IMPROVING COMMUNICATION EFFICIENCY AND CONVERGENCE IN FEDERATED LEARNING

IMPROVING COMMUNICATION EFFICIENCY AND CONVERGENCE IN FEDERATED LEARNING

BY YANGYI LIU, M.A.Sc.

A THESIS
SUBMITTED TO THE ELECTRICAL AND COMPUTER ENGINEERING
AND THE SCHOOL OF GRADUATE STUDIES
OF MCMASTER UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

McMaster University

DOCTOR OF PHILOSOPHY (2024)

Hamilton, Ontario, Canada (Electrical and Computer Engineering)

TITLE:	IMPROVING COMMUNICATION EFFICIENCY AND CONVERGENCE IN FEDERATED LEARNING
AUTHOR:	Yangyi Liu M.A.Sc. (Electrical and Computer Engineering), Georgia Institute of Technology, Atlanta, USA
SUPERVISOR:	Dr. Jun Chen
NUMBER OF PAGES:	xvii, 136

Lay Abstract

Federated Learning is a machine learning framework that allows remote clients to collaboratively train a model without raw data exchange, which ensures local data privacy. It differs from traditional machine learning scenarios where data needs to be stored centrally. This decentralized framework is advantageous in several respects including: data security, data diversity, real-time continual learning and hardware efficiency. However, the demand for frequent communication between clients and the server imposes tremendous communication challenges in applying Federated Learning to real-world scenarios. This thesis aims to tackle the problems in FL by theoretically characterizing the problem and developing practical methodologies. The theoretical results allow for systematic analysis of the communication cost and convergence rate. The experimental results validate the effectiveness of the proposed methods in improving communication efficiency and convergence in Federated Learning.

Abstract

Federated learning is an emerging field that has received tremendous attention as it enables training Deep Neural Networks in a distributed fashion. By keeping the data decentralized, Federated Learning enhances data privacy and security while maintaining the ability to train robust machine learning models. Unfortunately, despite these advantages, the communication overhead resulting from the demand for frequent communication between the central server and remote clients poses a serious challenge to the present-day communication infrastructure. As the size of the deep learning models and the number of devices participating in the training are ever increasing, the model gradient transmission between the remote clients and the central server orchestrating the training process becomes the critical performance bottleneck. In this thesis, we investigate and address the problems related to improving the communication efficiency while maintaining convergence speed and accuracy in Federated Learning.

To characterize the trade-off between communication cost and convergence in Federated Learning, an innovative formulation utilizing the clients' correlation is proposed, which considers gradient transmission and reconstruction problems as a multi-terminal source coding problem. Leveraging this formulation, the model update problem in Federated Learning is converted to a convex optimization problem

from a rate-distortion perspective. Technical results, including an iterative algorithm to solve for the upper bound and lower bound of the sum-rate, as well as the rate allocation schemes, are provided. Additionally, a correlation-aware client selection strategy is proposed and evaluated against the state-of-the-art methods. Extensive simulations are conducted to validate our theoretical analysis and the effectiveness of the proposed approaches.

Furthermore, based on the statistical insights about the model gradient, we propose a gradient compression algorithm also inspired by rate-distortion theory. More specifically, the proposed algorithm adopts model-wise sparsification for preliminary gradient dimension reduction and then performs layer-wise gradient quantization for further compression. The experimental results show that our approach achieves compression as aggressive as 1-bit while maintaining proper model convergence speed and final accuracy.

To my family and my wife

Acknowledgements

To begin, I would like to express my deepest gratitude to my supervisor, Dr. Jun Chen, for his unwavering support, invaluable guidance, and profound expertise that helped me finish my Ph.D. study. His commitment to excellence and dedication to his students are truly inspiring, and I feel incredibly fortunate to have had the opportunity to learn under his mentorship.

I would like to thank my committee members Dr. Sorina Dumitrescu, Dr. Ratnasingham Tharmarasa, Dr. Dongmei Zhao and Dr. Tim Davidson, as well as Ms. Cheryl Gies, for giving me guidance and support over the years. I would also like to thank Dr. Yongyi Mao for being my external examiner.

My gratitude goes to my friends and fellow collaborators, especially to Huan, Stefano, Siyao, Jingjing, Liangyan, Zewei, Ruibin and Yunzhe for supporting me in both study and life. I appreciate our friendship and believe it will last even beyond this journey.

Last but not least, I would like to thank my family for their unwavering support and love. To my loving wife, I could not complete this journey without you. To them I dedicate this thesis.

Table of Contents

Lay Abstract **iii**

Abstract **iv**

Acknowledgements **vii**

Abbreviations and Notation **xvi**

Declaration of Academic Achievement **xviii**

1 Introduction **1**

1.1 Federated Learning 1

1.2 Rate-Distortion Theory 4

1.3 Multi-terminal Source Coding in FL 6

1.4 Generalization Error in FL 9

1.5 Contributions and Thesis Organization 11

2 GCFed: Exploiting Clients Correlation in Federated Learning to Improve Communication Efficiency and Convergence **13**

2.1 Abstract 13

2.2 Introduction 14
2.3 Related Work 17
2.4 Problem Formulation and Technical Results 20
2.5 Proposed Methods 32
2.6 Experiments 37
2.7 Conclusion 49

3 M22: A Communication-Efficient Algorithm for Federated Learning Inspired by Rate-Distortion 50

3.1 Abstract 50
3.2 Introduction 51
3.3 System Model 56
3.4 A Rate-distortion Approach to DNN Gradient Compression 64
3.5 M22: the Proposed Approach 73
3.6 Numerical Evaluations 79
3.7 Conclusion 92

4 Conclusion and Future Work 95

4.1 Conclusion 95
4.2 Future Work 96

A Supplement of GCFed: Exploiting Clients Correlation in Federated Learning to Improve Communication Efficiency and Convergence 99

A.1 Detailed Introduction About the Variables in Upper and Lower Bounds 99
A.2 Algorithm to Solve the Optimization Problem 101
A.3 Generalizations of the Problem Formulation and Results 108

