehat{\mathbf{g}}(\mathbf{w}^t) - \nabla F(\mathbf{w}^t)\|_2,$$

and

$$\begin{aligned}\|\mathbf{w}^t - \eta\nabla F(\mathbf{w}^t) - \mathbf{w}^*\|_2^2 &= \|\mathbf{w}^t - \mathbf{w}^*\|_2^2 - 2\eta\langle\nabla F(\mathbf{w}^t), \mathbf{w}^t - \mathbf{w}^*\rangle + \eta^2\|\nabla F(\mathbf{w}^t)\|_2^2 \\ &\leq \|\mathbf{w}^t - \mathbf{w}^*\|_2^2 - 2\eta\frac{1}{L_F}\|\nabla F(\mathbf{w}^t)\|_2^2 + \eta^2\|\nabla F(\mathbf{w}^t)\|_2^2 \\ &= \|\mathbf{w}^t - \mathbf{w}^*\|_2^2 - \frac{1}{L_F^2}\|\nabla F(\mathbf{w}^t)\|_2^2 \\ &\leq \|\mathbf{w}^t - \mathbf{w}^*\|_2^2\end{aligned}$$

where the inequality is due to the co-coercivity of convex functions. Thus, we get

$$\|\mathbf{w}^{t+1} - \mathbf{w}^*\|_2 \leq \|\mathbf{w}^t - \mathbf{w}^*\|_2 + \frac{\Delta}{L_F},$$

and since $T = \frac{L_F D_0}{\Delta}$, according to Assumption 3.4 we know that $\mathbf{w}^t \in \mathcal{W}$ for all $t = 0, 1, \dots, T$. Then, we proceed to study the algorithm without projection. Here, we define $D_t := \|\mathbf{w}^0 - \mathbf{w}^*\|_2 + \frac{t\Delta}{L_F}$ for $t = 0, 1, \dots, T$.

Using the smoothness of $F(\mathbf{w})$, we have

$$\begin{aligned} F(\mathbf{w}^{t+1}) &\le F(\mathbf{w}^t) + \langle \nabla F(\mathbf{w}^t), \mathbf{w}^{t+1} - \mathbf{w}^t \rangle + \frac{L_F}{2}\|\mathbf{w}^{t+1} - \mathbf{w}^t\|_2^2 \\ &= F(\mathbf{w}^t) + \eta\langle \nabla F(\mathbf{w}^t), -\widehat{\mathbf{g}}(\mathbf{w}^t) + \nabla F(\mathbf{w}^t) - \nabla F(\mathbf{w}^t)\rangle \\ &\quad + \eta^2 \frac{L_F}{2}\|\widehat{\mathbf{g}}(\mathbf{w}^t) - \nabla F(\mathbf{w}^t) + \nabla F(\mathbf{w}^t)\|_2^2. \end{aligned}$$

Since $\eta = \frac{1}{L_F}$ and $\|\widehat{\mathbf{g}}(\mathbf{w}^t) - \nabla F(\mathbf{w}^t)\|_2 \le \Delta$, by simple algebra, we obtain

$$F(\mathbf{w}^{t+1}) \le F(\mathbf{w}^t) - \frac{1}{2L_F}\|\nabla F(\mathbf{w}^t)\|_2^2 + \frac{1}{2L_F}\Delta^2. \tag{3.49}$$

We now prove the following lemma.

Lemma 3.2. *Condition on the event that* (3.27) *holds for all* $\mathbf{w} \in \mathcal{W}$. *When* $F(\mathbf{w})$ *is convex, by running* $T = \frac{L_F D_0}{\Delta}$ *parallel iterations, there exists* $t \in \{0, 1, 2, \dots, T\}$ *such that*

$$F(\mathbf{w}^t) - F(\mathbf{w}^*) \le 16 D_0 \Delta.$$

Proof. We first notice that since $T = \frac{L_F D_0}{\Delta}$, we have $D_t \le 2D_0$ for all $t = 0, 1, \dots, T$. According to the first order optimality of convex functions, for any $\mathbf{w}$,

$$F(\mathbf{w}) - F(\mathbf{w}^*) \le \langle \nabla F(\mathbf{w}), \mathbf{w} - \mathbf{w}^* \rangle \le \|\nabla F(\mathbf{w})\|_2 \|\mathbf{w} - \mathbf{w}^*\|_2,$$

and thus

$$\|\nabla F(\mathbf{w})\|_2 \ge \frac{F(\mathbf{w}) - F(\mathbf{w}^*)}{\|\mathbf{w} - \mathbf{w}^*\|_2}. \tag{3.50}$$

Suppose that there exists $t \in \{0, 1, \dots, T-1\}$ such that $\|\nabla F(\mathbf{w}^t)\|_2 < \sqrt{2}\Delta$. Then we have

$$F(\mathbf{w}^t) - F(\mathbf{w}^*) \le \|\nabla F(\mathbf{w}^t)\|_2 \|\mathbf{w}^t - \mathbf{w}^*\|_2 \le 2\sqrt{2} D_0 \Delta.$$

Otherwise, for all $t \in \{0, 1, \dots, T-1\}$, $\|\nabla F(\mathbf{w}^t)\|_2 \ge \sqrt{2}\Delta$. Then, according to (3.49) and (3.50), we have for all $t < T$,

$$\begin{aligned} F(\mathbf{w}^{t+1}) - F(\mathbf{w}^*) &\le F(\mathbf{w}^t) - F(\mathbf{w}^*) - \frac{1}{4L_F}\|\nabla F(\mathbf{w}^t)\|_2^2 \\ &\le F(\mathbf{w}^t) - F(\mathbf{w}^*) - \frac{1}{4L_F D_t^2}(F(\mathbf{w}^t) - F(\mathbf{w}^*))^2. \end{aligned}$$

Multiplying both sides by $[(F(\mathbf{w}^{t+1}) - F(\mathbf{w}^*))(F(\mathbf{w}^t) - F(\mathbf{w}^*)]^{-1}$ and rearranging the terms, we obtain

$$\begin{aligned}\frac{1}{F(\mathbf{w}^{t+1}) - F(\mathbf{w}^*)} \geq & \frac{1}{F(\mathbf{w}^t) - F(\mathbf{w}^*)} \\ & + \frac{1}{4L_F D_t^2} \frac{F(\mathbf{w}^t) - F(\mathbf{w}^*)}{F(\mathbf{w}^{t+1}) - F(\mathbf{w}^*)} \geq \frac{1}{F(\mathbf{w}^t) - F(\mathbf{w}^*)} + \frac{1}{16L_F D_0^2},\end{aligned}$$

which implies

$$\frac{1}{F(\mathbf{w}^T) - F(\mathbf{w}^*)} \geq \frac{1}{F(\mathbf{w}^0) - F(\mathbf{w}^*)} + \frac{T}{16L_F D_0^2} \geq \frac{T}{16L_F D_0^2}.$$

Then, we obtain $F(\mathbf{w}^T) - F(\mathbf{w}^*) \leq 16D_0\Delta$ using the fact that $T = \frac{L_F D_0}{\Delta}$. □

Next, we show that $F(\mathbf{w}^T) - F(\mathbf{w}^*) \leq 16D_0\Delta + \frac{1}{2L_F}\Delta^2$. More specifically, let $t = t_0$ be the first time that $F(\mathbf{w}^t) - F(\mathbf{w}^*) \leq 16D_0\Delta$, and we show that for any $t > t_0$, $F(\mathbf{w}^t) - F(\mathbf{w}^*) \leq 16D_0\Delta + \frac{1}{2L_F}\Delta^2$. If this statement is not true, then we let $t_1 > t_0$ be the first time that $F(\mathbf{w}^t) - F(\mathbf{w}^*) > 16D_0\Delta + \frac{1}{2L_F}\Delta^2$. Then there must be $F(\mathbf{w}^{t_1-1}) < F(\mathbf{w}^{t_1})$. According to (3.49), there should also be

$$F(\mathbf{w}^{t_1-1}) - F(\mathbf{w}^*) \geq F(\mathbf{w}^{t_1}) - F(\mathbf{w}^*) - \frac{1}{2L_F}\Delta^2 > 16D_0\Delta.$$

Then, according to (3.50), we have

$$\|\nabla F(\mathbf{w}^{t_1-1})\|_2 \geq \frac{F(\mathbf{w}^{t_1-1}) - F(\mathbf{w}^*)}{\|\mathbf{w}^{t_1-1} - \mathbf{w}^*\|_2} > 8\Delta.$$

Then according to (3.49), this implies $F(\mathbf{w}^{t_1}) \leq F(\mathbf{w}^{t_1-1})$, which contradicts with the fact that $F(\mathbf{w}^{t_1-1}) < F(\mathbf{w}^{t_1})$.

Proof of Theorem 3.3

Since Theorem 3.8 holds without assuming the convexity of $F(\mathbf{w})$, when $F(\mathbf{w})$ is non-convex, the event that (3.27) holds for all $\mathbf{w} \in \mathcal{W}$ still happens with probability at least $1 - \frac{4d}{(1+nm\widehat{L}D)^d}$. We condition on this event. We first show that when Assumption 3.5 is satisfied and we choose $\eta = \frac{1}{L_F}$, the iterates $\mathbf{w}^t$ stays in $\mathcal{W}$ without using projection. Since we have

$$\|\mathbf{w}^{t+1} - \mathbf{w}^*\|_2 \leq \|\mathbf{w}^t - \mathbf{w}^*\|_2 + \eta(\|\nabla F(\mathbf{w}^t)\|_2 + \|\widehat{\mathbf{g}}(\mathbf{w}^t) - \nabla F(\mathbf{w}^t)\|_2) \leq \|\mathbf{w}^t - \mathbf{w}^*\|_2 + \frac{1}{L_F}(M + \Delta).$$

Then, we know that by running $T = \frac{2L_F}{\Delta^2}(F(\mathbf{w}^0) - F(\mathbf{w}^*))$ parallel iterations, using Assumption 3.5, we know that $\mathbf{w}^t \in \mathcal{W}$ for $t = 0, 1, \ldots, T$ without projection.

We proceed to study the convergence rate of the algorithm. By the smoothness of $F(\mathbf{w})$, we know that when choosing $\eta = \frac{1}{L_F}$, the inequality (3.49) still holds. More specifically, for all $t = 0, 1, \ldots, T-1$,

$$F(\mathbf{w}^{t+1}) - F(\mathbf{w}^*) \leq F(\mathbf{w}^t) - F(\mathbf{w}^*) - \frac{1}{2L_F}\|\nabla F(\mathbf{w}^t)\|_2^2 + \frac{1}{2L_F}\Delta^2. \tag{3.51}$$

Sum up (3.51) for $t = 0, 1, \ldots, T-1$. Then, we get

$$0 \leq F(\mathbf{w}^T) - F(\mathbf{w}^*) \leq F(\mathbf{w}^0) - F(\mathbf{w}^*) - \frac{1}{2L_F}\sum_{t=0}^{T-1}\|\nabla F(\mathbf{w}^t)\|_2^2 + \frac{T}{2L_F}\Delta^2.$$

This implies that

$$\min_{t=0,1,\ldots,T}\|\nabla F(\mathbf{w}^t)\|_2^2 \leq 2\frac{L_F}{T}(F(\mathbf{w}^0) - F(\mathbf{w}^*)) + \Delta^2,$$

which completes the proof.

Proof of Theorem 3.4

The proof of Theorem 3.4 consists of two parts: 1) the analysis of coordinate-wise trimmed mean of means estimator of the population gradients, and 2) the convergence analysis of the robustified gradient descent algorithm. Since the second part is essentially the same as the proof of Theorem 3.1, we mainly focus on the first part here.

Theorem 3.11. *Define*

$$\widehat{\mathbf{g}}^i(\mathbf{w}) = \begin{cases} \nabla F_i(\mathbf{w}) & i \in [m] \setminus \mathcal{B}, \\ * & i \in \mathcal{B}. \end{cases} \tag{3.52}$$

and the coordinate-wise trimmed mean of $\widehat{\mathbf{g}}^i(\mathbf{w})$*:*

$$\widehat{\mathbf{g}}(\mathbf{w}) = \textsf{trmean}_\beta\{\widehat{\mathbf{g}}^i(\mathbf{w}) : i \in [m]\}. \tag{3.53}$$

Suppose that Assumptions 3.1 and 3.6 are satisfied, and that $\alpha \leq \beta \leq \frac{1}{2} - \epsilon$*. Then, with probability at least* $1 - \frac{2d(m+1)}{(1+nm\widehat{L}D)^d}$*,*

$$\|\widehat{\mathbf{g}}(\mathbf{w}) - \nabla F(\mathbf{w})\|_2 \leq \frac{v}{\epsilon}\left(\frac{3\sqrt{2}\beta d}{\sqrt{n}} + \frac{2d}{\sqrt{nm}}\right)\sqrt{\log(1 + nm\widehat{L}D) + \frac{1}{d}\log m} + \widetilde{\mathcal{O}}(\frac{\beta}{n} + \frac{1}{nm})$$

for all $\mathbf{w} \in \mathcal{W}$*.*

The rest of the proof is essentially the same as the proof of Theorem 3.1. In fact, we essentially analyze a gradient descent algorithm with bounded noise in the gradients. In the proof of Theorem 3.1 in Section 3.9. The bound on the noise in the gradients is

$$\Delta = \sqrt{2}\frac{C_\epsilon}{\sqrt{n}}V(\alpha + \sqrt{\frac{d\log(1+nm\widehat{L}D)}{m(1-\alpha)}} + 0.4748\frac{S}{\sqrt{n}}) + 2\sqrt{2}\frac{1}{nm},$$

while here we replace Δ with Δ':

$$\Delta' := \frac{v}{\epsilon}\left(\frac{3\sqrt{2}\beta d}{\sqrt{n}} + \frac{2d}{\sqrt{nm}}\right)\sqrt{\log(1+nm\widehat{L}D) + \frac{1}{d}\log m} + \widetilde{\mathcal{O}}(\frac{\beta}{n} + \frac{1}{nm}),$$

and the same analysis can still go through. Therefore, we omit the details of the analysis here.

Remark 3.1. *The same arguments still go through when the population risk function $F(\mathbf{w})$ is non-strongly convex or non-convex. One can simply replace the bound on the noise in the gradients Δ in Theorems 3.2 and 3.3 with Δ' here. Thus we omit the details here.*

Proof of Theorem 3.11

The proof of Theorem 3.11 relies on the analysis of the trimmed mean of means estimator in the presence of adversarial data and a covering net argument. We first consider a general problem of robust estimation of a one dimensional random variable. Suppose that there are m worker machines, and q of them are Byzantine machines, which store n adversarial data (recall that $\alpha := q/m$). Each of the other $m(1-\alpha)$ normal worker machines stores n i.i.d. samples of some one dimensional random variable $x \sim \mathcal{D}$. Suppose that x is v-sub-exponential and let $\mu := \mathbb{E}[x]$. Denote the j-th sample in the i-th worker machine by $x^{i,j}$. In addition, define $\bar{x}^i$ as the average of samples in the i-th machine, i.e., $\bar{x}^i = \frac{1}{n}\sum_{j=1}^n x^{i,j}$. We have the following result on the trimmed mean of $\bar{x}^i$, $i \in [m]$.

Lemma 3.3. *Suppose that the one dimensional samples on all the normal machines are i.i.d. v-sub-exponential with mean μ. Then, we have for any $t \geq 0$,*

$$\mathbb{P}\{|\frac{1}{(1-\alpha)m}\sum_{i\in[m]\setminus\mathcal{B}}\bar{x}^i - \mu| \geq t\} \leq 2\exp\{-(1-\alpha)mn\min\{\frac{t}{2v}, \frac{t^2}{2v^2}\}\},$$

and for any $s \geq 0$,

$$\mathbb{P}\{\max_{i\in[m]\setminus\mathcal{B}}\{|\bar{x}^i - \mu|\} \geq s\} \leq 2(1-\alpha)m\exp\{-n\min\{\frac{s}{2v}, \frac{s^2}{2v^2}\}\},$$

and when $\beta \geq \alpha$, $|\frac{1}{(1-\alpha)m}\sum_{i\in[m]\setminus\mathcal{B}}\bar{x}^i - \mu| \leq t$, and $\max_{i\in[m]\setminus\mathcal{B}}\{|\bar{x}^i - \mu|\} \leq s$, we have

$$|\mathsf{trmean}_\beta\{\bar{x}^i : i \in [m]\} - \mu| \leq \frac{t + 3\beta s}{1-2\beta}.$$

Lemma 3.3 can be directly applied to the k-th partial derivative of the loss functions. Since we assume that for any $k \in [d]$ and $\mathbf{w} \in \mathcal{W}$, $\partial_k f(\mathbf{w}; \mathbf{z})$ is v-sub-exponential, we have for any $t \geq 0$, $s \geq 0$,

$$\mathbb{P}\{|\frac{1}{(1-\alpha)m} \sum_{i\in[m]\setminus\mathcal{B}} g_k^i(\mathbf{w}) - \partial_k F(\mathbf{w})| \geq t\} \leq 2\exp\{-(1-\alpha)mn \min\{\frac{t}{2v}, \frac{t^2}{2v^2}\}\}, \tag{3.54}$$

$$\mathbb{P}\{\max_{i\in[m]\setminus\mathcal{B}} \{|g_k^i(\mathbf{w}) - \partial_k F(\mathbf{w})|\} \geq s\} \leq 2(1-\alpha)m \exp\{-n \min\{\frac{s}{2v}, \frac{s^2}{2v^2}\}\}, \tag{3.55}$$

and consequently with probability at least

$$1 - 2\exp\{-(1-\alpha)mn \min\{\frac{t}{2v}, \frac{t^2}{2v^2}\}\} - 2(1-\alpha)m \exp\{-n \min\{\frac{s}{2v}, \frac{s^2}{2v^2}\}\},$$

we have

$$|g_k(\mathbf{w}) - \partial_k F(\mathbf{w})| = |\mathsf{trmean}_\beta\{g_k^i(\mathbf{w}) : i \in [m]\} - \partial_k F(\mathbf{w})| \leq \frac{t + 3\beta s}{1 - 2\beta}. \tag{3.56}$$

To extend this result to all $\mathbf{w} \in \mathcal{W}$ and all the d coordinates, we need to use union bound and a covering net argument. Let $\mathcal{W}_\delta = \{\mathbf{w}^1, \mathbf{w}^2, \ldots, \mathbf{w}^{N_\delta}\}$ be a finite subset of $\mathcal{W}$ such that for any $\mathbf{w} \in \mathcal{W}$, there exists $\mathbf{w}^\ell \in \mathcal{W}_\delta$ such that $\|\mathbf{w}^\ell - \mathbf{w}\|_2 \leq \delta$. According to the standard covering net results [182], we know that $N_\delta \leq (1 + \frac{D}{\delta})^d$. By union bound, we know that with probability at least

$$1 - 2dN_\delta \exp\{-(1-\alpha)mn \min\{\frac{t}{2v}, \frac{t^2}{2v^2}\}\},$$

the bound $|\frac{1}{(1-\alpha)m} \sum_{i\in[m]\setminus\mathcal{B}} g_k^i(\mathbf{w}) - \partial_k F(\mathbf{w})| \leq t$ holds for all $\mathbf{w} = \mathbf{w}^\ell \in \mathcal{W}_\delta$, and $k \in [d]$, and with probability at least

$$1 - 2(1-\alpha)dmN_\delta \exp\{-n \min\{\frac{s}{2v}, \frac{s^2}{2v^2}\}\}$$

the bound $\max_{i\in[m]\setminus\mathcal{B}}\{|g_k^i(\mathbf{w}) - \partial_k F(\mathbf{w})|\} \leq s$ holds for all $\mathbf{w} = \mathbf{w}^\ell \in \mathcal{W}_\delta$, and $k \in [d]$. By gathering all the k coordinates, we know that this implies for all $\mathbf{w}^\ell \in \mathcal{W}_\delta$,

$$\|\widehat{\mathbf{g}}(\mathbf{w}^\ell) - \nabla F(\mathbf{w}^\ell)\|_2 \leq \sqrt{d}\frac{t + 3\beta s}{1 - 2\beta}. \tag{3.57}$$

Then, consider an arbitrary $\mathbf{w} \in \mathcal{W}$. Suppose that $\|\mathbf{w}^\ell - \mathbf{w}\|_2 \leq \delta$. Since by Assumption 3.1, we assume that for each $k \in [d]$, the partial derivative $\partial_k f(\mathbf{w}; \mathbf{z})$ is L_k-Lipschitz for all $\mathbf{z}$, we know that for every normal machine $i \in [m] \setminus \mathcal{B}$,

$$\left|g_k^i(\mathbf{w}) - g_k^i(\mathbf{w}^\ell)\right| \leq L_k\delta, \quad \left|\partial_k F(\mathbf{w}) - \partial_k F(\mathbf{w}^\ell)\right| \leq L_k\delta.$$

This means that if

$$|\frac{1}{(1-\alpha)m}\sum_{i\in[m]\setminus\mathcal{B}} g_k^i(\mathbf{w}^\ell) - \partial_k F(\mathbf{w}^\ell)| \le t$$

and

$$\max_{i\in[m]\setminus\mathcal{B}}\{|g_k^i(\mathbf{w}^\ell) - \partial_k F(\mathbf{w}^\ell)|\} \le$$

hold for all $\mathbf{w}^\ell \in \mathcal{W}_\delta$, and $k \in [d]$, then

$$|\frac{1}{(1-\alpha)m}\sum_{i\in[m]\setminus\mathcal{B}} g_k^i(\mathbf{w}) - \partial_k F(\mathbf{w})| \le t + 2L_k\delta,$$

and

$$\max_{i\in[m]\setminus\mathcal{B}}\{|g_k^i(\mathbf{w}) - \partial_k F(\mathbf{w})|\} \le s + 2L_k\delta$$

hold for all $\mathbf{w} \in \mathcal{W}$. This implies that for all $\mathbf{w} \in \mathcal{W}$ and $k \in [d]$,

$$|g_k(\mathbf{w}) - \partial_k F(\mathbf{w})| = |\mathsf{trmean}_\beta\{g_k^i(\mathbf{w}) : i \in [m]\} - \partial_k F(\mathbf{w})| \le \frac{t + 3\beta s}{1-2\beta} + \frac{2(1+3\beta)}{1-2\beta}\delta L_k,$$

which yields

$$\|\widehat{\mathbf{g}}(\mathbf{w}) - \nabla F(\mathbf{w})\|_2 \le \sqrt{2d}\frac{t+3\beta s}{1-2\beta} + \sqrt{2}\frac{2(1+3\beta)}{1-2\beta}\delta\widehat{L}.$$

The proof is completed by choosing $\delta = \frac{1}{nm\widehat{L}}$,

$$t = v\max\{\frac{8d}{nm}\log(1+nm\widehat{L}D), \sqrt{\frac{8d}{nm}\log(1+nm\widehat{L}D)}\},$$
$$s = v\max\{\frac{4}{n}(d\log(1+nm\widehat{L}D)+\log m), \sqrt{\frac{4}{n}(d\log(1+nm\widehat{L}D)+\log m)}\},$$

and using the fact that $\beta \le \frac{1}{2} - \epsilon$.

Proof of Lemma 3.3

We first recall Bernstein's inequality for sub-exponential random variables.

Theorem 3.12 (Bernstein's inequality)**.** *Suppose that $X_1, X_2, \dots, X_n$ are i.i.d. random variables with mean μ and are v-sub-exponential. Then for any $t \ge 0$,*

$$\mathbb{P}\{|\frac{1}{n}\sum_{i=1}^n X_i - \mu| \ge t\} \le 2\exp\{-n\min\{\frac{t}{2v}, \frac{t^2}{2v^2}\}\}.$$

Thus, for any $t \geq 0$

$$\mathbb{P}\{|\frac{1}{(1-\alpha)m}\sum_{i\in[m]\setminus\mathcal{B}}\bar{x}^i - \mu| \geq t\} \leq 2\exp\{-(1-\alpha)mn\min\{\frac{t}{2v}, \frac{t^2}{2v^2}\}\}. \tag{3.58}$$

Similarly, for any $i \in [m]\setminus\mathcal{B}$, and any $s \geq 0$

$$\mathbb{P}\{|\bar{x}^i - \mu| \geq s\} \leq 2\exp\{-n\min\{\frac{s}{2v}, \frac{s^2}{2v^2}\}\}.$$

Then, by union bound we know that

$$\mathbb{P}\{\max_{i\in[m]\setminus\mathcal{B}}\{|\bar{x}^i - \mu|\} \geq s\} \leq 2(1-\alpha)m\exp\{-n\min\{\frac{s}{2v}, \frac{s^2}{2v^2}\}\}. \tag{3.59}$$

We proceed to analyze the trimmed mean of means estimator. To simplify notation, we define $\mathcal{M} = [m]\setminus\mathcal{B}$ as the set of all normal worker machines, $\mathcal{U} \subseteq [m]$ as the set of all untrimmed machines, and $\mathcal{T} \subseteq [m]$ as the set of all trimmed machines. The trimmed mean of means estimator simply computes

$$\mathsf{trmean}_\beta\{\bar{x}^i : i \in [m]\} = \frac{1}{(1-2\beta)m}\sum_{i\in\mathcal{U}}\bar{x}^i.$$

We further have

$$\begin{aligned}
&|\mathsf{trmean}_\beta\{\bar{x}^i : i \in [m]\} - \mu| \\
=&\left|\frac{1}{(1-2\beta)m}\sum_{i\in\mathcal{U}}\bar{x}^i - \mu\right| \\
=&\frac{1}{(1-2\beta)m}\left|\sum_{i\in\mathcal{M}}(\bar{x}^i - \mu) - \sum_{i\in\mathcal{M}\cap\mathcal{T}}(\bar{x}^i - \mu) + \sum_{i\in\mathcal{B}\cap\mathcal{U}}(\bar{x}^i - \mu)\right| \\
=&\frac{1}{(1-2\beta)m}\big(|\sum_{i\in\mathcal{M}}(\bar{x}^i - \mu)| + |\sum_{i\in\mathcal{M}\cap\mathcal{T}}(\bar{x}^i - \mu)| + |\sum_{i\in\mathcal{B}\cap\mathcal{U}}(\bar{x}^i - \mu)|\big)
\end{aligned} \tag{3.60}$$

We also know that $|\sum_{i\in\mathcal{M}\cap\mathcal{T}}(\bar{x}^i - \mu)| \leq 2\beta m\max_{i\in\mathcal{M}}\{|\bar{x}^i - \mu|\}$. In addition, since $\beta \geq \alpha$, without loss of generality, we assume that $\mathcal{M}\cap\mathcal{T} \neq \emptyset$, and then $|\sum_{i\in\mathcal{B}\cap\mathcal{U}}(\bar{x}^i - \mu)| \leq \alpha m\max_{i\in\mathcal{M}}\{|\bar{x}^i - \mu|\}$. Then we directly obtain the desired result.

Proof of Theorem 3.7

Since the loss functions are quadratic, we denote the loss function $f(\mathbf{w};\mathbf{z}^{i,j})$ by

$$f(\mathbf{w};\mathbf{z}^{i,j}) = \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{H}_{i,j}\mathbf{w} + \mathbf{p}_{i,j}^{\mathrm{T}}\mathbf{w} + c_{i,j}.$$

We further define $\mathbf{H}_i := \frac{1}{n}\sum_{j=1}^n \mathbf{H}_{i,j}$, $\mathbf{p}_i := \frac{1}{n}\sum_{j=1}^n \mathbf{p}_{i,j}$, and $c_i := \frac{1}{n}\sum_{j=1}^n c_{i,j}$. Thus the empirical risk function on the i-th machine is

$$F_i(\mathbf{w}) = \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{H}_i\mathbf{w} + \mathbf{p}_i^{\mathrm{T}}\mathbf{w} + c_i.$$

Then, for any worker machine $i \in [m] \setminus \mathcal{B}$, $\widehat{\mathbf{w}}^i = -\mathbf{H}_i^{-1}\mathbf{p}_i$. In addition, the population risk minimizer is $\mathbf{w}^* = -\mathbf{H}_F^{-1}\mathbf{p}_F$. We further define $\mathbf{U}_{i,j} := \mathbf{H}_{i,j} - \mathbf{H}_F$, $\mathbf{U}_i = \mathbf{H}_i - \mathbf{H}_F$, $\mathbf{v}_{i,j} = \mathbf{p}_{i,j} - \mathbf{p}_F$, and $\mathbf{v}_i = \mathbf{p}_i - \mathbf{p}_F$. Then

$$\widehat{\mathbf{w}}^i = -(\mathbf{U}_i + \mathbf{H}_F)^{-1}(\mathbf{v}_i + \mathbf{p}_F).$$

Let $\mathbf{e}_k$ be the k-th vector in the standard basis, i.e., the k-th column of the $d \times d$ identity matrix. We proceed to study the distribution of the k-th coordinate of $\widehat{\mathbf{w}}^i - \mathbf{w}^*$, $i \in [m] \setminus \mathcal{B}$, i.e.,

$$\widehat{w}_k^i - w_k^* = \mathbf{e}_k^{\mathrm{T}}\mathbf{H}_F^{-1}\mathbf{p}_F - \mathbf{e}_k^{\mathrm{T}}(\mathbf{U}_i + \mathbf{H}_F)^{-1}(\mathbf{v}_i + \mathbf{p}_F).$$

Similar to the proof of Theorem 3.1, we need to obtain a Berry-Esseen type bound for $\widehat{w}_k^i - w_k^*$. However, here, $\widehat{w}_k^i$ is not a sample mean of n i.i.d. random variables, and thus we cannot directly apply the vanilla Berry-Esseen bound. Instead, we apply the following bound in [150] on functions of sample means.

Theorem 3.13 (Theorem 2.11 in [150], simplified)**.** *Let $\mathcal{X}$ be a Hilbert space equipped with norm $\|\cdot\|$. Let $f : \mathcal{X} \to \mathbb{R}$ be a function on $\mathcal{X}$. Suppose that there exists linear functions $\ell : \mathcal{X} \to \mathbb{R}$, $\theta > 0$, $M_\theta > 0$ such that*

$$|f(X) - \ell(X)| \le \frac{M_\theta}{2}\|X\|^2, \ \forall \ \|X\| \le \theta. \tag{3.61}$$

Suppose that there is a probability distribution $\mathcal{D}_X$ over $\mathcal{X}$, and let $X, X_1, X_2, \ldots, X_n$ be i.i.d. random variables drawn from $\mathcal{D}_X$. Assume that $\mathbb{E}[X] = 0$, and define

$$\widetilde{\sigma} := (\mathbb{E}[\ell(X)^2])^{1/2}, \quad \nu_p := (\mathbb{E}[\|X\|^p])^{1/p}, p = 2, 3, \quad \varsigma := \frac{(\mathbb{E}[|\ell(X)|^3])^{1/3}}{\widetilde{\sigma}}.$$

Let $\bar{X} = \frac{1}{n}\sum_{i=1}^n X_i$. Then for any $z \in \mathbb{R}$, we have

$$\left|\mathbb{P}\left\{\frac{f(\bar{X})}{\widetilde{\sigma}/\sqrt{n}} \le z\right\} - \Phi(z)\right| \le \frac{C}{\sqrt{n}}, \tag{3.62}$$

where $C = C_0 + C_1\varsigma^3 + (C_{20} + C_{21}\varsigma)\nu_2^2 + (C_{30} + C_{31}\varsigma)\nu_3^2 + C_4$, with

$$\begin{aligned} &C_0 = 0.1393, \quad C_1 = 2.3356 \\ &(C_{20}, C_{21}, C_{30}, C_{31}) = \frac{M_\theta}{2\widetilde{\sigma}}\left(2(\frac{2}{\pi})^{1/6}, 2 + \frac{2^{2/3}}{n^{1/6}}, \frac{(8/\pi)^{1/6}}{n^{1/3}}, \frac{2}{n^{1/2}}\right) \\ &C_4 = \min\{\frac{\nu_2^2}{\theta^2 n^{1/2}}, \frac{2\nu_2^3 + \nu_3^3/n^{1/2}}{\theta^3 n}\}. \end{aligned} \tag{3.63}$$

Define the function $\psi_k(\mathbf{U}, \mathbf{v}) : \mathbb{R}^{d\times d} \times \mathbb{R} \to \mathbb{R}$:

$$\psi_k(\mathbf{U}, \mathbf{v}) := \mathbf{e}_k^{\mathrm{T}} \mathbf{H}_F^{-1} \mathbf{p}_F - \mathbf{e}_k^{\mathrm{T}} (\mathbf{U} + \mathbf{H}_F)^{-1} (\mathbf{v} + \mathbf{p}_F),$$

and thus

$$\widehat{w}_k^i - w_k^* = \psi_k(\mathbf{U}_i, \mathbf{v}_i) = \psi_k(\frac{1}{n}\sum_{j=1}^n \mathbf{U}_{i,j}, \frac{1}{n}\sum_{j=1}^n \mathbf{v}_{i,j}).$$

On the product space $\mathbb{R}^{d\times d} \times \mathbb{R}$, define the element-wise inner product:

$$\langle (\mathbf{U}, \mathbf{v}), (\mathbf{X}, \mathbf{y}) \rangle = \sum_{i,j=1}^d U_{i,j} X_{i,j} + \sum_{i=1}^d v_i y_i,$$

and thus $\mathbb{R}^{d\times d} \times \mathbb{R}$ is associated with the norm

$$\|(\mathbf{U}, \mathbf{v})\| = \sqrt{\|\mathbf{U}\|_F^2 + \|\mathbf{v}\|_2^2},$$

where $\|\cdot\|_F$ denotes the Frobenius norm of matrices. We then provide the following lemma on $\psi_k(\mathbf{U}, \mathbf{v})$.

Lemma 3.4. *There exits a linear function* $\ell_k(\mathbf{U}, \mathbf{v}) = \mathbf{e}_k^T \mathbf{H}_F^{-1} \mathbf{U} \mathbf{H}_F^{-1} \mathbf{p}_F - \mathbf{e}_k^T \mathbf{H}_F^{-1} \mathbf{v}$ *such that for any* $\mathbf{U}$, $\mathbf{v}$ *with*

$$\|\mathbf{U}\|_F^2 + \|\mathbf{v}\|_2^2 \le \frac{\lambda_F^2}{4},$$

we have

$$|\psi_k(\mathbf{U}, \mathbf{v}) - \ell_k(\mathbf{U}, \mathbf{v})| \le \frac{\lambda_F + 2\|\mathbf{p}_F\|_2}{\lambda_F^3} (\|\mathbf{U}\|_F^2 + \|\mathbf{v}\|_2^2).$$

Lemma 3.4 tells us that the condition (3.61) is satisfied with $\theta = \frac{\lambda_F}{2}$ and $M_\theta = \frac{2\lambda_F + 4\|\mathbf{p}_F\|_2}{\lambda_F^3}$. For all normal worker machine $i \in [m] \setminus \mathcal{B}$, denote the distribution of $\mathbf{U}_{i,j}$ and $\mathbf{v}_{i,j}$ by $\mathcal{D}_U$ and $\mathcal{D}_v$, respectively. Since $\widehat{w}_k^i - w_k^* = \psi_k(\frac{1}{n}\sum_{j=1}^n \mathbf{U}_{i,j}, \frac{1}{n}\sum_{j=1}^n \mathbf{v}_{i,j})$, Theorem 3.13 directly gives us the following lemma.

Lemma 3.5. *Let* $\mathbf{U} \sim \mathcal{D}_U$, $\mathbf{v} \sim \mathcal{D}_v$, *and* $\ell_k(\mathbf{U}, \mathbf{v}) = \mathbf{e}_k^T \mathbf{H}_F^{-1} \mathbf{U} \mathbf{H}_F^{-1} \mathbf{p}_F - \mathbf{e}_k^T \mathbf{H}_F^{-1} \mathbf{v}$. *Define*

$$\begin{aligned}
\widetilde{\sigma}_k &:= (\mathbb{E}[\ell_k(\mathbf{U}, \mathbf{v})^2])^{1/2}, \\
\nu_p &:= (\mathbb{E}[(\|\mathbf{U}\|_F^2 + \|\mathbf{v}\|_2^2)^{p/2}])^{1/p}, p = 2, 3, \\
\varsigma_k &:= \frac{(\mathbb{E}[|\ell_k(\mathbf{U}, \mathbf{v})|^3])^{1/3}}{\widetilde{\sigma}_k}.
\end{aligned}$$

Then for any $z \in \mathbb{R}$, $i \in [m] \setminus \mathcal{B}$, *we have*

$$\left| \mathbb{P}\left\{ \frac{\widehat{w}_k^i - w_k^*}{\widetilde{\sigma}_k / \sqrt{n}} \le z \right\} - \Phi(z) \right| \le \frac{C_k}{\sqrt{n}}, \tag{3.64}$$

where

$$C_k = \widehat{C}_0 + \widehat{C}_1 \varsigma_k^3 + \frac{1}{\widetilde{\sigma}_k}\left[(\widehat{C}_{20} + \widehat{C}_{21}\varsigma_k)\nu_2^2 + (\widehat{C}_{30} + \widehat{C}_{31}\varsigma_k)\nu_3^2\right] + \widehat{C}_4,$$

with

$$\begin{aligned} &\widehat{C}_0 = 0.1393, \quad \widehat{C}_1 = 2.3356 \\ &(\widehat{C}_{20}, \widehat{C}_{21}, \widehat{C}_{30}, \widehat{C}_{31}) = \frac{\lambda_F + 2\|\mathbf{p}_F\|_2}{\lambda_F^3}\left(2(\frac{2}{\pi})^{1/6}, 2 + \frac{2^{2/3}}{n^{1/6}}, \frac{(8/\pi)^{1/6}}{n^{1/3}}, \frac{2}{n^{1/2}}\right) \\ &\widehat{C}_4 = \min\{\frac{4\nu_2^2}{\lambda_F^2 n^{1/2}}, \frac{16\nu_2^3 + 8\nu_3^3/n^{1/2}}{\lambda_F^3 n}\}. \end{aligned} \tag{3.65}$$

Then, we proceed to bound $\mathsf{med}\{\widehat{w}_k^i : i \in [m]\} - w_k^*$, the technique is similar to what we use in the proof of Theorem 3.8. For every $z \in \mathbb{R}$, $k \in [d]$, define

$$\widetilde{p}(z;k) = \frac{1}{m(1-\alpha)} \sum_{i \in [m] \setminus \mathcal{B}} \mathbb{1}(\widehat{w}_k^i - w_k^* \le z).$$

We have the following lemma on $\widetilde{p}(z;k)$.

Lemma 3.6. *Suppose that for a fixed $t > 0$, we have*

$$\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + \frac{C_k}{\sqrt{n}} \le \frac{1}{2} - \epsilon, \tag{3.66}$$

for some $\epsilon > 0$. Then, with probability at least $1 - 4e^{-2t}$, we have

$$\widetilde{p}\left(C_\epsilon \frac{\widetilde{\sigma}_k}{\sqrt{n}}(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + \frac{C_k}{\sqrt{n}}); k\right) \ge \frac{1}{2} + \alpha, \tag{3.67}$$

and

$$\widetilde{p}\left(-C_\epsilon \frac{\widetilde{\sigma}_k}{\sqrt{n}}(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + \frac{C_k}{\sqrt{n}}); k\right) \le \frac{1}{2} - \alpha, \tag{3.68}$$

where C_ϵ is defined as in (3.4).