A.4 Client Selection Strategy Details 111
A.5 Rate Allocation and Quantization Approximation 113
A.6 An Example of the Rate-Distortion Curve from Our Correlation Estimation 117
A.7 Experiment Setting Details 117

B Supplement of M22: A Communication-Efficient Algorithm for Federated Learning Inspired by Rate-Distortion 120

List of Figures

1.1 Typical rate-distortion trade-off curve. 5

1.2 Lossy compression basic model. 6

1.3 The direct multi-terminal source coding problem with K terminals. . 8

1.4 Multi-terminal source coding problem with K agents in FL. 9

2.1 The Gaussian multi-terminal source coding problem in FL as introduced in Sec. 2.4.6. See Remark 1 for notations. 26

2.2 Example dominant faces (colored) that describe the optimal rate allocation schemes for $K = 2$. 31

2.3 Example dominant faces (colored) that describe the optimal rate allocation schemes for $K = 3$. 32

2.4 Top: under the same distortion constraint, the rate gain in terms of number of bits through the converging process; Bottom: under the same rate constraint, the distortion gain in terms of percentage through the converging process. 40

2.5 Convergence comparison between our client selection strategy and the SOTA. Each figure represents one heterogeneous setting. All experiments in one figure share the same hyperparameters except for the client selection strategy. 43

2.6 Ablation study on layer-wise client selection implementations. 45
2.7 The achievable min sum-rate VS. the number of subsamples on the gradient layer with 16000 elements. 47
2.8 The achievable min sum-rate VS. the number of subsamples on the gradient layer with 500 elements. 48
3.1 The positive region of the scalar quantizer described in Section 3.4.3 for the GenNorm distribution. Each row corresponds to a different value of M with M increasing with the row index. Note that the quantizer is symmetric around zero. Also, the first quantization region corresponds to the positive half of the sparsification region. 72
3.2 Comparison across different gradient compression approaches under four different settings: (i) row-wise: 2 or 5 remote users, atd (ii) column-wise: $d\mathsf{R}$ = 332k bits or $d\mathsf{R}$ = 996k bits of total communication rate. 84
3.3 Ablation study of the effect of M value. 10-round accuracy comparison in 2 remote clients setting (top left); first 4-round accuracy comparison in 2 remote clients setting (top right); 10-round accuracy comparison in 5 remote clients setting (bottom left); first 5-round accuracy comparison in 5 remote clients setting (bottom right). 86
3.4 Compare the 3 non-uniform compression algorithms on ResNet18 (left); Compare the scheme where no quantization is applied with our M22 algorithm under different communication overhead constraints on VGG16 (right). 88

3.5 Test the M22 algorithm in the case where the total communication overhead is fixed, while individual users are under different communication constraints with distinct compressors employed. 90

3.6 In a heterogeneous local data setting using FMNIST dataset for various compression rates. Each line corresponds to a heterogeneity level: $\epsilon = \{0.8, 0.5, 0.2\}$. 93

A.1 Example Rate Distortion curve. Upper bound is described in (2.4.12) and lower bound is described in (2.4.13) 117

List of Tables

2.1 Test accuracy comparison to SOTA methods in six heterogeneous settings on CIFAR-10. 41

2.2 Test accuracy comparison to SOTA methods in six heterogeneous settings on Fashion-MNIST. 42

2.3 Ablation study of layer-wise client selection on CIFAR-10. 44

2.4 Ablation study of layer-wise client selection on Fashion-MNIST. . . . 44

2.5 Test accuracy resulting from integrating our client selection and quantization approaches. 46

3.1 A summary of the parameter information of the models in Section 3.3.4. 62

3.2 Parameters and hyperparameters used for the training of the DNN models. 62

3.3 Examples of the sample gradient distribution and its fitting as discussed in Section 3.4.1 for CNN in Appendix B. 67

3.4 Experiment parameters information of methods in Section 3.6.1 with two remote clients. 83

A.1 CNN model information for Fashion-MNIST dataset. 119

B.1 CNN model information. 121

B.2 ResNet18 model information. 122

B.3 VGG16 model information. 123

Abbreviations and Notation

Abbreviations

AI	Artificial intelligence
CNN	Convolutional Neural Network
DNN	Deep Neural Network
ML	Machine Learning
MSE	Mean Squared Error
MMSE	Minimum Mean Square Error
FL	Federated Learning
CEO	Chief Executive Officer
SGD	Stochastic Gradient Decent
SOTA	State of the Art
UB	Upper Bound

LB	Lower Bound
RV	Random Variable

Notation

$\mathbb{E}[\,\cdot\,]$	Expectation operator
$(\cdot)^T$	Transpose operator
$\mathbf{tr}(\cdot)$	Trace operator
$\mathbf{diag}(x_1, ..., x_L)$	$L \times L$ diagonal matrix with diagonal entries $x_1, ..., x_L$
$\lvert\,\cdot\,\rvert$	Determinant operator
$\mathbb{1}$	All ones vector
$\mathbf{0}$	All zeros vector
$\lVert\,\cdot\,\rVert_2$	2-norm
$\lVert\,\cdot\,\rVert_p$	p-norm
$\lvert\mathcal{X}\rvert$	Cardinality of a set $\mathcal{X}$
Σ_X	Covariance matrix of X
$I(X;Y)$	Mutual information btween X and Y
$H(\cdot)$	Entropy
$H(\cdot\lvert\cdot)$	Conditional Entropy

Declaration of Academic Achievement

1. Y. Liu, S. Rini, and J. Chen. "Exploiting clients correlation benefits communication efficiency and convergence in FL", to be submitted to AAAI 2025.

2. Y. Liu, S. Rini, S. Salehkalaibar and J. Chen. "M22: A Communication-Efficient Algorithm for Federated Learning Inspired by Rate-Distortion," in IEEE Transactions on Communications, vol. 72, no. 2, pp. 845-860, Feb. 2024, doi: 10.1109/TCOMM.2023.3327778.

3. Y. Liu, S. Salehkalaibar, S. Rini and J. Chen. "M22: Rate-Distortion Inspired Gradient Compression," ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Rhodes Island, Greece, 2023, pp. 1-5, doi: 10.1109/ICASSP49357.2023.10097231.

4. Y. Liu, H. Liu, L. Li, Z. Wu, and J. Chen. "A data-centric solution to nonhomogeneous dehazing via vision transformer". In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2023.

5. H. Zhou, W. Dong, Y. Liu, and J. Chen. "Breaking through the haze: An

advanced non-homogeneous dehazing method based on fast fourier convolution and convnext". In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2023.

6. J. He, C. Wang, D. Jiang, Z. Li, Y. Liu and T. Zhang. "CycleGAN With an Improved Loss Function for Cell Detection Using Partly Labeled Images," in IEEE Journal of Biomedical and Health Informatics, vol. 24, no. 9, pp. 2473-2480, Sept. 2020, doi: 10.1109/JBHI.2020.2970091.

Chapter 1

Introduction

1.1 Federated Learning

Federated Learning, as a decentralized machine learning method, has attracted considerable attention in recent years. Under this framework, multiple remote clients collaboratively train machine learning models without exchanging their raw data. This approach offers several advantages, including reduced data collection costs, lowered risks associated with data breaches, and ensured compliance with data protection regulations. However, FL faces several challenges, such as data heterogeneity, system heterogeneity, and privacy-preserving techniques, which complicate the training process. Among these challenges, communication efficiency receives significant attention because the FL framework inevitably demands frequent data exchanges between the clients and the central server. Therefore, a primary focus of investigation is reducing communication costs while maintaining satisfactory convergence.

Following is a brief description of the FL framework. Given a system with K clients, each having kn data samples, the federated learning framework aims to solve:

$$\min_{\mathbf{w}\in R^m} \mathbb{E}_{k\sim\mathcal{P}}[F_k(\mathbf{w})], \tag{1.1.1}$$

, where $\mathbf{w}$ denotes the weight of the model of dimensionality, m. Specifically, $F_k(\mathbf{w}) = \mathbb{E}_{\mathbf{d}_{kn}\sim\mathcal{D}_k}[f_k(\mathbf{w};\mathbf{d}_{kn})]$ denotes the loss function of client k and $\mathcal{P}$ is the distribution over the remote clients. At the start of each iteration $t = 1, 2, ...$, the server will broadcast the current global model $\mathbf{w}_t$ to all the clients. After receiving the model, each remote client trains the model using their local data based on $\mathbf{w}_t$, a process referred to as a local update. Each client will generate a unique update because of the uniqueness of their local data. When the local update is finished, clients send back the unique update, also known as the local gradients, $\mathrm{g}(t)$ to the server. The server aggregates all the local gradients, to form a new global model $\mathbf{w}_{t+1}$ and then broadcasts it to all the clients in the next iteration. This broadcast and aggregation process is repeated continuously until the model converges.

One of the key obstacles in this approach is the communication channel capacity between remote clients and the server. Whether due to a noisy communication channel or corrupted source, the server is unable to receive the true local gradient $\mathrm{g}(t)$, thus cannot directly compute the optimal global model $\mathbf{w}_{t+1}$. Therefore, it is of fundamental interest to investigate methods that estimate based on noisy local gradients $\tilde{\mathrm{g}}(t)$ and form a global model $\hat{\mathbf{w}}_{t+1}$ as close as possible to the optimal global model $\mathbf{w}_{t+1}$.

Significant efforts have been made in two key areas: theoretically characterizing the relationship between communication efficiency and model convergence, and proposing practical methodologies to reduce communication overhead in real-world FL scenarios. Reducing the total communication cost in FL can be divided into two

parts [86]: (i) reducing the number of communication iterations between the server and clients, and (ii) lowering the communication cost at each iteration. Most theoretical analyses couple these two aspects together to study the convergence rate of FL versus the total communication cost [51, 66, 50]. Consequently, the convergence rate is parameterized by various indices and hyperparameters in FL settings. Although these results provide valuable insights for researchers to develop new FL algorithms, such as client selection strategies [23, 68] and aggregation methods [29, 66], they usually fail to address the reduction of communication costs per iteration due to their problem formulations. To account for ii), [86, 82] propose modeling the single iteration communication problem using source coding techniques from information theory, formulating the FL problem with rate-distortion functions. However, their results are limited to scenarios where the local data of all clients is uniformly distributed, which does not accurately reflect real-world conditions. In this paper, we aim to study the rate-distortion function and derive the corresponding rate allocation schemes for heterogeneous data settings.

Rather than being driven by theoretical analysis, some works leverage statistical insights from the transmitted data to develop methodologies. Data compression techniques, such as sparsification and quantization, are effective in reducing the dimensionality of client local gradients, making them widely adopted in designing practical FL compression solutions. However, the absence of a well-defined problem formulation often makes it difficult to theoretically justify these methods.

Additionally, most existing works neglect to account for the correlation between clients. For example, in the compression literature, the same compressor [19, 76, 4] is applied across different clients, resulting in suboptimal rate allocation. Although some

works address this issue by proposing to exploit correlation in terms of clients' local losses [51, 50], none have addressed modeling the clients' local gradients as random variables and considering the correlation in terms of clients' local gradient layer-wise. In light of this, we adopt the direct multi-terminal source coding formulation [73, 72] and aim to propose a correlation-aware solution that strategically selects participating clients and achieves optimal rate allocation.

1.2 Rate-Distortion Theory

To study the compression and reconstruction of gradients being transmitted in FL, lossy compression algorithms are typically investigated within the framework of rate-distortion theory, which is a key area of information theory. In the case of lossy compression, in contrast to its lossless counterpart, the reconstruction is not exactly the same as the original source, but only an approximation of it. Therefore, some distortion measurements need to be defined to evaluate the difference between the original source and the reconstruction. When evaluating the lossy compression performance, rate, often characterized by the number of bits per symbol, is the other key factor. It is acknowledged and proven that there exists a trade-off between the rate and distortion, meaning more rate is required for less distortion and vice versa. Rate-distortion theory [15] theoretically characterizes this trade-off by formulating it as the minimal rate that should be communicated over a channel so that the source can be reconstructed at the receiver without exceeding an expected distortion. A typical rate-distortion trade-off curve is shown in Figure 1.1.