Proof. The proof is essentially the same as the proof of Lemma 3.1. One can simply replace σ in Lemma 3.1 with $\widetilde{\sigma}_k$ and $0.4748\gamma(x)$ in Lemma 3.1 with C_k. Then the same arguments still apply. Thus, we skip the details of this proof. □

Then, define $\widehat{p}(z;k) = \frac{1}{m}\sum_{i \in [m]} \mathbb{1}(\widehat{w}_k^i - w_k^* \le z)$. Using the same arguments as in Corollary 3.1, we know that

$$\widehat{p}\left(C_\epsilon \frac{\widetilde{\sigma}_k}{\sqrt{n}}(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + \frac{C_k}{\sqrt{n}}); k\right) \ge \frac{1}{2},$$

and

$$\widehat{p}\left(-C_\epsilon \frac{\widetilde{\sigma}_k}{\sqrt{n}}(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + \frac{C_k}{\sqrt{n}}); k\right) \leq \frac{1}{2},$$

which implies that $|\mathsf{med}\{\widehat{w}_k^i : i \in [m]\} - w_k^*| \leq C_\epsilon \frac{\widetilde{\sigma}_k}{\sqrt{n}}(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + \frac{C_k}{\sqrt{n}})$. Then, let

$$\widetilde{\sigma} := \sqrt{\sum_{k=1}^{d} \widetilde{\sigma}_k^2} = \sqrt{\mathbb{E}[\|\mathbf{H}_F^{-1}(\mathbf{U}\mathbf{H}_F^{-1}\mathbf{p}_F - \mathbf{v})\|_2^2]},$$

and $\widetilde{C} = \max_{k\in[d]} C_k$, we have with probability at least $1 - 4de^{-2t}$,

$$\|\mathsf{med}\{\widehat{\mathbf{w}}^i : i \in [m]\} - \mathbf{w}^*\|_2 \leq \frac{C_\epsilon}{\sqrt{n}}\widetilde{\sigma}\big(\alpha + \sqrt{\frac{t}{m(1-\alpha)}} + \frac{\widetilde{C}}{\sqrt{n}}\big).$$

We complete the proof by choosing $t = \frac{1}{2}\log(nmd)$.

Explicit expression of $\widetilde{C}$. To summarize, we provide an explicit expression of $\widetilde{C}$. Let $\mathbf{e}_k$ be the k-th vector in the standard basis, i.e., the k-th column of the $d \times d$ identity matrix, and define $\ell_k(\mathbf{U}, \mathbf{v}) : \mathbb{R}^{d\times d} \times \mathbb{R} \to \mathbb{R}$ as

$$\ell_k(\mathbf{U}, \mathbf{v}) = \mathbf{e}_k^{\mathrm{T}}\mathbf{H}_F^{-1}\mathbf{U}\mathbf{H}_F^{-1}\mathbf{p}_F - \mathbf{e}_k^{\mathrm{T}}\mathbf{H}_F^{-1}\mathbf{v}.$$

Let $\mathbf{H} \sim \mathcal{D}_H$ and $\mathbf{p} \sim \mathcal{D}_p$ and define

$$\widetilde{\sigma}_k := (\mathbb{E}[\ell_k(\mathbf{H} - \mathbf{H}_F, \mathbf{p} - \mathbf{p}_F)^2])^{1/2}, \quad \varsigma_k := \frac{(\mathbb{E}[|\ell_k(\mathbf{H} - \mathbf{H}_F, \mathbf{p} - \mathbf{p}_F)|^3])^{1/3}}{\widetilde{\sigma}_k}.$$
$$\nu_p := (\mathbb{E}[(\|\mathbf{H} - \mathbf{H}_F\|_F^2 + \|\mathbf{p} - \mathbf{p}_F\|_2^2)^{p/2}])^{1/p}, p = 2, 3$$

Then, $\widetilde{C} = \max_{k\in[d]} C_k$, with where

$$C_k = \widehat{C}_0 + \widehat{C}_1\varsigma_k^3 + \frac{1}{\widetilde{\sigma}_k}\big[(\widehat{C}_{20} + \widehat{C}_{21}\varsigma_k)\nu_2^2 + (\widehat{C}_{30} + \widehat{C}_{31}\varsigma_k)\nu_3^2\big] + \widehat{C}_4,$$

with

$$\widehat{C}_0 = 0.1393, \quad \widehat{C}_1 = 2.3356$$
$$(\widehat{C}_{20}, \widehat{C}_{21}, \widehat{C}_{30}, \widehat{C}_{31}) = \frac{\lambda_F + 2\|\mathbf{p}_F\|_2}{\lambda_F^3}\left(2(\frac{2}{\pi})^{1/6}, 2 + \frac{2^{2/3}}{n^{1/6}}, \frac{(8/\pi)^{1/6}}{n^{1/3}}, \frac{2}{n^{1/2}}\right)$$
$$\widehat{C}_4 = \min\{\frac{4\nu_2^2}{\lambda_F^2 n^{1/2}}, \frac{16\nu_2^3 + 8\nu_3^3/n^{1/2}}{\lambda_F^3 n}\}.$$

Proof of Lemma 3.4

We use $\|\cdot\|_2$ and $\|\cdot\|_F$ to denote the operator norm and the Frobenius norm of matrices, respectively. We have the identity

$$(\mathbf{I}+\mathbf{A})^{-1} = \sum_{r=0}^{\infty}(-1)^r\mathbf{A}^r, \quad \forall \|\mathbf{A}\|_2 < 1.$$

Then, we have for all $\mathbf{U}\in\mathbb{R}^{d\times d}$ such that $\|\mathbf{H}_F^{-1}\mathbf{U}\|_2<1$,

$$(\mathbf{U}+\mathbf{H}_F)^{-1} = (\mathbf{I}+\mathbf{H}_F^{-1}\mathbf{U})^{-1}\mathbf{H}_F^{-1} = \mathbf{H}_F^{-1} - \mathbf{H}_F^{-1}\mathbf{U}\mathbf{H}_F^{-1} + \sum_{r=2}^{\infty}(-1)^r(\mathbf{H}_F^{-1}\mathbf{U})^r\mathbf{H}_F^{-1}. \tag{3.69}$$

Let us consider the set of matrices such that $\|\mathbf{U}\|_F \le \frac{\lambda_F}{2}$. One can check that for any such matrix, we have $\|\mathbf{H}_F^{-1}\mathbf{U}\|_2 \le \frac{1}{2}$. Let

$$\ell_k(\mathbf{U},\mathbf{v}) = \mathbf{e}_k^{\mathrm{T}}\mathbf{H}_F^{-1}\mathbf{U}\mathbf{H}_F^{-1}\mathbf{p}_F - \mathbf{e}_k^{\mathrm{T}}\mathbf{H}_F^{-1}\mathbf{v}.$$

Then, we know that

$$|\psi_k(\mathbf{U},\mathbf{v}) - \ell_k(\mathbf{U},\mathbf{v})| = \left|\mathbf{e}_k^{\mathrm{T}}\mathbf{H}_F^{-1}\mathbf{U}\mathbf{H}_F^{-1}\mathbf{v} - \sum_{r=2}^{\infty}(-1)^r\mathbf{e}_k^{\mathrm{T}}(\mathbf{H}_F^{-1}\mathbf{U})^r\mathbf{H}_F^{-1}(\mathbf{v}+\mathbf{p}_F)\right|. \tag{3.70}$$

Denote the operator norm of matrices by $\|\cdot\|_2$. We further have for any $r\ge 1$,

$$|\mathbf{e}_k^{\mathrm{T}}(\mathbf{H}_F^{-1}\mathbf{U})^r\mathbf{H}_F^{-1}\mathbf{v}| \le \frac{1}{2}\|\mathbf{H}_F^{-1}\mathbf{U}\|_2^{r-1}(\|\mathbf{H}_F^{-1}\mathbf{U}\|_2^2 + \|\mathbf{H}_F^{-1}\mathbf{v}\|_2^2) \le \frac{1}{2^r\lambda_F^2}(\|\mathbf{U}\|_F^2 + \|\mathbf{v}\|_2^2), \tag{3.71}$$

where we use the fact $\|\mathbf{U}\|_2 \le \|\mathbf{U}\|_F$. In addition, for any $r \ge 2$,

$$|\mathbf{e}_k^{\mathrm{T}}(\mathbf{H}_F^{-1}\mathbf{U})^r\mathbf{H}_F^{-1}\mathbf{p}_F| \le \|\mathbf{H}_F^{-1}\mathbf{U}\|_2^{r-2}\|\mathbf{H}_F^{-1}\|_2^3\|\mathbf{U}\|_2^2\|\mathbf{p}_F\|_2 \le \frac{\|\mathbf{p}_F\|_2}{2^{r-2}\lambda_F^3}\|\mathbf{U}\|_F^2. \tag{3.72}$$

Then, we plug (3.71) and (3.72) into (3.70), and obtain

$$|\psi_k(\mathbf{U},\mathbf{v}) - \ell_k(\mathbf{U},\mathbf{v})| \le \frac{1}{\lambda_F^2}(\|\mathbf{U}\|_F^2 + \|\mathbf{v}\|_2^2) + \frac{2\|\mathbf{p}_F\|_2}{\lambda_F^3}\|\mathbf{U}\|_F^2,$$

which completes the proof.

Proof of Observation 3.1

This proof is essentially the same as the lower bound in the robust mean estimation literature [39, 113]. We reproduce this result for the purpose of completeness. For a d dimensional

Gaussian distribution $P = \mathcal{N}(\boldsymbol{\mu}, \sigma^2\mathbf{I})$, we denote by P^n the joint distribution of n i.i.d. samples of P. Obviously P^n is equivalent to a dn dimensional Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}^+, \sigma^2\mathbf{I})$, where $\boldsymbol{\mu}^+ \in \mathbb{R}^{dn}$ is a vector generated by repeating $\boldsymbol{\mu}$ n times, i.e., $\boldsymbol{\mu}^+ = [\boldsymbol{\mu}^{\mathrm{T}} \ \boldsymbol{\mu}^{\mathrm{T}} \ \cdots \ \boldsymbol{\mu}^{\mathrm{T}}]^{\mathrm{T}}$.

We show that for two d dimensional distributions $P_1 = \mathcal{N}(\boldsymbol{\mu}_1, \sigma^2\mathbf{I})$ and $P_2 = \mathcal{N}(\boldsymbol{\mu}_2, \sigma^2\mathbf{I})$, there exist two dn dimensional distributions Q_1 and Q_2 such that

$$(1-\alpha)P_1^n + \alpha Q_1 = (1-\alpha)P_2^n + \alpha Q_2. \tag{3.73}$$

If this happens, then no algorithm can distinguish between P_1 and P_2. Let ϕ_1 and ϕ_2 be the PDF of P_1^n and P_2^n, respectively. Let $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ be such that the total variation distance between P_1^n and P_2^n is

$$\frac{1}{2}\int \|\phi_1 - \phi_2\|_1 = \frac{\alpha}{1-\alpha}.$$

By the results of the total variation distance between Gaussian distributions, we know that

$$\|\boldsymbol{\mu}_1^+ - \boldsymbol{\mu}_2^+\|_2 \geq \frac{2\alpha\sigma}{1-\alpha}. \tag{3.74}$$

Let Q_1 be the distribution with PDF $\frac{1-\alpha}{\alpha}(\phi_2 - \phi_1)\mathbb{1}_{\phi_2 \geq \phi_1}$ and Q_2 be the distribution with PDF $\frac{1-\alpha}{\alpha}(\phi_1 - \phi_2)\mathbb{1}_{\phi_1 \geq \phi_2}$. One can verify that (3.73) is satisfied. In this case, by the lower bound in (3.74), we get

$$\|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|_2 \geq \frac{2\alpha\sigma}{\sqrt{n}(1-\alpha)} \geq \frac{2\alpha\sigma}{\sqrt{n}}.$$

This implies that for two Gaussian distributions such that $\|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|_2 = \Omega(\frac{\alpha}{\sqrt{n}})$, in the worst case it can be impossible to distinguish these two distributions due to the existence of the adversary. Thus, to estimate the mean $\boldsymbol{\mu}$ of a Gaussian distribution in the distributed setting with α fraction of Byzantine machines, any algorithm that computes an estimation $\widehat{\boldsymbol{\mu}}$ of the mean has a constant probability of error $\|\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_2 = \Omega(\frac{\alpha}{\sqrt{n}})$.

Further, according to the standard results from minimax theory [191], we know that using $\mathcal{O}(nm)$ data, there is a constant probability that $\|\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_2 = \Omega(\sqrt{\frac{d}{nm}})$. Combining these two results, we know that $\|\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_2 = \Omega(\frac{\alpha}{\sqrt{n}} + \sqrt{\frac{d}{nm}})$.

Chapter 4

Saddle Point Attack in Byzantine-Robust Distributed Learning

We study robust distributed learning that involves minimizing a non-convex loss function with saddle points. We continue to consider the Byzantine setting where some worker machines have abnormal or even arbitrary and adversarial behavior. In this setting, the Byzantine machines may create fake local minima near a saddle point that is far away from any true local minimum, even when robust gradient estimators are used. We develop *ByzantinePGD*, a robust first-order algorithm that can provably escape saddle points and fake local minima, and converge to an approximate true local minimizer with low iteration complexity. As a by-product, we give a simpler algorithm and analysis for escaping saddle points in the usual non-Byzantine setting. We further discuss three robust gradient estimators that can be used in ByzantinePGD, including median, trimmed mean, and iterative filtering. We characterize their performance in concrete statistical settings, and argue for their near-optimality in low and high dimensional regimes.

4.1 Introduction

In this chapter, we still focus on robust distributed optimization for statistical learning problems, and consider a standard worker-server distributed computing framework, where a single master machine is in charge of maintaining and updating the parameter of interest, and a set of worker machines store the data, perform local computation and communicate with the master. In addition, we still consider the Byzantine setting, where a subset of machines behave completely arbitrarily—even in a way that depends on the algorithm used and the data on the other machines—thereby capturing the unpredictable nature of the errors. Here, we assume that the data points are generated from some unknown distribution $\mathcal{D}$ and stored locally in m worker machines, each storing n data points; the goal is to minimize a population loss function $F : \mathcal{W} \to \mathbb{R}$ defined as an expectation over $\mathcal{D}$, where $\mathcal{W} \subseteq \mathbb{R}^d$ is the parameter space. We assume that $\alpha \in (0, 1/2)$ fraction of the worker machines are Byzantine; that is, their behavior is arbitrary. As mentioned in Chapter 3, this Byzantine-robust distributed learning problem has attracted attention in a recent line of work [4, 23, 41, 68, 172, 173]. This body of work develops robust algorithms that are guaranteed to output an approximate minimizer of F when it is convex, or an approximate stationary point in the non-convex case.

However, fitting complicated machine learning models often requires finding a *local minimum* of *non-convex* functions, as exemplified by training deep neural networks and other high-capacity learning architectures [169, 73, 72]. It is well-known that many of the stationary points of these problems are in fact saddle points and far away from any local minimum [99, 72]. These tasks hence require algorithms capable of efficiently *escaping saddle points* and converging approximately to a local minimizer. In the centralized setting without Byzantine adversaries, this problem has been studied actively and recently [71, 94, 33, 96].

A main observation of this work is that the interplay between non-convexity and Byzantine errors makes escaping saddle points much more challenging. In particular, by orchestrating their messages sent to the master machine, *the Byzantine machines can create fake local minima near a saddle point of F that is far away from any true local minimizer.* Such a strategy, which may be referred to as **saddle point attack**, foils existing algorithms as we elaborate below:

- **Challenges due to non-convexity:** When F is convex, gradient descent (GD) equipped with a robust gradient estimator is guaranteed to find an approximate global minimizer (with accuracy depending on the fraction of Byzantine machines) [41, 4]. However, when F is non-convex, such algorithms may be trapped in the neighborhood of a saddle point; see Example 1 in Section 4.7.

- **Challenges due to Byzantine machines:** Without Byzantine machines, vanilla GD [119], as well as its more efficient variants such as perturbed gradient descent (PGD) [94], are known to converge to a local minimizer with high probability. However, Byzantine machines can manipulate PGD and GD (even robustified) into fake local minimum near a saddle point; see Example 2 in Section 4.7.

We illustrate these challenges in Figure 4.1. We discuss and compare with existing work in more details in Section 4.2. The observations above show that existing robust and saddle-escaping algorithms, as well as their naive combination, are insufficient against saddle point attack. Addressing these challenges requires the development of new robust distributed optimization algorithms.

Our Contributions

In this chapter, we develop ***ByzantinePGD***, a computation- and communication-efficient first-order algorithm that is able to escape saddle points and the fake local minima created

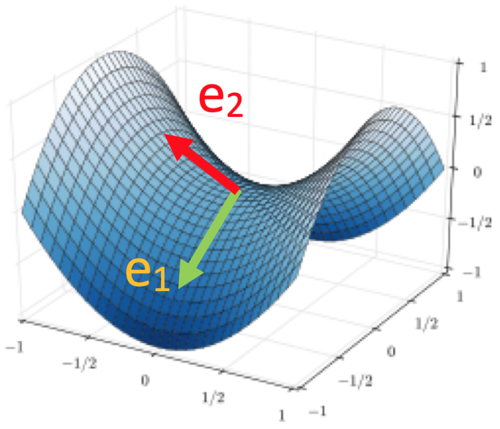


Figure 4.1: Escaping saddle point is more difficult in the Byzantine setting. Consider a saddle point with $\mathbf{e_1}$ and $\mathbf{e_2}$ being the "good" and "bad" directions, respectively. Without Byzantine machines, as long as the gradient direction has a small component on $\mathbf{e_1}$, the optimization algorithm can escape the saddle point. However, in the Byzantine setting, the adversary can eliminate the component on the $\mathbf{e_1}$ direction and make the learner stuck at the saddle point.

by Byzantine machines, and converge to an approximate local minimizer of a non-convex loss. To the best of our knowledge, our algorithm is the first to achieve such guarantees under *adversarial* noise.

Specifically, ByzantinePGD aggregates the empirical gradients received from the normal and Byzantine machines, and computes a robust estimate $\widehat{\mathbf{g}}(\mathbf{w})$ of the true gradient $\nabla F(\mathbf{w})$ of the population loss F. Crucial to our algorithm is the injection of random perturbation to the iterates $\mathbf{w}$, which serves the dual purpose of escaping saddling point and fake local minima. Our use of perturbation thus plays a more signified role than in existing algorithms such as PGD [94], as it also serves to combat the effect of Byzantine errors. To achieve this goal, we incorporate two crucial innovations: (i) we use multiple rounds of larger, yet carefully calibrated, amount of perturbation that is necessary to survive saddle point attack, (ii) we use the moving distance in the parameter space as the criterion for successful escape, eliminating the need of (robustly) evaluating function values. Consequently, our analysis is significantly different, and arguably simpler, than that of PGD.

We develop our algorithmic and theoretical results in a flexible, two-part framework, decomposing the *optimization* and *statistical* components of the problem.

The optimization part: We consider a general problem of optimizing a population loss function F given an *inexact gradient oracle*. For each query point $\mathbf{w}$, the Δ-inexact gradient oracle returns a vector $\widehat{\mathbf{g}}(\mathbf{w})$ (possibly chosen adversarially) that satisfies $\|\widehat{\mathbf{g}}(\mathbf{w})-\nabla F(\mathbf{w})\|_2 \leq \Delta$, where Δ is non-zero but bounded. Given access to such an inexact oracle, we show that

ByzantinePGD outputs an approximate local minimizer; moreover, *no* other algorithm can achieve significantly better performance in this setting in terms of the dependence on Δ:

Theorem 4.1 (Informal; see Sec. 4.4)**.** *Within $\widetilde{\mathcal{O}}(\frac{1}{\Delta^2})$ iterations, ByzantinePGD outputs an approximate local minimizer $\widetilde{\mathbf{w}}$ that satisfies $\|\nabla F(\widetilde{\mathbf{w}})\|_2 \lesssim \Delta$ and $\lambda_{\min}\big(\nabla^2 F(\widetilde{\mathbf{w}})\big) \gtrsim -\Delta^{2/5}$, where $\lambda_{\min}$ is the minimum eigenvalue. In addition, given only access to Δ-inexact gradient oracle, **no** algorithm is guaranteed to find a point $\widetilde{\mathbf{w}}$ with $\|\nabla F(\widetilde{\mathbf{w}})\|_2 < \Delta/2$ **or** $\lambda_{\min}\big(\nabla^2 F(\widetilde{\mathbf{w}})\big) > -\Delta^{1/2}/2$.*

Our algorithm is communication-efficient: it only sends gradients, and the number of parallel iterations in our algorithm *matches* the well-known iteration complexity of GD for non-convex problems in non-Byzantine setting [144] (up to log factors). In the exact gradient setting, a variant of the above result in fact matches the guarantees for PGD [94]—as mentioned, our proof is simpler.

Additionally, beyond Byzantine distributed learning, our results apply to any non-convex optimization problems (distributed or not) with inexact information for the gradients, including those with noisy but non-adversarial gradients. Thus, we believe our results are of independent interest in broader settings.

The statistical part: The optimization guarantee above can be applied whenever one has a robust aggregation procedure that serves as an inexact gradient oracle with a bounded error Δ. We consider three concrete examples of such robust procedures: median, trimmed mean, and iterative filtering [58, 57]. Under statistical settings for the data, we provide explicit bounds on their errors Δ as a function of the number of worker machines m, the number of data points on each worker machine n, the fraction of Byzantine machines α, and the dimension of the parameter space d. Combining these bounds with the optimization result above, we obtain concrete statistical guarantees on the output $\widetilde{\mathbf{w}}$. Furthermore, we argue that our first-order guarantees on $\|\nabla F(\widetilde{\mathbf{w}})\|_2$ are often nearly *optimal* when compared against a universal statistical lower bound. This is summarized below:

Theorem 4.2 (Informal; see Sec. 4.5)**.** *When combined with each of following three robust aggregation procedures, ByzantinePGD achieves the statistical guarantees:*
(i) median/; $\|\nabla F(\widetilde{\mathbf{w}})\|_2 \lesssim \frac{\alpha\sqrt{d}}{\sqrt{n}} + \frac{d}{\sqrt{nm}} + \frac{\sqrt{d}}{n}$;
(ii) trimmed mean: $\|\nabla F(\widetilde{\mathbf{w}})\|_2 \lesssim \frac{\alpha d}{\sqrt{n}} + \frac{d}{\sqrt{nm}}$;
(iii) iterative filtering: $\|\nabla F(\widetilde{\mathbf{w}})\|_2 \lesssim \frac{\sqrt{\alpha}}{\sqrt{n}} + \frac{\sqrt{d}}{\sqrt{nm}}$.
*Moreover, **no** algorithm can achieve $\|\nabla F(\widetilde{\mathbf{w}})\|_2 = o\big(\frac{\alpha}{\sqrt{n}} + \frac{\sqrt{d}}{\sqrt{nm}}\big)$.*

We emphasize that the above results are established under a very strong adversary model: the Byzantine machines are allowed to send messages that depend arbitrarily on each other and on the data on the normal machines; they may even behave adaptively during the iterations of our algorithm. Consequently, this setting requires robust *functional* estimation (of the gradient function), which is a much more challenging problem than the robust *mean* estimation setting considered by existing work on median, trimmed mean and iterative filtering. To overcome this difficulty, we make use of careful covering net arguments to establish certain error bounds that hold *uniformly* over the parameter space, regardless of the behavior of the Byzantine machines. Importantly, our inexact oracle framework allows such arguments to be implemented in a transparent and modular manner.

Notation

For an integer $N > 0$, define the set $[N] := \{1, 2, \ldots, N\}$. For matrices, denote the operator norm by $\|\cdot\|_2$; for symmetric matrices, denote the largest and smallest eigenvalues by $\lambda_{\max}(\cdot)$ and $\lambda_{\min}(\cdot)$, respectively. The d-dimensional ℓ_2 ball centered at $\mathbf{w}$ with radius r is denoted by $\mathbb{B}_{\mathbf{w}}^{(d)}(r)$, or $\mathbb{B}_{\mathbf{w}}(r)$ when it is clear from the context.

4.2 Related Work

Algorithm	PGD	Neon+GD	Neon2+GD	**ByzantinePGD**
Byzantine-robust?	no	no	no	yes
Purpose of perturbation	escape SP	escape SP	escape SP	escape SP & robustness
Escaping method	GD	NC search	NC search	inexact GD
Termination criterion	decrease in F	decrease in F	distance in $\mathcal{W}$	distance in $\mathcal{W}$
Multiple rounds?	no	no	no	yes

Table 4.1: Comparison with PGD, Neon+GD, and Neon2+GD. SP = saddle point.

	Robust Aggregation Method	Non-convex Guarantee
Feng, Xu, and Mannor [68]	geometric median	no
Chen, Su, and Xu [41]	geometric median	no
Blanchard et al. [23]	Krum	first-order
Yin et al. [197]	median, trimmed mean	first-order
Xie, Koyejo, and Gupta [192]	mean-around-median marginal median	first-order
Alistarh, Allen-Zhu, and Li [4]	martingale-based	no
Su and Xu [174]	iterative filtering	no
This work	median, trimmed mean iterative filtering	second-order

Table 4.2: Comparison with other Byzantine-robust distributed learning algorithms.

Efficient first-order algorithms for escaping saddle points Our algorithm is related to a recent line of work which develops efficient first-order algorithms for escaping saddle points. Although vanilla GD converges to local minimizers almost surely [119, 118], achieving convergence in polynomial time requires more a careful algorithmic design [61]. Such convergence guarantees are enjoyed by several GD-based algorithms; examples include PGD [94], Neon+GD [196], and Neon2+GD [7]. The general idea of these algorithms is to run GD and add perturbation to the iterate when the gradient is small. While our algorithm also uses this idea, the design and analysis techniques of our algorithm are significantly different from the work above in the following aspects (also summarized in Table 4.1).

- In our algorithm, besides helping with escaping saddle points, the random perturbation has the additional role of defending against adversarial errors.
- The perturbation used in our algorithm needs to be larger, yet carefully calibrated, in order to account for the influence of the inexactness of gradients across the iterations, especially iterations for escaping saddle points.
- We run inexact GD after the random perturbation, while Neon+GD and Neon2+GD use negative curvature (NC) search. It is not immediately clear whether NC search can be robustified against Byzantine failures. Compared to PGD, our analysis is arguably *simpler* and more *straightforward*.
- Our algorithm does not use the value of the loss function (hence no need for robust function value estimation); PGD and Neon+GD assume access to the (exact) function values.
- We employed *multiple* rounds of perturbation to boost the probability of escaping saddle points; this technique is not used in PGD, Neon+GD, or Neon2+GD.

Inexact oracles Optimization with an inexact oracle (e.g. noisy gradients) has been studied in various settings such as general convex optimization [20, 56], robust estimation [152], and structured non-convex problems [14, 42, 29, 205]. Particularly relevant to us is the recent work by Jin et al. [95], who consider the problem of minimizing F when only given access to the gradients of another *smooth* function $\widehat{F}$ satisfying $\|\nabla\widehat{F}(\mathbf{w}) - \nabla F(\mathbf{w})\|_\infty \le \Delta/\sqrt{d}$, $\forall\mathbf{w}$. Their algorithm uses Gaussian smoothing on $\widehat{F}$. We emphasize that the inexact gradient setting considered by them is much more benign than our Byzantine setting, since (i) their inexactness is defined in terms of ℓ_∞ norm whereas the inexactness in our problem is in ℓ_2 norm, and (ii) we assume that the inexact gradient can be *any* vector within Δ error, and thus the smoothing technique is not applicable in our problem. Moreover, the iteration complexity obtained by Jin et al. [95] may be a high-degree polynomial of the problem parameters and thus not suitable for distributed implementation.

Byzantine-robust distributed learning Solving large scale learning problems in distributed systems has received much attention in recent years, where communication efficiency and Byzantine robustness are two important topics [165, 117, 23, 38, 49]. Here, we compare with existing Byzantine-robust distributed learning algorithms that are most relevant to our work, and summarize the comparison in Table 4.2. A general idea of designing Byzantine-robust algorithms is to combine optimization algorithms with a robust aggregation (or outlier removal) subroutine. For convex losses, the aggregation subroutines analyzed in the literature include geometric median [68, 41], iterative filtering for the high dimensional setting [174], and martingale-based methods for the SGD setting [4]. For non-convex losses, to the best of our knowledge, existing works only provide first-order convergence guarantee (i.e., small gradients), by using aggregation subroutines such as the Krum function [23], median and trimmed mean as we discussed in Chapter 3, mean-around-median and marginal median [192]. In this chapter, we make use of subroutines based on median, trimmed mean, and iterative filtering. Our results based on the iterative filtering subroutine, on the other hand, are new:

- The problem that we tackle is harder than what is considered in the original iterative filtering papers [58, 57]. There they only consider robust estimation of a single mean parameter, where as we guarantee robust gradient estimation over the parameter space.
- Recent work by Su and Xu [174] also makes use of the iterative filtering subroutine for the Byzantine setting. They only study strongly convex loss functions, and assume that the gradients are sub-exponential and $d \leq \mathcal{O}(\sqrt{mn})$. Our results apply to the non-convex case and do not require the aforementioned condition on d (which may therefore scale, for example, linearly with the sample size mn), but we impose the stronger assumption of sub-Gaussian gradients.

Other non-convex optimization algorithms Besides first-order GD-based algorithms, many other non-convex optimization methods that can provably converge to approximate local minimum have received much attention in recent years. For specific problems such as phase retrieval [29], low-rank estimation [42, 211], and dictionary learning [2, 176], many algorithms are developed by leveraging the particular structure of the problems, and the either use a smart initialization [29, 181] or initialize randomly [43, 36]. Other algorithms are developed for general non-convex optimization, and they can be classified into gradient-based [71, 123, 196, 6, 7, 96], Hessian-vector-product-based [33, 3, 161, 160], and Hessian-based [145, 48] methods. While algorithms using Hessian information can usually achieve better convergence rates—for example, $\mathcal{O}(\frac{1}{\epsilon^{3/2}})$ by Curtis, Robinson, and Samadi [48], and $\mathcal{O}(\frac{1}{\epsilon^{7/4}})$ by Carmon et al. [33]— gradient-based methods are easier to implement in practice, especially in the distributed setting we are interested in.

Robust statistics Outlier-robust estimation is a classical topic in statistics [90]. The coordinate-wise median aggregation subroutine that we consider is related to the median-of-means estimator [143, 93], which has been applied to various robust inference problems [138, 129, 139].

A recent line of work develops efficient robust estimation algorithms in high-dimensional settings [22, 58, 113, 35, 170, 124, 21, 102, 127]. In the centralized setting, the recent work [59] proposes a scheme, similar to the iterative filtering procedure, that iteratively removes outliers for gradient-based optimization.

4.3 Problem Setup

We consider empirical risk minimization for a statistical learning problem where each data point $\mathbf{z}$ is sampled from an unknown distribution $\mathcal{D}$ over the sample space $\mathcal{Z}$. Let $f(\mathbf{w};\mathbf{z})$ be the loss function of a parameter vector $\mathbf{w} \in \mathcal{W} \subseteq \mathbb{R}^d$, where $\mathcal{W}$ is the parameter space. The population loss function is therefore given by $F(\mathbf{w}) := \mathbb{E}_{\mathbf{z}\sim\mathcal{D}}[f(\mathbf{w};\mathbf{z})]$.

We consider a distributed computing system with one *master* machine and m *worker* machines, αm of which are Byzantine machines and the other $(1-\alpha)m$ are normal. Each worker machine has n data points sampled i.i.d. from $\mathcal{D}$. Denote by $\mathbf{z}_{i,j}$ the j-th data point on the i-th worker machine, and let $F_i(\mathbf{w}) := \frac{1}{n}\sum_{j=1}^{n} f(\mathbf{w};\mathbf{z}_{i,j})$ be the empirical loss function on the i-th machine. The master machine and worker machines can send and receive messages via the following communication protocol: In each parallel iteration, the master machine sends a parameter vector $\mathbf{w}$ to all the worker machines, and then each *normal* worker machine computes the gradient of its empirical loss $F_i(\cdot)$ at $\mathbf{w}$ and sends the gradient to the master machine. The Byzantine machines may be jointly controlled by an adversary and send arbitrary or even malicious messages. We denote the unknown set of Byzantine machines by $\mathcal{B}$, where $|\mathcal{B}| = \alpha m$. With this notation, the gradient sent by the i-th worker machine is

$$\widehat{\mathbf{g}}_i(\mathbf{w}) = \begin{cases} \nabla F_i(\mathbf{w}) & i \in [m] \setminus \mathcal{B}, \\ * & i \in \mathcal{B}, \end{cases} \tag{4.1}$$

where the symbol $*$ denotes an arbitrary vector. As mentioned, the adversary is assumed to have complete knowledge of the algorithm used and the data stored on all machines, and the Byzantine machines may collude [131] and adapt to the output of the master and normal worker machines. We only make the mild assumption that the adversary cannot *predict* the random numbers generated by the master machine.

We consider the scenario where $F(\mathbf{w})$ is non-convex, and our goal to find an approximate local minimizer of $F(\mathbf{w})$. Note that a first-order stationary point (i.e., one with a small gradient) is not necessarily close to a local minimizer, since the point may be a *saddle point* whose Hessian matrix has a large negative eigenvalue. Accordingly, we seek to find a *second-order stationary point* $\widetilde{\mathbf{w}}$, namely, one with a small gradient and a nearly positive semidefinite Hessian:

Definition 4.1 (Second-order stationarity)**.** *We say that $\widetilde{\mathbf{w}}$ is an (ϵ_g, ϵ_H)-second-order stationary point of a twice differentiable function $F(\cdot)$ if $\|\nabla F(\widetilde{\mathbf{w}})\|_2 \le \epsilon_g$ and $\lambda_{\min}\big(\nabla^2 F(\widetilde{\mathbf{w}})\big) \ge -\epsilon_H$.*

In the sequel, we make use of several standard concepts from continuous optimization.

Definition 4.2 (Smooth and Hessian-Lipschitz functions)**.** *A function h is called L-smooth if $\sup_{\mathbf{w}\neq\mathbf{w}'} \frac{\|\nabla h(\mathbf{w})-\nabla h(\mathbf{w}')\|_2}{\|\mathbf{w}-\mathbf{w}'\|_2} \le L$, and ρ-Hessian Lipschitz if $\sup_{\mathbf{w}\neq\mathbf{w}'} \frac{\|\nabla^2 h(\mathbf{w})-\nabla^2 h(\mathbf{w}')\|_2}{\|\mathbf{w}-\mathbf{w}'\|_2} \le \rho$.*

Throughout this chapter, the above properties are imposed on the *population* loss function $F(\cdot)$.

Assumption 4.1. *F is L_F-smooth, and ρ_F-Hessian Lipschitz on $\mathcal{W}$.*

4.4 Byzantine Perturbed Gradient Descent

In this section, we describe our algorithm, *Byzantine Perturbed Gradient Descent* (ByzantinePGD), which provably finds a second-order stationary point of the population loss $F(\cdot)$ in the distributed setting with Byzantine machines. As mentioned, ByzantinePGD robustly aggregates gradients from the worker machines, and performs multiple rounds of carefully calibrated perturbation to combat the effect of Byzantine machines. We now elaborate.

It is well-known that naively aggregating the workers' messages using standard averaging can be arbitrarily skewed in the presence of just a single Byzantine machine. In view of this, we introduce the subroutine $\mathsf{GradAGG}\{\widehat{\mathbf{g}}_i(\mathbf{w})\}_{i=1}^m$, which robustly aggregates the gradients $\{\widehat{\mathbf{g}}_i(\mathbf{w})\}_{i=1}^m$ collected from the m workers. We stipulate that $\mathsf{GradAGG}$ provides an estimate of the true population gradient $\nabla F(\cdot)$ with accuracy Δ, *uniformly* across $\mathcal{W}$. This property is formalized using the terminology of *inexact gradient oracle.*

Definition 4.3 (Inexact gradient oracle)**.** *We say that $\mathsf{GradAGG}$ provides a Δ-inexact gradient oracle for the population loss $F(\cdot)$ if, for every $\mathbf{w} \in \mathcal{W}$, we have $\|\mathsf{GradAGG}\{\widehat{\mathbf{g}}_i(\mathbf{w})\}_{i=1}^m - \nabla F(\mathbf{w})\|_2 \le \Delta$.*

Without loss of generality, we assume that $\Delta \le 1$ throughout the chapter. In this section, we treat $\mathsf{GradAGG}$ as a given black box; in Section 4.5, we discuss several robust aggregation algorithms and characterize their inexactness Δ. We emphasize that in the Byzantine setting, the output of $\mathsf{GradAGG}$ can take values adversarially within the error bounds; that is, $\mathsf{GradAGG}\{\widehat{\mathbf{g}}_i(\mathbf{w})\}_{i=1}^m$ may output an arbitrary vector in the ball $\mathbb{B}_{\nabla F(\mathbf{w})}(\Delta)$, and this vector can depend on the data in all the machines and all previous iterations of the algorithm.

The use of robust aggregation with bounded inexactness, however, is not yet sufficient to guarantee convergence to an approximate local minimizer. As mentioned, the Byzantine machines may create fake local minima that traps a vanilla gradient descent iteration. Our ByzantinePGD algorithm is designed to escape such fake minima as well as any existing saddle points of F.

Algorithm

We now describe the details of our algorithm, given in the left panel of Algorithm 3. We focus on unconstrained optimization, i.e., $\mathcal{W} = \mathbb{R}^d$. In Section 4.5, we show that the iterates $\mathbf{w}$ during the algorithm actually stay in a bounded ℓ_2 ball centered at the initial iterate $\mathbf{w}_0$, and we will discuss the statistical error rates within the bounded space.

In each parallel iteration, the master machine sends the current iterate $\mathbf{w}$ to all the worker machines, and the worker machines send back $\{\widehat{\mathbf{g}}_i(\mathbf{w})\}$. The master machine aggregates the workers' gradients using GradAGG and computes a robust estimate $\widehat{\mathbf{g}}(\mathbf{w})$ of the population gradient $\nabla F(\mathbf{w})$. The master machine then performs a gradient descent step using $\widehat{\mathbf{g}}(\mathbf{w})$. This procedure is repeated until it reaches a point $\widetilde{\mathbf{w}}$ with $\|\widehat{\mathbf{g}}(\mathbf{w})\|_2 \le \epsilon$ for a pre-specified threshold ϵ.