The basic model is depicted in Figure 1.2, where X is the original source and $\hat{X}$ represents the reconstruction the receiver approximates. And $R(d)$ represents the

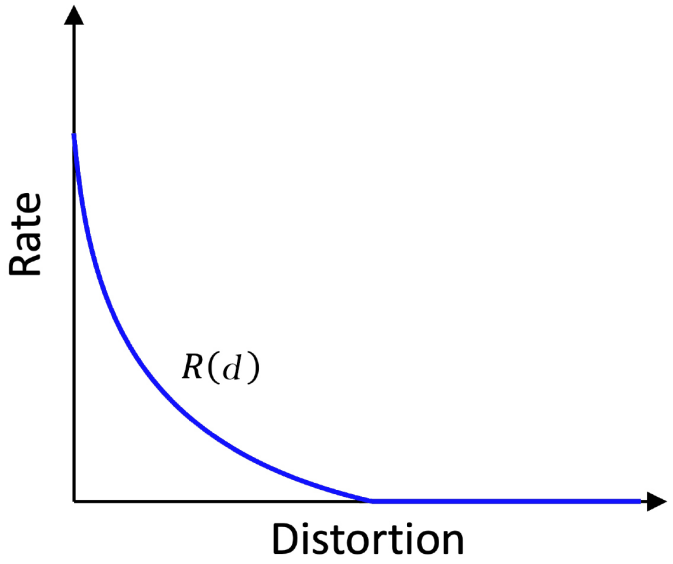

Figure 1.1: Typical rate-distortion trade-off curve.

rate that achieves a certain distortion level, d. Defining the distortion measurement using Mean Square Error, the rate-distortion function (trade-off) is as follows:

$$\begin{aligned} R(d) = \inf_{P_{\hat{X}|X}} \ & I(X;\hat{X}) \\ \text{s.t.} \quad & \mathbb{E}[(X-\hat{X})^2] \leq d \end{aligned} \tag{1.2.1}$$

Incorporating this rate-distortion function into FL helps us perform theoretical analysis but requires some modification of the original formulation. The source is no longer a single random variable. Instead, in FL, $X_1^n, X_2^n ..., X_K^n$ represent the local gradients (or weights) sequence obtained by K remote clients. Generally speaking, given a predefined aggregation function $f(\cdot)$, which takes the local gradients as input and outputs the global gradients to update the global model, some works aim to reconstruct the sources,with the reconstructions denoted as $\hat{X}_1^n, \hat{X}_2^n ..., \hat{X}_K^n$, such that the distortion between $f(X_1^n, X_2^n ..., X_K^n)$ and $f(\hat{X}_1^n, \hat{X}_2^n ..., \hat{X}_K^n)$ is minimized. In

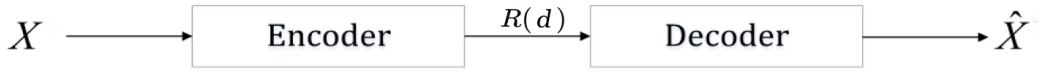

Figure 1.2: Lossy compression basic model.

contrast, most works in FL do not reconstruct each individual source. Instead, they directly target reconstructing $f(X_1^n, X_2^n ..., X_K^n)$ at the server side of FL. By incorporating some rate constraints into this reconstruction process, it becomes natural to treat FL within the framework of rate-distortion theory.

In this thesis, rate distortion theory is applied in both theoretical problem formulation and implementation. We formulate the FL model update problem using a sum-rate distortion function, and derive the upper bound and lower bound to the minimum sum-rate by adopting certain assumptions. In implementation, we employ quantizer designs following the rule that minimizes a chosen distortion under a certain rate constraint. In both theoretical analysis and practical application, rate-distortion theory serves the role of a solid foundation.

1.3 Multi-terminal Source Coding in FL

The work of Slepian and Wolf [62] builds the foundations for studying the multi-terminal source coding problem, where in [62] they consider two correlated sources and determine the minimum number of bits needed to accomplish lossless encoding and decoding. Wyner and Ziv [77, 78] extend this work by establishing information-theoretic bounds for lossy compression when side information is available at the decoder side. Later, Berger and Tung [8, 70] introduce the multi-terminal source coding

problem and investigate an inner bound and an outer bound of the region of all possible encoder rate tuples such that the sources are reconstructed under distortion constraints. Considering the multi-terminal source coding problem in the quadratic Gaussian case, Oohama [47] derives the rate-distortion region by relaxing the distortion constraint to be imposed on only one of the two sources. Together with the Berger-Tung inner bound obtained in [8, 70], the results provide a partial characterization of the boundary of the rate region of the quadratic Gaussian source coding problem in the two terminal case. Extending from $L = 2$ to more terminals, Wang, Chen and et al. [75, 72, 73] derive the upper and lower bounds on the minimum sum rate of the scalar and vector Gaussian multi-terminal source coding problem, which provides theoretical foundations of analyzing distributed learning from an information theory rate-distortion perspective.

Generally speaking, direct multi-terminal source coding, which will be referred to simply as multi-terminal source coding, considers the problem of separate compression and joint decompression of multiple correlated sources. The scheme with K separate encoders is depicted in Figure 1.3.