At this point, $\widetilde{\mathbf{w}}$ may lie near a saddle point whose Hessian has a large negative eigenvalue. To escape this potential saddle point, the algorithm invokes the Escape routine (right panel of Algorithm 3), which performs Q rounds of *perturbation-and-descent* operations. In each round, the master machine perturbs $\widetilde{\mathbf{w}}$ randomly and independently within the ball $\mathbb{B}_{\widetilde{\mathbf{w}}}(r)$. Let $\mathbf{w}'_0$ be the perturbed vector. Starting from the $\mathbf{w}'_0$, the algorithm conducts at most T_{th} parallel iterations of Δ-inexact gradient descent (using GradAGG as before):

$$\mathbf{w}'_t = \mathbf{w}'_{t-1} - \eta\widehat{\mathbf{g}}(\mathbf{w}'_{t-1}),\ t \le T_{\text{th}}. \tag{4.2}$$

During this process, once we observe that $\|\mathbf{w}'_t - \mathbf{w}'_0\|_2 \ge R$ for some pre-specified threshold R (this means the iterate moves by a sufficiently large distance in the parameter space), we claim that $\widetilde{\mathbf{w}}$ is a saddle point and the algorithm has escaped it; we then resume Δ-inexact gradient descent starting from $\mathbf{w}'_t$. If after Q rounds no sufficient move in the parameter space is ever observed, we claim that $\widetilde{\mathbf{w}}$ is a second-order stationary point of $F(\mathbf{w})$ and output $\widetilde{\mathbf{w}}$.

Convergence Guarantees

In this section, we provide the theoretical result guaranteeing that Algorithm 3 converges to a second-order stationary point. In Theorem 4.3, we let $F^* := \min_{\mathbf{w}\in\mathbb{R}^d} F(\mathbf{w})$, $\mathbf{w}_0$ be the initial iterate, and $F_0 := F(\mathbf{w}_0)$.

Theorem 4.3 (ByzantinePGD)**.** *Suppose that Assumptions 4.1 holds, and assume that* GradAGG *provides a Δ-inexact gradient oracle for $F(\cdot)$ with $\Delta \le 1$. Given any $\delta \in (0,1)$, choose the parameters for Algorithm 3 as follows: step-size $\eta = \frac{1}{L_F}$, $\epsilon = 3\Delta$, $r = 4\Delta^{3/5}d^{3/10}\rho_F^{-1/2}$, $R = \Delta^{2/5}d^{1/5}\rho_F^{-1/2}$,*

$$Q = 2\log\left(\frac{\rho_F(F_0 - F^*)}{48L_F\delta(\Delta^{6/5}d^{3/5} + \Delta^{7/5}d^{7/10})}\right),\ \textit{and}$$

$$T_{\text{th}} = \frac{L_F}{384(\rho_F^{1/2} + L_F)(\Delta^{2/5}d^{1/5} + \Delta^{3/5}d^{3/10})}.$$

```
ByzantinePGD($\mathbf{w}_0, \eta, \epsilon, r, Q, R, T_{\text{th}}$)
  $\mathbf{w} \leftarrow \mathbf{w}_0$
  while true do
    Master: send $\mathbf{w}$ to worker machines.
    for all $i \in [m]$ do in parallel
      Worker i: compute $\widehat{\mathbf{g}}_i(\mathbf{w})$
      send to master machine.
    end for
    Master:
    $\widehat{\mathbf{g}}(\mathbf{w}) \leftarrow \mathsf{GradAGG}\{\widehat{\mathbf{g}}_i(\mathbf{w})\}_{i=1}^m$.
    if $\|\widehat{\mathbf{g}}(\mathbf{w})\|_2 \le \epsilon$ then
      Master:
      $\widetilde{\mathbf{w}} \leftarrow \mathbf{w}$,
      $(\mathsf{esc}, \mathbf{w}, \widehat{\mathbf{g}}(\mathbf{w})) \leftarrow \mathsf{Escape}\ (\widetilde{\mathbf{w}}, \eta, r, Q, R, T_{\text{th}})$.
      if esc = false then
        return $\widetilde{\mathbf{w}}$.
      end if
    end if
    Master: $\mathbf{w} \leftarrow \mathbf{w} - \eta\widehat{\mathbf{g}}(\mathbf{w})$.
  end while
```

```
Escape($\widetilde{\mathbf{w}}, \eta, r, Q, R, T_{\text{th}}$)
  for $k = 1, 2, \ldots, Q$ do
    Master: sample $\mathbf{p}_k \sim \text{Unif}(\mathbb{B}_0(r))$,
    $\mathbf{w}' \leftarrow \widetilde{\mathbf{w}} + \mathbf{p}_k$, $\mathbf{w}'_0 \leftarrow \mathbf{w}'$.
    for $t = 0, 1, \ldots, T_{\text{th}}$ do
      Master: send $\mathbf{w}'$ to worker machines.
      for all $i \in [m]$ do in parallel
        Worker i: compute $\widehat{\mathbf{g}}_i(\mathbf{w}')$
        send to master machine.
      end for
      Master:
      $\widehat{\mathbf{g}}(\mathbf{w}') \leftarrow \mathsf{GradAGG}\{\widehat{\mathbf{g}}_i(\mathbf{w}')\}_{i=1}^m$.
      if $\|\mathbf{w}' - \mathbf{w}'_0\|_2 \ge R$ then
        return $(\text{true}, \mathbf{w}', \widehat{\mathbf{g}}(\mathbf{w}'))$.
      else
        $\mathbf{w}' \leftarrow \mathbf{w}' - \eta\widehat{\mathbf{g}}(\mathbf{w}')$
      end if
    end for
  end for
  return $(\text{false}, \mathbf{w}', \widehat{\mathbf{g}}(\mathbf{w}'))$.
```

Algorithm 3: Byzantine Perturbed Gradient Descent (ByzantinePGD)

Then, with probability at least $1-\delta$, the output of Algorithm 3, denoted by $\widetilde{\mathbf{w}}$, satisfies the bounds

$$\begin{aligned} \|\nabla F(\widetilde{\mathbf{w}})\|_2 &\le 4\Delta, \\ \lambda_{\min}\big(\nabla^2 F(\widetilde{\mathbf{w}})\big) &\ge -1900\big(\rho_F^{1/2} + L_F\big)\Delta^{2/5} d^{1/5} \log\Big(\frac{10}{\Delta}\Big), \end{aligned} \tag{4.3}$$

and the algorithm terminates within $\frac{2(F_0 - F^*)L_F}{3\Delta^2}Q$ parallel iterations.

Below let us parse the above theorem and discuss its implications.

Focusing on the scaling with Δ, we may read off from Theorem 4.3 the following result:

Observation 4.1. *Under the above setting, within $\widetilde{\mathcal{O}}(\frac{1}{\Delta^2})$ parallel iterations, ByzantinePGD outputs an $(\mathcal{O}(\Delta), \widetilde{\mathcal{O}}(\Delta^{2/5}))$-second-order stationary point $\widetilde{\mathbf{w}}$ of $F(\cdot)$;*[1] *that is,*

$$\|\nabla F(\widetilde{\mathbf{w}})\|_2 \le 4\Delta \quad \textit{and} \quad \lambda_{\min}(\nabla^2 F(\widetilde{\mathbf{w}})) \ge -\widetilde{\mathcal{O}}(\Delta^{2/5}).$$

[1]Here, by using the symbol $\widetilde{\mathcal{O}}$, we ignore logarithmic factors and only consider the dependence on Δ.

In terms of the iteration complexity, it is well-known that for a smooth non-convex $F(\cdot)$, gradient descent requires at least $\frac{1}{\Delta^2}$ iterations to achieve $\|\nabla F(\widetilde{\mathbf{w}})\|_2 \leq \mathcal{O}(\Delta)$ [144]; up to logarithmic factors, our result matches this complexity bound. In addition, our $\mathcal{O}(\Delta)$ first-order guarantee is clearly order-wise *optimal*, as the gradient oracle is Δ-inexact. It is currently unclear to us whether our $\widetilde{\mathcal{O}}(\Delta^{2/5})$ second-order guarantee is optimal. We provide a converse result showing that one cannot hope to achieve a second-order guarantee better than $\mathcal{O}(\Delta^{1/2})$.

Proposition 4.1. *There exists a class of real-valued 1-smooth and 1-Hessian Lipschitz differentiable functions $\mathcal{F}$ such that, for any algorithm that only uses a Δ-inexact gradient oracle, there exists $f \in \mathcal{F}$ such that the output of the algorithm $\widetilde{\mathbf{w}}$ must satisfy $\|\nabla F(\widetilde{\mathbf{w}})\|_2 > \Delta/2$ and $\lambda_{\min}(\nabla^2 F(\widetilde{\mathbf{w}})) < -\Delta^{1/2}/2$.*

Again, we emphasize that our results above are in fact not restricted to the Byzantine distributed learning setting. They apply to any non-convex optimization problems (distributed or not) with inexact information for the gradients, including those with noisy but non-adversarial gradients; see Section 4.2 for comparison with related work in such settings.

As a byproduct, we can show that with a different choice of parameters, ByzantinePGD can be used in the standard (non-distribued) setting with access to the *exact* gradient $\nabla F(\mathbf{w})$, and the algorithm converges to an $(\epsilon, \widetilde{\mathcal{O}}(\sqrt{\epsilon}))$-second-order stationary point within $\mathcal{O}(\frac{1}{\epsilon^2})$ iterations:

Theorem 4.4 (Exact gradient oracle)**.** *Suppose that Assumptions 4.1 holds, and assume that for any query point $\mathbf{w}$ we can obtain exact gradient, i.e., $\widehat{\mathbf{g}}(\mathbf{w}) \equiv \nabla F(\mathbf{w})$. For any $\epsilon \in (0, \min\{\frac{1}{\rho_F}, \frac{4}{L_F^2 \rho_F}\})$ and $\delta \in (0,1)$, we choose the parameters in Algorithm 3 as follows: step-size $\eta = 1/L_F$, $Q = 1$, $r = \epsilon$, and $R = \sqrt{\epsilon/\rho_F}$, $T_{\text{th}} = \frac{L}{12\rho_F(R+r)}$. Then, with probability at least $1-\delta$, Algorithm 3 outputs a $\widetilde{\mathbf{w}}$ satisfying the bounds*

$$\begin{aligned} \|\nabla F(\widetilde{\mathbf{w}})\|_2 \leq & \epsilon, \\ \lambda_{\min}(\nabla^2 F(\widetilde{\mathbf{w}})) \geq & -60\sqrt{\rho_F \epsilon} \log\Big(\frac{8\rho_F\sqrt{d}(F_0 - F^*)}{\delta\epsilon^2}\Big), \end{aligned}$$

and the algorithm terminates within $\frac{2L_F(F_0 - F^*)}{\epsilon^2}$ iterations.

The convergence guarantee above matches that of the original PGD algorithm [94] up to logarithmic factors. Moreover, our proof is considerably simpler, and our algorithm only requires gradient information, whereas the original PGD algorithm also needs function values.

4.5 Robust Estimation of Gradients

The results in the previous section can be applied as long as one has a robust aggregation subroutine GradAGG that provides a Δ-inexact gradient oracle of the population loss F. In this section, we discuss three concrete examples of GradAGG: *median*, *trimmed mean*, and a high-dimension robust estimator based on the *iterative filtering* algorithm [58, 57, 170]. We characterize their inexactness Δ under the statistical setting in Section 4.3, where the data points are sampled independently according to an unknown distribution $\mathcal{D}$.

To describe our statistical results, we need the standard notions of sub-Gaussian and sub-exponential random vectors.

Definition 4.4 (sub-Gaussianity and sub-exponentiality)**.** *A random vector* $\mathbf{x}$ *with mean* $\boldsymbol{\mu}$ *is said to be* ζ*-sub-Gaussian if* $\mathbb{E}[\exp(\lambda\langle \mathbf{x}-\boldsymbol{\mu}, \mathbf{u}\rangle)] \leq e^{\frac{1}{2}\zeta^2\lambda^2\|\mathbf{u}\|_2^2}, \forall\ \lambda, \mathbf{u}$*. It is said to be* ξ*-sub-exponential if* $\mathbb{E}[\exp(\lambda\langle \mathbf{x}-\boldsymbol{\mu}, \mathbf{u}\rangle)] \leq e^{\frac{1}{2}\xi^2\lambda^2\|\mathbf{u}\|_2^2},\ \forall\ |\lambda| < \frac{1}{\xi}, \mathbf{u}$*.*

We also need the following result, which shows that the iterates of ByzantinePGD in fact stay in a bounded set around the initial iterate $\mathbf{w}_0$.

Proposition 4.2. *Under the choice of algorithm parameters in Theorem 4.3, all the iterates* $\mathbf{w}$ *in ByzantinePGD stay in the* ℓ_2 *ball* $\mathbb{B}_{\mathbf{w}_0}(D/2)$ *with* $D := C\frac{F_0-F^*}{\Delta}$*, where* $C > 0$ *is a number that only depends on* L_F *and* ρ_F*.*

Consequently, for the convergence guarantees of ByzantinePGD to hold, we only need GradAGG to satisfy the inexact oracle property (Definition 4.3) within the bounded set $\mathcal{W} = \mathbb{B}_{\mathbf{w}_0}(D/2)$, with D given in Proposition 4.2. As shown below, the three aggregation procedures indeed satisfy this property, with their inexactness Δ depends mildly (logarithmically) on the radius D.

Iterative Filtering Algorithm

We start with a recently developed high-dimension robust estimation technique called the *iterative filtering* algorithm [58, 57, 170] and use it to build the subroutine GradAGG. As can be seen below, iterative filtering can tolerate a constant fraction of Byzantine machines even when the dimension grows—an advantage over simpler algorithms such as median and trimmed mean.

We relegate the details of the iterative filtering algorithm to Section 4.7. Again, we emphasize that the original iterative filtering algorithm is proposed to robustly estimate a single parameter vector, whereas in our setting, since the Byzantine machines may produce unspecified probabilistic dependency across the iterations, we need to prove an error bound for robust gradient estimation uniformly across the parameter space $\mathcal{W}$. We prove such a bound for iterative filtering under the following two assumptions on the gradients and the smoothness of each loss function $f(\cdot;\mathbf{z})$.

Assumption 4.2. *For each* $\mathbf{w} \in \mathcal{W}$*,* $\nabla f(\mathbf{w};\mathbf{z})$ *is* ζ*-sub-Gaussian.*

Assumption 4.3. *For each* $\mathbf{z} \in \mathcal{Z}$*,* $f(\cdot;\mathbf{z})$ *is* L*-smooth.*

Let $\boldsymbol{\Sigma}(\mathbf{w})$ be the covariance matrix of $\nabla f(\mathbf{w};\mathbf{z})$, and define $\sigma := \sup_{\mathbf{w}\in\mathcal{W}} \|\boldsymbol{\Sigma}(\mathbf{w})\|_2^{1/2}$. We have the following bounds on the inexactness parameter of iterative filtering.

Theorem 4.5 (Iterative Filtering). *Suppose that Assumptions 4.2 and 4.3 hold. Use the iterative filtering algorithm described in Section 4.7 for* $\mathsf{GradAGG}$*, and assume that* $\alpha \leq \frac{1}{4}$*. With probability* $1 - o(1)$*,* $\mathsf{GradAGG}$ *provides a* Δ_{ftr}*-inexact gradient oracle with*

$$\Delta_{\mathsf{ftr}} \leq c \left((\sigma + \zeta)\sqrt{\frac{\alpha}{n}} + \zeta\sqrt{\frac{d}{nm}} \right) \sqrt{\log(nmDL)},$$

where c *is an absolute constant.*

The proof of Theorem 4.5 is given in Section 4.7. Assuming bounded σ and ζ, we see that iterative filtering provides an $\widetilde{\mathcal{O}}\big(\sqrt{\frac{\alpha}{n}} + \sqrt{\frac{d}{nm}}\big)$-inexact gradient oracle.

Median and Trimmed Mean

The median and trimmed mean operations are two widely used robust estimation methods. While the dependence of their performance on d is not optimal, they are conceptually simple and computationally fast, and still have good performance in low dimensional settings. We apply these operations in a coordinate-wise fashion to build $\mathsf{GradAGG}$.

Formally, for a set of vectors $\mathbf{x}^i \in \mathbb{R}^d$, $i \in [m]$, their coordinate-wise median $\mathbf{u} := \mathsf{med}\{\mathbf{x}^i\}_{i=1}^m$ is a vector with its k-th coordinate being $u_k = \mathsf{med}\{x_k^i\}_{i=1}^m$ for each $k \in [d]$, where med is the usual (one-dimensional) median. The coordinate-wise β-trimmed mean $\mathbf{u} := \mathsf{trmean}_\beta\{\mathbf{x}^i\}_{i=1}^m$ is a vector with $u_k = \frac{1}{(1-2\beta)m}\sum_{x \in \mathcal{U}_k} x$ for each $k \in [d]$, where $\mathcal{U}_k$ is a subset of $\{x_k^1, \dots, x_k^m\}$ obtained by removing the largest and smallest β fraction of its elements.

For robust estimation of the gradient in the Byzantine setting, the error bounds of median and trimmed mean have been studied in Chapter 3. For completeness, we record their results below as an informal theorem.

Theorem 4.6 (Informal). *Under appropriate smoothness and probabilistic assumptions,*[2] *with high probability, the median operation provides a* Δ_{med}*-inexact gradient oracle with* $\Delta_{\mathsf{med}} \lesssim \frac{\alpha\sqrt{d}}{\sqrt{n}} + \frac{d}{\sqrt{nm}} + \frac{\sqrt{d}}{n}$*, and the trimmed mean operation provides a* Δ_{tm}*-inexact gradient oracle with* $\Delta_{\mathsf{tm}} \lesssim \frac{\alpha d}{\sqrt{n}} + \frac{d}{\sqrt{nm}}$*.*

Comparison and Optimality

In Table 4.3, we compare the above three algorithms in terms of the dependence of their gradient inexactness Δ on the problem parameters α, n, m, and d . We see that when $d = \mathcal{O}(1)$, the median and trimmed mean algorithms have better inexactness due to a better scaling with α. When d is large, iterative filtering becomes preferable.

[2]Specifically, for median we assume that gradients have bounded skewness, and for trimmed mean we assume that the gradients are sub-exponentially distributed.

	Gradient inexactness Δ
median	$\widetilde{\mathcal{O}}\big(\frac{\alpha\sqrt{d}}{\sqrt{n}} + \frac{d}{\sqrt{nm}} + \frac{\sqrt{d}}{n}\big)$
trimmed mean	$\widetilde{\mathcal{O}}\big(\frac{\alpha d}{\sqrt{n}} + \frac{d}{\sqrt{nm}}\big)$
iterative filtering	$\widetilde{\mathcal{O}}\big(\frac{\sqrt{\alpha}}{\sqrt{n}} + \frac{\sqrt{d}}{\sqrt{nm}}\big)$

Table 4.3: Statistical bounds on gradient inexactness Δ.

Recall that according to Observation 4.1, with Δ-inexact gradients the ByzantinePGD algorithm converges to an $(\mathcal{O}(\Delta), \widetilde{\mathcal{O}}(\Delta^{2/5}))$-second-order stationary point. Combining this general result with the bounds in Table 4.3, we obtain explicit statistical guarantees on the output of ByzantinePGD. To understand the statistical optimality of these guarantees, we provide a converse result below.

Observation 4.2. *There exists a statistical learning problem in the Byzantine setting such that the output* $\widetilde{\mathbf{w}}$ *of* any *algorithm must satisfy* $\|\nabla F(\widetilde{\mathbf{w}})\|_2 = \Omega\big(\frac{\alpha}{\sqrt{n}} + \frac{\sqrt{d}}{\sqrt{nm}}\big)$ *with a constant probability.*

In view of this observation, we see that in terms of the first-order guarantee (i.e., on $\|\nabla F(\widetilde{\mathbf{w}})\|_2$) and up to logarithmic factors, trimmed mean is optimal if $d = \mathcal{O}(1)$, the median is optimal if $d = \mathcal{O}(1)$ and $n \gtrsim m$, and iterative filtering is optimal if $\alpha = \Theta(1)$. The statistical optimality of their second-order guarantees (i.e., on $\lambda_{\min}(\nabla^2 F(\widetilde{\mathbf{w}}))$) is currently unclear to us, and we believe this is an interesting problem for future investigation.

4.6 Conclusions

In this chapter, we study security issues that arise in large-scale distributed learning because of the presence of saddle points in non-convex loss functions. We observe that in the presence of non-convexity and Byzantine machines, escaping saddle points becomes much more challenging. We develop ByzantinePGD, a computation- and communication-efficient algorithm that is able to provably escape saddle points and converge to a second-order stationary point, even in the presence of Byzantine machines. We also discuss three different choices of the robust gradient and function value aggregation subroutines in ByzantinePGD—median, trimmed mean, and the iterative filtering algorithm. We characterize their performance in statistical settings, and argue for their near-optimality in different regimes including the high dimensional setting.

4.7 Proofs

Challenges of Escaping Saddle Points in the Adversarial Setting

We provide two examples showing that in non-convex setting with saddle points, inexact oracle can lead to much worse sub-optimal solutions than in the convex setting, and that in the adversarial setting, escaping saddle points can be inherently harder than the adversary-free case.

Consider standard gradient descent using exact or Δ-inexact gradients. Our first example shows that Byzantine machines have a more severe impact in the non-convex case than in the convex case.

Example 1. Let $d = 1$ and consider the functions $F^{(1)}(w) = (w-1)^2$ and $F^{(2)}(w) = (w^2-1)^2/4$. Here $F^{(1)}$ is strongly convex with a unique local minimizer $w^* = 1$, whereas $F^{(2)}$ has two local (in fact, global) minimizers $w^* = \pm 1$ and a saddle point (in fact, a local maximum) $w = 0$. Proposition 4.3 below shows the following: for the convex $F^{(1)}$, gradient descent (GD) finds a near-optimal solution with sub-optimality proportional to Δ, regardless of initialization; for the nonconvex $F^{(2)}$, GD initialized near the saddle point $w = 0$ suffers from an $\Omega(1)$ sub-optimality gap.

Proposition 4.3. *Suppose that $\Delta \le 1/2$. Under the setting above, the following holds.*
(i) For $F^{(1)}$, starting from any w_0, GD using a Δ-inexact gradient oracle finds w with $F^{(1)}(w) - F^{(1)}(w^*) \le \mathcal{O}(\Delta)$.
(ii) For $F^{(2)}$, there exists an adversarial strategy such that starting from a w_0 sampled uniformly from $[-r, r]$, GD with a Δ-inexact gradient oracle outputs w with $F^{(2)}(w) - F^{(2)}(w^*) \ge \frac{9}{64}, \forall w^* = \pm 1$, with probability $\min\{1, \frac{\Delta}{r}\}$.

Proof. Since $F^{(2)}(w) = \frac{1}{4}(w^2-1)^2$, we have $\nabla F^{(2)}(w) = w^3 - w$. For any $w \in [-\Delta, \Delta]$, $|\nabla F^{(2)}(w)| \le \Delta$ (since $\Delta \le 1/2$). Thus, the adversarial oracle can always output $\widehat{g}(w) = 0$ when $w \in [-\Delta, \Delta]$, and we have $|\widehat{g}(w) - \nabla F^{(2)}(w)| \le \Delta$. Thus, if $w \in [-\Delta, \Delta]$, the iterate can no longer move with this adversarial strategy. Then, we have $F^{(2)}(w) - F^{(2)}(w^*) \ge F^{(2)}(\Delta) - 0 \ge \frac{9}{64}$ (since $\Delta \le 1/2$). The result for the convex function $F^{(1)}$ is a direct corollary of Theorem 1 in Chapter 3. $\square$

Our second example shows that escaping saddle points is much harder in the Byzantine setting than in the non-Byzantine setting.

Example 2. Let $d = 2$, and assume that in the neighborhood $\mathbb{B}_0(b)$ of the origin, F takes the quadratic form $F(\mathbf{w}) \equiv \frac{1}{2}w_1^2 - \frac{\lambda}{2}w_2^2$, with $\lambda > \epsilon_H$.[3] The origin $\mathbf{w}_0 = 0$ is not an (ϵ_g, ϵ_H)-second-order stationary point, but rather a saddle point. Proposition 4.4 below shows that exact GD escapes the saddle point almost surely, while GD with an inexact oracle fails to do so.

[3] $F(\mathbf{w}) \equiv \frac{1}{2}w_1^2 - \frac{\lambda}{2}w_2^2$ holds locally around the origin, not globally; otherwise $F(\mathbf{w})$ has no minimum.

Proposition 4.4. *Under the setting above, if one chooses $r < b$ and sample $\mathbf{w}$ from $\mathbb{B}_0(r)$ uniformly at random, then:*
(i) Using exact gradient descent, with probability 1, the iterate $\mathbf{w}$ eventually leaves $\mathbb{B}_0(r)$.
(ii) There exists an adversarial strategy such that, when we update $\mathbf{w}$ using Δ-inexact gradient oracle, if $\Delta \geq \lambda r$, with probability 1, the iterate $\mathbf{w}$ cannot leave $\mathbb{B}_0(r)$; otherwise with probability $\frac{2}{\pi}\Big(\arcsin\big(\frac{\Delta}{\lambda r}\big) + \frac{\Delta}{\lambda r}\sqrt{1-(\frac{\Delta}{\lambda r})^2}\Big)$ the iterate $\mathbf{w}$ cannot leave $\mathbb{B}_0(r)$.

Proof. Since $F(\mathbf{w}) = \frac{1}{2}w_1^2 - \frac{1}{2}\lambda w_2^2, \forall\, \mathbf{w} \in \mathbb{B}_0(r)$, we have $\nabla F(\mathbf{w}) = [w_1,\ -\lambda w_2]^\top$. Sample $\mathbf{w}_0$ uniformly at random from $\mathbb{B}_0(r)$, and we know that with probability 1, $w_{0,2} \neq 0$. Then, by running exact gradient descent $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta\nabla F(\mathbf{w}_t)$, we can see that the second coordinate of $\mathbf{w}_t$ is $w_{t,2} = (1+\eta\lambda)^t w_{0,2}$. When $w_{0,2}$, we know that as t gets large, we eventually have $w_{t,2} > r$, which implies that the iterate leaves $\mathbb{B}_0(r)$.

On the other hand, suppose that we run Δ-inexact gradient descent, i.e., $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta\widehat{\mathbf{g}}(\mathbf{w}_t)$ with $\|\widehat{\mathbf{g}}(\mathbf{w}_t) - \nabla F(\mathbf{w}_t)\|_2 \leq \Delta$. In the first step, if $|w_{0,2}| \leq \frac{\Delta}{\lambda}$, the adversary can simply replace $\nabla F(\mathbf{w}_0)$ with $\widehat{\mathbf{g}}(\mathbf{w}_0) = [w_{0,1},\ 0]^\top$ (one can check that here we have $\|\widehat{\mathbf{g}}(\mathbf{w}_0) - \nabla F(\mathbf{w}_0)\|_2 \leq \Delta$), and then the second coordinate of $\mathbf{w}_1$ does not change, i.e., $w_{1,2} = w_{0,2}$. In the following iterations, the adversary can keep using the same strategy and the second coordinate of $\mathbf{w}$ never changes, and then the iterates cannot escape $\mathbb{B}_0(r)$, since $F(\mathbf{w})$ is a strongly convex function in its first coordinate. To compute the probability of getting stuck at the saddle point, we only need to compute the area of the region $\{\mathbf{w} \in \mathbb{B}_0(r) : |w_2| \leq \frac{\Delta}{\lambda}\}$, which can be done via simple geometry. □

Remark. Even if we choose the largest possible perturbation in $\mathbb{B}_0(r)$, i.e., sample $\mathbf{w}$ from the circle $\{\mathbf{w} \in \mathbb{R}^2 : \|\mathbf{w}\|_2 = r\}$, the stuck region still exists. We can compute the length of the arc $\{\|\mathbf{w}\|_2 = r : |w_2| \leq \frac{\Delta}{\lambda}\}$ and find the probability of stuck. One can find that when $\Delta \geq \lambda r$, the probability of being stuck in $\mathbb{B}_0(r)$ is still 1, otherwise, the probability of being stuck is $\frac{2}{\pi}(\arcsin(\frac{\Delta}{\lambda r}))$.

The above examples show that the adversary can significantly alter the landscape of the function near a saddle point. We counter this by exerting a large perturbation on the iterate so that it escapes this bad region. The amount of perturbation is carefully calibrated to ensure that the algorithm finds a descent direction "steep" enough to be preserved under Δ-corruption, while not compromising the accuracy. Multiple rounds of perturbation are performed, boosting the escape probability exponentially.

Proof of Theorem 4.3

We first analyze the gradient descent step with Δ-inexact gradient oracle.

Lemma 4.1. *Suppose that $\eta = 1/L_F$. For any $\mathbf{w} \in \mathcal{W}$, if we run the following inexact gradient descent step:*

$$\mathbf{w}' = \mathbf{w} - \eta\widehat{\mathbf{g}}(\mathbf{w}), \tag{4.4}$$

with $\|\widehat{\mathbf{g}}(\mathbf{w}) - \nabla F(\mathbf{w})\|_2 \leq \Delta$. *Then, we have*

$$F(\mathbf{w}') \leq F(\mathbf{w}) - \frac{1}{2L_F}\|\nabla F(\mathbf{w})\|_2^2 + \frac{1}{2L_F}\Delta^2.$$

Proof. Since $F(\mathbf{w})$ is L_F smooth, we know that

$$\begin{aligned}F(\mathbf{w}') \leq & F(\mathbf{w}) + \langle \nabla F(\mathbf{w}), \mathbf{w}' - \mathbf{w}\rangle + \frac{L_F}{2}\|\mathbf{w}' - \mathbf{w}\|_2^2 \\ = & F(\mathbf{w}) - \langle \nabla F(\mathbf{w}), \frac{1}{L_F}(\widehat{\mathbf{g}}(\mathbf{w}) - \nabla F(\mathbf{w}))\rangle - \langle \nabla F(\mathbf{w}), \frac{1}{L_F}\nabla F(\mathbf{w})\rangle \\ & + \frac{1}{2L_F}\|\widehat{\mathbf{g}}(\mathbf{w}) - \nabla F(\mathbf{w}) + \nabla F(\mathbf{w})\|_2^2 \\ \leq & F(\mathbf{w}) - \frac{1}{2L_F}\|\nabla F(\mathbf{w})\|_2^2 + \frac{1}{2L_F}\Delta^2.\end{aligned}$$

□

Let ϵ be the threshold on $\|\widehat{\mathbf{g}}(\widetilde{\mathbf{w}})\|_2$ that the algorithm uses to determine whether or not to add perturbation. Choose $\epsilon = 3\Delta$. Suppose that at a particular iterate $\widetilde{\mathbf{w}}$, we observe $\|\widehat{\mathbf{g}}(\widetilde{\mathbf{w}})\|_2 > \epsilon$. Then, we know that

$$\|\nabla F(\widetilde{\mathbf{w}})\|_2 \geq \|\widehat{\mathbf{g}}(\widetilde{\mathbf{w}})\|_2 - \Delta \geq 2\Delta.$$

According to Lemma 4.1, by running one iteration of the inexact gradient descent step, the decrease in function value is at least

$$\frac{1}{2L_F}\|\nabla F(\widetilde{\mathbf{w}})\|_2^2 - \frac{1}{2L_F}\Delta^2 \geq \frac{3\Delta^2}{2L_F}. \tag{4.5}$$

We proceed to analyze the perturbation step, which happens when the algorithm arrives at an iterate $\widetilde{\mathbf{w}}$ with $\|\widehat{\mathbf{g}}(\widetilde{\mathbf{w}})\|_2 \leq \epsilon$. In this proof, we slightly abuse the notation. Recall that in equation (4.2) in Section 4.4 , we use $\mathbf{w}'_t$ ($0 \leq t \leq T_{\text{th}}$) to denote the iterates of the algorithm in the saddle point escaping process. Here, we simply use $\mathbf{w}_t$ to denote these iterates. We start with the definition of *stuck region* at $\widetilde{\mathbf{w}} \in \mathcal{W}$.

Definition 4.5. *Given* $\widetilde{\mathbf{w}} \in \mathcal{W}$, *and parameters* r, R, *and* T_{th}, *the stuck region*

$$\mathbb{W}_S(\widetilde{\mathbf{w}}, r, R, T_{\text{th}}) \subseteq \mathbb{B}_{\widetilde{\mathbf{w}}}(r)$$

is a set of $\mathbf{w}_0 \in \mathbb{B}_{\widetilde{\mathbf{w}}}(r)$ *which satisfies the following property: there exists an adversarial strategy such that when we start with* $\mathbf{w}_0$ *and run* T_{th} *gradient descent steps with* Δ*-inexact gradient oracle* $\widehat{\mathbf{g}}(\mathbf{w})$*:*

$$\mathbf{w}_t = \mathbf{w}_{t-1} - \eta\widehat{\mathbf{g}}(\mathbf{w}_{t-1}),\ t = 1, 2, \ldots, T_{\text{th}}, \tag{4.6}$$

we observe $\|\mathbf{w}_t - \mathbf{w}_0\|_2 < R$, $\forall\ t \leq T_{\text{th}}$.

When it is clear from the context, we may simply use the terminology stuck region $\mathbb{W}_S$ at $\widetilde{\mathbf{w}}$. The following lemma shows that if $\nabla^2 F(\widetilde{\mathbf{w}})$ has a large negative eigenvalue, then the stuck region has a small width along the direction of the eigenvector associated with this negative eigenvalue.

Lemma 4.2. *Assume that the smallest eigenvalue of* $\mathbf{H} := \nabla^2 F(\widetilde{\mathbf{w}})$ *satisfies* $\lambda_{\min}(\mathbf{H}) \leq -\gamma < 0$*, and let the unit vector* $\mathbf{e}$ *be the eigenvector associated with* $\lambda_{\min}(\mathbf{H})$*. Let* $\mathbf{u}_0, \mathbf{y}_0 \in \mathbb{B}_{\widetilde{\mathbf{w}}}(r)$ *be two points such that* $\mathbf{y}_0 = \mathbf{u}_0 + \mu_0 \mathbf{e}$ *with some* $\mu_0 \geq \mu \in (0, r)$*. Choose step size* $\eta = \frac{1}{L_F}$*, and consider the stuck region* $\mathbb{W}_S(\widetilde{\mathbf{w}}, r, R, T_{\text{th}})$*. Suppose that* r*,* R*,* T_{th}*, and* μ *satisfy* [4]

$$T_{\text{th}} = \frac{2}{\eta\gamma}\log_{9/4}(\frac{2(R+r)}{\mu}), \tag{4.7}$$

$$R \geq \mu, \tag{4.8}$$

$$\rho_F(R+r)\mu \geq \Delta, \tag{4.9}$$

$$\gamma \geq 24\rho_F(R+r)\log_{9/4}(\frac{2(R+r)}{\mu}). \tag{4.10}$$

Then, there must be either $\mathbf{u}_0 \notin \mathbb{W}_S$ *or* $\mathbf{y}_0 \notin \mathbb{W}_S$*.*

With this lemma, we proceed to analyze the probability that the algorithm escapes the saddle points. In particular, we bound the probability that $\mathbf{w}_0 \in \mathbb{W}_S(\widetilde{\mathbf{w}}, r, R, T_{\text{th}})$ when $\lambda_{\min}(\nabla^2 F(\widetilde{\mathbf{w}})) \leq -\gamma$ and $\mathbf{w}_0$ is drawn from $\mathbb{B}_{\widetilde{\mathbf{w}}}(r)$ uniform at random.

Lemma 4.3. *Assume that* $\lambda_{\min}(\nabla^2 F(\widetilde{\mathbf{w}})) \leq -\gamma < 0$*, and let the unit vector* $\mathbf{e}$ *be the eigenvector associated with* $\lambda_{\min}(\nabla^2 F(\widetilde{\mathbf{w}}))$*. Consider the stuck region* $\mathbb{W}_S(\widetilde{\mathbf{w}}, r, R, T_{\text{th}})$ *at* $\widetilde{\mathbf{w}}$*, and suppose that* r*,* R*,* T_{th}*, and* μ *satisfy the conditions in* (4.7)-(4.10)*. Then, when we sample* $\mathbf{w}_0$ *from* $\mathbb{B}_{\widetilde{\mathbf{w}}}(r)$ *uniformly at random, the probability that* $\mathbf{w}_0 \in \mathbb{W}_S(\widetilde{\mathbf{w}}, r, R, T_{\text{th}})$ *is at most* $\frac{2\mu\sqrt{d}}{r}$*.*

Proof. Since the starting point $\mathbf{w}_0$ is uniformly distributed in $\mathbb{B}_{\widetilde{\mathbf{w}}}(r)$, to bound the probability of getting stuck, it suffices to bound the volume of $\mathbb{W}_S$. Let $\mathbb{1}_{\mathbb{W}_S}(\mathbf{w})$ be the indicator function of the set $\mathbb{W}_S$. For any $\mathbf{w} \in \mathbb{R}^d$, let $w^{(1)}$ be the projection of $\mathbf{w}$ onto the $\mathbf{e}$ direction, and $\mathbf{w}^{(-1)} \in \mathbb{R}^{d-1}$ be the remaining component of $\mathbf{w}$. Then, we have

$$\begin{aligned}
\text{Vol}(\mathbb{W}_S) =& \int_{\mathbb{B}^{(d)}_{\widetilde{\mathbf{w}}}(r)} \mathbb{1}_{\mathbb{W}_S}(\mathbf{w})\text{d}\mathbf{w} \\
=& \int_{\mathbb{B}^{(d-1)}_{\widetilde{\mathbf{w}}}(r)} \text{d}\mathbf{w}^{(-1)} \int_{\widetilde{w}^{(1)}-\sqrt{r^2-\|\widetilde{\mathbf{w}}^{(-1)}-\mathbf{w}^{(-1)}\|_2^2}}^{\widetilde{w}^{(1)}+\sqrt{r^2-\|\widetilde{\mathbf{w}}^{(-1)}-\mathbf{w}^{(-1)}\|_2^2}} \mathbb{1}_{\mathbb{W}_S}(\mathbf{w})\text{d}\widetilde{w}^{(1)} \\
\leq& 2\mu \int_{\mathbb{B}^{(d-1)}_{\widetilde{\mathbf{w}}}(r)} \text{d}\mathbf{w}^{(-1)} \\
=& 2\mu\text{Vol}(\mathbb{B}^{(d-1)}_0(r)),
\end{aligned}$$

[4]Without loss of generality, here we assume that $\frac{2}{\eta\gamma}\log_{9/4}(\frac{2(R+r)}{\mu})$ is an integer, so that T_{th} is an integer.

where the inequality is due to Lemma 4.2. Then, we know that the probability of getting stuck is

$$\frac{\mathrm{Vol}(\mathbb{W}_S)}{\mathrm{Vol}(\mathbb{B}_0^{(d)}(r))} \le 2\mu\frac{\mathrm{Vol}(\mathbb{B}_0^{(d-1)}(r))}{\mathrm{Vol}(\mathbb{B}_0^{(d)}(r))} = \frac{2\mu}{\sqrt{\pi}r}\frac{\Gamma(\frac{d}{2}+1)}{\Gamma(\frac{d}{2}+\frac{1}{2})} \le \frac{2\mu}{\sqrt{\pi}r}\sqrt{\frac{d}{2}+\frac{1}{2}} \le \frac{2\mu\sqrt{d}}{r},$$

where we use the fact that $\frac{\Gamma(x+1)}{\Gamma(x+\frac{1}{2})} < \sqrt{x+\frac{1}{2}}$ for any $x \ge 0$. □

We then analyze the decrease of value of the population loss function $F(\cdot)$ when we conduct the perturbation step. Assume that we successfully escape the saddle point, i.e., there exists $t \le T_{\mathrm{th}}$ such that $\|\mathbf{w}_t - \mathbf{w}_0\|_2 \ge R$. The following lemma provides the decrease of $F(\cdot)$.

Lemma 4.4. *Suppose that* $\lambda_{\min}(\nabla^2 F(\widetilde{\mathbf{w}})) \le -\gamma < 0$*, and at* $\widetilde{\mathbf{w}}$*, we observe* $\|\widehat{\mathbf{g}}(\widetilde{\mathbf{w}})\|_2 \le \epsilon = 3\Delta$*. Assume that* $\mathbf{w}_0 \in \mathbb{B}_{\widetilde{\mathbf{w}}}(r)$ *and that* $\mathbf{w}_0 \notin \mathbb{W}_S(\widetilde{\mathbf{w}}, r, R, T_{\mathrm{th}})$*. Let* $t \le T_{\mathrm{th}}$ *be the step such that* $\|\mathbf{w}_t - \mathbf{w}_0\|_2 \ge R$*. Then, we have*

$$F(\widetilde{\mathbf{w}}) - F(\mathbf{w}_t) \ge \frac{L_F}{4T_{\mathrm{th}}}R^2 - \frac{\Delta^2 T_{\mathrm{th}}}{L_F} - 4\Delta r - \frac{L_F}{2}r^2. \tag{4.11}$$

Next, we choose the quantities μ, r, R, and γ such that (i) the conditions (4.7)-(4.10) in Lemma 4.2 are satisfied, (ii) the probability of escaping saddle point in Lemma 4.3 is at least a constant, and (iii) the decrease in function value in (4.11) is large enough. We first choose

$$\mu = \Delta^{3/5} d^{-1/5} \rho_F^{-1/2}, \tag{4.12}$$

$$r = 4\Delta^{3/5} d^{3/10} \rho_F^{-1/2}, \tag{4.13}$$

$$R = \Delta^{2/5} d^{1/5} \rho_F^{-1/2}. \tag{4.14}$$

One can simply check that, according to Lemma 4.3, when we drawn $\mathbf{w}_0$ from $\mathbb{B}_{\widetilde{\mathbf{w}}}(r)$ uniformly at random, the probability that $\mathbf{w}_0 \in \mathbb{W}_S(\widetilde{\mathbf{w}}, r, R, T_{\mathrm{th}})$ is at most $1/2$. Since we assume that $\Delta \le 1$, one can also check that the condition (4.8) is satisfied. In addition, since $\rho_F R\mu = \Delta$, the condition (4.9) is also satisfied. According to (4.7), we have

$$T_{\mathrm{th}} = \frac{2L_F}{\gamma}\log_{9/4}\left(\frac{2d^{2/5}}{\Delta^{1/5}} + 8d^{1/2}\right). \tag{4.15}$$

In the following, we choose

$$\gamma = 768(\rho_F^{1/2} + L_F)(\Delta^{2/5}d^{1/5} + \Delta^{3/5}d^{3/10})\log_{9/4}\left(\frac{2d^{2/5}}{\Delta^{1/5}} + 8d^{1/2}\right), \tag{4.16}$$

which implies

$$T_{\mathrm{th}} = \frac{L_F}{384(\rho_F^{1/2} + L_F)(\Delta^{2/5}d^{1/5} + \Delta^{3/5}d^{3/10})} \tag{4.17}$$

Then we check condition (4.10) holds. We have

$$24\rho_F(R+r)\log_{9/4}(\frac{2(R+r)}{\mu}) = 24\rho_F^{1/2}(\Delta^{2/5}d^{1/5}+4\Delta^{3/5}d^{3/10})\log_{9/4}(\frac{2d^{2/5}}{\Delta^{1/5}}+8d^{1/2}) \le \gamma.$$

Next, we consider the decrease in function value in (4.11). Using the equations (4.15) and (4.16), we can show the following three inequalities by direct algebra manipulation.

$$\frac{L_F}{4T_{\text{th}}}R^2 \ge 6\frac{\Delta^2 T_{\text{th}}}{L_F}, \tag{4.18}$$

$$\frac{L_F}{4T_{\text{th}}}R^2 \ge 24\Delta r, \tag{4.19}$$

$$\frac{L_F}{4T_{\text{th}}}R^2 \ge 3L_F r^2. \tag{4.20}$$

By adding up (4.18), (4.19), and (4.20), we obtain

$$\frac{L_F}{4T_{\text{th}}}R^2 \ge 2\frac{\Delta^2 T_{\text{th}}}{L_F} + 8\Delta r + L_F r^2,$$

which implies that when we successfully escape the saddle point, we have

$$F(\widetilde{\mathbf{w}}) - F(\mathbf{w}_t) \ge \frac{L_F}{8T_{\text{th}}}R^2 = 48(\rho_F^{-1/2}+L_F\rho_F^{-1})(\Delta^{6/5}d^{3/5}+\Delta^{7/5}d^{7/10}). \tag{4.21}$$

Then, one can simply check that, the average decrease in function value during the successful round of the Escape process is

$$\frac{F(\widetilde{\mathbf{w}}) - F(\mathbf{w}_t)}{t} \ge \frac{F(\widetilde{\mathbf{w}}) - F(\mathbf{w}_t)}{T_{\text{th}}} \ge \frac{2(\Delta^{8/5}d^{4/5}+\Delta^2 d)}{L_F} > \frac{3\Delta^2}{2L_F}. \tag{4.22}$$

Recall that according to (4.5), when the algorithm is not in the Escape process, the function value is decreased by at least $\frac{3\Delta^2}{2L_F}$ in each iteration. Therefore, if the algorithm successfully escapes the saddle point, during the Escape process, the average decrease in function value is *larger* than the iterations which are not in this process.

So far, we have chosen the algorithm parameters r, R, T_{th}, as well as the final second-order convergence guarantee γ. Now we proceed to analyze the total number of iterations and the failure probability of the algorithm. According to Lemma 4.3 and the choice of μ and r, we know that at each point with $\|\widehat{\mathbf{g}}(\widetilde{\mathbf{w}})\|_2 \le \epsilon$, the algorithm can successfully escape this saddle point with probability at least $1/2$. To boost the probability of escaping saddle points, we need to repeat the process Q rounds in Escape, independently. Since for each successful round, the function value decrease is at least

$$48(\rho_F^{-1/2}+L_F\rho_F^{-1})(\Delta^{6/5}d^{3/5}+\Delta^{7/5}d^{7/10}) \ge 48L_F\rho_F^{-1}(\Delta^{6/5}d^{3/5}+\Delta^{7/5}d^{7/10}),$$

and the function value can decrease at most $F_0 - F^*$. Therefore, the total number of saddle points that we need to escape is at most

$$\frac{\rho_F(F_0 - F^*)}{48L_F(\Delta^{6/5}d^{3/5} + \Delta^{7/5}d^{7/10})}. \tag{4.23}$$

Therefore, by union bound, the failure probability of the algorithm is at most

$$\frac{\rho_F(F_0 - F^*)}{48L_F(\Delta^{6/5}d^{3/5} + \Delta^{7/5}d^{7/10})}(\frac{1}{2})^Q,$$

and to make the failure probability at most δ, one can choose

$$Q \geq 2\log\left(\frac{\rho_F(F_0 - F^*)}{48L_F\delta(\Delta^{6/5}d^{3/5} + \Delta^{7/5}d^{7/10})}\right). \tag{4.24}$$

Again, due to the fact that the function value decrease is at most $F_0 - F^*$, and in each *effective* iteration, the function value is decreased by at least $\frac{3\Delta^2}{2L_F}$. (Here, the effective iterations are the iterations when the algorithm is not in the Escape process and the iterations when the algorithm successfully escapes the saddle points.) The total number of effective iterations is at most

$$\frac{2(F_0 - F^*)L_F}{3\Delta^2}. \tag{4.25}$$

Combing with (4.24), we know that the total number of parallel iterations is at most

$$\frac{4(F_0 - F^*)L_F}{3\Delta^2}\log\left(\frac{\rho_F(F_0 - F^*)}{48L_F\delta(\Delta^{6/5}d^{3/5} + \Delta^{7/5}d^{7/10})}\right).$$

When all the algorithm terminates, and the saddle point escaping process is successful, the output of the algorithm $\widetilde{\mathbf{w}}$ satisfies $\|\widehat{\mathbf{g}}(\widetilde{\mathbf{w}})\|_2 \leq \epsilon$, which implies that $\|\nabla F(\widetilde{\mathbf{w}})\|_2 \leq 4\Delta$, and

$$\begin{aligned}\lambda_{\min}(\nabla^2 F(\widetilde{\mathbf{w}})) \geq -\gamma &= -768(\rho_F^{1/2} + L_F)(\Delta^{2/5}d^{1/5} + \Delta^{3/5}d^{3/10})\log_{9/4}(\frac{2d^{2/5}}{\Delta^{1/5}} + 8d^{1/2}) \\ &\geq -950(\rho_F^{1/2} + L_F)(\Delta^{2/5}d^{1/5} + \Delta^{3/5}d^{3/10})\log(\frac{2d^{2/5}}{\Delta^{1/5}} + 8d^{1/2}).\end{aligned} \tag{4.26}$$

We next show that we can simplify the guarantee as

$$\lambda_{\min}(\nabla^2 F(\widetilde{\mathbf{w}})) \geq -1900(\rho_F^{1/2} + L_F)\Delta^{2/5}d^{1/5}\log(\frac{10}{\Delta}). \tag{4.27}$$

We can see that if $\Delta \leq \frac{1}{\sqrt{d}}$, then $\Delta^{3/5}d^{3/10} \leq \Delta^{2/5}d^{1/5}$ and $\frac{2d^{2/5}}{\Delta^{1/5}} + 8d^{1/2} \leq \frac{10}{\Delta}$. Thus, the bound (4.27) holds. On the other hand, when $\Delta > \frac{1}{\sqrt{d}}$, we have $\Delta^{2/5}d^{1/5} > 1$ and thus

$$1900(\rho_F^{1/2} + L_F)\Delta^{2/5}d^{1/5}\log(\frac{10}{\Delta}) > L_F.$$

By the smoothness of $F(\cdot)$, we know that $\lambda_{\min}(\nabla^2 F(\widetilde{\mathbf{w}})) \geq -L_F$. Therefore, the bound (4.27) still holds, and this completes the proof.

Proof of Lemma 4.2

We prove by contradiction. Suppose that $\mathbf{u}_0, \mathbf{y}_0 \in \mathbb{W}_S$. Let $\{\mathbf{u}_t\}$ and $\{\mathbf{y}_t\}$ be two sequences generated by the following two iterations:

$$\mathbf{u}_t = \mathbf{u}_{t-1} - \eta\widehat{\mathbf{g}}(\mathbf{u}_{t-1}), \tag{4.28}$$

$$\mathbf{y}_t = \mathbf{y}_{t-1} - \eta\widehat{\mathbf{g}}(\mathbf{y}_{t-1}), \tag{4.29}$$

respectively, where $\|\widehat{\mathbf{g}}(\mathbf{w}) - \nabla F(\mathbf{w})\|_2 \le \Delta$ for any $\mathbf{w} \in \mathcal{W}$. According to our assumption, we have $\forall\ t \le T_{\text{th}}$, $\|\mathbf{u}_t - \mathbf{u}_0\|_2 < R$ and $\|\mathbf{y}_t - \mathbf{y}_0\|_2 < R$.

Define $\mathbf{v}_t := \mathbf{y}_t - \mathbf{u}_t$, $\boldsymbol{\delta}_t := \widehat{\mathbf{g}}(\mathbf{u}_t) - \nabla F(\mathbf{u}_t)$, and $\boldsymbol{\delta}'_t := \widehat{\mathbf{g}}(\mathbf{y}_t) - \nabla F(\mathbf{y}_t)$. Then we have

$$\begin{aligned}
\mathbf{y}_{t+1} &= \mathbf{y}_t - \eta(\nabla F(\mathbf{y}_t) + \boldsymbol{\delta}'_t) \\
&= \mathbf{u}_t + \mathbf{v}_t - \eta(\nabla F(\mathbf{u}_t + \mathbf{v}_t) + \boldsymbol{\delta}'_t) \\
&= \mathbf{u}_t + \mathbf{v}_t - \eta\nabla F(\mathbf{u}_t) - \eta\left[\int_0^1 \nabla^2 F(\mathbf{u}_t + \theta\mathbf{v}_t)\right]\mathbf{v}_t - \eta\boldsymbol{\delta}'_t \\
&= \mathbf{u}_{t+1} + \eta\boldsymbol{\delta}_t + \mathbf{v}_t - \eta\left[\int_0^1 \nabla^2 F(\mathbf{u}_t + \theta\mathbf{v}_t)\mathrm{d}\theta\right]\mathbf{v}_t - \eta\boldsymbol{\delta}'_t,
\end{aligned}$$

which yields

$$\mathbf{v}_{t+1} = (\mathbf{I} - \eta\mathbf{H})\mathbf{v}_t - \eta\mathbf{Q}_t\mathbf{v}_t + \eta(\boldsymbol{\delta}_t - \boldsymbol{\delta}'_t), \tag{4.30}$$

where

$$\mathbf{Q}_t := \int_0^1 \nabla^2 F(\mathbf{u}_t + \theta\mathbf{v}_t)\mathrm{d}\theta - \mathbf{H}. \tag{4.31}$$

By the Hessian Lipschitz property, we know that

$$\begin{aligned}
\|\mathbf{Q}_t\|_2 \le& \rho_F(\|\mathbf{u}_t - \widetilde{\mathbf{w}}\|_2 + \|\mathbf{y}_t - \widetilde{\mathbf{w}}\|_2) \\
\le& \rho_F(\|\mathbf{u}_t - \mathbf{u}_0\|_2 + \|\mathbf{u}_0 - \widetilde{\mathbf{w}}\|_2 + \|\mathbf{y}_t - \mathbf{y}_0\|_2 + \|\mathbf{y}_0 - \widetilde{\mathbf{w}}\|_2) \\
\le& 2\rho_F(R + r).
\end{aligned} \tag{4.32}$$

We let ψ_t be the norm of the projection of $\mathbf{v}_t$ onto the $\mathbf{e}$ direction, and ϕ_t be the norm of the projection of $\mathbf{v}_t$ onto the remaining subspace. By definition, we have $\psi_0 = \mu_0 \ge \mu > 0$ and $\phi_0 = 0$. According to (4.30) and (4.32), we have

$$\psi_{t+1} \ge (1 + \eta\gamma)\psi_t - 2\eta\rho_F(R + r)\sqrt{\psi_t^2 + \phi_t^2} - 2\eta\Delta, \tag{4.33}$$

$$\phi_{t+1} \le (1 + \eta\gamma)\phi_t + 2\eta\rho_F(R + r)\sqrt{\psi_t^2 + \phi_t^2} + 2\eta\Delta. \tag{4.34}$$

In the following, we use induction to prove that $\forall\ t \le T_{\text{th}}$,

$$\psi_t \ge (1 + \frac{1}{2}\eta\gamma)\psi_{t-1} \quad \text{and} \quad \phi_t \le \frac{t}{T_{\text{th}}}\psi_t \tag{4.35}$$

We know that (4.35) holds when $t = 0$ since we have $\phi_0 = 0$. Then, assume that for some $t < T_{\text{th}}$, we have $\forall\ \tau \leq t$, $\psi_\tau \geq (1 + \frac{1}{2}\eta\gamma)\psi_{\tau-1}$ and $\phi_\tau \leq \frac{\tau}{T_{\text{th}}}\psi_\tau$. We show that (4.35) holds for $t + 1$.

First, we show that $\psi_{t+1} \geq (1 + \frac{1}{2}\eta\gamma)\psi_t$. Since $\forall\ \tau \leq t$, $\psi_\tau \geq \psi_{\tau-1}$, we know that $\psi_t \geq \psi_0 \geq \mu$. Therefore, according to (4.9), we have

$$\Delta \leq \rho_F(R + r)\mu \leq \rho_F(R + r)\psi_t. \tag{4.36}$$

In addition, since $t < T_{\text{th}}$, we have

$$\phi_t \leq \psi_t. \tag{4.37}$$

Combining (4.36), (4.37) and (4.33), (4.34), we get

$$\psi_{t+1} \geq (1 + \eta\gamma)\psi_t - 2\eta\rho_F(R + r)\sqrt{2\psi_t^2} - 2\eta\rho_F(R + r)\psi_t > (1 + \eta\gamma)\psi_t - 6\eta\rho_F(R + r)\psi_t, \tag{4.38}$$

$$\phi_{t+1} \leq (1 + \eta\gamma)\phi_t + 2\eta\rho_F(R + r)\sqrt{2\psi_t^2} + 2\eta\rho_F(R + r)\psi_t < (1 + \eta\gamma)\phi_t + 6\eta\rho_F(R + r)\psi_t. \tag{4.39}$$

According to (4.10), we have $\gamma \geq 24\rho_F(R + r)\log_{9/4}(\frac{2(R+r)}{\mu}) > 12\rho_F(R + r)$. Combining with (4.38), we know that $\psi_{t+1} \geq (1 + \frac{1}{2}\eta\gamma)\psi_t$.

Next, we show that $\phi_{t+1} \leq \frac{t+1}{T_{\text{th}}}\psi_{t+1}$. Combining with (4.38) and (4.39), we know that to show $\phi_{t+1} \leq \frac{t+1}{T_{\text{th}}}\psi_{t+1}$, it suffices to show

$$(1 + \eta\gamma)\phi_t + 6\eta\rho_F(R + r)\psi_t \leq \frac{t + 1}{T_{\text{th}}}[1 + \eta\gamma - 6\eta\rho_F(R + r)]\psi_t. \tag{4.40}$$

According to the induction assumption, we have $\phi_t \leq \frac{t}{T_{\text{th}}}\psi_t$. Then, to show (4.40), it suffices to show that

$$(1 + \eta\gamma)t + 6\eta\rho_F(R + r)T_{\text{th}} \leq (t + 1)[1 + \eta\gamma - 6\eta\rho_F(R + r)] \tag{4.41}$$

Since $t + 1 \leq T_{\text{th}}$, we know that to show (4.41), it suffices to show

$$12\eta\rho_F(R + r)T_{\text{th}} \leq 1. \tag{4.42}$$

Then, according to (4.7) and (4.10), we know that (4.42) holds, which completes the induction.

Next, according to (4.35), we know that

$$\begin{aligned}
\|\mathbf{u}_{T_{\text{th}}} - \mathbf{y}_{T_{\text{th}}}\|_2 &\geq \phi_{T_{\text{th}}} \geq (1 + \frac{1}{2}\eta\gamma)^{T_{\text{th}}}\mu_0 \\
&\geq (1 + \frac{1}{2}\eta\gamma)^{\frac{2}{\eta\gamma}\log_{9/4}(\frac{2(R+r)}{\mu})}\mu_0 \\
&\geq \frac{2(R + r)}{\mu} \cdot \mu_0 = 2(R + r),
\end{aligned}$$

where the last inequality is due to the fact that $\eta = \frac{1}{L_F}$ and thus $\eta\gamma \le 1$. On the other hand, since we assume that $\mathbf{u}_0, \mathbf{y}_0 \in \mathbb{W}_S$, we know that

$$\|\mathbf{u}_{T_{\text{th}}} - \mathbf{y}_{T_{\text{th}}}\|_2 \le \|\mathbf{u}_{T_{\text{th}}} - \mathbf{u}_0\|_2 + \|\mathbf{y}_{T_{\text{th}}} - \mathbf{y}_0\|_2 + \|\mathbf{u}_0 - \mathbf{y}_0\|_2 < 2(R+r),$$

which leads to contradiction and thus completes the proof.

Proof of Lemma 4.4

Recall that we have the iterations $\mathbf{w}_{\tau+1} = \mathbf{w}_\tau - \eta\widehat{\mathbf{g}}(\mathbf{w}_\tau)$ for all $\tau < t$. Let $\boldsymbol{\delta}_\tau = \nabla F(\mathbf{w}_\tau) - \widehat{\mathbf{g}}(\mathbf{w}_\tau)$, and then $\|\boldsymbol{\delta}_\tau\|_2 \le \Delta$. By the smoothness of $F(\cdot)$ and the fact that $\eta = \frac{1}{L_F}$, we have

$$\begin{aligned} F(\mathbf{w}_\tau) - F(\mathbf{w}_{\tau+1}) \ge & \langle \nabla F(\mathbf{w}_\tau), \mathbf{w}_\tau - \mathbf{w}_{\tau+1}\rangle - \frac{L_F}{2}\|\mathbf{w}_\tau - \mathbf{w}_{\tau+1}\|_2^2 \\ = & \left\langle \frac{\mathbf{w}_\tau - \mathbf{w}_{\tau+1}}{\eta} + \boldsymbol{\delta}_\tau, \mathbf{w}_\tau - \mathbf{w}_{\tau+1} \right\rangle - \frac{L_F}{2}\|\mathbf{w}_\tau - \mathbf{w}_{\tau+1}\|_2^2 \\ = & \frac{L_F}{2}\|\mathbf{w}_\tau - \mathbf{w}_{\tau+1}\|_2^2 + \langle \boldsymbol{\delta}_\tau, \mathbf{w}_\tau - \mathbf{w}_{\tau+1}\rangle \\ \ge & \frac{L_F}{4}\|\mathbf{w}_\tau - \mathbf{w}_{\tau+1}\|_2^2 - \frac{\|\boldsymbol{\delta}_\tau\|_2^2}{L_F} \\ \ge & \frac{L_F}{4}\|\mathbf{w}_\tau - \mathbf{w}_{\tau+1}\|_2^2 - \frac{\Delta^2}{L_F}. \end{aligned} \tag{4.43}$$

By summing up (4.43) for $\tau = 0, 1, \ldots, t-1$, we get

$$F(\mathbf{w}_0) - F(\mathbf{w}_t) \ge \frac{L_F}{4}\sum_{\tau=0}^{t-1}\|\mathbf{w}_\tau - \mathbf{w}_{\tau+1}\|_2^2 - \frac{\Delta^2 t}{L_F}. \tag{4.44}$$

Consider the k-th coordinate of $\mathbf{w}_\tau$ and $\mathbf{w}_{\tau+1}$, by Cauchy-Schwarz inequality, we have

$$\sum_{\tau=0}^{t-1}(w_{\tau,k} - w_{\tau+1,k})^2 \ge \frac{1}{t}(w_{0,k} - w_{t,k})^2,$$

which implies

$$\sum_{\tau=0}^{t-1}\|\mathbf{w}_\tau - \mathbf{w}_{\tau+1}\|_2^2 \ge \frac{1}{t}\|\mathbf{w}_0 - \mathbf{w}_t\|_2^2. \tag{4.45}$$

Combining (4.44) and (4.45), we obtain

$$F(\mathbf{w}_0) - F(\mathbf{w}_t) \ge \frac{L_F}{4t}\|\mathbf{w}_0 - \mathbf{w}_t\|_2^2 - \frac{\Delta^2 t}{L_F} \ge \frac{L_F}{4T_{\text{th}}}R^2 - \frac{\Delta^2 T_{\text{th}}}{L_F}. \tag{4.46}$$

On the other hand, by the smoothness of $F(\cdot)$, we have

$$F(\widetilde{\mathbf{w}}) - F(\mathbf{w}_0) \ge \langle \nabla F(\widetilde{\mathbf{w}}), \widetilde{\mathbf{w}} - \mathbf{w}_0\rangle - \frac{L_F}{2}\|\mathbf{w}_0 - \widetilde{\mathbf{w}}\|_2^2 \ge -(\epsilon + \Delta)r - \frac{L_F}{2}r^2. \tag{4.47}$$

Adding up (4.46) and (4.47), we obtain

$$F(\widetilde{\mathbf{w}}) - F(\mathbf{w}_t) \geq \frac{L_F}{4T_{\text{th}}}R^2 - \frac{\Delta^2 T_{\text{th}}}{L_F} - (\epsilon + \Delta)r - \frac{L_F}{2}r^2, \tag{4.48}$$

which completes the proof.

Proof of Theorem 4.4

First, when we run gradient descent iterations $\mathbf{w}' = \mathbf{w} - \eta\nabla F(\mathbf{w})$, according to Lemma 4.1, we have

$$F(\mathbf{w}') \leq F(\mathbf{w}) - \frac{1}{2L_F}\|\nabla F(\mathbf{w})\|_2^2. \tag{4.49}$$

Suppose at $\widetilde{\mathbf{w}}$, we observe that $\|\nabla F(\widetilde{\mathbf{w}})\|_2 \leq \epsilon$, and then we start the Escape process. When we have exact gradient oracle, we can still define the stuck region $\mathbb{W}_S$ at $\widetilde{\mathbf{w}}$ as in the definition of stuck region in Section 4.7, by simply replacing the inexact gradient oracle with the exact oracle. Then, we can analyze the size of the stuck region according to Lemma 4.2. Assume that the smallest eigenvalue of $\mathbf{H} := \nabla^2 F(\widetilde{\mathbf{w}})$ satisfies $\lambda_{\min}(\mathbf{H}) \leq -\gamma < 0$, and let the unit vector $\mathbf{e}$ be the eigenvector associated with $\lambda_{\min}(\mathbf{H})$. Let $\mathbf{u}_0, \mathbf{y}_0 \in \mathbb{B}_{\widetilde{\mathbf{w}}}(r)$ be two points such that $\mathbf{y}_0 = \mathbf{u}_0 + \mu_0\mathbf{e}$ with some $\mu_0 \geq \mu \in (0, r)$. Consider the stuck region $\mathbb{W}_S(\widetilde{\mathbf{w}}, r, R, T_{\text{th}})$. Suppose that r, R, T_{th}, and μ satisfy

$$T_{\text{th}} = \frac{2}{\eta\gamma}\log_{9/4}(\frac{2(R+r)}{\mu}), \tag{4.50}$$

$$R \geq \mu, \tag{4.51}$$

$$\gamma \geq 24\rho_F(R+r)\log_{9/4}(\frac{2(R+r)}{\mu}). \tag{4.52}$$

Then, there must be either $\mathbf{u}_0 \notin \mathbb{W}_S$ or $\mathbf{y}_0 \notin \mathbb{W}_S$. In addition, according to Lemma 4.3, if conditions (4.50)-(4.52) are satisfied, then, when we sample $\mathbf{w}_0$ from $\mathbb{B}_{\widetilde{\mathbf{w}}}(r)$ uniformly at random, the probability that $\mathbf{w}_0 \in \mathbb{W}_S(\widetilde{\mathbf{w}}, r, R, T_{\text{th}})$ is at most $\frac{2\mu\sqrt{d}}{r}$. In addition, according to (4.48) in the proof of Lemma 4.4, assume that $\mathbf{w}_0 \in \mathbb{B}_{\widetilde{\mathbf{w}}}(r)$ and that $\mathbf{w}_0 \notin \mathbb{W}_S(\widetilde{\mathbf{w}}, r, R, T_{\text{th}})$. Let $t \leq T_{\text{th}}$ be the step such that $\|\mathbf{w}_t - \mathbf{w}_0\|_2 \geq R$. Then, we have

$$F(\widetilde{\mathbf{w}}) - F(\mathbf{w}_t) \geq \frac{L_F}{4T_{\text{th}}}R^2 - \epsilon r - \frac{L_F}{2}r^2. \tag{4.53}$$

Combining (4.50) and (4.52), we know that the first term on the right hand side of (4.53) satisfies

$$\frac{L_F}{4T_{\text{th}}}R^2 \geq 3\rho_F R^3. \tag{4.54}$$

Choose $R = \sqrt{\epsilon/\rho_F}$ and $r = \epsilon$. Then, we know that when $\epsilon \leq \min\{\frac{1}{\rho_F}, \frac{4}{L_F^2\rho_F}\}$, we have $\epsilon r \leq \rho_F R^3$ and $\frac{1}{2}L_F r^2 \leq \rho_F R^3$. Combining these facts with (4.53) and (4.54), we know that, when the algorithm successfully escapes the saddle point, the decrease in function value satisfies

$$F(\widetilde{\mathbf{w}}) - F(\mathbf{w}_t) \geq \rho_F R^3. \tag{4.55}$$

Therefore, the average function value decrease during the Escape process is at least

$$\frac{F(\widetilde{\mathbf{w}}) - F(\mathbf{w}_t)}{T_{\text{th}}} \geq \frac{12}{L_F}\epsilon^2. \tag{4.56}$$

When we have exact gradient oracle, we choose $Q = 1$. According to (4.49) and (4.56), for the iterations that are not in the Escape process, the function value decrease in each iteration is at least $\frac{1}{2L_F}\epsilon^2$; for the iterations in the Escape process, the function value decrease on average is $\frac{12}{L_F}\epsilon^2$. Since the function value can decrease at most $F_0 - F^*$, the algorithm must terminate within $\frac{2L_F(F_0-F^*)}{\epsilon^2}$ iterations.

The we proceed to analyze the failure probability. We can see that the number of saddle points that the algorithm may need to escape is at most $\frac{F_0-F^*}{\rho_F R^3}$. Then, by union bound the probability that the algorithm fails to escape one of the saddle points is at most

$$\frac{2\mu\sqrt{d}}{r} \cdot \frac{F_0 - F^*}{\rho_F R^3}$$

By letting the above probability to be δ, we obtain

$$\mu = \frac{\delta \epsilon^{5/2}}{2\sqrt{\rho_F d}(F_0 - F^*)},$$

which completes the proof.

Proof of Proposition 4.1

We consider the following class of one-dimensional functions indexed by $s \in \mathbb{R}$:

$$\mathcal{F} = \{f_s(\cdot) : f_s(w) = \Delta^{3/2}\sin(\Delta^{-1/2}w + s), s \in \mathbb{R}\}.$$

Then, for each function $f_s(\cdot) \in \mathcal{F}$, we have

$$\nabla f_s(w) = \Delta \cos(\Delta^{-1/2}w + s),$$

and

$$\nabla^2 f_s(w) = -\Delta^{1/2}\sin(\Delta^{-1/2}w + s).$$

Thus, we always have $|\nabla f_s(w)| \leq \Delta, \forall w$. Therefore, the Δ-inexact gradient oracle can simply output 0 all the time. In addition, we verify that for all s and w, $|\nabla^2 f_s(w)| \leq \Delta^{1/2} \leq 1$ and $|\nabla^3 f_s(w)| = |-\cos(\Delta^{-1/2}w + s)| \leq 1$ under the assumption that $\Delta \leq 1$, so all the functions in $\mathcal{F}$ are 1-smooth and 1-Hessian Lipschitz as claimed.

In this case, the output of the algorithm does not depend on s, that is, the actual function that we aim to minimize. Consequently, for any output $\widetilde{w}$ of the algorithm, there exists $s \in \mathbb{R}$ such that $\Delta^{-1/2}\widetilde{w} + s = \pi/4$, and thus $|\nabla f_s(\widetilde{w})| = \Delta/\sqrt{2}$ and $\lambda_{\min}(\nabla^2 f_s(\widetilde{w})) = -\Delta^{1/2}/\sqrt{2}$.

Proof of Proposition 4.2

Suppose that during all the iterations, the Escape process is called $E+1$ times. In the first E times, the algorithm escapes the saddle points, and in the last Escape process, the algorithm does not escape and outputs $\widetilde{\mathbf{w}}$. For the first E processes, there might be up to Q rounds of perturb-and-descent operations, and we only consider the successful descent round. We can then partition the algorithm into $E+1$ segments. We denote the starting and ending iterates of the t-th segment by $\mathbf{w}_t$ and $\widetilde{\mathbf{w}}_t$, respectively, and denote the length (number of inexact gradient descent iterations) by T_t. When the algorithm reaches $\widetilde{\mathbf{w}}_t$, we randomly perturb $\widetilde{\mathbf{w}}_t$ to $\mathbf{w}_{t+1}$, and thus we have $\|\widetilde{\mathbf{w}}_t - \mathbf{w}_{t+1}\|_2 \le r$ for every $t = 0, 1, \dots, E-1$. According to (4.25), we know that

$$\sum_{t=0}^{E} T_t \le \frac{2(F_0 - F^*)L_F}{3\Delta^2} := \widetilde{T},$$

and according to (4.23), we have

$$E \le \frac{\rho_F(F_0 - F^*)}{48L_F(\Delta^{6/5}d^{3/5} + \Delta^{7/5}d^{7/10})}.$$

According to (4.46), we know that

$$F(\mathbf{w}_t) - F(\widetilde{\mathbf{w}}_t) \ge \frac{L_F}{4T_t}\|\mathbf{w}_t - \widetilde{\mathbf{w}}_t\|_2^2 - \frac{\Delta^2 T_t}{L_F},$$

which implies

$$\|\mathbf{w}_t - \widetilde{\mathbf{w}}_t\|_2 \le \frac{2}{\sqrt{L_F}}\sqrt{T_t(F(\mathbf{w}_t) - F(\widetilde{\mathbf{w}}_t))} + \frac{2\Delta T_t}{L_F}.$$

Then, by Cauchy-Schwarz inequality, we have

$$\sum_{t=0}^{E} \|\mathbf{w}_t - \widetilde{\mathbf{w}}_t\|_2 \le 2\sqrt{\frac{\widetilde{T}}{L_F}\sum_{t=0}^{E}(F(\mathbf{w}_t) - F(\widetilde{\mathbf{w}}_t))} + \frac{2\Delta\widetilde{T}}{L_F}. \tag{4.57}$$

On the other hand, we have

$$\sum_{t=0}^{E}(F(\mathbf{w}_t) - F(\widetilde{\mathbf{w}}_t)) + \sum_{t=0}^{E-1}(F(\widetilde{\mathbf{w}}_t) - F(\mathbf{w}_{t+1})) = F(\mathbf{w}_0) - F(\widetilde{\mathbf{w}}_E) \le F(\mathbf{w}_0) - F^*.$$

According to (4.47), we have

$$F(\widetilde{\mathbf{w}}_t) - F(\mathbf{w}_{t+1}) \ge -4\Delta r - \frac{L_F}{2}r^2,$$

and thus

$$\sum_{t=0}^{E}(F(\mathbf{w}_t) - F(\widetilde{\mathbf{w}}_t)) \le F(\mathbf{w}_0) - F^* + E(4\Delta r + \frac{L_F}{2}r^2) \tag{4.58}$$

Combining (4.57) and (4.58), and using the bounds for $\widetilde{T}$ and E, we obtain that

$$\sum_{t=0}^{E} \|\mathbf{w}_t - \widetilde{\mathbf{w}}_t\|_2 \le C_1 \frac{F(\mathbf{w}_0) - F^*}{\Delta}, \tag{4.59}$$

where $C_1 > 0$ is a quantity that only depends on L_F and ρ_F. In addition, we have

$$\sum_{t=0}^{E-1} \|\widetilde{\mathbf{w}}_t - \mathbf{w}_{t+1}\|_2 \le Er \le C_2 \frac{F(\mathbf{w}_0) - F^*}{\Delta^{3/5} d^{3/10} + \Delta^{4/5} d^{2/5}}, \tag{4.60}$$

where $C_2 > 0$ is a quantity that only depends on L_F and ρ_F. Combining (4.59) and (4.60), and using triangle inequality, we know that

$$\|\widetilde{\mathbf{w}}_E - \mathbf{w}_0\|_2 \le C_1 \frac{F(\mathbf{w}_0) - F^*}{\Delta} + C_2 \frac{F(\mathbf{w}_0) - F^*}{\Delta^{3/5} d^{3/10} + \Delta^{4/5} d^{2/5}} \le C \frac{F(\mathbf{w}_0) - F^*}{\Delta}.$$

Here, the last inequality is due to the fact that we consider the regime where $\Delta \to 0$, and C is a quantity that only depends on L_F and ρ_F. As a final note, the analysis above also applies to any iterate prior to the final output, and thus, all the iterates during the algorithm stays in the ℓ_2 ball centered at $\mathbf{w}_0$ with radius $C\frac{F(\mathbf{w}_0)-F^*}{\Delta}$.

Robust Estimation of Gradients

Iterative Filtering Algorithm

We describe an iterative filtering algorithm for robust mean estimation. The algorithm is originally proposed for robust mean estimation for Gaussian distribution in [58], and extended to sub-Gaussian distribution in [57]; then algorithm is reinterpreted in [170]. Here, we present the algorithm using the interpretation in [170]. Suppose that m random vectors $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m \in \mathbb{R}^d$ are drawn i.i.d. from some distribution with mean $\boldsymbol{\mu}$. An adversary observes all these vectors and changes an α fraction of them in an arbitrary fashion, and we only have access to the corrupted data points $\widehat{\mathbf{x}}_1, \widehat{\mathbf{x}}_2, \dots, \widehat{\mathbf{x}}_m$. The goal of the iterative filtering algorithm is to output an accurate estimate of the true mean $\boldsymbol{\mu}$ even when the dimension d is large. We provide the detailed procedure in Algorithm 4. Here, we note that the algorithm parameter σ needs to be chosen properly in order to achieve the best possible statistical error rate.

Proof of Theorem 4.5

To prove Theorem 4.5, we first state a result that bounds the error of the iterative filtering algorithm when the original data points $\{\mathbf{x}_i\}$ are deterministic. The following lemma is proved in [57, 170]; also see [174] for additional discussion.