[85] and [1] are the pioneering works that link the multi-terminal source coding problem with Federated Learning. They propose that FL can be viewed as a variant of the multi-terminal source coding problem, where the goal is to reconstruct an aggregation of the sources rather than reconstructing each source individually in the traditional formulation. They adopt relatively strong assumptions on the local gradients to be a noisy version of the true optimal gradients corrupted by independent additive white Gaussian random noise (AWGN). They propose to reconstruct the true optimal gradients. Such assumptions categorize their problem formulations as

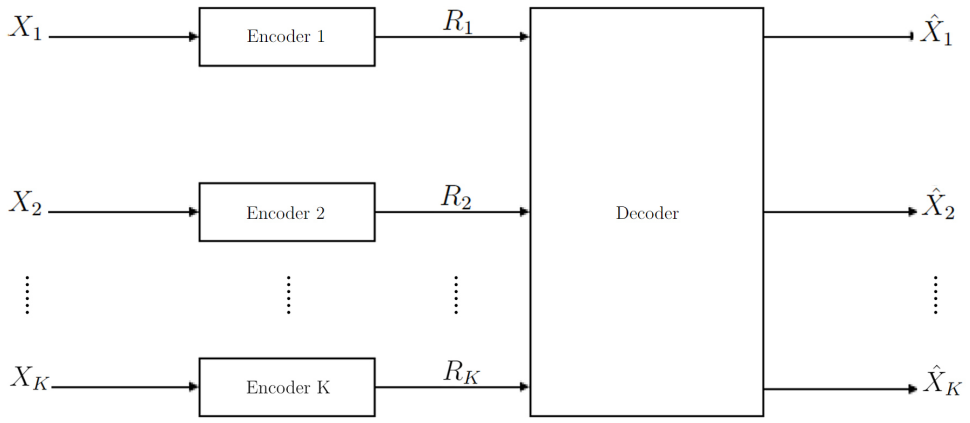

Figure 1.3: The direct multi-terminal source coding problem with K terminals.

another branch of multi-terminal source coding, referred to as indirect multi-terminal source coding, also known as the CEO problem [9]. Within this framework, [85] conducts convergence analysis and characterizes the rate region, but their results are only applicable for the i.i.d. local data case (the local datasets of each client follow the same Gaussian distribution). In contrast, we formulate the source coding problem in FL as the direct multi-terminal source coding problem. We present the basic scheme in Figure 1.4. We assume that each client obtains a random Gaussian sequence, which is their local gradients. Furthermore, we assume that within each sequence, the gradients are generated through an i.i.d. process. The correlation is assumed to exist across different sequences at the same position. The goal of the server is to reconstruct a sequence of the same length as the local gradients, such that the aggregation distortion is minimized.

In the later sections, we will detailedly introduce our problem formulation with the more relaxed assumption that we only require the local gradients to be jointly Gaussian distributed. Therefore, all our theoretical analysis is applicable to the real-world

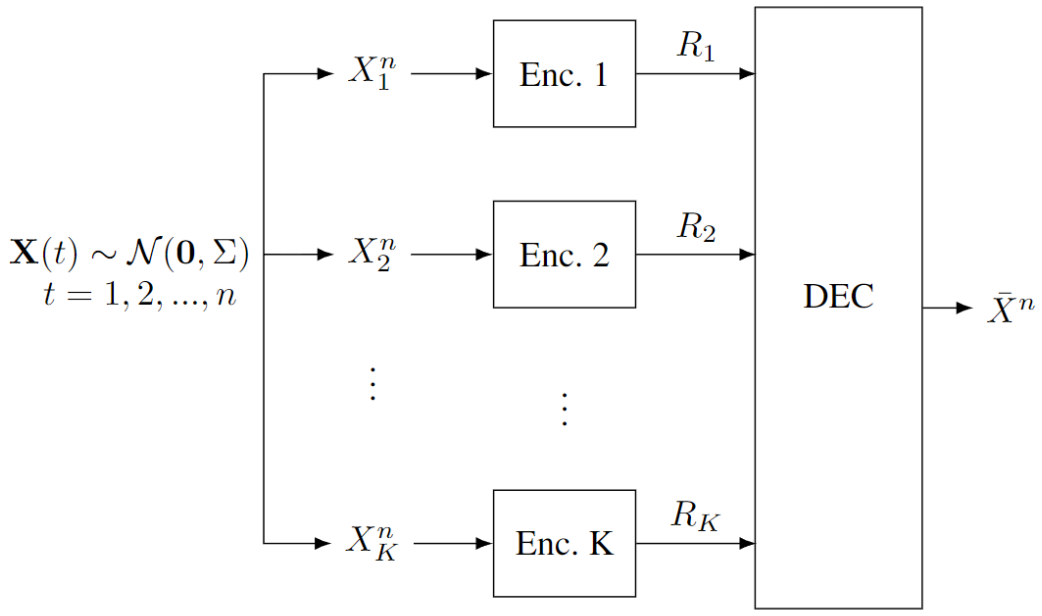

Figure 1.4: Multi-terminal source coding problem with K agents in FL.

FL scenario, where the local data is heterogeneously distributed. The experimental results validate our theory.

1.4 Generalization Error in FL

In this thesis, our primary focus is improving the communication efficiency and the convergence in the training part of FL. However, to complete the deployment of an FL algorithm, the trained model needs to be evaluated on unforeseen test data. The difference between i) the average loss (or error) of the model on the training data (also known as the model's empirical risk) and ii) the expected loss of the model across the entire distribution of possible data points, not just the training set (also known as the model's population risks) is referred to as the generalized error. Although our

problem formulations and technical solutions target on reducing the empirical risk, we provide experiment results of applying the proposed methods on test data, which demonstrates the model's population risks. Thus, we present a brief literature review on the generalization error in FL, which is one of our future work directions.

The study of the generalization error is essentially a difficult topic. The problem in the data-centralized learning settings is still poorly understood, let along in the distributed and multi-round cases like FL. Targeting on the multi-round property of FL, [63, 24] show that in general, smaller empirical risk is obtained with multi-round communication. For one-round communication, [7] establish a bound on the generalized error, for linear and location models and losses. Building upon this work, [55, 81, 14] apply information-theoretic and rate-distortion theoretic approaches, and provide tighter bound to a broader broader class of loss functions. However, these work neglect the study of how the generalization error of FL-type algorithms evolves with the number of rounds R. [56] and [21] are two outstanding recent works studying this relationship. [56] study the upper bound on the generalized error in FL, and propose that the population risk decays less rapidly with R than the empirical risk. They suggest that in practice, one may find the trade off between R and the risks, meaning larger R values help reduce the empirical risk, while smaller R values make the population risk smaller. A concurrent work by [21] support this advocacy that less communication leads to better generalization performance for FL. The authors of [21] demonstrate that less frequent aggregations, hence more local updates, for the representation extractor (often corresponds to initial layers) leads to the creation of models with better generalization capability, particularly for heterogeneous FL settings.