Algorithm 4 Iterative Filtering [58, 57, 170]

```
Require: corrupted data $\widehat{\mathbf{x}}_1, \widehat{\mathbf{x}}_2, \dots, \widehat{\mathbf{x}}_m \in \mathbb{R}^d$, $\alpha \in [0, \frac{1}{4})$, and algorithm parameter $\sigma > 0$.
  $\mathcal{A} \leftarrow [m]$, $c_i \leftarrow 1$, and $\tau_i \leftarrow 0$, $\forall\, i \in \mathcal{A}$.
  while true do
    Let $\mathbf{W} \in \mathbb{R}^{|\mathcal{A}|\times|\mathcal{A}|}$ be a minimizer of the convex optimization problem:
      $$\min_{\substack{0 \le W_{ji} \le \frac{3+\alpha}{(1-\alpha)(3-\alpha)m} \\ \sum_{j\in\mathcal{A}} W_{ji} = 1}} \max_{\substack{\mathbf{U} \succeq 0 \\ \mathrm{tr}(\mathbf{U}) \le 1}} \sum_{i\in\mathcal{A}} c_i (\widehat{\mathbf{x}}_i - \sum_{j\in\mathcal{A}} \widehat{\mathbf{x}}_j W_{ji})^\top \mathbf{U} (\widehat{\mathbf{x}}_i - \sum_{j\in\mathcal{A}} \widehat{\mathbf{x}}_j W_{ji}),$$
    and $\mathbf{U} \in \mathbb{R}^{d\times d}$ be a maximizer of the convex optimization problem:
      $$\max_{\substack{\mathbf{U} \succeq 0 \\ \mathrm{tr}(\mathbf{U}) \le 1}} \min_{\substack{0 \le W_{ji} \le \frac{3+\alpha}{(1-\alpha)(3-\alpha)m} \\ \sum_{j\in\mathcal{A}} W_{ji} = 1}} \sum_{i\in\mathcal{A}} c_i (\widehat{\mathbf{x}}_i - \sum_{j\in\mathcal{A}} \widehat{\mathbf{x}}_j W_{ji})^\top \mathbf{U} (\widehat{\mathbf{x}}_i - \sum_{j\in\mathcal{A}} \widehat{\mathbf{x}}_j W_{ji}).$$
    $\forall\, i \in \mathcal{A},\ \tau_i \leftarrow (\widehat{\mathbf{x}}_i - \sum_{j\in\mathcal{A}} \widehat{\mathbf{x}}_j W_{ji})^\top \mathbf{U} (\widehat{\mathbf{x}}_i - \sum_{j\in\mathcal{A}} \widehat{\mathbf{x}}_j W_{ji})$.
    if $\sum_{i\in\mathcal{A}} c_i \tau_i > 8m\sigma^2$ then
      $\forall\, i \in \mathcal{A},\ c_i \leftarrow (1 - \frac{\tau_i}{\tau_{\max}}) c_i$, where $\tau_{\max} = \max_{i\in\mathcal{A}} \tau_i$.
      $\mathcal{A} \leftarrow \mathcal{A} \setminus \{i : c_i \le \frac{1}{2}\}$.
    else
      return $\widehat{\boldsymbol{\mu}} = \frac{1}{|\mathcal{A}|} \sum_{i\in\mathcal{A}} \widehat{\mathbf{x}}_i$
    end if
  end while
```

Lemma 4.5. *[57, 170] Let $\mathcal{S} := \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m\}$ be the set of original data points and $\boldsymbol{\mu}_{\mathcal{S}} := \frac{1}{m}\sum_{i=1}^m \mathbf{x}_i$ be their sample mean. Let $\widehat{\mathbf{x}}_1, \widehat{\mathbf{x}}_2, \dots, \widehat{\mathbf{x}}_m$ be the corrupted data. If $\alpha \le \frac{1}{4}$, and the algorithm parameter σ is chosen such that*

$$\left\| \frac{1}{m} \sum_{i=1}^m (\mathbf{x}_i - \boldsymbol{\mu}_{\mathcal{S}})(\mathbf{x} - \boldsymbol{\mu}_{\mathcal{S}})^\top \right\|_2 \le \sigma^2, \tag{4.61}$$

then the output of the iterative filtering algorithm satisfies $\|\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}_{\mathcal{S}}\|_2 \le \mathcal{O}(\sigma\sqrt{\alpha})$.

By triangle inequality, we have

$$\|\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_2 \le \|\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}_{\mathcal{S}}\|_2 + \|\boldsymbol{\mu}_{\mathcal{S}} - \boldsymbol{\mu}\|_2, \tag{4.62}$$

and

$$\begin{aligned}\left\|\frac{1}{m}\sum_{i=1}^{m}(\mathbf{x}_i-\boldsymbol{\mu}_{\mathcal{S}})(\mathbf{x}-\boldsymbol{\mu}_{\mathcal{S}})^\top\right\|_2 =&\frac{1}{m}\left\|([\mathbf{x}_1,\cdots,\mathbf{x}_m]-\boldsymbol{\mu}_{\mathcal{S}}\mathbf{1}^\top)([\mathbf{x}_1,\cdots,\mathbf{x}_m]-\boldsymbol{\mu}_{\mathcal{S}}\mathbf{1}^\top)^\top\right\|_2 \\ =&\frac{1}{m}\left\|[\mathbf{x}_1,\cdots,\mathbf{x}_m]-\boldsymbol{\mu}_{\mathcal{S}}\mathbf{1}^\top\right\|_2^2 \\ \le&\frac{1}{m}\Big(\|[\mathbf{x}_1,\cdots,\mathbf{x}_m]-\boldsymbol{\mu}\mathbf{1}^\top\|_2+\sqrt{m}\|\boldsymbol{\mu}-\boldsymbol{\mu}_{\mathcal{S}}\|_2\Big)^2, \quad (4.63)\end{aligned}$$

where $\mathbf{1}$ denotes the all-one vector.[5] By choosing

$$\sigma=\Theta(\frac{1}{\sqrt{m}}\|[\mathbf{x}_1,\cdots,\mathbf{x}_m]-\boldsymbol{\mu}\mathbf{1}^\top\|_2+\|\boldsymbol{\mu}-\boldsymbol{\mu}_{\mathcal{S}}\|_2)$$

in Lemma 4.5 and combining with the bounds (4.62) and (4.63), we obtain that

$$\|\widehat{\boldsymbol{\mu}}-\boldsymbol{\mu}\|_2\lesssim\frac{\sqrt{\alpha}}{\sqrt{m}}\|[\mathbf{x}_1,\cdots,\mathbf{x}_m]-\boldsymbol{\mu}\mathbf{1}^\top\|_2+\|\boldsymbol{\mu}-\boldsymbol{\mu}_{\mathcal{S}}\|_2. \quad (4.64)$$

With the above bound in hand, we now turn to the robust gradient estimation problem, where the data points are drawn i.i.d. from some unknown distribution. Let $\widehat{\mathbf{g}}(\mathbf{w}) := \mathsf{filter}\{\widehat{\mathbf{g}}_i(\mathbf{w})\}_{i=1}^m$, where filter represents the iterative filtering algorithm. In light of (4.64), we know that in order to bound the gradient estimation error $\sup_{\mathbf{w}\in\mathcal{W}}\|\widehat{\mathbf{g}}(\mathbf{w})-\nabla F(\mathbf{w})\|_2$, it suffices to bound the quantities

$$\sup_{\mathbf{w}\in\mathcal{W}}\|[\nabla F_1(\mathbf{w}),\cdots,\nabla F_m(\mathbf{w})]-\nabla F(\mathbf{w})\mathbf{1}^\top\|_2$$

and

$$\sup_{\mathbf{w}\in\mathcal{W}}\|\frac{1}{m}\sum_{i=1}^{m}\nabla F_i(\mathbf{w})-\nabla F(\mathbf{w})\|_2.$$

Here, we recall that $\nabla F_i(\mathbf{w})$ is the true gradient of the empirical loss function on the i-th machine, and $\widehat{\mathbf{g}}_i(\mathbf{w})$ is the (possibly) corrupted gradient.

We first bound $\sup_{\mathbf{w}\in\mathcal{W}}\|\frac{1}{m}\sum_{i=1}^m\nabla F_i(\mathbf{w})-\nabla F(\mathbf{w})\|_2$. Note that we have

$$\frac{1}{m}\sum_{i=1}^{m}\nabla F_i(\mathbf{w})=\frac{1}{nm}\sum_{i=1}^{m}\sum_{j=1}^{n}\nabla f(\mathbf{w};\mathbf{z}_{i,j}).$$

Using the same method as in the proof of Lemma 6 in [41], we can show that for each fixed $\mathbf{w}$, with probability at least $1-\delta$,

$$\|\frac{1}{m}\sum_{i=1}^{m}\nabla F_i(\mathbf{w})-\nabla F(\mathbf{w})\|_2\le\frac{2\sqrt{2}\zeta}{\sqrt{nm}}\sqrt{d\log 6+\log\left(\frac{1}{\delta}\right)}.$$

[5]We note that similar derivation also appears in [174].

For some $\delta_0 > 0$ to be chosen later, let $\mathcal{W}_{\delta_0} = \{\mathbf{w}^1, \mathbf{w}^2, \ldots, \mathbf{w}^{N_{\delta_0}}\}$ be a finite subset of $\mathcal{W}$ such that for any $\mathbf{w} \in \mathcal{W}$, there exists some $\mathbf{w}^\ell \in \mathcal{W}_{\delta_0}$ such that $\|\mathbf{w}^\ell - \mathbf{w}\|_2 \le \delta_0$. Standard ϵ-net results from [182] ensure that $N_{\delta_0} \le (1 + \frac{D}{\delta_0})^d$. Then, by the union bound, we have with probability $1-\delta$, for all $\mathbf{w}^\ell \in \mathcal{W}_{\delta_0}$,

$$\|\frac{1}{m}\sum_{i=1}^m \nabla F_i(\mathbf{w}^\ell) - \nabla F(\mathbf{w}^\ell)\|_2 \le \frac{2\sqrt{2}\zeta}{\sqrt{nm}}\sqrt{d\log 6 + \log\left(\frac{N_{\delta_0}}{\delta}\right)}. \tag{4.65}$$

When (4.65) holds, by the smoothness of $f(\cdot;\mathbf{z})$ we know that for all $\mathbf{w} \in \mathcal{W}$,

$$\|\frac{1}{m}\sum_{i=1}^m \nabla F_i(\mathbf{w}) - \nabla F(\mathbf{w})\|_2 \le \frac{2\sqrt{2}\zeta}{\sqrt{nm}}\sqrt{d\log 6 + \log\left(\frac{N_{\delta_0}}{\delta}\right)} + 2L\delta_0.$$

By choosing $\delta_0 = \frac{1}{nmL}$ and $\delta = \frac{1}{(1+mnDL)^d}$, we obtain that with probability at least $1 - \frac{1}{(1+mnDL)^d}$, for all $\mathbf{w} \in \mathcal{W}$,

$$\|\frac{1}{m}\sum_{i=1}^m \nabla F_i(\mathbf{w}) - \nabla F(\mathbf{w})\|_2 \lesssim \frac{\zeta}{\sqrt{nm}}\sqrt{d\log(1+nmDL)}. \tag{4.66}$$

We next bound $\sup_{\mathbf{w}\in\mathcal{W}} \|[\nabla F_1(\mathbf{w}), \cdots, \nabla F_m(\mathbf{w})] - \nabla F(\mathbf{w})\mathbf{1}^\top\|_2$. We note that when the gradients are sub-Gaussian distributed, similar results for the centralized setting have been established in [35]. One can check that for every i, $\nabla F_i(\mathbf{w}) - \nabla F(\mathbf{w})$ is $\frac{\zeta}{\sqrt{n}}$-sub-Gaussian. Define $\mathbf{G}(\mathbf{w}) := [\nabla F_1(\mathbf{w}), \cdots, \nabla F_m(\mathbf{w})] - \nabla F(\mathbf{w})\mathbf{1}^\top$. Using a standard concentration inequality for the norm of a matrix with independent sub-Gaussian columns [182], we obtain that for each fixed $\mathbf{w}$, with probability at least $1-\delta$,

$$\|\frac{1}{m}\mathbf{G}(\mathbf{w})\mathbf{G}(\mathbf{w})^\top - \frac{1}{n}\mathbf{\Sigma}(\mathbf{w})\|_2 \lesssim \frac{\zeta^2}{n}\left(\sqrt{\frac{d}{m}} + \frac{d}{m} + \frac{1}{m}\log\left(\frac{1}{\delta}\right) + \sqrt{\frac{1}{m}\log\left(\frac{1}{\delta}\right)}\right),$$

which implies that

$$\frac{1}{\sqrt{m}}\|\mathbf{G}(\mathbf{w})\|_2 \lesssim \frac{\sigma}{\sqrt{n}} + \frac{\zeta}{\sqrt{n}}\left(\sqrt{\frac{d}{m}} + \frac{d}{m} + \frac{1}{m}\log\left(\frac{1}{\delta}\right) + \sqrt{\frac{1}{m}\log\left(\frac{1}{\delta}\right)}\right)^{1/2}.$$

Recall the δ_0-net $\mathcal{W}_{\delta_0} = \{\mathbf{w}^1, \mathbf{w}^2, \ldots, \mathbf{w}^{N_{\delta_0}}\}$ as defined above. Then, we have with probability at least $1-\delta$, for all $\mathbf{w}^\ell \in \mathcal{W}_{\delta_0}$

$$\frac{1}{\sqrt{m}}\|\mathbf{G}(\mathbf{w}^\ell)\|_2 \lesssim \frac{\sigma}{\sqrt{n}} + \frac{\zeta}{\sqrt{n}}\left(\sqrt{\frac{d}{m}} + \frac{d}{m} + \frac{1}{m}\log\left(\frac{N_{\delta_0}}{\delta}\right) + \sqrt{\frac{1}{m}\log\left(\frac{N_{\delta_0}}{\delta}\right)}\right)^{1/2}. \tag{4.67}$$

For each $\mathbf{w}$ with $\|\mathbf{w}^\ell - \mathbf{w}\|_2 \le \delta_0$, we have

$$\begin{aligned}\|\mathbf{G}(\mathbf{w}^\ell) - \mathbf{G}(\mathbf{w})\|_2 \le & \|\mathbf{G}(\mathbf{w}^\ell) - \mathbf{G}(\mathbf{w})\|_F \\ \le & \left(\sum_{i=1}^m \|(\nabla F_i(\mathbf{w}^\ell) - \nabla F(\mathbf{w}^\ell)) - (\nabla F_i(\mathbf{w}) - \nabla F(\mathbf{w}))\|_2^2 \right)^{1/2} \\ \le & 2L\delta_0\sqrt{m}.\end{aligned}$$

This implies that when the bound (4.67) holds, we have for all $\mathbf{w} \in \mathcal{W}$,

$$\frac{1}{\sqrt{m}}\|\mathbf{G}(\mathbf{w})\|_2 \lesssim \frac{\sigma}{\sqrt{n}} + \frac{\zeta}{\sqrt{n}} \left(\sqrt{\frac{d}{m}} + \frac{d}{m} + \frac{1}{m}\log\left(\frac{N_{\delta_0}}{\delta}\right) + \sqrt{\frac{1}{m}\log\left(\frac{N_{\delta_0}}{\delta}\right)} \right)^{1/2} + 2L\delta_0. \tag{4.68}$$

Choose $\delta_0 = \frac{1}{nmL}$, in which case the last term above is a high order term. In this case, choosing $\delta = \frac{1}{(1+mnDL)^d}$, we have with probability at least $1 - \frac{1}{(1+mnDL)^d}$, for all $\mathbf{w} \in \mathcal{W}$,

$$\begin{aligned}\frac{1}{\sqrt{m}}\|\mathbf{G}(\mathbf{w})\|_2 \lesssim & \frac{\sigma}{\sqrt{n}} + \frac{\zeta}{\sqrt{n}} \left(\left(\frac{d}{m} + \sqrt{\frac{d}{m}}\right) \log(1 + nmDL) \right)^{1/2} \\ \lesssim & \frac{\sigma}{\sqrt{n}} + \frac{\zeta}{\sqrt{n}} \left(1 + \sqrt{\frac{d}{m}} \right) \sqrt{\log(1 + nmDL)}.\end{aligned} \tag{4.69}$$

Combining the bounds (4.64), (4.66), and (4.69), we obtain that with probability at least $1 - \frac{2}{(1+mnDL)^d}$,

$$\sup_{\mathbf{w}\in\mathcal{W}} \|\widehat{\mathbf{g}}(\mathbf{w}) - \nabla F(\mathbf{w})\|_2 \lesssim \left((\sigma + \zeta)\sqrt{\frac{\alpha}{n}} + \zeta\sqrt{\frac{d}{nm}} \right) \sqrt{\log(1 + nmDL)},$$

which completes the proof.

Median and Trimmed Mean

In this section, we present the error bounds of median and trimmed mean operations in the Byzantine setting in Chapter 3 for completeness.

Assumption 4.4. *For any $\mathbf{z} \in \mathcal{Z}$, the k-th partial derivative $\partial_k f(\cdot;\mathbf{z})$ is L_k-Lipschitz for each $k \in [d]$. Let $\widehat{L} := (\sum_{k=1}^d L_k^2)^{1/2}$.*

For the median-based algorithm, one needs to use the notion of the *absolute skewness* of a one-dimensional random variable X, defined as $S(X) := \mathbb{E}[|X - \mathbb{E}[X]|^3]/\mathrm{Var}(X)^{3/2}$. Define the following upper bounds on the standard deviation and absolute skewness of the gradients:

$$v := \sup_{\mathbf{w}\in\mathcal{W}} \left(\mathbb{E}[\|\nabla f(\mathbf{w};\mathbf{z}) - \nabla F(\mathbf{w})\|_2^2]\right)^{1/2}, \quad s := \sup_{\mathbf{w}\in\mathcal{W}} \max_{k\in[d]} S\big(\partial f_k(\mathbf{w};\mathbf{z})\big).$$

Then one has the following guarantee for the median-based algorithm.

Theorem 4.7 (median)**.** *Suppose that Assumption 4.4 holds. Assume that*

$$\alpha + \left(\frac{d\log(1+nmD\widehat{L})}{m(1-\alpha)}\right)^{1/2} + c_1\frac{s}{\sqrt{n}} \le \frac{1}{2} - c_2$$

for some constant $c_1, c_2 > 0$. Then, with probability $1-o(1)$, GradAGG $\equiv$ *med provides a $\Delta_{\textsf{med}}$-inexact gradient oracle with*

$$\Delta_{\textsf{med}} \le \frac{c_3}{\sqrt{n}}v\big(\alpha + (\frac{d\log(nmD\widehat{L})}{m})^{1/2} + \frac{s}{\sqrt{n}}\big) + \mathcal{O}(\frac{1}{nm}),$$

where c_3 is an absolute constant.

Therefore, the median operation provides a $\widetilde{\mathcal{O}}(v(\frac{\alpha}{\sqrt{n}} + \sqrt{\frac{d}{nm}} + \frac{s}{n}))$-inexact gradient oracle. If each partial derivative is of size $\mathcal{O}(1)$, the quantity v is of the order $\mathcal{O}(\sqrt{d})$ and thus one has $\Delta_{\mathsf{med}} \lesssim \frac{\alpha\sqrt{d}}{\sqrt{n}} + \frac{d}{\sqrt{nm}} + \frac{\sqrt{d}}{n}$.

For the trimmed mean algorithm, one needs to assume that the gradients of the loss functions are sub-exponential.

Assumption 4.5. *For any $\mathbf{w} \in \mathcal{W}$, $\nabla f(\mathbf{w};\mathbf{z})$ is ξ-sub-exponential.*

In this setting, there is the following guarantee.

Theorem 4.8 (trimmed mean)**.** *Suppose that Assumptions 4.4 and 4.5 hold. Choose $\beta = c_4\alpha \le \frac{1}{2} - c_5$ with some constant $c_4 \ge 1$, $c_5 > 0$. Then, with probability $1-o(1)$,* GradAGG $\equiv$ *trmean$_\beta$ provides a $\Delta_{\textsf{tm}}$-inexact gradient oracle with*

$$\Delta_{\textsf{tm}} \le c_6\xi d\Big(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}}\Big)\sqrt{\log(nmD\widehat{L})},$$

where c_6 is an absolute constant.

Therefore, the trimmed mean operation provides a $\widetilde{\mathcal{O}}(\xi d(\frac{\alpha}{\sqrt{n}} + \frac{1}{\sqrt{nm}}))$-inexact gradient oracle.

Lower Bound for First-Order Guarantee

In this section we prove Observation 4.2. We consider the simple mean estimation problem with random vector $\mathbf{z}$ drawn from a distribution $\mathcal{D}$ with mean $\boldsymbol{\mu}$. The loss function associated with $\mathbf{z}$ is $f(\mathbf{w};\mathbf{z}) = \frac{1}{2}\|\mathbf{w}-\mathbf{z}\|_2^2$. The population loss is $F(\mathbf{w}) = \frac{1}{2}(\|\mathbf{w}\|_2^2 - 2\boldsymbol{\mu}^\top\mathbf{w} + \mathbb{E}[\|\mathbf{z}\|_2^2])$, and $\nabla F(\mathbf{w}) = \mathbf{w} - \boldsymbol{\mu}$. We first provide a lower bound for distributed mean estimation in the Byzantine setting, which is proved in Chapter 3.

Lemma 4.6. *Suppose that* $\mathbf{z}$ *is Gaussian distributed with mean* $\boldsymbol{\mu}$ *and covariance* $\sigma^2\mathbf{I}$*. Then, any algorithm that outputs an estimate* $\widetilde{\mathbf{w}}$ *of* $\boldsymbol{\mu}$ *has a constant probability such that*

$$\|\widetilde{\mathbf{w}} - \boldsymbol{\mu}\|_2 = \Omega(\frac{\alpha}{\sqrt{n}} + \sqrt{\frac{d}{nm}}).$$

Since $\nabla F(\widetilde{\mathbf{w}}) = \widetilde{\mathbf{w}} - \boldsymbol{\mu}$, the above bound directly implies the lower bound on $\|\nabla F(\widetilde{\mathbf{w}})\|_2$ in Observation 4.2.

Chapter 5

Rademacher Complexity for Adversarially Robust Generalization

Many machine learning models are vulnerable to adversarial attacks; for example, adding adversarial perturbations that are imperceptible to humans can often make machine learning models produce wrong predictions with high confidence. Moreover, although we may obtain robust models on the training dataset via adversarial training, in some problems the learned models cannot generalize well to the test data. In this chapter, we focus on ℓ_∞ attacks, and study the adversarially robust generalization problem through the lens of Rademacher complexity. For binary linear classifiers, we prove tight bounds for the adversarial Rademacher complexity, and show that the adversarial Rademacher complexity is never smaller than its natural counterpart, and it has an unavoidable dimension dependence, unless the weight vector has bounded ℓ_1 norm. The results also extend to multi-class linear classifiers. For (nonlinear) neural networks, we show that the dimension dependence in the adversarial Rademacher complexity also exists. We further consider a surrogate adversarial loss for one-hidden layer ReLU network and prove margin bounds for this setting. Our results indicate that having ℓ_1 norm constraints on the weight matrices might be a potential way to improve generalization in the adversarial setting. We demonstrate experimental results that validate our theoretical findings.

5.1 Introduction

In recent years, many modern machine learning models, in particular, deep neural networks, have achieved success in tasks such as image classification [84], speech recognition [81], machine translation [13], game playing [167], etc. However, although these models achieve the state-of-the-art performance in many standard benchmarks or competitions, it has been observed that by adversarially adding some perturbation to the input of the model (images, audio signals), the machine learning models can make wrong predictions with high confidence. These adversarial inputs are often called the *adversarial examples*. Typical methods of generating adversarial examples include adding small perturbations that are imperceptible to humans [177], changing surrounding areas of the main objects in images [76], and even simple rotation and translation [63]. This phenomenon was first discovered by Szegedy et al. [177] in image classification problems, and similar phenomena have been observed in other areas [31, 110]. Adversarial examples bring serious challenges in many security-critical applications, such as medical diagnosis and autonomous driving—the existence of these examples shows that many state-of-the-art machine learning models are actually unreliable in the presence of adversarial attacks.

Since the discovery of adversarial examples, there has been a race between designing robust models that can defend against adversarial attacks and designing attack algorithms that can generate adversarial examples and fool the machine learning models [79, 82, 32, 30]. As of now, it seems that the attackers are winning this game. For example, a recent work shows that many of the defense algorithms fail when the attacker uses a carefully designed gradient-based method [11]. Meanwhile, *adversarial training* [88, 163, 132] seems to be the most effective defense method. Adversarial training takes a robust optimization [18] perspective to the problem, and the basic idea is to minimize some *adversarial loss* over the training data. We elaborate below.

Suppose that data points $(\mathbf{x}, y)$ are drawn according to some unknown distribution $\mathcal{D}$ over the feature-label space $\mathcal{X} \times \mathcal{Y}$, and $\mathcal{X} \subseteq \mathbb{R}^d$. Let $\mathcal{F}$ be a hypothesis class (e.g., a class of neural networks with a particular architecture), and $\ell(f(\mathbf{x}), y)$ be the loss associated with $f \in \mathcal{F}$. Consider the ℓ_∞ white-box adversarial attack where an adversary is allowed to observe the trained model and choose some $\mathbf{x}'$ such that $\|\mathbf{x}' - \mathbf{x}\|_\infty \leq \epsilon$ and $\ell(f(\mathbf{x}'), y)$ is maximized. Therefore, to better defend against adversarial attacks, during training, the learner should aim to solve the empirical adversarial risk minimization problem

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \max_{\|\mathbf{x}_i' - \mathbf{x}_i\|_\infty \leq \epsilon} \ell(f(\mathbf{x}_i'), y_i), \tag{5.1}$$

where $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ are n i.i.d. training examples drawn according to $\mathcal{D}$. This minimax formulation raises many interesting theoretical and practical questions. For example, we need to understand how to efficiently solve the optimization problem in (5.1), and in addition, we need to characterize the generalization property of the adversarial risk, i.e., the gap between the empirical adversarial risk in (5.1) and the population adversarial risk $\mathbb{E}_{(\mathbf{x},y)\sim D}[\max_{\|\mathbf{x}'-\mathbf{x}\|_\infty \leq \epsilon} \ell(f(\mathbf{x}'), y)]$. In fact, for deep neural networks, both questions are still wide open. In particular, for the generalization problem, it has been observed that even if we can minimize the adversarial training error, the adversarial test error can still be large. For example, for a Resnet [84] model on CIFAR10, using the PGD adversarial training algorithm by Madry et al. [132], one can achieve about 96% adversarial training accuracy, but the adversarial test accuracy is only 47%. This generalization gap is significantly larger than that in the natural setting (without adversarial attacks), and thus it has become increasingly important to better understand the generalization behavior of machine learning models in the adversarial setting.

In this chapter, we focus on the adversarially robust generalization property and make a first step towards deeper understanding of this problem. We focus on ℓ_∞ adversarial attacks and analyze generalization through the lens of Rademacher complexity. We study both linear classifiers and nonlinear feedforward neural networks, and both binary and multi-class classification problems. Intuitively, our results provide rigorous evidence showing that adversarially robust learning is more difficult than natural learning, illustrated in Figure 5.1. [1]

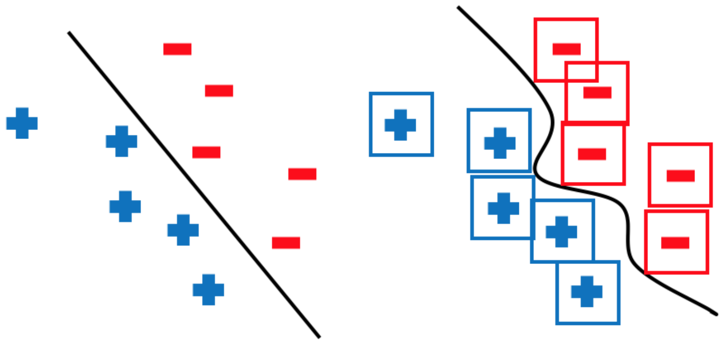


Figure 5.1: Adversarially robust learning is more difficult than natural learning.

We summarize our contributions as follows.

Our Contributions

- For binary linear classifiers, we prove tight upper and lower bounds for the adversarial Rademacher complexity. We show that the adversarial Rademacher complexity is never smaller than its counterpart in the natural setting, which provides theoretical evidence for the empirical observation that adversarially robust generalization can be hard. We also show that under an ℓ_∞ adversarial attack, when the weight vector of the linear classifier has bounded ℓ_p norm ($p \geq 1$), a polynomial dimension dependence in the adversarial Rademacher complexity is unavoidable, unless $p = 1$. For multi-class linear classifiers, we prove margin bounds in the adversarial setting. Similar to binary classifiers, the margin bound also exhibits polynomial dimension dependence when the weight vector for each class has bounded ℓ_p norm ($p > 1$).

- For neural networks, we show that in contrast to the margin bounds derived by Bartlett, Foster, and Telgarsky [16] and Golowich, Rakhlin, and Shamir [78] which depend only on the norms of the weight matrices and the data points, the adversarial Rademacher complexity has a lower bound with an explicit dimension dependence, which is also an effect of the ℓ_∞ attack. We further consider a *surrogate adversarial loss* for one hidden layer ReLU networks, based on the SDP relaxation proposed by Raghunathan,

[1]Similar figure appears in [132].

Steinhardt, and Liang [154]. We prove margin bounds using the surrogate loss and show that if the weight matrix of the first layer has bounded ℓ_1 norm, the margin bound does not have explicit dimension dependence. This suggests that in the adversarial setting, controlling the ℓ_1 norms of the weight matrices may be a way to improve generalization.

- We conduct experiments on linear classifiers and neural networks to validate our theoretical findings; more specifically, our experiments show that ℓ_1 regularization could reduce the adversarial generalization error, and the adversarial generalization gap increases as the dimension of the feature spaces increases.

Notation

We define the set $[N] := \{1, 2, \dots, N\}$. For two sets $\mathcal{A}$ and $\mathcal{B}$, we denote by $\mathcal{B}^{\mathcal{A}}$ the set of all functions from $\mathcal{A}$ to $\mathcal{B}$. We denote the indicator function of a event A as $\mathbb{1}(A)$. Unless otherwise stated, we denote vectors by boldface lowercase letters such as $\mathbf{w}$, and the elements in the vector are denoted by italics letters with subscripts, such as w_k. All-one vectors are denoted by $\mathbf{1}$. Matrices are denoted by boldface uppercase letters such as $\mathbf{W}$. For a matrix $\mathbf{W} \in \mathbb{R}^{d\times m}$ with columns $\mathbf{w}_i$, $i \in [m]$, the (p,q) matrix norm of $\mathbf{W}$ is defined as $\|\mathbf{W}\|_{p,q} = \|[\|\mathbf{w}_1\|_p, \|\mathbf{w}_2\|_p, \cdots, \|\mathbf{w}_m\|_p]\|_q$, and we may use the shorthand notation $\|\cdot\|_p \equiv \|\cdot\|_{p,p}$. We denote the spectral norm of matrices by $\|\cdot\|_\sigma$ and the Frobenius norm of matrices by $\|\cdot\|_F$ (i.e., $\|\cdot\|_F \equiv \|\cdot\|_2$). We use $\mathbb{B}^\infty_{\mathbf{x}}(\epsilon)$ to denote the ℓ_∞ ball centered at $\mathbf{x} \in \mathbb{R}^d$ with radius ϵ, i.e., $\mathbb{B}^\infty_{\mathbf{x}}(\epsilon) = \{\mathbf{x}' \in \mathbb{R}^d : \|\mathbf{x}' - \mathbf{x}\|_\infty \le \epsilon\}$.

Organization

The rest of this chapter is organized as follows: in Section 5.2, we discuss related work; in Section 5.3, we describe the formal problem setup; we present our main results for linear classifiers and neural networks in Sections 5.4 and 5.5, respectively. We demonstrate our experimental results in Section 5.6, make conclusions in Section 5.7, and provide the proofs in Section 5.8.

5.2 Related Work

During the preparation of the initial draft of this chapter, we become aware of another independent and concurrent work by Khim and Loh [101], which studies a similar problem. In this section, we first compare our work with Khim and Loh [101] and then discuss other related work. We make the comparison in the following aspects.

- For binary classification problems, the adversarial Rademacher complexity *upper bound* by Khim and Loh [101] is similar to ours. However, we show an adversarial Rademacher complexity *lower bound* that *matches* the upper bound. Our lower bound shows that the adversarial Rademacher complexity is never smaller than that in the natural setting, indicating the hardness of adversarially robust generalization. As mentioned, although our lower bound is for Rademacher complexity rather than generalization, Rademacher complexity is a tight bound for the rate of uniform convergence of a loss function class [104] and thus in many settings can be a tight bound for generalization. In addition, we provide a lower bound for the adversarial Rademacher complexity for neural networks. These lower bounds do not appear in the work by Khim and Loh [101].

- We discuss the generalization bounds for the multi-class setting, whereas Khim and Loh [101] focus only on binary classification.
- Both our work and Khim and Loh [101] prove adversarial generalization bound using surrogate adversarial loss (upper bound for the actual adversarial loss). Khim and Loh [101] use a method called *tree transform* whereas we use the SDP relaxation proposed by [154]. These two approaches are based on different ideas and thus we believe that they are not directly comparable.

We proceed to discuss other related work.

Adversarially robust generalization As discussed in Section 5.1, it has been observed by Madry et al. [132] that there might be a significant generalization gap when training deep neural networks in the adversarial setting. This generalization problem has been further studied by Schmidt et al. [162], who show that to correctly classify two separated d-dimensional spherical Gaussian distributions, in the natural setting one only needs $\mathcal{O}(1)$ training data, but in the adversarial setting one needs $\Theta(\sqrt{d})$ data. Getting distribution agnostic generalization bounds (also known as the PAC-learning framework) for the adversarial setting is proposed as an open problem by Schmidt et al. [162]. In a subsequent work, Cullina, Bhagoji, and Mittal [47] study PAC-learning guarantees for binary linear classifiers in the adversarial setting via VC-dimension, and show that the VC-dimension does not increase in the adversarial setting. This result does not provide explanation to the empirical observation that adversarially robust generalization may be hard. In fact, although VC-dimension and Rademacher complexity can both provide valid generalization bounds, VC-dimension usually depends on the number of parameters in the model while Rademacher complexity usually depends on the norms of the weight matrices and data points, and can often provide tighter generalization bounds [15]. Suggala et al. [175] discuss a similar notion of adversarial risk but do not prove explicit generalization bounds. Attias, Kontorovich, and Mansour [12] prove adversarial generalization bounds in a setting where the number of potential adversarial perturbations is finite, which is a weaker notion than the ℓ_∞ attack that we consider. Sinha, Namkoong, and Duchi [168] analyze the convergence and generalization of an adversarial training algorithm under the notion of distributional robustness. Farnia, Zhang, and Tse [65] study the generalization problem when the attack algorithm of the adversary is provided to the learner, which is also a weaker notion than our problem. In earlier work, robust optimization has been studied in Lasso [193] and SVM [194] problems. Xu and Mannor [195] make the connection between algorithmic robustness and generalization property in the natural setting, whereas our work focus on generalization in the adversarial setting.

Provable defense against adversarial attacks Besides generalization property, another recent line of work aim to design provable defense against adversarial attacks. Two examples of provable defense are SDP relaxation [154, 155] and LP relaxation [105, 190]. The idea of these methods is to construct *upper bounds* of the adversarial risk that can be efficiently evaluated and optimized. The analyses of these algorithms usually focus on minimizing training error and do not have generalization guarantee; in contrast, we focus on generalization property in this chapter.

Other theoretical analysis of adversarial examples A few other lines of work have conducted theoretical analysis of adversarial examples. Wang, Jha, and Chaudhuri [187] analyze the adversarial robustness of nearest neighbors estimator. Papernot et al. [149] try to demonstrate the unavoidable trade-offs between accuracy in the natural setting and the resilience to adversarial attacks, and this trade-off is further studied by Tsipras et al. [180] through some constructive examples of distributions. Fawzi, Moosavi-Dezfooli, and Frossard [67] analyze adversarial robustness of fixed classifiers, in contrast to our generalization analysis. Fawzi, Fawzi, and Fawzi [66] construct examples of distributions with large latent variable space such that adversarially robust classifiers do not exist; here we argue that these examples may not explain the fact that adversarially perturbed images can usually be recognized by humans. Bubeck, Price, and Razenshteyn [28] try to explain the hardness of learning an adversarially robust model from the computational constraints under the statistical query model. Another recent line of work explains the existence of adversarial examples via high dimensional geometry and concentration of measure [77, 60, 133]. These works provide examples where adversarial examples provably exist as long as the test error of a classifier is non-zero.

Generalization of neural networks Generalization of neural networks has been an important topic, even in the natural setting where there is no adversary. The key challenge is to understand why deep neural networks can generalize to unseen data despite the high capacity of the model class. The problem has received attention since the early stage of neural network study [15, 9]. Recently, understanding generalization of deep nets is raised as an open problem since traditional techniques such as VC-dimension, Rademacher complexity, and algorithmic stability are observed to produce vacuous generalization bounds [203]. Progress has been made more recently. In particular, it has been shown that when properly normalized by the margin, using Rademacher complexity or PAC-Bayesian analysis, one can obtain generalization bounds that tend to match the experimental results [16, 146, 10, 78]. In addition, in this chapter, we show that when the weight vectors or matrices have bounded

ℓ_1 norm, there is no dimension dependence in the adversarial generalization bound. This result is consistent and related to several previous works [121, 15, 137, 207].

5.3 Problem Setup

We start with the general statistical learning framework. Let $\mathcal{X}$ and $\mathcal{Y}$ be the feature and label spaces, respectively, and suppose that there is an unknown distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$. In this chapter, we assume that the feature space is a subset of the d dimensional Euclidean space, i.e., $\mathcal{X} \subseteq \mathbb{R}^d$. Let $\mathcal{F} \subseteq \mathcal{V}^{\mathcal{X}}$ be the hypothesis class that we use to make predictions, where $\mathcal{V}$ is another space that might be different from $\mathcal{Y}$. Let $\ell : \mathcal{V} \times \mathcal{Y} \to [0, B]$ be the loss function. Throughout this chapter we assume that ℓ is bounded, i.e., B is a positive constant. In addition, we introduce the function class $\ell_{\mathcal{F}} \subseteq [0, B]^{\mathcal{X} \times \mathcal{Y}}$ by composing the functions in $\mathcal{F}$ with $\ell(\cdot, y)$, i.e., $\ell_{\mathcal{F}} := \{(\mathbf{x}, y) \mapsto \ell(f(\mathbf{x}), y) : f \in \mathcal{F}\}$. The goal of the learning problem is to find $f \in \mathcal{F}$ such that the *population risk* $R(f) := \mathbb{E}_{(\mathbf{x},y)\in\mathcal{D}}[\ell(f(\mathbf{x}), y)]$ is minimized.

We consider the supervised learning setting where one has access to n i.i.d. training examples drawn according to $\mathcal{D}$, denoted by $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n)$. A learning algorithm maps the n training examples to a hypothesis $f \in \mathcal{F}$. In this chapter, we are interested in the gap between the *empirical risk* $R_n(f) := \frac{1}{n}\sum_{i=1}^n \ell(f(\mathbf{x}_i), y_i)$ and the population risk $R(f)$, known as the generalization error.

Rademacher complexity [17] is one of the classic measures of generalization error. Here, we present its formal definition. For any function class $\mathcal{H} \subseteq \mathbb{R}^{\mathcal{Z}}$, given a sample $\mathcal{S} = \{\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_n\}$ of size n, and *empirical Rademacher complexity* is defined as

$$\mathfrak{R}_{\mathcal{S}}(\mathcal{H}) := \frac{1}{n}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{h\in\mathcal{H}} \sum_{i=1}^n \sigma_i h(\mathbf{z}_i)\right],$$

where $\sigma_1, \dots, \sigma_n$ are i.i.d. Rademacher random variables with $\mathbb{P}\{\sigma_i = 1\} = \mathbb{P}\{\sigma_i = -1\} = \frac{1}{2}$. In our learning problem, denote the training sample by $\mathcal{S}$, i.e.,

$$\mathcal{S} := \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n)\}.$$

We then have the following theorem which connects the population and empirical risks via Rademacher complexity.

Theorem 5.1. *[17, 140] Suppose that the range of $\ell(f(\mathbf{x}), y)$ is $[0, B]$. Then, for any $\delta \in (0, 1)$, with probability at least $1 - \delta$, the following holds for all $f \in \mathcal{F}$,*

$$R(f) \le R_n(f) + 2B\mathfrak{R}_{\mathcal{S}}(\ell_{\mathcal{F}}) + 3B\sqrt{\frac{\log\frac{2}{\delta}}{2n}}.$$

As we can see, Rademacher complexity measures the rate that the empirical risk converges to the population risk *uniformly* across $\mathcal{F}$. In fact, according to the anti-symmetrization lower bound by Koltchinskii et al. [104], one can show that $\mathfrak{R}_{\mathcal{S}}(\ell_{\mathcal{F}}) \lesssim \sup_{f\in\mathcal{F}} R(f) - R_n(f) \lesssim \mathfrak{R}_{\mathcal{S}}(\ell_{\mathcal{F}})$. Therefore, Rademacher complexity is a tight bound for the rate of uniform convergence of a loss function class, and in many settings can be a tight bound for generalization error.

The above discussions can be extended to the adversarial setting. In this chapter, we focus on the ℓ_∞ adversarial attack. In this setting, the learning algorithm still has access to n i.i.d. uncorrupted training examples drawn according to $\mathcal{D}$. Once the learning procedure finishes, the output hypothesis f is revealed to an adversary. For any data point $(\mathbf{x}, y)$ drawn according to $\mathcal{D}$, the adversary is allowed to perturb $\mathbf{x}$ within some ℓ_∞ ball to maximize the loss. Our goal is to minimize the *adversarial population risk*, i.e.,

$$\widetilde{R}(f) := \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left[\max_{\mathbf{x}'\in\mathbb{B}_{\mathbf{x}}^{\infty}(\epsilon)} \ell(f(\mathbf{x}'), y)\right],$$

and to this end, a natural way is to conduct *adversarial training*—minimizing the adversarial empirical risk

$$\widetilde{R}_n(f) := \frac{1}{n}\sum_{i=1}^{n} \max_{\mathbf{x}_i'\in\mathbb{B}_{\mathbf{x}_i}^{\infty}(\epsilon)} \ell(f(\mathbf{x}_i'), y_i).$$

Let us define the adversarial loss $\widetilde{\ell}(f(\mathbf{x}), y) := \max_{\mathbb{B}_{\mathbf{x}}^{\infty}(\epsilon)} \ell(f(\mathbf{x}'), y)$ and the function class $\widetilde{\ell}_{\mathcal{F}} \subseteq [0, B]^{\mathcal{X}\times\mathcal{Y}}$ as $\widetilde{\ell}_{\mathcal{F}} := \{\widetilde{\ell}(f(\mathbf{x}), y) : f \in \mathcal{F}\}$. Since the range of $\widetilde{\ell}(f(\mathbf{x}), y)$ is still $[0, B]$, we have the following direct corollary of Theorem 5.1.

Corollary 5.1. *For any $\delta \in (0, 1)$, with probability at least $1 - \delta$, the following holds for all $f \in \mathcal{F}$,*

$$\widetilde{R}(f) \le \widetilde{R}_n(f) + 2B\mathfrak{R}_{\mathcal{S}}(\widetilde{\ell}_{\mathcal{F}}) + 3B\sqrt{\frac{\log\frac{2}{\delta}}{2n}}.$$

As we can see, the Rademacher complexity of the adversarial loss function class $\widetilde{\ell}_{\mathcal{F}}$, i.e., $\mathfrak{R}_{\mathcal{S}}(\widetilde{\ell}_{\mathcal{F}})$ is again the key quantity for the generalization ability of the learning problem. A natural problem of interest is to compare the Rademacher complexities in the natural and the adversarial settings, and obtain generalization bounds for the adversarial loss.

5.4 Linear Classifiers

Binary Classification

We start with binary linear classifiers. In this setting, we define $\mathcal{Y} = \{-1, +1\}$, and let the hypothesis class $\mathcal{F} \subseteq \mathbb{R}^{\mathcal{X}}$ be a set of linear functions of $\mathbf{x} \in \mathcal{X}$. More specifically, we define $f_{\mathbf{w}}(\mathbf{x}) := \langle \mathbf{w}, \mathbf{x}\rangle$, and consider prediction vector $\mathbf{w}$ with ℓ_p norm constraint ($p \ge 1$), i.e.,

$$\mathcal{F} = \{f_{\mathbf{w}}(\mathbf{x}) : \|\mathbf{w}\|_p \le W\}. \tag{5.2}$$

We predict the label with the sign of $f_{\mathbf{w}}(\mathbf{x})$; more specifically, we assume that the loss function $\ell(f_{\mathbf{w}}(\mathbf{x}), y)$ can be written as $\ell(f_{\mathbf{w}}(\mathbf{x}), y) \equiv \phi(y\langle \mathbf{w}, \mathbf{x}\rangle)$, where $\phi : \mathbb{R} \to [0, B]$ is monotonically nonincreasing and L_ϕ-Lipschitz. In fact, if $\phi(0) \geq 1$, we can obtain a bound on the classification error according to Theorem 5.1. That is, with probability at least $1 - \delta$, for all $f_{\mathbf{w}} \in \mathcal{F}$,

$$\mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}}\{\mathrm{sgn}(f_{\mathbf{w}}(\mathbf{x})) \neq y\} \leq \frac{1}{n}\sum_{i=1}^{n} \ell(f_{\mathbf{w}}(\mathbf{x}_i), y_i) + 2B\mathfrak{R}_{\mathcal{S}}(\ell_{\mathcal{F}}) + 3B\sqrt{\frac{\log\frac{2}{\delta}}{2n}}.$$

In addition, recall that according to the Ledoux-Talagrand contraction inequality [116], we have $\mathfrak{R}_{\mathcal{S}}(\ell_{\mathcal{F}}) \leq L_\phi \mathfrak{R}_{\mathcal{S}}(\mathcal{F})$.

For the adversarial setting, we have

$$\widetilde{\ell}(f_{\mathbf{w}}(\mathbf{x}), y) = \max_{\mathbf{x}' \in \mathbb{B}^\infty_{\mathbf{x}}(\epsilon)} \ell(f_{\mathbf{w}}(\mathbf{x}'), y) = \phi(\min_{\mathbf{x}' \in \mathbb{B}^\infty_{\mathbf{x}}(\epsilon)} y\langle \mathbf{w}, \mathbf{x}'\rangle).$$

Let us define the following function class $\widetilde{\mathcal{F}} \subseteq \mathbb{R}^{\mathcal{X}\times\{\pm 1\}}$:

$$\widetilde{\mathcal{F}} = \left\{ \min_{\mathbf{x}' \in \mathbb{B}^\infty_{\mathbf{x}}(\epsilon)} y\langle \mathbf{w}, \mathbf{x}'\rangle : \|\mathbf{w}\|_p \leq W \right\}. \tag{5.3}$$

Again, we have $\mathfrak{R}_{\mathcal{S}}(\widetilde{\ell}_{\mathcal{F}}) \leq L_\phi \mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}})$. The first major contribution of our work is the following theorem, which provides a comparison between $\mathfrak{R}_{\mathcal{S}}(\mathcal{F})$ and $\mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}})$.

Theorem 5.2 (Main Result 1). *Let*

$$\begin{aligned}\mathcal{F} :=& \{f_{\mathbf{w}}(\mathbf{x}) : \|\mathbf{w}\|_p \leq W\}, \\ \widetilde{\mathcal{F}} :=& \{\min_{\mathbf{x}' \in \mathbb{B}^\infty_{\mathbf{x}}(\epsilon)} y\langle \mathbf{w}, \mathbf{x}'\rangle : \|\mathbf{w}\|_p \leq W\}.\end{aligned}$$

Suppose that $\frac{1}{p} + \frac{1}{q} = 1$. Then, there exists a universal constant $c \in (0,1)$ such that

$$\max\{\mathfrak{R}_{\mathcal{S}}(\mathcal{F}), c\epsilon W \frac{d^{\frac{1}{q}}}{\sqrt{n}}\} \leq \mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}}) \leq \mathfrak{R}_{\mathcal{S}}(\mathcal{F}) + \epsilon W \frac{d^{\frac{1}{q}}}{\sqrt{n}}.$$

We can see that the adversarial Rademacher complexity, i.e., $\mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}})$ is always at least as large as the Rademacher complexity in the natural setting. This implies that uniform convergence in the adversarial setting is at least as hard as that in the natural setting. In addition, since $\max\{a, b\} \geq \frac{1}{2}(a + b)$, we have

$$\frac{c}{2}\Big(\mathfrak{R}_{\mathcal{S}}(\mathcal{F}) + \epsilon W \frac{d^{\frac{1}{q}}}{\sqrt{n}}\Big) \leq \mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}}) \leq \mathfrak{R}_{\mathcal{S}}(\mathcal{F}) + \epsilon W \frac{d^{\frac{1}{q}}}{\sqrt{n}}.$$

Therefore, we have a tight bound for $\mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}})$ up to a constant factor. Further, if $p > 1$ the adversarial Rademacher complexity has an unavoidable polynomial dimension dependence, i.e., $\mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}})$ is always at least as large as $\mathcal{O}(\epsilon W \frac{d^{1/q}}{\sqrt{n}})$. On the other hand, one can easily show that in the natural setting, $\mathfrak{R}_{\mathcal{S}}(\mathcal{F}) = \frac{W}{n}\mathbb{E}_{\boldsymbol{\sigma}}[\| \sum_{i=1}^n \sigma_i \mathbf{x}_i \|_q]$, which implies that $\mathfrak{R}_{\mathcal{S}}(\mathcal{F})$ depends on the distribution of $\mathbf{x}_i$ and the norm constraint W, but does not have an explicit dimension dependence. This means that $\mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}})$ could be order-wise larger than $\mathfrak{R}_{\mathcal{S}}(\mathcal{F})$, depending on the distribution of the data. An interesting fact is that, if we have an ℓ_1 norm constraint on the prediction vector $\mathbf{w}$, we can avoid the dimension dependence in $\mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}})$.

Multi-class Classification

Margin Bounds for Multi-class Classification

We proceed to study multi-class linear classifiers. We start with the standard margin bound framework for multi-class classification. In K-class classification problems, we choose $\mathcal{Y} = [K]$, and the functions in the hypothesis class $\mathcal{F}$ map $\mathcal{X}$ to $\mathbb{R}^K$, i.e., $\mathcal{F} \subseteq (\mathbb{R}^K)^{\mathcal{X}}$. Intuitively, the k-th coordinate of $f(\mathbf{x})$ is the score that f gives to the k-th class, and we make prediction by choosing the class with the highest score. We define the margin operator $M(\mathbf{z}, y) : \mathbb{R}^K \times [K] \to \mathbb{R}$ as $M(\mathbf{z}, y) = z_y - \max_{y' \neq y} z_{y'}$. For a training example $(\mathbf{x}, y)$, a hypothesis f makes a correct prediction if and only if $M(f(\mathbf{x}), y) > 0$. We also define function class $M_{\mathcal{F}} := \{(\mathbf{x}, y) \mapsto M(f(\mathbf{x}), y) : f \in \mathcal{F}\} \subseteq \mathbb{R}^{\mathcal{X} \times [K]}$. For multi-class classification problems, we consider a particular loss function $\ell(f(\mathbf{x}), y) = \phi_\gamma(M(f(\mathbf{x}), y))$, where $\gamma > 0$ and $\phi_\gamma : \mathbb{R} \to [0, 1]$ is the ramp loss:

$$\phi_\gamma(t) = \begin{cases} 1 & t \leq 0 \\ 1 - \frac{t}{\gamma} & 0 < t < \gamma \\ 0 & t \geq \gamma. \end{cases} \tag{5.4}$$

One can check that $\ell(f(\mathbf{x}), y)$ satisfies:

$$\mathbb{1}(y \neq \arg\max_{y' \in [K]} [f(\mathbf{x})]_{y'}) \leq \ell(f(\mathbf{x}), y) \leq \mathbb{1}([f(\mathbf{x})]_y \leq \gamma + \max_{y' \neq y} [f(\mathbf{x})]_{y'}). \tag{5.5}$$

Let $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n \in (\mathcal{X} \times [K])^n$ be the i.i.d. training examples, and define the function class $\ell_{\mathcal{F}} := \{(\mathbf{x}, y) \mapsto \phi_\gamma(M(f(\mathbf{x}), y)) : f \in \mathcal{F}\} \subseteq \mathbb{R}^{\mathcal{X} \times [K]}$. Since $\phi_\gamma(t) \in [0, 1]$ and $\phi_\gamma(\cdot)$ is $1/\gamma$-Lipschitz, by combining (5.5) with Theorem 5.1, we can obtain the following direct corollary as the generalization bound in the multi-class setting [140].

Corollary 5.2. *Consider the above multi-class classification setting. For any fixed $\gamma > 0$, we have with probability at least $1 - \delta$, for all $f \in \mathcal{F}$,*

$$\mathbb{P}_{(\mathbf{x}, y) \sim \mathcal{D}}\{y \neq \arg\max_{y' \in [K]} [f(\mathbf{x})]_{y'}\} \leq \frac{1}{n}\sum_{i=1}^n \mathbb{1}([f(\mathbf{x}_i)]_{y_i} \leq \gamma + \max_{y' \neq y} [f(\mathbf{x}_i)]_{y'}) + 2\mathfrak{R}_{\mathcal{S}}(\ell_{\mathcal{F}}) + 3\sqrt{\frac{\log \frac{2}{\delta}}{2n}}.$$

In the adversarial setting, the adversary tries to maximize the loss

$$\ell(f(\mathbf{x}), y) = \phi_\gamma(M(f(\mathbf{x}), y))$$

around an ℓ_∞ ball centered at $\mathbf{x}$. We have the adversarial loss

$$\widetilde{\ell}(f(\mathbf{x}), y) := \max_{\mathbf{x}' \in \mathbb{B}_{\mathbf{x}}^{\infty}(\epsilon)} \ell(f(\mathbf{x}'), y),$$

and the function class $\widetilde{\ell}_{\mathcal{F}} := \{(\mathbf{x}, y) \mapsto \widetilde{\ell}(f(\mathbf{x}), y) : f \in \mathcal{F}\} \subseteq \mathbb{R}^{\mathcal{X} \times [K]}$. Thus, we have the following generalization bound in the adversarial setting.

Corollary 5.3. *Consider the above adversarial multi-class classification setting. For any fixed $\gamma > 0$, we have with probability at least $1 - \delta$, for all $f \in \mathcal{F}$,*

$$\begin{aligned}&\mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{\exists\ \mathbf{x}' \in \mathbb{B}_{\mathbf{x}}^{\infty}(\epsilon)\ s.t.\ y \neq \arg\max_{y' \in [K]} [f(\mathbf{x}')]_{y'}\right\} \\ \leq & \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}(\exists\ \mathbf{x}_i' \in \mathbb{B}_{\mathbf{x}_i}^{\infty}(\epsilon)\ s.t.\ [f(\mathbf{x}_i')]_{y_i} \leq \gamma + \max_{y' \neq y} [f(\mathbf{x}_i')]_{y'}) + 2\mathfrak{R}_{\mathcal{S}}(\widetilde{\ell}_{\mathcal{F}}) + 3\sqrt{\frac{\log\frac{2}{\delta}}{2n}}.\end{aligned}$$

Multi-class Linear Classifiers

We now focus on multi-class linear classifiers. For linear classifiers, each function in the hypothesis class is linearly parametrized by a matrix $\mathbf{W} \in \mathbb{R}^{K \times d}$, i.e., $f(\mathbf{x}) \equiv f_{\mathbf{W}}(\mathbf{x}) = \mathbf{W}\mathbf{x}$. Let $\mathbf{w}_k \in \mathbb{R}^d$ be the k-th column of $\mathbf{W}^\top$, then we have $[f_{\mathbf{W}}(\mathbf{x})]_k = \langle \mathbf{w}_k, \mathbf{x} \rangle$. We assume that each $\mathbf{w}_k$ has ℓ_p norm ($p \geq 1$) upper bounded by W, which implies that $\mathcal{F} = \{f_{\mathbf{W}}(\mathbf{x}) : \|\mathbf{W}^\top\|_{p,\infty} \leq W\}$. In the natural setting, we have the following margin bound for linear classifiers as a corollary of the multi-class margin bounds by Kuznetsov, Mohri, and Syed [112] and Maximov and Reshetova [134].

Theorem 5.3. *Consider the multi-class linear classifiers in the above setting, and suppose that $\frac{1}{p} + \frac{1}{q} = 1$, $p, q \geq 1$. For any fixed $\gamma > 0$ and $W > 0$, we have with probability at least $1 - \delta$, for all $\mathbf{W}$ such that $\|\mathbf{W}^\top\|_{p,\infty} \leq W$,*

$$\begin{aligned}\mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{y \neq \arg\max_{y' \in [K]} \langle \mathbf{w}_{y'}, \mathbf{x} \rangle\right\} \leq & \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}(\langle \mathbf{w}_{y_i}, \mathbf{x}_i \rangle \leq \gamma + \max_{y' \neq y_i} \langle \mathbf{w}_{y'}, \mathbf{x}_i \rangle) \\ & + \frac{4KW}{\gamma n}\mathbb{E}_{\boldsymbol{\sigma}}\left[\|\sum_{i=1}^{n} \sigma_i \mathbf{x}_i\|_q\right] + 3\sqrt{\frac{\log\frac{2}{\delta}}{2n}}.\end{aligned}$$

In the adversarial setting, we have the following margin bound.

Theorem 5.4 (Main Result 2). *Consider the multi-class linear classifiers in the adversarial setting, and suppose that $\frac{1}{p} + \frac{1}{q} = 1$, $p, q \geq 1$. For any fixed $\gamma > 0$ and $W > 0$, we* *have with probability at least* $1-\delta$*, for all* $\mathbf{W}$ *such that* $\|\mathbf{W}^\top\|_{p,\infty} \le W$*,*

$$\begin{aligned}
&\mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{\exists\ \mathbf{x}' \in \mathbb{B}^\infty_{\mathbf{x}}(\epsilon)\ s.t.\ y \neq \arg\max_{y'\in[K]}\langle \mathbf{w}_{y'}, \mathbf{x}'\rangle\right\}\\
\le&\frac{1}{n}\sum_{i=1}^n \mathbb{1}(\langle \mathbf{w}_{y_i}, \mathbf{x}_i\rangle \le \gamma + \max_{y'\neq y_i}(\langle \mathbf{w}_{y'}, \mathbf{x}_i\rangle + \epsilon\|\mathbf{w}_{y'} - \mathbf{w}_{y_i}\|_1))\\
&+\frac{2WK}{\gamma}\left[\frac{\epsilon\sqrt{K}d^{\frac{1}{q}}}{\sqrt{n}} + \frac{1}{n}\sum_{y=1}^K \mathbb{E}_{\boldsymbol{\sigma}}\left[\|\sum_{i=1}^n \sigma_i \mathbf{x}_i \mathbb{1}(y_i = y)\|_q\right]\right] + 3\sqrt{\frac{\log\frac{2}{\delta}}{2n}}.
\end{aligned}$$

As we can see, similar to the binary classification problems, if $p > 1$, the margin bound in the adversarial setting has an explicit polynomial dependence on d, whereas in the natural setting, the margin bound does not have dimension dependence. This shows that, at least for the generalization upper bound that we obtain, the dimension dependence in the adversarial setting also exists in the multi-class classification problems.

5.5 Neural Networks

We proceed to consider feedforward neural networks with ReLU activation. Here, each function f in the hypothesis class $\mathcal{F}$ is parametrized by a sequence of matrices $\mathbf{W} = (\mathbf{W}_1, \mathbf{W}_2, \ldots, \mathbf{W}_L)$, i.e., $f \equiv f_{\mathbf{W}}$. Assume that $\mathbf{W}_h \in \mathbb{R}^{d_h \times d_{h-1}}$, and $\rho(\cdot)$ be the ReLU function, i.e., $\rho(t) = \max\{t, 0\}$ for $t \in \mathbb{R}$. For vectors, $\rho(\mathbf{x})$ is vector generated by applying $\rho(\cdot)$ on each coordinate of $\mathbf{x}$, i.e., $[\rho(\mathbf{x})]_i = \rho(x_i)$. We have[2]

$$f_{\mathbf{W}}(\mathbf{x}) = \mathbf{W}_L\rho(\mathbf{W}_{L-1}\rho(\cdots\rho(\mathbf{W}_1\mathbf{x})\cdots)).$$

For K-class classification, we have $d_L = K$, $f_{\mathbf{W}}(\mathbf{x}) : \mathbb{R}^d \to \mathbb{R}^K$, and $[f_{\mathbf{W}}(\mathbf{x})]_k$ is the score for the k-th class. In the special case of binary classification, as discussed in Section 5.4, we can have $\mathcal{Y} = \{+1, -1\}$, $d_L = 1$, and the loss function can be written as

$$\ell(f_{\mathbf{W}}(\mathbf{x}), y) = \phi(yf_{\mathbf{W}}(\mathbf{x})),$$

where $\phi : \mathbb{R} \to [0, B]$ is monotonically nonincreasing and L_ϕ-Lipschitz.

Comparison of Rademacher Complexity Bounds

We start with a comparison of Rademacher complexities of neural networks in the natural and adversarial settings. Although naively applying the definition of Rademacher complexity may provide a loose generalization bound [203], when properly normalized by the margin, one can still derive generalization bound that matches experimental observations via Rademacher

[2]This implies that $d_0 \equiv d$.

complexity [16]. Our comparison shows that, when the weight matrices of the neural networks have bounded norms, in the natural setting, the Rademacher complexity is upper bounded by a quantity which only has logarithmic dependence on the dimension; however, in the adversarial setting, the Rademacher complexity is lower bounded by a quantity with explicit $\sqrt{d}$ dependence.

We focus on the binary classification. For the natural setting, we review the results by Bartlett, Foster, and Telgarsky [16]. Let $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n \in (\mathcal{X} \times \{-1, +1\})^n$ be the i.i.d. training examples, and define

$$\mathbf{X} := [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n] \in \mathbb{R}^{d\times n}, \quad \text{and} \quad d_{\max} = \max\{d, d_1, d_2, \dots, d_L\}.$$

Theorem 5.5. *[16] Consider the neural network hypothesis class*

$$\mathcal{F} = \{f_{\mathbf{W}}(\mathbf{x}) : \mathbf{W} = (\mathbf{W}_1, \mathbf{W}_2, \dots, \mathbf{W}_L), \|\mathbf{W}_h\|_\sigma \le s_h, \|\mathbf{W}_h^\top\|_{2,1} \le b_h, h \in [L]\} \subseteq \mathbb{R}^{\mathcal{X}}.$$

Then, we have

$$\mathfrak{R}_{\mathcal{S}}(\mathcal{F}) \le \frac{4}{n^{3/2}} + \frac{26\log(n)\log(2d_{\max})}{n}\|\mathbf{X}\|_F\left(\prod_{h=1}^L s_h\right)\left(\sum_{j=1}^L (\frac{b_j}{s_j})^{2/3}\right)^{3/2}.$$

On the other hand, in this work, we prove the following result which shows that when the product of the spectral norms of all the weight matrices is bounded, the Rademacher complexity of the adversarial loss function class is lower bounded by a quantity with an explicit $\sqrt{d}$ factor. More specifically, for binary classification problems, since

$$\widetilde{\ell}(f_{\mathbf{W}}(\mathbf{x}), y) = \max_{\mathbf{x}' \in \mathbb{B}_{\mathbf{x}}^\infty(\epsilon)} \ell(f_{\mathbf{W}}(\mathbf{x}'), y) = \phi(\min_{\mathbf{x}' \in \mathbb{B}_{\mathbf{x}}^\infty(\epsilon)} y f_{\mathbf{W}}(\mathbf{x}')),$$

and $\phi(\cdot)$ is Lipschitz, we consider the function class

$$\widetilde{\mathcal{F}} = \{(\mathbf{x}, y) \mapsto \min_{\mathbf{x}' \in \mathbb{B}_{\mathbf{x}}^\infty(\epsilon)} y f_{\mathbf{W}}(\mathbf{x}') : \mathbf{W} = (\mathbf{W}_1, \mathbf{W}_2, \dots, \mathbf{W}_L), \prod_{h=1}^L \|\mathbf{W}_h\|_\sigma \le r\} \subseteq \mathbb{R}^{\mathcal{X}\times\{-1,+1\}}. \tag{5.6}$$

Then we have the following result.

Theorem 5.6 (**Main Result 3**)**.** *Let $\widetilde{\mathcal{F}}$ be defined as in* (5.6)*. Then, there exists a universal constant $c > 0$ such that*

$$\mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}}) \ge cr\left(\frac{1}{n}\|\mathbf{X}\|_F + \epsilon\sqrt{\frac{d}{n}}\right).$$

This result shows that if we aim to study the Rademacher complexity of the function class defined as in (5.6), a $\sqrt{d}$ dimension dependence may be unavoidable, in contrast to the natural setting where the dimension dependence is only logarithmic.

Generalization Bound for Surrogate Adversarial Loss

For neural networks, even if there is only one hidden layer, for a particular data point $(\mathbf{x}, y)$, evaluating the adversarial loss $\widetilde{\ell}(f_{\mathbf{W}}(\mathbf{x}), y) = \max_{\mathbf{x}' \in \mathbb{B}_{\mathbf{x}}^{\infty}(\epsilon)} \ell(f_{\mathbf{W}}(\mathbf{x}'), y)$ can be hard, since it requires maximizing a non-concave function in a bounded set. A recent line of work tries to find upper bounds for $\widetilde{\ell}(f_{\mathbf{W}}(\mathbf{x}), y)$ that can be computed in polynomial time. More specifically, we would like to find *surrogate adversarial loss* $\widehat{\ell}(f_{\mathbf{W}}(\mathbf{x}), y)$ such that $\widehat{\ell}(f_{\mathbf{W}}(\mathbf{x}), y) \geq \widetilde{\ell}(f_{\mathbf{W}}(\mathbf{x}), y)$, $\forall\ \mathbf{x}, y, \mathbf{W}$. These surrogate adversarial loss can thus provide *certified* defense against adversarial examples, and can be computed efficiently. In addition, the surrogate adversarial loss $\widehat{\ell}(f_{\mathbf{W}}(\mathbf{x}), y)$ should be as tight as possible—it should be close enough to the original adversarial loss $\widetilde{\ell}(f_{\mathbf{W}}(\mathbf{x}), y)$, so that the surrogate adversarial loss can indeed represent the robustness of the model against adversarial attacks. Recently, a few approaches to designing surrogate adversarial loss have been developed, and SDP relaxation [154, 155] and LP relaxation [105, 190] are two major examples.

In this section, we focus on the SDP relaxation for one hidden layer neural network with ReLU activation proposed by Raghunathan, Steinhardt, and Liang [154]. We prove a generalization bound regarding the surrogate adversarial loss, and show that this generalization bound does not have explicit dimension dependence if the weight matrix of the first layer has bounded ℓ_1 norm. We consider K-class classification problems in this section (i.e., $\mathcal{Y} = [K]$), and start with the definition and property of the SDP surrogate loss. Since we only have one hidden layer, $f_{\mathbf{W}}(\mathbf{x}) = \mathbf{W}_2 \rho(\mathbf{W}_1 \mathbf{x})$. Let $\mathbf{w}_{2,k}$ be the k-th column of $\mathbf{W}_2^\top$. Then, we have the following results according to Raghunathan, Steinhardt, and Liang [154].

Theorem 5.7. *[154] For any $(\mathbf{x}, y)$, $\mathbf{W}_1$, $\mathbf{W}_2$, and $y' \neq y$,*

$$\max_{\mathbf{x}' \in \mathbb{B}_{\mathbf{x}}^{\infty}(\epsilon)} ([f_{\mathbf{W}}(\mathbf{x}')]_{y'} - [f_{\mathbf{W}}(\mathbf{x}')]_{y}) \leq [f_{\mathbf{W}}(\mathbf{x})]_{y'} - [f_{\mathbf{W}}(\mathbf{x})]_{y} + \frac{\epsilon}{4} \max_{\mathbf{P} \succeq 0, \mathrm{diag}(\mathbf{P}) \leq 1} \langle Q(\mathbf{w}_{2,y'} - \mathbf{w}_{2,y}, \mathbf{W}_1), \mathbf{P} \rangle,$$

where $Q(\mathbf{v}, \mathbf{W})$ is defined as

$$Q(\mathbf{v}, \mathbf{W}) := \begin{bmatrix} 0 & 0 & \mathbf{1}^\top \mathbf{W}^\top \mathrm{diag}(\mathbf{v}) \\ 0 & 0 & \mathbf{W}^\top \mathrm{diag}(\mathbf{v}) \\ \mathrm{diag}(\mathbf{v})^\top \mathbf{W} \mathbf{1} & \mathrm{diag}(\mathbf{v})^\top \mathbf{W} & 0 \end{bmatrix}. \tag{5.7}$$

Since we consider multi-class classification problems in this section, we use the ramp loss ϕ_γ defined in (5.4) composed with the margin operator as our loss function. Thus, we have $\ell(f_{\mathbf{W}}(\mathbf{x}), y) = \phi_\gamma(M(f_{\mathbf{W}}(\mathbf{x}), y))$ and $\widetilde{\ell}(f_{\mathbf{W}}(\mathbf{x}), y) = \max_{\mathbf{x}' \in \mathbb{B}_{\mathbf{x}}^{\infty}(\epsilon)} \phi_\gamma(M(f_{\mathbf{W}}(\mathbf{x}'), y))$. Here, we design a surrogate loss $\widehat{\ell}(f_{\mathbf{W}}(\mathbf{x}), y)$ based on Theorem 5.7.

Lemma 5.1. *Define*

$$\widehat{\ell}(f_{\mathbf{W}}(\mathbf{x}), y) := \phi_\gamma\Big(M(f_{\mathbf{W}}(\mathbf{x}), y) - \frac{\epsilon}{2} \max_{k \in [K], z = \pm 1} \max_{\mathbf{P} \succeq 0, \mathrm{diag}(\mathbf{P}) \leq 1} \langle z Q(\mathbf{w}_{2,k}, \mathbf{W}_1), \mathbf{P} \rangle\Big).$$

Then, we have

$$\max_{\mathbf{x}'\in\mathbb{B}_{\mathbf{x}}^{\infty}(\epsilon)} \mathbb{1}(y \neq \arg\max_{y'\in[K]}[f_{\mathbf{W}}(\mathbf{x}')]_{y'}) \leq \widehat{\ell}(f_{\mathbf{W}}(\mathbf{x}), y)$$
$$\leq \mathbb{1}\Big(M(f_{\mathbf{W}}(\mathbf{x}), y) - \frac{\epsilon}{2} \max_{k\in[K],z=\pm 1} \max_{\mathbf{P}\succeq 0,\mathrm{diag}(\mathbf{P})\leq 1} \langle zQ(\mathbf{w}_{2,k}, \mathbf{W}_1), \mathbf{P}\rangle \leq \gamma\Big).$$

With this surrogate adversarial loss in hand, we can develop the following margin bound for adversarial generalization. In this theorem, we use $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n] \in \mathbb{R}^{d\times n}$, and $d_{\max} = \max\{d, d_1, K\}$.

Theorem 5.8. *Consider the neural network hypothesis class*

$$\mathcal{F} = \{f_{\mathbf{W}}(\mathbf{x}) : \mathbf{W} = (\mathbf{W}_1, \mathbf{W}_2), \|\mathbf{W}_h\|_\sigma \leq s_h, h = 1, 2, \|\mathbf{W}_1\|_1 \leq b_1, \|\mathbf{W}_2^\top\|_{2,1} \leq b_2\}.$$

Then, for any fixed $\gamma > 0$, with probability at least $1-\delta$, we have for all $f_{\mathbf{W}}(\cdot) \in \mathcal{F}$,

$$\mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{\exists\ \mathbf{x}' \in \mathbb{B}_{\mathbf{x}}^{\infty}(\epsilon)\ s.t.\ y \neq \arg\max_{y'\in[K]}[f_{\mathbf{W}}(\mathbf{x}')]_{y'}\right\}$$
$$\leq \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\Big([f_{\mathbf{W}}(\mathbf{x}_i)]_{y_i} \leq \gamma + \max_{y'\neq y_i}[f_{\mathbf{W}}(\mathbf{x}_i)]_{y'} + \frac{\epsilon}{2} \max_{k\in[K],z=\pm 1} \max_{\mathbf{P}\succeq 0,\mathrm{diag}(\mathbf{P})\leq 1} \langle zQ(\mathbf{w}_{2,k}, \mathbf{W}_1), \mathbf{P}\rangle\Big)$$
$$+ \frac{1}{\gamma}\left(\frac{4}{n^{3/2}} + \frac{60\log(n)\log(2d_{\max})}{n} s_1 s_2 \Big((\frac{b_1}{s_1})^{2/3} + (\frac{b_2}{s_2})^{2/3}\Big)^{3/2} \|\mathbf{X}\|_F + \frac{2\epsilon b_1 b_2}{\sqrt{n}}\right) + 3\sqrt{\frac{\log\frac{2}{\delta}}{2n}}.$$

Similar to linear classifiers, in the adversarial setting, if we have an ℓ_1 norm constraint on the matrix matrix $\mathbf{W}_1$, then the generalization bound of the surrogate adversarial loss does not have an explicit dimension dependence.

5.6 Experiments

In this section, we validate our theoretical findings for linear classifiers and neural networks via experiments. Our experiments are implemented with Tensorflow [1] on the MNIST dataset [115].

Linear Classifiers

We validate two theoretical findings for linear classifiers: (i) controlling the ℓ_1 norm of the model parameters can reduce the adversarial generalization error, and (ii) there is a dimension dependence in adversarial generalization, i.e., adversarially robust generalization is harder when the dimension of the feature space is higher. We train the multi-class linear classifier using the following objective function:

$$\min_{\mathbf{W}} \frac{1}{n}\sum_{i=1}^{n} \max_{\mathbf{x}_i' \in \mathbb{B}_{\mathbf{x}_i}^{\infty}(\epsilon)} \ell(f_{\mathbf{W}}(\mathbf{x}_i'), y_i) + \lambda\|\mathbf{W}\|_1, \tag{5.8}$$

where $\ell(\cdot)$ is cross entropy loss and $f_{\mathbf{W}}(\mathbf{x}) \equiv \mathbf{W}\mathbf{x}$. Since we focus on the generalization property, we use a small number of training data so that the generalization gap is more significant. More specifically, in each run of the training algorithm, we randomly sample $n = 1000$ data points from the training set of MNIST as the training data, and run adversarial training to minimize the objective (5.8). Our training algorithm alternates between mini-batch stochastic gradient descent with respect to $\mathbf{W}$ and computing adversarial examples on the chosen batch in each iteration. Here, we note that since we consider linear classifiers, the adversarial examples can be analytically computed.

In our first experiment, we vary the values of ϵ and λ, and for each (ϵ, λ) pair, we conduct 10 runs of the training algorithm, and in each run we sample the 1000 training data independently. In Figure 5.2, we plot the adversarial generalization error as a function of ϵ and λ, and the error bars show the standard deviation of the 10 runs. As we can see, when λ increases, the generalization gap decreases, and thus we conclude that ℓ_1 regularization is helpful for reducing adversarial generalization error.

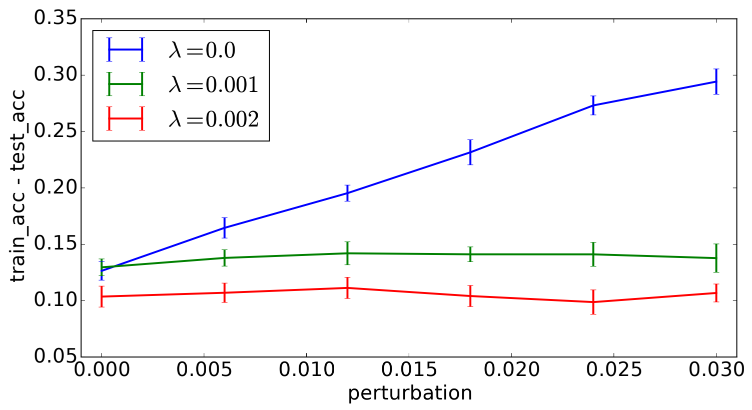


Figure 5.2: Linear classifiers. Adversarial generalization error vs ℓ_∞ perturbation ϵ and regularization coefficient λ.

In our second experiment, we choose $\lambda = 0$ and study the dependence of adversarial generalization error on the dimension of the feature space. Recall that each data point in the original MNIST dataset is a 28×28 image, i.e., $d = 784$. We construct two additional image datasets with $d = 196$ (downsampled) and $d = 3136$ (expanded), respectively. To construct the downsampled image, we replace each 2×2 patch—say, with pixel values a, b, c, d—on the original image with a single pixel with value $\sqrt{a^2 + b^2 + c^2 + d^2}$. To construct the expanded image, we replace each pixel—say, with value a—on the original image with a 2×2 patch, with the value of each pixel in the patch being $a/2$. Such construction keeps the ℓ_2 norm of the every single image the same across the three datasets, and thus leads a fair comparison. The adversarial generalization error is plotted in Figure 5.3, and as we can see, when the dimension d increases, the generalization gap also increases.

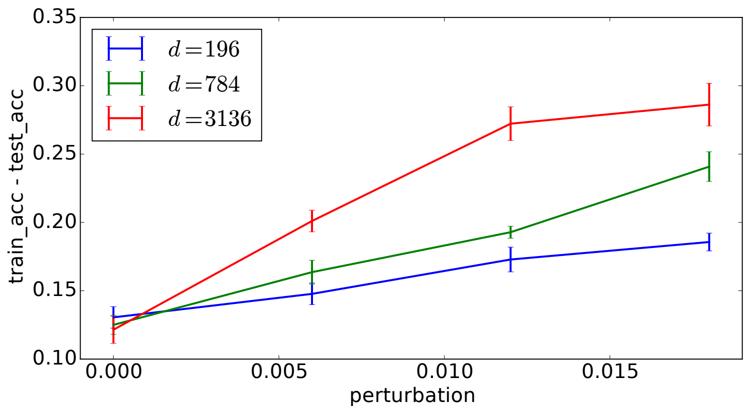


Figure 5.3: Linear classifiers. Adversarial generalization error vs ℓ_∞ perturbation ϵ and dimension of feature space d.

Neural Networks

In this experiment, we validate our theoretical result that ℓ_1 regularization can reduce the adversarial generalization error on a four-layer ReLU neural network, where the first two layers are convolutional and the last two layers are fully connected. We use PGD attack [132] adversarial training to minimize the ℓ_1 regularized objective (5.8). We use the whole training set of MNIST, and once the model is obtained, we use PGD attack to check the adversarial training and test error. We present the adversarial generalization errors under the PGD attack in Figure 5.4. As we can see, the adversarial generalization error decreases as we increase the regularization coefficient λ; thus ℓ_1 regularization indeed reduces the adversarial generalization error under the PGD attack.