Since the generalization error study is so important to quantitatively assess the generalization characteristics of trained models, one of our future work is to integrate it with our proposed methodologies. In this way, we may provide reliable assurances regarding their expected performance quality and to develop new models and systems.

1.5 Contributions and Thesis Organization

The thesis consists of three published/unpublished articles that address the communication efficiency and convergence problems in Federated Learning. The contributions are listed in the abstract sections of Chapter 2 and Chapter 3. Here is the summarized reference information:

- Y. Liu, S. Rini, and J. Chen. 2023. "Exploiting clients correlation benefits communication efficiency and convergence in FL", to be submitted to AAAI 2025.

- Y. Liu, S. Rini, S. Salehkalaibar and J. Chen, "M22: A Communication-Efficient Algorithm for Federated Learning Inspired by Rate-Distortion," in IEEE Transactions on Communications, vol. 72, no. 2, pp. 845-860, Feb. 2024, doi: 10.1109/TCOMM.2023.3327778.

- Y. Liu, S. Salehkalaibar, S. Rini and J. Chen, "M22: Rate-Distortion Inspired Gradient Compression," ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Rhodes Island, Greece, 2023, pp. 1-5, doi: 10.1109/ICASSP49357.2023.10097231.

The rest of the thesis is organized as follows:

- **Chapter 2** introduces GCFed: Exploiting Clients Correlation in Federated Learning to Improve Communication Efficiency and Convergence.
- **Chapter 3** introduces M22: A Communication-Efficient Algorithm for Federated Learning Inspired by Rate-Distortion Theory.
- **Chapter 4** provides the conclusion of this thesis and the discussion of future works.

Chapter 2

GCFed: Exploiting Clients Correlation in Federated Learning to Improve Communication Efficiency and Convergence

2.1 Abstract

Federated Learning has gained considerable attention as it allows for the utilization of diverse data from various sources, enhancing model robustness and generalizability. Extensive research has focused on reducing communication costs and improving convergence. However, existing works fail to comprehensively leverage client correlation, which we fully exploit in this work. By adopting a multi-terminal source coding formulation, we propose GCFed (Gradient Correlation- aware Federated Learning) – an

FL framework that exploits gradient correlation to achieve optimal rate allocation and client selection. Specifically, we formulate the model update in Federated Learning as a multi-terminal source coding problem in a rate-distortion perspective. Leveraging the gradient correlation, the formulations are further converted to convex optimization problems that calculate the upper and lower bounds of the sum-rate-distortion function, which intrinsically determine the optimal rate for each client. Furthermore, we propose a layer-wise client selection strategy that is seamlessly guided by the estimated correlation. We conduct extensive experiments to demonstrate the effectiveness of the proposed approaches, both independently and jointly. The results validate the superior performance of our framework compared to state-of-the-art methods.

2.2 Introduction

Among many distributed deep learning paradigms, Federated Learning (FL) has gained considerable attention in the research community due to its ability to train models collaboratively without sharing raw data, thereby enhancing data privacy and security. This framework allows for the utilization of diverse data from various sources, improving model robustness and generalizability. However, FL faces several challenges, including data heterogeneity, system heterogeneity, and the implementation of privacy-preserving techniques, all of which complicate the training process. Among these challenges, communication efficiency receives significant attention because the FL framework requires frequent data exchange between remote clients and the central server. Therefore, a primary focus of investigation is reducing communication costs while maintaining satisfactory convergence.

Significant efforts have been made in two key areas: theoretically characterizing the relationship between communication efficiency and model convergence, and proposing practical methodologies to reduce communication overhead in real-world FL scenarios. Reducing the total communication cost in FL can be divided into two parts [86]: (i) reducing the number of communication iterations between the server and clients, and (ii) lowering the communication cost at each iteration. Most theoretical analyses couple these two aspects together to study the convergence rate of FL versus the total communication cost [51, 66, 50]. Consequently, the convergence rate is parameterized by various indices and hyperparameters in FL settings. Although these results provide valuable insights for researchers to develop new FL algorithms, such as client selection strategies [23, 68] and aggregation methods [29, 66], they often fail to address the reduction of communication costs per iteration due to their problem formulations. To account for (ii), [86, 82] propose modeling the single iteration communication problem using source coding techniques from information theory, formulating the FL problem with rate-distortion functions. However, their results are limited to scenarios where the local data of all clients are uniformly distributed, which does not accurately reflect real-world conditions. In this paper, we aim to study the rate-distortion function and derive the corresponding rate allocation schemes for heterogeneous data settings.

Rather than being driven by theoretical analysis, some works leverage statistical insights from the transmitted data to develop methodologies. Data compression techniques, such as sparsification and quantization, are effective in reducing the dimensionality of client local gradients, making them widely adopted in designing practical

FL compression solutions. However, the absence of a well-defined problem formulation often makes it difficult to theoretically justify these methods.

Additionally, most existing works neglect to account for the correlation between clients. For example, in the compression literature, the same compressor [19, 76, 4] is applied across different clients, resulting in suboptimal rate allocation. Although some works scratch this issue by proposing to exploit correlation in terms of clients' local losses [51, 50], none have addressed modeling the clients' local gradient as random variables and considering the correlation in terms of clients' local gradient layer-wise. In light of this, we adopt the direct multi-terminal source coding formulation [73, 72] and aim to propose a correlation-aware solution that strategically selects participating clients and achieves optimal rate allocation.

In this paper, we present a rigorous study aimed at improving communication efficiency and convergence in FL from both theoretical and implementation perspectives. We begin by formulating the model update problem in a single FL iteration as a direct multi-terminal source coding problem. We introduce a method to estimate the clients' gradient correlation which facilitates converting the problem into a convex optimization problem. Based on this formulation, we characterize the upper and lower bounds to the sum-rate-distortion function, derive optimal rate allocation, and simulate the corresponding quantization process. Leveraging the correlation analysis, we also develop a client selection strategy that allows for more flexible, layer-wise application. Furthermore, we unify these two approaches into a comprehensive methodology, the effectiveness of which is verified through extensive experimental results. Our contributions can be outlined as follows:

- We formulate the single iteration source coding problem in FL using the direct

Gaussian multi-terminal source coding problem, and derive a convex optimization problem to calculate the upper and lower bounds of the sum-rate-distortion function by estimating clients' gradient correlation.