5.7 Conclusions

We study the adversarially robust generalization properties of linear classifiers and neural networks through the lens of Rademacher complexity. For binary linear classifiers, we prove tight bounds for the adversarial Rademacher complexity, and show that in the adversarial setting, Rademacher complexity is never smaller than that in the natural setting, and it has an unavoidable dimension dependence, unless the weight vector has bounded ℓ_1 norm. The results also extends to multi-class linear classifiers. For neural networks, we prove a lower bound of the Rademacher complexity of the adversarial loss function class and show that

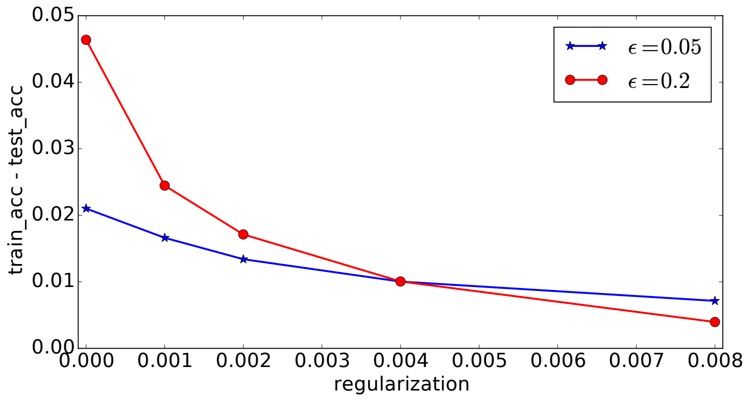


Figure 5.4: Neural networks. Adversarial generalization error vs regularization coefficient λ.

there is also an unavoidable dimension dependence due to ℓ_∞ adversarial attack. We further consider a surrogate adversarial loss and prove margin bound for this setting. Our results indicate that having ℓ_1 norm constraints on the weight matrices might be a potential way to improve generalization in the adversarial setting. Our experimental results validate our theoretical findings.

5.8 Proofs

Proof of Theorem 5.2

First, we have

$$\mathfrak{R}_{\mathcal{S}}(\mathcal{F}) := \frac{1}{n}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{\|\mathbf{w}\|_p\le W}\sum_{i=1}^n \sigma_i\langle \mathbf{w},\mathbf{x}_i\rangle\right] = \frac{W}{n}\mathbb{E}_{\boldsymbol{\sigma}}\left[\left\|\sum_{i=1}^n \sigma_i\mathbf{x}_i\right\|_q\right]. \tag{5.9}$$

We then analyze $\mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}})$. Define $\widetilde{f}_{\mathbf{w}}(\mathbf{x},y) := \min_{\mathbf{x}'\in\mathbb{B}^\infty_{\mathbf{x}}(\epsilon)} y\langle \mathbf{w},\mathbf{x}'\rangle$. Then, we have

$$\widetilde{f}_{\mathbf{w}}(\mathbf{x},y) = \begin{cases}\min_{\mathbf{x}'\in\mathbb{B}^\infty_{\mathbf{x}}(\epsilon)}\langle \mathbf{w},\mathbf{x}'\rangle & y=1,\\ -\max_{\mathbf{x}'\in\mathbb{B}^\infty_{\mathbf{x}}(\epsilon)}\langle \mathbf{w},\mathbf{x}'\rangle & y=-1.\end{cases}$$

When $y=1$, we have

$$\begin{aligned}\widetilde{f}_{\mathbf{w}}(\mathbf{x},y) = \widetilde{f}_{\mathbf{w}}(\mathbf{x},1) &= \min_{\mathbf{x}'\in\mathbb{B}^\infty_{\mathbf{x}}(\epsilon)}\langle \mathbf{w},\mathbf{x}'\rangle = \min_{\mathbf{x}'\in\mathbb{B}^\infty_{\mathbf{x}}(\epsilon)}\sum_{i=1}^d w_i x_i'\\ &= \sum_{i=1}^d w_i\left[\mathbb{1}(w_i\ge 0)(x_i-\epsilon)+\mathbb{1}(w_i<0)(x_i+\epsilon)\right] = \sum_{i=1}^d w_i(x_i-\operatorname{sgn}(w_i)\epsilon)\\ &= \langle \mathbf{w},\mathbf{x}\rangle - \epsilon\|\mathbf{w}\|_1.\end{aligned}$$

Similarly, when $y = -1$, we have

$$\begin{aligned}\widetilde{f}_{\mathbf{w}}(\mathbf{x}, y) = \widetilde{f}_{\mathbf{w}}(\mathbf{x}, -1) &= -\max_{\mathbf{x}' \in \mathbb{B}_{\mathbf{x}}^{\infty}(\epsilon)} \langle \mathbf{w}, \mathbf{x}' \rangle = -\max_{\mathbf{x}' \in \mathbb{B}_{\mathbf{x}}^{\infty}(\epsilon)} \sum_{i=1}^{d} w_i x_i' \\ &= -\sum_{i=1}^{d} w_i \left[\mathbb{1}(w_i \geq 0)(x_i + \epsilon) + \mathbb{1}(w_i < 0)(x_i - \epsilon)\right] = -\sum_{i=1}^{d} w_i (x_i + \operatorname{sgn}(w_i)\epsilon) \\ &= -\langle \mathbf{w}, \mathbf{x} \rangle - \epsilon \|\mathbf{w}\|_1.\end{aligned}$$

Thus, we conclude that $\widetilde{f}_{\mathbf{w}}(\mathbf{x}, y) = y\langle \mathbf{w}, \mathbf{x} \rangle - \epsilon \|\mathbf{w}\|_1$, and therefore

$$\mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}}) = \frac{1}{n} \mathbb{E}_{\boldsymbol{\sigma}} \left[\sup_{\|\mathbf{w}\|_2 \leq W} \sum_{i=1}^{n} \sigma_i (y_i \langle \mathbf{w}, \mathbf{x}_i \rangle - \epsilon \|\mathbf{w}\|_1) \right].$$

Define $\mathbf{u} := \sum_{i=1}^{n} \sigma_i y_i \mathbf{x}_i$ and $v := \epsilon \sum_{i=1}^{n} \sigma_i$. Then we have

$$\mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}}) = \frac{1}{n} \mathbb{E}_{\boldsymbol{\sigma}} \left[\sup_{\|\mathbf{w}\|_p \leq W} \langle \mathbf{w}, \mathbf{u} \rangle - v \|\mathbf{w}\|_1 \right]$$

Since the supremum of $\langle \mathbf{w}, \mathbf{u} \rangle - v \|\mathbf{w}\|_1$ over $\mathbf{w}$ can only be achieved when $\operatorname{sgn}(w_i) = \operatorname{sgn}(u_i)$, we know that

$$\begin{aligned}\mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}}) =& \frac{1}{n} \mathbb{E}_{\boldsymbol{\sigma}} \left[\sup_{\|\mathbf{w}\|_p \leq W} \langle \mathbf{w}, \mathbf{u} \rangle - v \langle \mathbf{w}, \operatorname{sgn}(\mathbf{w}) \rangle \right] \\ =& \frac{1}{n} \mathbb{E}_{\boldsymbol{\sigma}} \left[\sup_{\|\mathbf{w}\|_p \leq W} \langle \mathbf{w}, \mathbf{u} \rangle - v \langle \mathbf{w}, \operatorname{sgn}(\mathbf{u}) \rangle \right] \\ =& \frac{1}{n} \mathbb{E}_{\boldsymbol{\sigma}} \left[\sup_{\|\mathbf{w}\|_p \leq W} \langle \mathbf{w}, \mathbf{u} - v \operatorname{sgn}(\mathbf{u}) \rangle \right] \\ =& \frac{W}{n} \mathbb{E}_{\boldsymbol{\sigma}} \left[\|\mathbf{u} - v \operatorname{sgn}(\mathbf{u})\|_q \right] \\ =& \frac{W}{n} \mathbb{E}_{\boldsymbol{\sigma}} \left[\left\| \sum_{i=1}^{n} \sigma_i y_i \mathbf{x}_i - \big(\epsilon \sum_{i=1}^{n} \sigma_i\big) \operatorname{sgn}(\sum_{i=1}^{n} \sigma_i y_i \mathbf{x}_i) \right\|_q \right]. \qquad (5.10)\end{aligned}$$

Now we prove an upper bound for $\mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}})$. By triangle inequality, we have

$$\begin{aligned}\mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}}) \leq & \frac{W}{n}\mathbb{E}_{\boldsymbol{\sigma}}\left[\left\|\sum_{i=1}^n \sigma_i y_i \mathbf{x}_i\right\|_q\right] + \frac{\epsilon W}{n}\mathbb{E}_{\boldsymbol{\sigma}}\left[\left\|(\sum_{i=1}^n \sigma_i)\operatorname{sgn}(\sum_{i=1}^n \sigma_i y_i \mathbf{x}_i)\right\|_q\right] \\ = & \mathfrak{R}_{\mathcal{S}}(\mathcal{F}) + \epsilon W \frac{d^{\frac{1}{q}}}{n}\mathbb{E}_{\boldsymbol{\sigma}}\left[\left|\sum_{i=1}^n \sigma_i\right|\right] \\ \leq & \mathfrak{R}_{\mathcal{S}}(\mathcal{F}) + \epsilon W \frac{d^{\frac{1}{q}}}{\sqrt{n}},\end{aligned}$$

where the last step is due to Khintchine's inequality.

We then proceed to prove a lower bound for $\mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}})$. According to (5.10) and by symmetry, we know that

$$\begin{aligned}\mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}}) = & \frac{W}{n}\mathbb{E}_{\boldsymbol{\sigma}}\left[\left\|\sum_{i=1}^n (-\sigma_i) y_i \mathbf{x}_i - \big(\epsilon\sum_{i=1}^n (-\sigma_i)\big)\operatorname{sgn}(\sum_{i=1}^n (-\sigma_i) y_i \mathbf{x}_i)\right\|_q\right] \\ = & \frac{W}{n}\mathbb{E}_{\boldsymbol{\sigma}}\left[\left\|\sum_{i=1}^n \sigma_i y_i \mathbf{x}_i + \big(\epsilon\sum_{i=1}^n \sigma_i\big)\operatorname{sgn}(\sum_{i=1}^n \sigma_i y_i \mathbf{x}_i)\right\|_q\right]. \qquad (5.11)\end{aligned}$$

Then, combining (5.10) and (5.11) and using triangle inequality, we have

$$\begin{aligned}\mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}}) = & \frac{W}{2n}\mathbb{E}_{\boldsymbol{\sigma}}\Bigg[\left\|\sum_{i=1}^n \sigma_i y_i \mathbf{x}_i - \big(\epsilon\sum_{i=1}^n \sigma_i\big)\operatorname{sgn}(\sum_{i=1}^n \sigma_i y_i \mathbf{x}_i)\right\|_q \\ & + \left\|\sum_{i=1}^n \sigma_i y_i \mathbf{x}_i + \big(\epsilon\sum_{i=1}^n \sigma_i\big)\operatorname{sgn}(\sum_{i=1}^n \sigma_i y_i \mathbf{x}_i)\right\|_q\Bigg] \\ \geq & \frac{W}{n}\mathbb{E}_{\boldsymbol{\sigma}}\left[\left\|\sum_{i=1}^n \sigma_i y_i \mathbf{x}_i\right\|_q\right] = \mathfrak{R}_{\mathcal{S}}(\mathcal{F}). \qquad (5.12)\end{aligned}$$

Similarly, we have

$$\begin{aligned}\mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}}) \geq & \frac{W}{n}\mathbb{E}_{\boldsymbol{\sigma}}\left[\left\|\big(\epsilon\sum_{i=1}^n \sigma_i\big)\operatorname{sgn}(\sum_{i=1}^n \sigma_i y_i \mathbf{x}_i)\right\|_q\right] \\ = & \frac{W}{n}\mathbb{E}_{\boldsymbol{\sigma}}\left[\epsilon\left|\sum_{i=1}^n \sigma_i\right|\left\|\operatorname{sgn}(\sum_{i=1}^n \sigma_i y_i \mathbf{x}_i)\right\|_q\right] \\ = & \epsilon W \frac{d^{\frac{1}{q}}}{n}\mathbb{E}_{\boldsymbol{\sigma}}\left[\left|\sum_{i=1}^n \sigma_i\right|\right].\end{aligned}$$

By Khintchine's inequality, we know that there exists a universal constant $c > 0$ such that $\mathbb{E}_{\boldsymbol{\sigma}}[|\sum_{i=1}^n \sigma_i|] \geq c\sqrt{n}$. Therefore, we have $\mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}}) \geq c\epsilon W \frac{d^{\frac{1}{q}}}{\sqrt{n}}$. Combining with (5.12), we complete the proof.

Proof of Theorem 5.3

According to the multi-class margin bound in [112], for any fixed γ, with probability at least $1-\delta$, we have

$$\begin{aligned}\mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{y \neq \arg\max_{y'\in[K]}[f(\mathbf{x})]_{y'}\right\} \leq & \frac{1}{n}\sum_{i=1}^n \mathbb{1}([f(\mathbf{x}_i)]_{y_i} \leq \gamma + \max_{y'\neq y}[f(\mathbf{x}_i)]_{y'}) \\ & + \frac{4K}{\gamma}\mathfrak{R}_{\mathcal{S}}(\Pi_1(\mathcal{F})) + 3\sqrt{\frac{\log\frac{2}{\delta}}{2n}},\end{aligned}$$

where $\Pi_1(\mathcal{F}) \subseteq \mathbb{R}^{\mathcal{X}}$ is defined as

$$\Pi_1(\mathcal{F}) := \{\mathbf{x} \mapsto [f(\mathbf{x})]_k : f \in \mathcal{F}, k \in [K]\}.$$

In the special case of linear classifiers $\mathcal{F} = \{f_{\mathbf{W}}(\mathbf{x}) : \|\mathbf{W}^\top\|_{p,\infty} \leq W\}$, we can see that

$$\Pi_1(\mathcal{F}) = \{\mathbf{x} \mapsto \langle \mathbf{w}, \mathbf{x}\rangle : \|\mathbf{w}\|_p \leq W\}.$$

Thus, we have

$$\mathfrak{R}_{\mathcal{S}}(\Pi_1(\mathcal{F})) = \frac{1}{n}\mathbb{E}_{\boldsymbol{\sigma}}\left[\left\|\sum_{i=1}^n \sigma_i \mathbf{x}_i\right\|_q\right],$$

which completes the proof.

Proof of Theorem 5.4

Since the loss function in the adversarial setting is

$$\widetilde{\ell}(f_{\mathbf{W}}(\mathbf{x}), y) = \max_{\mathbf{x}'\in\mathbb{B}_{\mathbf{x}}^\infty(\epsilon)} \phi_\gamma(M(f_{\mathbf{W}}(\mathbf{x}), y)) = \phi_\gamma(\min_{\mathbf{x}'\in\mathbb{B}_{\mathbf{x}}^\infty(\epsilon)} M(f_{\mathbf{W}}(\mathbf{x}), y)).$$

Since we consider linear classifiers, we have

$$\begin{aligned}\min_{\mathbf{x}'\in\mathbb{B}_{\mathbf{x}}^\infty(\epsilon)} M(f_{\mathbf{W}}(\mathbf{x}), y) &= \min_{\mathbf{x}'\in\mathbb{B}_{\mathbf{x}}^\infty(\epsilon)} \min_{y'\neq y}(\mathbf{w}_y - \mathbf{w}_{y'})^\top \mathbf{x}' \\ &= \min_{y'\neq y} \min_{\mathbf{x}'\in\mathbb{B}_{\mathbf{x}}^\infty(\epsilon)} (\mathbf{w}_y - \mathbf{w}_{y'})^\top \mathbf{x}' \\ &= \min_{y'\neq y}(\mathbf{w}_y - \mathbf{w}_{y'})^\top \mathbf{x} - \epsilon\|\mathbf{w}_y - \mathbf{w}_{y'}\|_1 \qquad (5.13)\end{aligned}$$

Define

$$h_{\mathbf{W}}^{(k)}(\mathbf{x}, y) := (\mathbf{w}_y - \mathbf{w}_k)^\top \mathbf{x} - \epsilon \|\mathbf{w}_y - \mathbf{w}_k\|_1 + \gamma \mathbb{1}(y = k).$$

We now show that

$$\widetilde{\ell}(f_{\mathbf{W}}(\mathbf{x}), y) = \max_{k \in [K]} \phi_\gamma(h_{\mathbf{W}}^{(k)}(\mathbf{x}, y)). \tag{5.14}$$

To see this, we can see that according to (5.13),

$$\min_{\mathbf{x}' \in \mathbb{B}_{\mathbf{x}}^\infty(\epsilon)} M(f_{\mathbf{W}}(\mathbf{x}), y) = \min_{k \neq y} h_{\mathbf{W}}^{(k)}(\mathbf{x}, y).$$

If $\min_{k \neq y} h_{\mathbf{W}}^{(k)}(\mathbf{x}, y) \le \gamma$, we have $\min_{k \neq y} h_{\mathbf{W}}^{(k)}(\mathbf{x}, y) = \min_{k \in [K]} h_{\mathbf{W}}^{(k)}(\mathbf{x}, y)$, since $h_{\mathbf{W}}^{(y)}(\mathbf{x}, y) = \gamma$. On the other hand, if $\min_{k \neq y} h_{\mathbf{W}}^{(k)}(\mathbf{x}, y) > \gamma$, then $\min_{k \in [K]} h_{\mathbf{W}}^{(k)}(\mathbf{x}, y) = \gamma$. In this case, we have $\phi_\gamma(\min_{k \neq y} h_{\mathbf{W}}^{(k)}(\mathbf{x}, y)) = \phi_\gamma(\min_{k \in [K]} h_{\mathbf{W}}^{(k)}(\mathbf{x}, y)) = 0$. Therefore, we can see that (5.14) holds.

Define the K function classes $\mathcal{F}_k := \{h_{\mathbf{W}}^{(k)}(\mathbf{x}, y) : \|\mathbf{W}^\top\|_{p,\infty} \le W\} \subseteq \mathbb{R}^{\mathcal{X} \times \mathcal{Y}}$. Since $\phi_\gamma(\cdot)$ is $1/\gamma$-Lipschitz, according to the Ledoux-Talagrand contraction inequality [116] and Lemma 8.1 in [140], we have

$$\mathfrak{R}_{\mathcal{S}}(\widetilde{\ell}_{\mathcal{F}}) \le \frac{1}{\gamma} \sum_{k=1}^{K} \mathfrak{R}_{\mathcal{S}}(\mathcal{F}_k). \tag{5.15}$$

We proceed to analyze $\mathfrak{R}_{\mathcal{S}}(\mathcal{F}_k)$. The basic idea is similar to the proof of Theorem 5.2. We define $\mathbf{u}_y = \sum_{i=1}^n \sigma_i \mathbf{x}_i \mathbb{1}(y_i = y)$ and $v_y = \sum_{i=1}^n \sigma_i \mathbb{1}(y_i = y)$. Then, we have

$$
\begin{aligned}
&\mathfrak{R}_{\mathcal{S}}(\mathcal{F}_k)\\
=&\frac{1}{n}\mathbb{E}_{\boldsymbol{\sigma}}\Big[\sup_{\|\mathbf{W}^\top\|_{p,\infty}\le W}\sum_{i=1}^n \sigma_i((\mathbf{w}_{y_i}-\mathbf{w}_k)^\top \mathbf{x}_i - \epsilon\|\mathbf{w}_{y_i}-\mathbf{w}_k\|_1 + \gamma\mathbb{1}(y_i=k))\Big]\\
=&\frac{1}{n}\mathbb{E}_{\boldsymbol{\sigma}}\Big[\sup_{\|\mathbf{W}^\top\|_{p,\infty}\le W}\sum_{i=1}^n\sum_{y=1}^K \sigma_i((\mathbf{w}_{y_i}-\mathbf{w}_k)^\top \mathbf{x}_i - \epsilon\|\mathbf{w}_{y_i}-\mathbf{w}_k\|_1 + \gamma\mathbb{1}(y_i=k))\mathbb{1}(y_i=y)\Big]\\
=&\frac{1}{n}\mathbb{E}_{\boldsymbol{\sigma}}\Big[\sup_{\|\mathbf{W}^\top\|_{p,\infty}\le W}\sum_{y=1}^K\sum_{i=1}^n \sigma_i((\mathbf{w}_{y}-\mathbf{w}_k)^\top \mathbf{x}_i\mathbb{1}(y_i=y) - \epsilon\|\mathbf{w}_{y}-\mathbf{w}_k\|_1\mathbb{1}(y_i=y)\\
&+\gamma\mathbb{1}(y_i=k)\mathbb{1}(y_i=y))\Big]\\
=&\frac{1}{n}\mathbb{E}_{\boldsymbol{\sigma}}\Big[\gamma\sum_{i=1}^n\sigma_i\mathbb{1}(y_i=k) + \sup_{\|\mathbf{W}^\top\|_{p,\infty}\le W}\sum_{y\neq k}(\langle \mathbf{w}_y-\mathbf{w}_k, \mathbf{u}_y\rangle - \epsilon v_y\|\mathbf{w}_y-\mathbf{w}_k\|_1)\Big]\\
\le&\frac{1}{n}\mathbb{E}_{\boldsymbol{\sigma}}\Big[\sum_{y\neq k}\sup_{\|\mathbf{w}_k\|_p,\|\mathbf{w}_y\|_p\le W}(\langle \mathbf{w}_y-\mathbf{w}_k, \mathbf{u}_y\rangle - \epsilon v_y\|\mathbf{w}_y-\mathbf{w}_k\|_1)\Big]\\
=&\frac{1}{n}\mathbb{E}_{\boldsymbol{\sigma}}\Big[\sum_{y\neq k}\sup_{\|\mathbf{w}\|_p\le 2W}(\langle \mathbf{w}, \mathbf{u}_y\rangle - \epsilon v_y\|\mathbf{w}\|_1)\Big]\\
=&\frac{2W}{n}\mathbb{E}_{\boldsymbol{\sigma}}\Big[\sum_{y\neq k}\|\mathbf{u}_y - \epsilon v_y\,\mathrm{sgn}(\mathbf{u}_y)\|_q\Big],
\end{aligned}
$$

where the last equality is due to the same derivation as in the proof of Theorem 5.2. Let $n_y = \sum_{i=1}^n \mathbb{1}(y_i = y)$. Then, we apply triangle inequality and Khintchine's inequality and obtain

$$\mathfrak{R}_{\mathcal{S}}(\mathcal{F}_k) \le \frac{2W}{n}\sum_{y\neq k}\mathbb{E}_{\boldsymbol{\sigma}}[\|\mathbf{u}_y\|_2] + \epsilon d^{\frac{1}{q}}\sqrt{n_y}.$$

Combining with (5.15), we obtain

$$\mathfrak{R}_{\mathcal{S}}(\widetilde{\ell}_{\mathcal{F}}) \le \frac{2WK}{\gamma n}(\sum_{y=1}^K\mathbb{E}_{\boldsymbol{\sigma}}[\|\mathbf{u}_y\|_2] + \epsilon d^{\frac{1}{q}}\sqrt{n_y}) \le \frac{2WK}{\gamma}\left[\frac{\epsilon\sqrt{K}d^{\frac{1}{q}}}{\sqrt{n}} + \frac{1}{n}\sum_{y=1}^K\mathbb{E}_{\boldsymbol{\sigma}}[\|\mathbf{u}_y\|_2]\right],$$

where the last step is due to Cauchy-Schwarz inequality.

Proof of Theorem 5.6

We first review a Rademacher complexity lower bound in [16].

Lemma 5.2. *[16] Define the function class*

$$\widehat{\mathcal{F}} = \{\mathbf{x} \mapsto f_{\mathbf{W}}(\mathbf{x}) : \mathbf{W} = (\mathbf{W}_1, \mathbf{W}_2, \dots, \mathbf{W}_L), \prod_{h=1}^{L} \|\mathbf{W}_h\|_\sigma \le r\},$$

and $\widehat{\mathcal{F}}' = \{\mathbf{x} \mapsto \langle \mathbf{w}, \mathbf{x}\rangle : \|\mathbf{w}\|_2 \le \frac{r}{2}\}$*. Then we have* $\widehat{\mathcal{F}}' \subseteq \widehat{\mathcal{F}}$*, and thus there exists a universal constant* $c > 0$ *such that*

$$\mathfrak{R}_{\mathcal{S}}(\widehat{\mathcal{F}}) \ge \frac{cr}{n}\|\mathbf{X}\|_F.$$

According to Lemma 5.2, in the adversarial setting, by defining

$$\widetilde{\mathcal{F}}' = \{\mathbf{x} \mapsto \min_{\mathbf{x}' \in \mathbb{B}^\infty_{\mathbf{x}}(\epsilon)} y\langle \mathbf{w}, \mathbf{x}'\rangle : \|\mathbf{w}\|_2 \le \frac{r}{2}\} \subseteq \mathbb{R}^{\mathcal{X}\times\{-1,+1\}},$$

we have $\widetilde{\mathcal{F}}' \subseteq \widetilde{\mathcal{F}}$. Therefore, there exists a universal constant $c > 0$ such that

$$\mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}}) \ge \mathfrak{R}_{\mathcal{S}}(\widetilde{\mathcal{F}}') \ge cr\left(\frac{1}{n}\|\mathbf{X}\|_F + \epsilon\sqrt{\frac{d}{n}}\right),$$

where the last inequality is due to Theorem 5.2.

Proof of Lemma 5.1

Since $Q(\cdot,\cdot)$ is a linear function in its first argument, we have for any $y, y' \in [K]$,

$$\begin{aligned}
&\max_{\mathbf{P}\succeq 0, \mathrm{diag}(\mathbf{P})\le 1} \langle Q(\mathbf{w}_{2,y'} - \mathbf{w}_{2,y}, \mathbf{W}_1), \mathbf{P}\rangle \\
\le &\max_{\mathbf{P}\succeq 0, \mathrm{diag}(\mathbf{P})\le 1} \langle Q(\mathbf{w}_{2,y'}, \mathbf{W}_1), \mathbf{P}\rangle + \max_{\mathbf{P}\succeq 0, \mathrm{diag}(\mathbf{P})\le 1} \langle -Q(\mathbf{w}_{2,y}, \mathbf{W}_1), \mathbf{P}\rangle \\
\le &2\max_{k\in[K], z=\pm 1}\max_{\mathbf{P}\succeq 0, \mathrm{diag}(\mathbf{P})\le 1} \langle zQ(\mathbf{w}_{2,k}, \mathbf{W}_1), \mathbf{P}\rangle.
\end{aligned} \tag{5.16}$$

Then, for any $(\mathbf{x}, y)$, we have

$$\begin{aligned}
&\max_{\mathbf{x}' \in \mathbb{B}^\infty_{\mathbf{x}}(\epsilon)} \mathbb{1}(y \ne \arg\max_{y'\in[K]} [f_{\mathbf{W}}(\mathbf{x}')]_{y'}) \\
\le &\phi_\gamma(\min_{\mathbf{x}' \in \mathbb{B}^\infty_{\mathbf{x}}(\epsilon)} M(f_{\mathbf{W}}(\mathbf{x}'), y)) \\
\le &\phi_\gamma(\min_{y'\ne y}\min_{\mathbf{x}' \in \mathbb{B}^\infty_{\mathbf{x}}(\epsilon)} [f_{\mathbf{W}}(\mathbf{x}')]_y - [f_{\mathbf{W}}(\mathbf{x}')]_{y'}) \\
\le &\phi_\gamma\Big(\min_{y'\ne y}[f_{\mathbf{W}}(\mathbf{x})]_y - [f_{\mathbf{W}}(\mathbf{x})]_{y'} - \frac{\epsilon}{4}\max_{y'\ne y}\max_{\mathbf{P}\succeq 0, \mathrm{diag}(\mathbf{P})\le 1} \langle Q(\mathbf{w}_{2,y'} - \mathbf{w}_{2,y}, \mathbf{W}_1), \mathbf{P}\rangle\Big) \\
\le &\phi_\gamma\Big(\min_{y'\ne y}[f_{\mathbf{W}}(\mathbf{x})]_y - [f_{\mathbf{W}}(\mathbf{x})]_{y'} - \frac{\epsilon}{2}\max_{k\in[K], z=\pm 1}\max_{\mathbf{P}\succeq 0, \mathrm{diag}(\mathbf{P})\le 1} \langle zQ(\mathbf{w}_{2,k}, \mathbf{W}_1), \mathbf{P}\rangle\Big) \\
\le &\phi_\gamma\Big(M(f_{\mathbf{W}}(\mathbf{x}), y) - \frac{\epsilon}{2}\max_{k\in[K], z=\pm 1}\max_{\mathbf{P}\succeq 0, \mathrm{diag}(\mathbf{P})\le 1} \langle zQ(\mathbf{w}_{2,k}, \mathbf{W}_1), \mathbf{P}\rangle\Big) := \widehat{\ell}(f_{\mathbf{W}}(\mathbf{x}), y) \\
\le &\mathbb{1}\Big(M(f_{\mathbf{W}}(\mathbf{x}), y) - \frac{\epsilon}{2}\max_{k\in[K], z=\pm 1}\max_{\mathbf{P}\succeq 0, \mathrm{diag}(\mathbf{P})\le 1} \langle zQ(\mathbf{w}_{2,k}, \mathbf{W}_1), \mathbf{P}\rangle \le \gamma\Big),
\end{aligned}$$

where the first inequality is due to the property of ramp loss, the second inequality is by the definition of the margin, the third inequality is due to Theorem 5.7, the fourth inequality is due to (5.16), the fifth inequality is by the definition of the margin and the last inequality is due to the property of ramp loss.

Proof of Theorem 5.8

We study the Rademacher complexity of the function class

$$\widehat{\ell}_{\mathcal{F}} := \{(\mathbf{x}, y) \mapsto \widehat{\ell}(f_{\mathbf{W}}(\mathbf{x}), y) : f_{\mathbf{W}} \in \mathcal{F}\}.$$

Define $M_{\mathcal{F}} := \{(\mathbf{x}, y) \mapsto M(f_{\mathbf{W}}(\mathbf{x}), y) : f_{\mathbf{W}} \in \mathcal{F}\}$. Then we have

$$\mathfrak{R}_{\mathcal{S}}(\widehat{\ell}_{\mathcal{F}}) \leq \frac{1}{\gamma}\Big(\mathfrak{R}_{\mathcal{S}}(M_{\mathcal{F}}) + \frac{\epsilon}{2n}\mathbb{E}_{\boldsymbol{\sigma}}\Big[\sup_{f_{\mathbf{W}}\in\mathcal{F}} \sum_{i=1}^{n} \sigma_i \max_{k\in[K], z=\pm 1} \max_{\mathbf{P}\succeq 0, \mathrm{diag}(\mathbf{P})\leq 1} \langle zQ(\mathbf{w}_{2,k}, \mathbf{W}_1), \mathbf{P}\rangle\Big]\Big), \tag{5.17}$$

where we use the Ledoux-Talagrand contraction inequality and the convexity of the supreme operation. For the first term, since we have $\|\mathbf{W}_1\|_1 \leq b_1$, we have $\|\mathbf{W}_1^{\top}\|_{2,1} \leq b_1$. Then, we can apply the Rademacher complexity bound in [16] and obtain

$$\mathfrak{R}_{\mathcal{S}}(M_{\mathcal{F}}) \leq \frac{4}{n^{3/2}} + \frac{60\log(n)\log(2d_{\max})}{n} s_1 s_2 \left((\frac{b_1}{s_1})^{2/3} + (\frac{b_2}{s_2})^{2/3}\right)^{3/2} \|\mathbf{X}\|_F. \tag{5.18}$$

Now consider the second term in (5.17). According to [154], we always have

$$\max_{\mathbf{P}\succeq 0, \mathrm{diag}(\mathbf{P})\leq 1} \langle zQ(\mathbf{w}_{2,k}, \mathbf{W}_1), \mathbf{P}\rangle \geq 0. \tag{5.19}$$

In addition, we know that when $\mathbf{P} \succeq 0$ and $\mathrm{diag}(\mathbf{P}) \leq 1$, we have

$$\|\mathbf{P}\|_{\infty} \leq 1. \tag{5.20}$$

Moreover, we have

$$\|\mathbf{W}_2\|_{\infty} \leq \|\mathbf{W}_2^{\top}\|_{2,1} \leq b_2. \tag{5.21}$$

Then, we obtain

$$\begin{aligned}
&\frac{\epsilon}{2n}\mathbb{E}_{\boldsymbol{\sigma}}\Big[\sup_{f_{\mathbf{W}}\in\mathcal{F}}\sum_{i=1}^{n}\sigma_i \max_{k\in[K],z=\pm 1}\max_{\mathbf{P}\succeq 0,\mathrm{diag}(\mathbf{P})\leq 1}\langle zQ(\mathbf{w}_{2,k},\mathbf{W}_1),\mathbf{P}\rangle\Big] \\
\leq&\frac{\epsilon}{2n}\Big(\sup_{f_{\mathbf{W}}\in\mathcal{F}}\max_{k\in[K],z=\pm 1}\max_{\mathbf{P}\succeq 0,\mathrm{diag}(\mathbf{P})\leq 1}\langle zQ(\mathbf{w}_{2,k},\mathbf{W}_1),\mathbf{P}\rangle\Big)\mathbb{E}_{\boldsymbol{\sigma}}\Big[\big|\sum_{i=1}^{n}\sigma_i\big|\Big] \\
\leq&\frac{\epsilon}{2\sqrt{n}}\sup_{f_{\mathbf{W}}\in\mathcal{F}}\max_{k\in[K],z=\pm 1}\max_{\mathbf{P}\succeq 0,\mathrm{diag}(\mathbf{P})\leq 1}\langle zQ(\mathbf{w}_{2,k},\mathbf{W}_1),\mathbf{P}\rangle \\
\leq&\frac{\epsilon}{2\sqrt{n}}\sup_{f_{\mathbf{W}}\in\mathcal{F}}\max_{k\in[K],z=\pm 1}\max_{\mathbf{P}\succeq 0,\mathrm{diag}(\mathbf{P})\leq 1}\|zQ(\mathbf{w}_{2,k},\mathbf{W}_1)\|_1\|\mathbf{P}\|_\infty \\
\leq&\frac{2\epsilon}{\sqrt{n}}\sup_{f_{\mathbf{W}}\in\mathcal{F}}\max_{k\in[K]}\|\mathrm{diag}(\mathbf{w}_{2,k})^\top\mathbf{W}_1\|_1 \\
\leq&\frac{2\epsilon}{\sqrt{n}}\sup_{f_{\mathbf{W}}\in\mathcal{F}}\|\mathbf{W}_1\|_1\|\mathbf{W}_2\|_\infty \\
\leq&\frac{2\epsilon b_1 b_2}{\sqrt{n}},
\end{aligned} \tag{5.22}$$

where the first inequality is due to (5.19), the second inequality is due to Khintchine's inequality, the third inequality is due to Hölder's inequality, and the fourth inequality is due to the definition of $Q(\cdot,\cdot)$ and (5.20), the fifth inequality is a direct upper bound, and the last inequality is due to (5.21).

Now we can combine (5.18) and (5.22) and get an upper bound for $\mathfrak{R}_{\mathcal{S}}(\widehat{\ell}_{\mathcal{F}})$ in (5.17). Then, Theorem 5.8 is a direct consequence of Theorem 5.1 and Lemma 5.1.

Chapter 6

Conclusions

In this dissertation, we consider several topics on the scalability and robustness of modern machine learning algorithms. We focus on the following specific problems: how to efficiently leverage distributed computing systems to speedup training algorithms, how to make distributed learning algorithms robust to Byzantine failures, and how to understand and overcome the adversarial examples in machine learning.

For the first problem, we study the speedup saturation problem in large batch SGD algorithms. We introduce gradient diversity, a quantity that measures the dissimilarity between concurrent gradient updates, and show its key role in the convergence and generalization performance of mini-batch SGD. We also introduce several diversity-inducing mechanisms, and provide experimental evidence indicating that these mechanisms can enable the use of larger batches without sacrificing the final accuracy, and lead to faster training in distributed learning.

For the second problem, we design statistically and computationally efficient algorithms for Byzantine-robust distributed learning. We show that, combining the distributed gradient descent algorithm with robust estimation subroutines such as median, trimmed mean, and iterative filtering leads to an efficient Byzantine-robust distributed learning algorithm. We establish the statistical rates and iteration complexities of the algorithm with these subroutines and prove their optimality in various regimes, including the high dimensional setting. We also design ByzantinePGD, an efficient robust distributed learning algorithm that can provably escape saddle points and converge to second-order stationary points in the Byzantine setting.

For the third problem, we focus on the adversarially robust generalization properties of linear classifiers and neural networks through the lens of Rademacher complexity. For binary linear classifiers, we establish tight bounds for the adversarial Rademacher complexity, and show that in the adversarial setting, Rademacher complexity is never smaller than that in the natural setting, and it has an unavoidable dimension dependence, unless the weight vector has bounded ℓ_1 norm. The result also extends to multi-class linear classifiers. For neural networks, we prove a lower bound of the Rademacher complexity of the adversarial loss function class and show that there is also an unavoidable dimension dependence due to

ℓ_∞ adversarial attack. We further consider a surrogate adversarial loss and prove margin bound for this setting.

Building scalable and robust machine learning algorithms and systems is a long-standing goal. We demonstrate some progress in this dissertation, and there are many other future directions. Here, we discuss three of them.

First, in this dissertation, the results that we establish for distributed learning are mainly for the data parallelism setting. However, for many modern machine learning applications, especially natural language processing [55], the size of the model is also significantly larger than many traditional models, and the most efficient way to store these models is using distributed devices. However, in the distributed learning research community, model parallelism has not received enough attention compared to data parallelism. We believe that designing scalable, robust and provably efficient algorithms for model parallelism setting is a very important future direction in this area.

Second, we believe that training models that are robust to adversarial perturbations is still a widely open problem. In this dissertation, we discuss the difficulty of adversarially robust generalization. In the literature, there are many other discussions on this topic, including the fundamental trade-offs between robustness and generalization [204] and potential ways to improve adversarial robustness of machine learning [34, 45, 154]. We believe that getting a deeper understanding of black-box adversarial attacks and the relationship between over-parameterization in deep learning models and the robustness to adversarial perturbations are two important problems to study.

Third, studying robustness problems that are more likely to appear in real-world applications, such as common corruptions to images [85] is also a crucial research topic. For these problems, the test examples are not adversarially perturbed, but there exists distribution shift between training and test data. We believe that understanding the generalization properties of machine learning models and algorithms under distributional shift is fundamentally important research direction, and deep theoretical understanding of these problems can provide us with more principled approaches to training robust models in practice.

Bibliography

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. "Tensorflow: a system for large-scale machine learning". In: *USENIX Symposium on Operating Systems Design and Implementation.* Vol. 16. 2016, pp. 265–283.

[2] Alekh Agarwal, Animashree Anandkumar, Prateek Jain, Praneeth Netrapalli, and Rashish Tandon. "Learning sparsely used overcomplete dictionaries". In: *Conference on Learning Theory.* 2014, pp. 123–137.

[3] Naman Agarwal, Zeyuan Allen-Zhu, Brian Bullins, Elad Hazan, and Tengyu Ma. "Finding approximate local minima for nonconvex optimization in linear time". In: *arXiv preprint arXiv:1611.01146* (2016).

[4] Dan Alistarh, Zeyuan Allen-Zhu, and Jerry Li. "Byzantine stochastic gradient descent". In: *arXiv preprint arXiv:1803.08917* (2018).