- Based on the sum-rate-distortion function results, we calculate and employ rate allocation in the quantization simulations under various FL schemes. The results demonstrate that our rate allocation method produces lower distortion under the same rate constraint. Conversely, it requires a lower rate to achieve the same level of distortion compared to the scenario that does not account for client correlation.

- Leveraging the correlation estimation, we develop a flexible correlation-aware layer-wise client selection strategy that outperforms state-of-the-art (SOTA) methods in terms of convergence speed and final accuracy. Furthermore, we combine the client selection with rate allocation and conduct extensive experiments to demonstrate the promising performance of the integrated approach.

2.3 Related Work

In this section, we briefly review the theoretical advancements and practical applications related to communication efficiency and convergence in FL.

2.3.1 Theoretical Analysis

Considerable efforts have been devoted to conducting theoretical analyses of the trade-off between communication cost and convergence in FL. Many studies focus on characterizing the global model convergence rate in relation to FL parameters, such as the

number of clients, the degree of local data heterogeneity, the number of local updates, and the frequency of communication between clients and the server. [88, 63, 74] lay the foundation for FL convergence analysis by adopting basic assumptions, including uniformly distributed local data and full client participation. [37] first examines the convergence of FedAvg on heterogeneous local data, deriving a convergence rate under the assumption of a strongly convex and smooth problem. Recent works have built on similar problem formulations to further investigate convergence rates in more realistic FL settings. For instance, [66, 29] analyze the convergence of FedAvg under partial client participation. [51] considers the temporal and spatial correlation in clients' availability dynamics, proposing a trade-off between maximizing convergence speed and minimizing model bias. [50] assesses intermittent client availability and suggests that by learning clients' long-term participation rates, improvements could be made in FL convergence by reducing model bias and minimizing sampling variance.

While these studies provide an intuitive understanding of factors affecting convergence in FL, they do not guide the design of encoding and decoding schemes for data transmission and aggregation in each iteration. Therefore, some researchers have shifted their focus from the entire convergence process to analyzing the communication cost in a single FL iteration.

The essence of the single iteration FL problem lies in the server's estimation of the local gradient (more precisely, a function of the local gradient) obtained by the clients based on corrupted observations. This inspires [86, 82] to formulate the problem as a distributed source coding problem. Based on rate-distortion theory, [82] puts forth a framework for model aggregation performance analysis, deriving the inner bound

of the rate region that can be achieved under a uniform data setting. [86] first connects the model update problem in FL with multi-terminal source coding. Their assumption that local gradient sequences are noisy versions of the global gradient, corrupted by additive Gaussian white noise (AWGN), places their problem formulation in the category of indirect Gaussian multi-terminal source coding, also known as the CEO problem [9, 48]. They depict the rate-region and characterize a trade-off between communication cost and convergence guarantees. However, their analysis is constrained to uniform data settings due to their assumptions. In contrast, by relaxing the assumptions about the local gradient, we adopt a direct multi-terminal source coding formulation. We aim to derive theoretical results that are applicable to both uniform and heterogeneous local data distributions.

2.3.2 Practical Implementation

Among practical implementations aimed at reducing FL communication costs, those involving gradient compression and client selection show the most promising potential.

Common techniques employed in gradient compression include sparsification and quantization. Gradient sparsification [3, 28] reduces the communication load by transmitting only a subset of gradient entries, exploiting the resulting vector's sparsity. Recent efforts focus on optimizing thresholding strategies to select the most impactful gradients. Gradient quantization [4, 76, 19, 40] involves mapping gradients to a restricted support representation using handcrafted scalar or vector quantizers. Both sparsification and quantization, whether applied separately or jointly, significantly enhance communication efficiency.

Within the literature on client selection methods, [66, 51] propose dynamically

assigning weights to participating clients to mitigate aggregation bias in partial client schemes, each grounded in different theoretical bases ([51] considers temporal and spatial client correlation, while [66] does not). In contrast, [29, 23] develop a biased client selection strategy favoring clients with higher local losses. [50] proposes algorithms tailored for positively correlated and uncorrelated clients based on client availability. [68] employs Gaussian Process modeling to collectively analyze and leverage changes in local losses for client selection based on correlation. [80] introduces a heterogeneity-aware approach promoting diversity through top$K$ sparsification gradient compression and client selection strategies.

Instead of relying solely on empirical observations, our methodologies are rigorously derived from client gradient correlation estimation within our problem formulation. Our approach integrates gradient quantization simulation and client selection, validated through experimental results to demonstrate its effectiveness.

2.4 Problem Formulation and Technical Results

In this section, we first present the FL problem formulation with the communication constraints. Then we introduce our problem formulation that considers the multi-terminal source coding problem in each iteration of FL as the multi-terminal source coding problem. Finally, the problem is converted into a convex optimization problem, and we derive the solutions.

2.4.1 Notations

Lowercase boldface letters (e.g., $\mathbf{z}$) are used for vectors, uppercase letters for random variables (e.g., X), and calligraphic uppercase symbols for sets (e.g., $\mathcal{A}$). Given the set $\mathcal{A}$, $|\mathcal{A}|$ indicates the cardinality of the set. We also adopt the short-hands $[m:n] \triangleq \{m, \ldots, n\}$ and $[n] \triangleq \{1, \ldots, n\}$. Let M_1 and M_2 be two square matrices of size $\mathbb{R}^{m \times m}$. The notation $\text{diag}(M_1, M_2)$ represents the diagonal operator that places M_1 and M_2 on the diagonal of a matrix of size $2m \times 2m$, with the remaining entries zero-padded. We use "$\preceq$" to denote positive semidefinite (PSD) partial ordering.