[5] Dan Alistarh, Jerry Li, Ryota Tomioka, and Milan Vojnovic. "QSGD: Randomized quantization for communication-optimal stochastic gradient descent". In: *arXiv preprint arXiv:1610.02132* (2016).

[6] Zeyuan Allen-Zhu. "Natasha 2: Faster non-convex optimization than SGD". In: *arXiv preprint arXiv:1708.08694* (2017).

[7] Zeyuan Allen-Zhu and Yuanzhi Li. "Neon2: Finding local minima via first-order oracles". In: *arXiv preprint arXiv:1711.06673* (2017).

[8] Noga Alon, Yossi Matias, and Mario Szegedy. "The space complexity of approximating the frequency moments". In: *Journal of Computer and system sciences* 58.1 (1999), pp. 137–147.

[9] Martin Anthony and Peter L Bartlett. *Neural network learning: Theoretical foundations.* Cambridge University Press, 1999.

[10] Sanjeev Arora, Rong Ge, Behnam Neyshabur, and Yi Zhang. "Stronger generalization bounds for deep nets via a compression approach". In: *International Conference on Machine Learning.* 2018, pp. 254–263.

[11] Anish Athalye, Nicholas Carlini, and David Wagner. "Obfuscated gradients give a false sense of security: circumventing defenses to adversarial examples". In: *Proceedings of the 35th International Conference on Machine Learning.* 2018, pp. 274–283.

[12] Idan Attias, Aryeh Kontorovich, and Yishay Mansour. "Improved generalization bounds for robust learning". In: *arXiv preprint arXiv:1810.02180* (2018).

[13] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate". In: *arXiv preprint arXiv:1409.0473* (2014).

[14] Sivaraman Balakrishnan, Martin J. Wainwright, and Bin Yu. "Statistical guarantees for the EM algorithm: From population to sample-based analysis". In: *arXiv preprint arXiv:1408.2156* (2014).

[15] Peter L Bartlett. "The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network". In: *IEEE Transactions on Information Theory* 44.2 (1998), pp. 525–536.

[16] Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. "Spectrally-normalized margin bounds for neural networks". In: *Advances in Neural Information Processing Systems*. 2017.

[17] Peter L Bartlett and Shahar Mendelson. "Rademacher and Gaussian complexities: Risk bounds and structural results". In: *Journal of Machine Learning Research* 3.Nov (2002), pp. 463–482.

[18] Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust optimization*. Princeton University Press, 2009.

[19] Andrew C Berry. "The accuracy of the Gaussian approximation to the sum of independent variates". In: *Transactions of the American Mathematical Society* 49.1 (1941), pp. 122–136.

[20] Dimitri P. Bertsekas and John N. Tsitsiklis. "Gradient convergence in gradient methods with errors". In: *SIAM Journal on Optimization* 10.3 (2000), pp. 627–642.

[21] Kush Bhatia, Prateek Jain, Parameswaran Kamalaruban, and Purushottam Kar. "Consistent robust regression". In: *Advances in Neural Information Processing Systems*. 2017, pp. 2107–2116.

[22] Kush Bhatia, Prateek Jain, and Purushottam Kar. "Robust regression via hard thresholding". In: *Advances in Neural Information Processing Systems*. 2015, pp. 721–729.

[23] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. "Byzantine-tolerant machine learning". In: *arXiv preprint arXiv:1703.02757* (2017).

[24] Léon Bottou, Frank E Curtis, and Jorge Nocedal. "Optimization methods for large-scale machine learning". In: *arXiv preprint arXiv:1606.04838* (2016).

[25] Olivier Bousquet and André Elisseeff. "Stability and generalization". In: *Journal of Machine Learning Research* 2.Mar (2002), pp. 499–526.

[26] Sébastien Bubeck et al. "Convex optimization: Algorithms and complexity". In: *Foundations and Trends® in Machine Learning* 8.3-4 (2015), pp. 231–357.

[27] Sébastien Bubeck, Nicolo Cesa-Bianchi, and Gábor Lugosi. "Bandits with heavy tail". In: *IEEE Transactions on Information Theory* 59.11 (2013), pp. 7711–7717.

[28] Sébastien Bubeck, Eric Price, and Ilya Razenshteyn. "Adversarial examples from computational constraints". In: *arXiv preprint arXiv:1805.10204* (2018).

[29] Emmanuel J Candes, Xiaodong Li, and Mahdi Soltanolkotabi. "Phase retrieval via Wirtinger flow: Theory and algorithms". In: *IEEE Transactions on Information Theory* 61.4 (2015), pp. 1985–2007.

[30] Nicholas Carlini and David Wagner. "Adversarial examples are not easily detected: Bypassing ten detection methods". In: *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. ACM. 2017, pp. 3–14.

[31] Nicholas Carlini and David Wagner. "Audio adversarial examples: Targeted attacks on speech-to-text". In: *arXiv preprint arXiv:1801.01944* (2018).

[32] Nicholas Carlini and David Wagner. "Defensive distillation is not robust to adversarial examples". In: *arXiv preprint arXiv:1607.04311* (2016).

[33] Yair Carmon, John C Duchi, Oliver Hinder, and Aaron Sidford. "Accelerated methods for non-convex optimization". In: *arXiv preprint arXiv:1611.00756* (2016).

[34] Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, Percy Liang, and John C Duchi. "Unlabeled data improves adversarial robustness". In: *arXiv preprint arXiv:1905.13736* (2019).

[35] Moses Charikar, Jacob Steinhardt, and Gregory Valiant. "Learning from untrusted data". In: *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*. ACM. 2017, pp. 47–60.

[36] Niladri Chatterji and Peter L Bartlett. "Alternating minimization for dictionary learning with random initialization". In: *Advances in Neural Information Processing Systems*. 2017, pp. 1994–2003.

[37] Jianmin Chen, Rajat Monga, Samy Bengio, and Rafal Jozefowicz. "Revisiting distributed synchronous SGD". In: *arXiv preprint arXiv:1604.00981* (2016).

[38] Lingjiao Chen, Zachary Charles, Dimitris Papailiopoulos, et al. "DRACO: Robust distributed training via redundant gradients". In: *arXiv preprint arXiv:1803.09877* (2018).

[39] Mengjie Chen, Chao Gao, and Zhao Ren. "Robust covariance matrix estimation via matrix depth". In: *arXiv preprint arXiv:1506.00691* (2015).

[40] Tianqi Chen, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. "Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems". In: *arXiv preprint arXiv:1512.01274* (2015).

[41] Yudong Chen, Lili Su, and Jiaming Xu. "Distributed statistical machine learning in adversarial settings: Byzantine gradient descent". In: *arXiv preprint arXiv:1705.05491* (2017).

[42] Yudong Chen and Martin J Wainwright. "Fast low-rank estimation by projected gradient descent: General statistical and algorithmic guarantees". In: *arXiv preprint arXiv:1509.03025* (2015).

[43] Yuxin Chen, Yuejie Chi, Jianqing Fan, and Cong Ma. "Gradient descent with random initialization: Fast global convergence for nonconvex phase retrieval". In: *arXiv preprint arXiv:1803.07726* (2018).

[44] Trishul Chilimbi, Yutaka Suzue, Johnson Apacible, and Karthik Kalyanaraman. "Project adam: Building an efficient and scalable deep learning training system". In: *11th USENIX Symposium on Operating Systems Design and Implementation*. 2014, pp. 571–582.

[45] Jeremy M Cohen, Elan Rosenfeld, and J Zico Kolter. "Certified adversarial robustness via randomized smoothing". In: *arXiv preprint arXiv:1902.02918* (2019).

[46] Andrew Cotter, Ohad Shamir, Nati Srebro, and Karthik Sridharan. "Better mini-batch algorithms via accelerated gradient methods". In: *Advances in Neural Information Processing Systems*. 2011, pp. 1647–1655.

[47] Daniel Cullina, Arjun Nitin Bhagoji, and Prateek Mittal. "PAC-learning in the presence of evasion adversaries". In: *arXiv preprint arXiv:1806.01471* (2018).

[48] Frank E Curtis, Daniel P Robinson, and Mohammadreza Samadi. "A trust region algorithm with a worst-case iteration complexity of $\epsilon^{-3/2}$ for nonconvex optimization". In: *Mathematical Programming* 162.1-2 (2017), pp. 1–32.

[49] Georgios Damaskinos, El Mahdi El Mhamdi, Rachid Guerraoui, Rhicheek Patra, and Mahsa Taziki. "Asynchronous Byzantine machine learning". In: *arXiv preprint arXiv:1802.07928* (2018).

[50] Hadi Daneshmand, Aurelien Lucchi, and Thomas Hofmann. "Starting small-learning with adaptive sample sizes". In: *International conference on machine learning*. 2016, pp. 1463–1471.

[51] Soham De, Abhay Yadav, David Jacobs, and Tom Goldstein. "Big batch SGD: Automated inference using adaptive batch sizes". In: *arXiv preprint arXiv:1610.05792* (2016).

[52] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Andrew Senior, Paul Tucker, Ke Yang, Quoc V Le, et al. "Large scale distributed deep networks". In: *Advances in Neural Information Processing Systems*. 2012, pp. 1223–1231.

[53] Alexandre Défossez and Francis Bach. "Averaged least-mean-squares: Bias-variance trade-offs and optimal sampling distributions". In: *Artificial Intelligence and Statistics*. 2015, pp. 205–213.

[54] Ofer Dekel, Ran Gilad-Bachrach, Ohad Shamir, and Lin Xiao. "Optimal distributed online prediction using mini-batches". In: *Journal of Machine Learning Research* 13.Jan (2012), pp. 165–202.

[55] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *arXiv preprint arXiv:1810.04805* (2018).

[56] Olivier Devolder, François Glineur, and Yurii Nesterov. "First-order methods of smooth convex optimization with inexact oracle". In: *Mathematical Programming* 146.1-2 (2014), pp. 37–75.

[57] Ilias Diakonikolas, Gautam Kamath, Daniel M Kane, Jerry Li, Ankur Moitra, and Alistair Stewart. "Being robust (in high dimensions) can be practical". In: *arXiv preprint arXiv:1703.00893* (2017).

[58] Ilias Diakonikolas, Gautam Kamath, Daniel M Kane, Jerry Li, Ankur Moitra, and Alistair Stewart. "Robust estimators in high dimensions without the computational intractability". In: *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*. IEEE. 2016, pp. 655–664.

[59] Ilias Diakonikolas, Gautam Kamath, Daniel M Kane, Jerry Li, Jacob Steinhardt, and Alistair Stewart. "Sever: a robust meta-algorithm for stochastic optimization". In: *arXiv preprint arXiv:1803.02815* (2018).

[60] Elvis Dohmatob. "Limitations of adversarial robustness: strong No Free Lunch Theorem". In: *arXiv preprint arXiv:1810.04065* (2018).

[61] Simon S Du, Chi Jin, Jason D Lee, Michael I Jordan, Aarti Singh, and Barnabas Poczos. "Gradient descent can take exponential time to escape saddle points". In: *Advances in Neural Information Processing Systems*. 2017, pp. 1067–1077.

[62] John Duchi, Michael I Jordan, and Brendan McMahan. "Estimation, optimization, and parallelism when data is sparse". In: *Advances in Neural Information Processing Systems*. 2013, pp. 2832–2840.

[63] Logan Engstrom, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. "A rotation and a translation suffice: Fooling CNNs with simple transformations". In: *arXiv preprint arXiv:1712.02779* (2017).

[64] Carl-Gustaf Esseen. *On the Liapounoff limit of error in the theory of probability.* Almqvist & Wiksell, 1942.

[65] Farzan Farnia, Jesse M Zhang, and David Tse. "Generalizable adversarial training via spectral normalization". In: *arXiv preprint arXiv:1811.07457* (2018).

[66] Alhussein Fawzi, Hamza Fawzi, and Omar Fawzi. "Adversarial vulnerability for any classifier". In: *arXiv preprint arXiv:1802.08686* (2018).

[67] Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. "Robustness of classifiers: From adversarial to random noise". In: *Advances in Neural Information Processing Systems*. 2016.

[68] Jiashi Feng, Huan Xu, and Shie Mannor. "Distributed robust learning". In: *arXiv preprint arXiv:1409.5937* (2014).

[69] Michael P Friedlander and Mark Schmidt. "Hybrid deterministic-stochastic methods for data fitting". In: *SIAM Journal on Scientific Computing* 34.3 (2012), A1380–A1405.

[70] Roy Frostig, Rong Ge, Sham M Kakade, and Aaron Sidford. "Competing with the empirical risk minimizer in a single pass". In: *Conference on learning theory*. 2015, pp. 728–763.

[71] Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. "Escaping from saddle points - online stochastic gradient for tensor decomposition". In: *Conference on Learning Theory*. 2015, pp. 797–842.

[72] Rong Ge, Chi Jin, and Yi Zheng. "No spurious local minima in nonconvex low rank problems: A unified geometric analysis". In: *arXiv preprint arXiv:1704.00708* (2017).

[73] Rong Ge, Jason D Lee, and Tengyu Ma. "Matrix completion has no spurious local minimum". In: *Advances in Neural Information Processing Systems*. 2016, pp. 2973–2981.

[74] Rainer Gemulla, Erik Nijkamp, Peter J Haas, and Yannis Sismanis. "Large-scale matrix factorization with distributed stochastic gradient descent". In: *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. 2011, pp. 69–77.

[75] Saeed Ghadimi and Guanghui Lan. "Accelerated gradient methods for nonconvex nonlinear and stochastic programming". In: *Mathematical Programming* 156.1-2 (2016), pp. 59–99.

[76] Justin Gilmer, Ryan P Adams, Ian Goodfellow, David Andersen, and George E Dahl. "Motivating the rules of the game for adversarial example research". In: *arXiv preprint arXiv:1807.06732* (2018).

[77] Justin Gilmer, Luke Metz, Fartash Faghri, Samuel S Schoenholz, Maithra Raghu, Martin Wattenberg, and Ian Goodfellow. "Adversarial spheres". In: *arXiv preprint arXiv:1801.02774* (2018).

[78] Noah Golowich, Alexander Rakhlin, and Ohad Shamir. "Size-independent sample complexity of neural networks". In: *arXiv preprint arXiv:1712.06541* (2017).

[79] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples". In: *arXiv preprint arXiv:1412.6572* (2014).

[80] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. "Accurate, large minibatch SGD: Training ImageNet in 1 hour". In: *arXiv preprint arXiv:1706.02677* (2017).

[81] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. "Speech recognition with deep recurrent neural networks". In: *International Conference on Acoustics, Speech and Signal Processing*. IEEE. 2013.

[82] Shixiang Gu and Luca Rigazio. "Towards deep neural network architectures robust to adversarial examples". In: *arXiv preprint arXiv:1412.5068* (2014).

[83] Moritz Hardt, Benjamin Recht, and Yoram Singer. "Train faster, generalize better: Stability of stochastic gradient descent". In: *arXiv preprint arXiv:1509.01240* (2015).

[84] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 770–778.

[85] Dan Hendrycks and Thomas Dietterich. "Benchmarking neural network robustness to common corruptions and perturbations". In: *arXiv preprint arXiv:1903.12261* (2019).

[86] Elad Hoffer, Itay Hubara, and Daniel Soudry. "Train longer, generalize better: closing the generalization gap in large batch training of neural networks". In: *arXiv preprint arXiv:1705.08741* (2017).

[87] Daniel Hsu and Sivan Sabato. "Loss minimization and parameter estimation with heavy tails". In: *Journal of Machine Learning Research* 17.1 (2016), pp. 543–582.

[88] Ruitong Huang, Bing Xu, Dale Schuurmans, and Csaba Szepesvári. "Learning with a strong adversary". In: *arXiv preprint arXiv:1511.03034* (2015).

[89] Peter J Huber. "Robust estimation of a location parameter". In: *The annals of mathematical statistics* (1964), pp. 73–101.

[90] Peter J Huber. "Robust statistics". In: *International Encyclopedia of Statistical Science*. Springer, 2011, pp. 1248–1251.

[91] Martin Jaggi, Virginia Smith, Martin Takác, Jonathan Terhorst, Sanjay Krishnan, Thomas Hofmann, and Michael I Jordan. "Communication-efficient distributed dual coordinate ascent". In: *Advances in Neural Information Processing Systems*. 2014, pp. 3068–3076.

[92] Prateek Jain, Sham M. Kakade, Rahul Kidambi, Praneeth Netrapalli, and Aaron Sidford. "Parallelizing stochastic gradient descent for least squares regression: Minibatching, averaging, and model misspecification". In: *Journal of Machine Learning Research* 18 (2018), pp. 1–42.

[93] Mark R Jerrum, Leslie G Valiant, and Vijay V Vazirani. "Random generation of combinatorial structures from a uniform distribution". In: *Theoretical Computer Science* 43 (1986), pp. 169–188.

[94] Chi Jin, Rong Ge, Praneeth Netrapalli, Sham M Kakade, and Michael I Jordan. "How to escape saddle points efficiently". In: *arXiv preprint arXiv:1703.00887* (2017).

[95] Chi Jin, Lydia T Liu, Rong Ge, and Michael I Jordan. "Minimizing nonconvex population risk from rough empirical risk". In: *arXiv preprint arXiv:1803.09357* (2018).

[96] Chi Jin, Praneeth Netrapalli, and Michael I Jordan. "Accelerated gradient descent escapes saddle points faster than gradient descent". In: *Conference on Learning Theory.* 2018, pp. 1042–1085.

[97] Thorsten Joachims. "Training linear SVMs in linear time". In: *12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* ACM. 2006, pp. 217–226.

[98] Hamed Karimi, Julie Nutini, and Mark Schmidt. "Linear convergence of gradient and proximal-gradient methods under the Polyak-Lojasiewicz condition". In: *Joint Eur. Conf. on ML and Knowledge Disc. in Databases.* Springer. 2016, pp. 795–811.

[99] Kenji Kawaguchi. "Deep learning without poor local minima". In: *Advances in Neural Information Processing Systems.* 2016, pp. 586–594.

[100] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. "On large-batch training for deep learning: Generalization gap and sharp minima". In: *arXiv preprint arXiv:1609.04836* (2016).

[101] Justin Khim and Po-Ling Loh. "Adversarial risk bounds for binary classification via function transformation". In: *arXiv preprint arXiv:1810.09519* (2018).

[102] Adam Klivans, Pravesh K Kothari, and Raghu Meka. "Efficient algorithms for outlier-robust regression". In: *arXiv preprint arXiv:1803.03241* (2018).

[103] Alexander Kogler and Patrick Traxler. "Efficient and robust median-of-means algorithms for location and regression". In: *Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), 2016 18th International Symposium on.* IEEE. 2016, pp. 206–213.

[104] Vladimir Koltchinskii et al. "Local Rademacher complexities and oracle inequalities in risk minimization". In: *The Annals of Statistics* 34.6 (2006), pp. 2593–2656.

[105] J Zico Kolter and Eric Wong. "Provable defenses against adversarial examples via the convex outer adversarial polytope". In: *arXiv preprint arXiv:1711.00851* (2017).

[106] Jakub Konečný, Jie Liu, Peter Richtárik, and Martin Takáč. "Mini-batch semi - stochastic gradient descent in the proximal setting". In: *IEEE Journal of Selected Topics in Signal Processing* 10.2 (2016), pp. 242–255.

[107] Jakub Konečný, Brendan McMahan, and Daniel Ramage. "Federated optimization: Distributed optimization beyond the datacenter". In: *arXiv preprint arXiv:1511.03575* (2015).

[108] Jakub Konečný, H Brendan McMahan, Daniel Ramage, and Peter Richtárik. "Federated optimization: Distributed machine learning for on-device intelligence". In: *arXiv preprint arXiv:1610.02527* (2016).

[109] Jakub Konečný, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. "Federated learning: Strategies for improving communication efficiency". In: *arXiv preprint arXiv:1610.05492* (2016).

[110] Jernej Kos, Ian Fischer, and Dawn Song. "Adversarial examples for generative models". In: *2018 IEEE Security and Privacy Workshops (SPW)*. IEEE. 2018, pp. 36–42.

[111] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in Neural Information Processing Systems*. 2012, pp. 1097–1105.

[112] Vitaly Kuznetsov, Mehryar Mohri, and U Syed. "Rademacher complexity margin bounds for learning with a large number of classes". In: *ICML Workshop on Extreme Classification: Learning with a Very Large Number of Labels*. 2015.

[113] Kevin A Lai, Anup B Rao, and Santosh Vempala. "Agnostic estimation of mean and covariance". In: *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*. IEEE. 2016, pp. 665–674.

[114] Leslie Lamport, Robert Shostak, and Marshall Pease. "The Byzantine generals problem". In: *ACM Transactions on Programming Languages and Systems (TOPLAS)* 4.3 (1982), pp. 382–401.

[115] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[116] Michel Ledoux and Michel Talagrand. *Probability in Banach Spaces: Isoperimetry and processes*. Springer Science & Business Media, 2013.

[117] Jason D Lee, Qihang Lin, Tengyu Ma, and Tianbao Yang. "Distributed stochastic variance reduced gradient methods and a lower bound for communication complexity". In: *arXiv preprint arXiv:1507.07595* (2015).

[118] Jason D Lee, Ioannis Panageas, Georgios Piliouras, Max Simchowitz, Michael I Jordan, and Benjamin Recht. "First-order methods almost always avoid saddle points". In: *arXiv preprint arXiv:1710.07406* (2017).

[119] Jason D Lee, Max Simchowitz, Michael I Jordan, and Benjamin Recht. "Gradient descent converges to minimizers". In: *arXiv preprint arXiv:1602.04915* (2016).

[120] Kangwook Lee, Maximilian Lam, Ramtin Pedarsani, Dimitris Papailiopoulos, and Kannan Ramchandran. "Speeding up distributed machine learning using codes". In: *IEEE Transactions on Information Theory* 64.3 (2017), pp. 1514–1529.

[121] Wee Sun Lee, Peter L Bartlett, and Robert C Williamson. "Efficient agnostic learning of neural networks with bounded fan-in". In: *IEEE Transactions on Information Theory* 42.6 (1996), pp. 2118–2132.

[122] Matthieu Lerasle and Roberto I Oliveira. "Robust empirical mean estimators". In: *arXiv preprint arXiv:1112.3914* (2011).

[123] Kfir Y Levy. "The power of normalization: Faster evasion of saddle points". In: *arXiv preprint arXiv:1611.04831* (2016).

[124] Jerry Li. "Robust sparse estimation tasks in high dimensions". In: *arXiv preprint arXiv:1702.05860* (2017).

[125] Mu Li, Tong Zhang, Yuqiang Chen, and Alexander J Smola. "Efficient mini-batch training for stochastic optimization". In: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. 2014, pp. 661–670.

[126] Ji Liu, Steve Wright, Christopher Re, Victor Bittorf, and Srikrishna Sridhar. "An asynchronous parallel stochastic coordinate descent algorithm". In: *Proceedings of International Conference on Machine Learning*. 2014, pp. 469–477.

[127] Liu Liu, Yanyao Shen, Tianyang Li, and Constantine Caramanis. "High dimensional robust sparse regression". In: *arXiv preprint arXiv:1805.11643* (2018).

[128] Stanislaw Lojasiewicz. "A topological property of real analytic subsets". In: *Coll. du CNRS, Les ´equations aux d´eriv´ees partielles* (1963), pp. 87–89.

[129] Gabor Lugosi and Shahar Mendelson. "Risk minimization by median-of-means tournaments". In: *arXiv preprint arXiv:1608.00757* (2016).

[130] Gábor Lugosi and Shahar Mendelson. "Sub-Gaussian estimators of the mean of a random vector". In: *arXiv preprint arXiv:1702.00482* (2017).

[131] Nancy A. Lynch. *Distributed Algorithms*. Elsevier, 1996.

[132] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. "Towards deep learning models resistant to adversarial attacks". In: *arXiv preprint arXiv:1706.06083* (2017).

[133] Saeed Mahloujifar, Dimitrios I Diochnos, and Mohammad Mahmoody. "The curse of concentration in robust learning: Evasion and poisoning attacks from concentration of measure". In: *arXiv preprint arXiv:1809.03063* (2018).

[134] Yu Maximov and Daria Reshetova. "Tight risk bounds for multi-class margin classifiers". In: *Pattern Recognition and Image Analysis* 26.4 (2016), pp. 673–680.

[135] Brendan McMahan and Daniel Ramage. *Federated learning: Collaborative machine learning without centralized training data*. `https://research.googleblog.com/2017/04/federated-learning-collaborative.html`. 2017.

[136] H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al. "Communication efficient learning of deep networks from decentralized data". In: *arXiv preprint arXiv:1602.05629* (2016).

[137] Song Mei, Andrea Montanari, and Phan-Minh Nguyen. "A mean field view of the landscape of two-layers neural networks". In: *arXiv preprint arXiv:1804.06561* (2018).

[138] Stanislav Minsker et al. "Geometric median and robust estimation in Banach spaces". In: *Bernoulli* 21.4 (2015), pp. 2308–2335.

[139] Stanislav Minsker and Nate Strawn. "Distributed statistical estimation and rates of convergence in normal approximation". In: *arXiv preprint arXiv:1704.02658* (2017).

[140] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning.* MIT press, 2012.

[141] Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. "Learning with noisy labels". In: *Advances in neural information processing systems.* 2013, pp. 1196–1204.

[142] Deanna Needell and Rachel Ward. "Batched stochastic gradient descent with weighted sampling". In: *arXiv preprint arXiv:1608.07641* (2016).

[143] Arkadii Nemirovskii, David Borisovich Yudin, and Edgar Ronald Dawson. *Problem complexity and method efficiency in optimization.* Wiley, 1983.

[144] Yurii Nesterov. "Introductory lectures on convex programming volume i: Basic course". In: *Lecture notes* (1998).

[145] Yurii Nesterov and Boris T Polyak. "Cubic regularization of Newton method and its global performance". In: *Mathematical Programming* 108.1 (2006), pp. 177–205.

[146] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nathan Srebro. "A PAC-bayesian approach to spectrally-normalized margin bounds for neural networks". In: *arXiv preprint arXiv:1707.09564* (2017).

[147] Atsushi Nitanda. "Stochastic proximal gradient descent with acceleration techniques". In: *Advances in Neural Information Processing Systems.* 2014, pp. 1574–1582.

[148] Feng Niu, Benjamin Recht, Christopher Re, and Stephen Wright. "Hogwild: A lock-free approach to parallelizing stochastic gradient descent". In: *Advances in Neural Information Processing Systems.* 2011, pp. 693–701.

[149] Nicolas Papernot, Patrick McDaniel, Arunesh Sinha, and Michael Wellman. "Towards the science of security and privacy in machine learning". In: *IEEE European Symposium on Security and Privacy.* 2018.

[150] Iosif Pinelis and Raymond Molzon. "Optimal-order bounds on the rate of convergence to normality in the multivariate delta method". In: *Electronic Journal of Statistics* 10.1 (2016), pp. 1001–1063.

[151] Boris Teodorovich Polyak. “Gradient methods for minimizing functionals”. In: *Zhurnal Vychislitel’noi Matematiki i Matematicheskoi Fiziki* 3.4 (1963), pp. 643–653.

[152] Adarsh Prasad, Arun Sai Suggala, Sivaraman Balakrishnan, and Pradeep Ravikumar. “Robust estimation via robust gradient estimation”. In: *stat* 1050 (2018), p. 20.

[153] Hang Qi, Evan R Sparks, and Ameet Talwalkar. “Paleo: A performance model for deep neural networks”. In: (2017).

[154] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. “Certified defenses against adversarial examples”. In: *arXiv preprint arXiv:1801.09344* (2018).

[155] Aditi Raghunathan, Jacob Steinhardt, and Percy S Liang. “Semidefinite relaxations for certifying robustness to adversarial examples”. In: *Advances in Neural Information Processing Systems*. 2018.

[156] Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. “Hogwild: A lock-free approach to parallelizing stochastic gradient descent”. In: *Advances in Neural Information Processing Systems*. 2011, pp. 693–701.

[157] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. “Do ImageNet classifiers generalize to ImageNet?” In: *arXiv preprint arXiv:1902.10811* (2019).

[158] Sashank J Reddi, Jakub Konečný, Peter Richtárik, Barnabás Póczós, and Alex Smola. “AIDE: Fast and communication efficient distributed optimization”. In: *arXiv preprint arXiv:1608.06879* (2016).

[159] Jonathan D Rosenblatt and Boaz Nadler. “On the optimality of averaging in distributed statistical learning”. In: *Information and Inference: A Journal of the IMA* 5.4 (2016), pp. 379–404.

[160] Clément W Royer, Michael O’Neill, and Stephen J Wright. “A Newton-CG algorithm with complexity guarantees for smooth unconstrained optimization”. In: *arXiv preprint arXiv:1803.02924* (2018).

[161] Clément W Royer and Stephen J Wright. “Complexity analysis of second-order line-search algorithms for smooth nonconvex optimization”. In: *SIAM Journal on Optimization* 28.2 (2018), pp. 1448–1477.

[162] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. “Adversarially robust generalization requires more data”. In: *arXiv preprint arXiv:1804.11285* (2018).

[163] Uri Shaham, Yutaro Yamada, and Sahand Negahban. “Understanding adversarial training: Increasing local stability of neural nets through robust optimization”. In: *arXiv preprint arXiv:1511.05432* (2015).

[164] Shai Shalev-Shwartz and Tong Zhang. “Accelerated mini-batch stochastic dual coordinate ascent”. In: *Advances in Neural Information Processing Systems*. 2013, pp. 378–385.

[165] Ohad Shamir, Nati Srebro, and Tong Zhang. “Communication-efficient distributed optimization using an approximate Newton-type method”. In: *International Conference on Machine Learning*. 2014, pp. 1000–1008.

[166] IG Shevtsova. “On the absolute constants in the Berry-Esseen-type inequalities”. In: *Doklady Mathematics*. Vol. 89. 3. Springer. 2014, pp. 378–381.

[167] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. “Mastering the game of Go with deep neural networks and tree search”. In: *Nature* 529.7587 (2016), p. 484.

[168] Aman Sinha, Hongseok Namkoong, and John Duchi. “Certifying some distributional robustness with principled adversarial training”. In: *International Conference on Learning Representations*. 2018.

[169] Daniel Soudry and Yair Carmon. “No bad local minima: Data independent training error guarantees for multilayer neural networks”. In: *arXiv preprint arXiv:1605.08361* (2016).

[170] Jacob Steinhardt, Moses Charikar, and Gregory Valiant. “Resilience: A criterion for learning in the presence of arbitrary outliers”. In: *arXiv preprint arXiv:1703.04940* (2017).

[171] Jacob Steinhardt, Pang Wei W Koh, and Percy S Liang. “Certified defenses for data poisoning attacks”. In: *Advances in neural information processing systems*. 2017, pp. 3517–3529.

[172] Lili Su and Nitin H Vaidya. “Fault-tolerant multi-agent optimization: optimal iterative distributed algorithms”. In: *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing*. ACM. 2016, pp. 425–434.

[173] Lili Su and Nitin H Vaidya. “Non-Bayesian learning in the presence of Byzantine agents”. In: *Proceedings of the International Symposium on Distributed Computing*. Springer. 2016, pp. 414–427.

[174] Lili Su and Jiaming Xu. “Securing distributed machine learning in high dimensions”. In: *arXiv preprint arXiv:1804.10140* (2018).

[175] Arun Sai Suggala, Adarsh Prasad, Vaishnavh Nagarajan, and Pradeep Ravikumar. “On adversarial risk and training”. In: *arXiv preprint arXiv:1806.02924* (2018).

[176] J. Sun, Q. Qu, and J. Wright. “Complete dictionary recovery using nonconvex optimization”. In: *Proceedings of the 32nd International Conference on Machine Learning*. 2015, pp. 2351–2360.

[177] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. “Intriguing properties of neural networks”. In: *arXiv preprint arXiv:1312.6199* (2013).

[178] Martin Takác, Avleen Singh Bijral, Peter Richtárik, and Nati Srebro. "Mini-batch primal and dual methods for SVMs." In: *International Conference on Machine Learning (3)*. 2013, pp. 1022–1030.

[179] Martin Takáč, Peter Richtárik, and Nathan Srebro. "Distributed mini-batch SDCA". In: *preprint arXiv:1507.08322* (2015).

[180] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. "Robustness may be at odds with accuracy". In: *International Conference on Learning Representations*. 2019.

[181] Stephen Tu, Ross Boczar, Max Simchowitz, Mahdi Soltanolkotabi, and Benjamin Recht. "Low-rank solutions of linear matrix equations via procrustes flow". In: *arXiv preprint arXiv:1507.03566* (2015).

[182] Roman Vershynin. "Introduction to the non-asymptotic analysis of random matrices". In: *arXiv preprint arXiv:1011.3027* (2010).

[183] Li Wan, Matthew Zeiler, Sixin Zhang, Yann L Cun, and Rob Fergus. "Regularization of neural networks using dropconnect". In: *Proceedings of the 30th international conference on machine learning (International Conference on Machine Learning)*. 2013, pp. 1058–1066.

[184] Jialei Wang, Weiran Wang, and Nathan Srebro. "Memory and communication efficient distributed stochastic optimization with minibatch prox". In: *arXiv preprint arXiv:1702.06269* (2017).

[185] Shusen Wang, Farbod Roosta-Khorasani, Peng Xu, and Michael W. Mahoney. "GIANT: Globally improved approximate Newton method for distributed optimization". In: *arXiv preprint arXiv:1709.03528* (2017).

[186] Weiran Wang and Nathan Srebro. "Stochastic nonconvex optimization with large minibatches". In: *arXiv preprint arXiv:1709.08728* (2017).

[187] Yizhen Wang, Somesh Jha, and Kamalika Chaudhuri. "Analyzing the robustness of nearest neighbors to adversarial examples". In: *arXiv preprint arXiv:1706.03922* (2017).

[188] Max Welling and Yee W Teh. "Bayesian learning via stochastic gradient Langevin dynamics". In: *Proceedings of International Conference on Machine Learning*. 2011, pp. 681–688.

[189] Wei Wen, Cong Xu, Feng Yan, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. "TernGrad: Ternary gradients to reduce communication in distributed deep learning". In: *arXiv preprint arXiv:1705.07878* (2017).

[190] Eric Wong, Frank Schmidt, Jan Hendrik Metzen, and J Zico Kolter. "Scaling provable adversarial defenses". In: *arXiv preprint arXiv:1805.12514* (2018).

[191] Yihong Wu. *Lecture notes for ECE598YW: Information-theoretic methods for high-dimensional statistics.* `http://www.stat.yale.edu/~yw562/teaching/it-stats.pdf`. 2017.

[192] Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta. “Generalized Byzantine-tolerant SGD”. In: *arXiv preprint arXiv:1802.10116* (2018).

[193] Huan Xu, Constantine Caramanis, and Shie Mannor. “Robust regression and Lasso”. In: *Advances in Neural Information Processing Systems*. 2009.

[194] Huan Xu, Constantine Caramanis, and Shie Mannor. “Robustness and regularization of support vector machines”. In: *Journal of Machine Learning Research* 10.Jul (2009), pp. 1485–1510.

[195] Huan Xu and Shie Mannor. “Robustness and generalization”. In: *Machine learning* 86.3 (2012), pp. 391–423.

[196] Yi Xu and Tianbao Yang. “First-order stochastic algorithms for escaping from saddle points in almost linear time”. In: *arXiv preprint arXiv:1711.01944* (2017).

[197] Dong Yin, Yudong Chen, Kannan Ramchandran, and Peter Bartlett. “Byzantine-robust distributed learning: Towards optimal statistical rates”. In: *Proceedings of the 35th International Conference on Machine Learning*. Vol. 80. PMLR, 2018, pp. 5650–5659.

[198] Dong Yin, Yudong Chen, Kannan Ramchandran, and Peter Bartlett. “Defending against saddle point attack in Byzantine-robust distributed learning”. In: *Proceedings of the 36th International Conference on Machine Learning*. Vol. 97. PMLR, 2019, pp. 7074–7084.

[199] Dong Yin, Ashwin Pananjady, Max Lam, Dimitris Papailiopoulos, Kannan Ramchandran, and Peter Bartlett. “Gradient diversity: A key ingredient for scalable distributed learning”. In: *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*. Vol. 84. PMLR, 2018, pp. 1998–2007.

[200] Dong Yin, Kannan Ramchandran, and Peter Bartlett. “Rademacher complexity for adversarially robust generalization”. In: *Proceedings of the 36th International Conference on Machine Learning*. Vol. 97. PMLR, 2019, pp. 7085–7094.

[201] Hyokun Yun, Hsiang-Fu Yu, Cho-Jui Hsieh, SVN Vishwanathan, and Inderjit Dhillon. “NOMAD: Non-locking, stochastic multi-machine algorithm for asynchronous and decentralized matrix completion”. In: *arXiv:1312.0193* (2013).

[202] Cheng Zhang, Hedvig Kjellstrom, and Stephan Mandt. “Stochastic learning on imbalanced data: determinantal point processes for mini-batch diversification”. In: *arXiv preprint arXiv:1705.00607* (2017).

[203] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. “Understanding deep learning requires rethinking generalization”. In: *arXiv preprint arXiv:1611.03530* (2016).

[204] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P Xing, Laurent El Ghaoui, and Michael I Jordan. "Theoretically principled trade-off between robustness and accuracy". In: *arXiv preprint arXiv:1901.08573* (2019).

[205] Huishuai Zhang, Yuejie Chi, and Yingbin Liang. "Provable non-convex phase retrieval with outliers: Median-truncated wirtinger flow". In: *International Conference on Machine Learning*. 2016, pp. 1022–1031.

[206] Yuchen Zhang, John Duchi, and Martin Wainwright. "Divide and conquer kernel ridge regression: A distributed algorithm with minimax optimal rates". In: *Journal of Machine Learning Research* 16.1 (2015), pp. 3299–3340.

[207] Yuchen Zhang, Jason D Lee, and Michael I Jordan. "ℓ_1-regularized neural networks are improperly learnable in polynomial time". In: *International Conference on Machine Learning*. 2016.

[208] Yuchen Zhang and Xiao Lin. "DiSCO: Distributed optimization for self-concordant empirical loss". In: *International Conference on Machine Learning*. 2015, pp. 362–370.

[209] Yuchen Zhang, Martin J Wainwright, and John C Duchi. "Communication-efficient algorithms for statistical optimization". In: *Advances in Neural Information Processing Systems*. 2012, pp. 1502–1510.

[210] Peilin Zhao and Tong Zhang. "Accelerating minibatch stochastic gradient descent using stratified sampling". In: *arXiv preprint arXiv:1405.3080* (2014).

[211] Tuo Zhao, Zhaoran Wang, and Han Liu. "A nonconvex optimization framework for low rank matrix estimation". In: *Advances in Neural Information Processing Systems*. 2015, pp. 559–567.

[212] Martin Zinkevich, Markus Weimer, Lihong Li, and Alex J Smola. "Parallelized stochastic gradient descent". In: *Advances in Neural Information Processing Systems*. 2010, pp. 2595–2603.