2.4.2 Preliminary

Consider the FL setting with K clients, each possessing a local dataset $\mathcal{D}_k \in \mathcal{D}$ for $k = 1, ..., K$ wishing to minimize the *loss function* $\mathcal{L}$ as evaluated across all the clients and over the model weights $\mathbf{w} \in \mathbb{R}^m$, where m denotes the dimensionality of the model parameter. This minimization is coordinated by a parameter server (PS) as follows: at discrete points in round $t \in [T]$, (i) the clients transmit a model update to the PS, the PS, (ii) the PS aggregates the model updates and produces a model update, and (iii) the updated model is transmitted to the clients. The above steps are repeated for T times: the model obtained at time T is declared as the optimal model.

Mathematically, the loss function $\mathcal{L}$ is defined as:

$$\mathcal{L}(\mathbf{w}) = \frac{1}{|\mathcal{D}|} \sum_{k \in [K]} \mathcal{L}_k(\mathcal{D}_k, \mathbf{w}), \tag{2.4.1}$$

where $\mathcal{L}_k(\mathcal{D}_k, \mathbf{w})$ is the local loss function quantifying the prediction error of the k-th client's model. A common approach for numerically finding the optimal value of $\mathbf{w}$ is

through the iterative application of (synchronous) stochastic gradient descent (SGD). We define the local gradients calculated at communication round t as:

$$\mathbb{E}[\mathrm{g}_{kt}] = \mathbb{E}[\nabla \mathcal{L}_k(\mathcal{D}_k, \mathbf{w}_t)], \tag{2.4.2}$$

where $\nabla \mathcal{L}_k(\mathcal{D}_k, \mathbf{w}_r)$ denotes the local gradients of the model examined at the local dataset of the k-th client by minimizing the local loss function. Note that the expectation in (2.4.2) is taken over the randomness in evaluating the gradients – e.g., mini-batch effects.

In many scenarios of practical relevance, the communication between the PS is limited in some manner, while the communication from the PS to the client is unconstrained. One common constraint considered in the literature on communication between clients and the PS is the *rate constraint*– in which the communication is restricted to $d\mathrm{R}_k$ bits from each client to the PS. To meet this constraint, before communication, each client k compresses the d-dimensional gradient vector g_{kt} to a $d\mathsf{R}_k$ binary vector through a compressor $\mathrm{comp}_{\mathrm{R}_k} : \mathbb{R}^d \to [2^{d\mathsf{R}_k}]$, where the sum over all R_k is constrained, and the client is dropped (unselected) if assigned a rate of $\mathrm{R}_k = 0$. The PS aggregates all the compressed gradients and forms the new global weights

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \hat{\mathbf{g}}_t, \tag{2.4.3}$$

for $t \in [T]$ and where

$$\hat{\mathbf{g}}_t = \frac{1}{K} \sum_{k \in [K]} \mathrm{comp}^{-1}_{\mathrm{R}_k}(\mathrm{comp}_{\mathrm{R}_k}(\mathbf{g}_{kt})), \tag{2.4.4}$$

and where $\mathbf{w}_0$ is randomly drawn according to some distribution. The ultimate goal in a single FL iteration is to minimize the distortion in gradient reconstruction, i.e., the distortion between $\mathbf{g}_t$ and $\hat{\mathbf{g}}_t$. We assume that minimizing the gradient reconstruction distortion leads to minimum perturbation of the accuracy.

2.4.3 Data Heterogeneity

In many practical scenarios, the local dataset present at the client is intrinsically heterogeneous. A possible way to describe this data heterogeneity is by assuming that the data entries are i.i.d. draws from certain distribution present at a client k. Mathematically, let a dataset i-the dateset entry at user k – including features and labels – be denoted as the vector $\mathbf{m}_{ik}$. The distribution of $\mathbf{m}_{ik}$ is obtained as:

$$\mathbf{m}_{ik} \sim P_{\mathbf{m}}^{k} = \sum_{j \in [J]} \lambda_{jk} P_{\mathbf{m}}^{(j)}. \tag{2.4.5}$$

In other words, we assume that there exists a kernel of J distributions

$$\left\{ P_{\mathbf{m}}^{(j)} \right\}_{j \in [J]}, \tag{2.4.6}$$

and each client observes a data distribution which is obtained as mixture of these kernel distributions according the coefficients

$$\boldsymbol{\lambda}_k = \left\{ \lambda_{kj} \right\}_{j \in [J]}. \tag{2.4.7}$$

2.4.4 Deep Neural Networks (DNNs)

While the problem formulation in Sec. 2.4.2 is rather general, in the remainder of the paper, we shall only consider the scenario in which the model weight models a DNN N. In the context of DNN training, it is often convenient to re-parameterize the vector of parameters $\mathbf{w}$ int the matrix $\mathbf{W}$. For simplicity, let us consider a classic Multi-Layer Perceptron (MLP) with L layers, each of j neurons, with activation σ and without bias. For this setting, there are j^2 weights connecting the neurons at layer l to those of layers $l+1$. Accordingly, let us re-define the matrix $\mathbf{W}$ as a matrix of size $L \times j^2$ in which each row contains the weights between two given layers.

2.4.5 Relevant Assumptions

In general, the iterations in (2.4.3) for the learning problem in (2.4.1) give rise to a random process when accounting for the (i) randomness in the model initialization and (ii) in the evaluation of the stochastic gradient. The characterization of the properties of this process is generally considered a hard problem. In the following, we adopt a set of assumptions on the distribution of the gradients that make it possible to develop a rigorous theoretical and numerical approach. Although we do not validate these assumptions, we argue that the numerical results show the effectiveness of the approach developed from them. These gradient assumptions are as follows:

A1 **well-defined random process:** Consider a given learning problem as in (2.4.3) and a given DNN N: then (i) the heterogeneity in the data distribution, (ii) the randomness in the network initialization and (iii) the randomness

in the gradient evaluation give rise to the well-defined random process

$$\mathcal{G} = \{\mathbf{g}_{kt}\}_{k\in[K],t\in[T]}. \tag{2.4.8}$$

A2 **gradient independence over time:** We assume that the process $\mathcal{G}$ is such that $\mathbf{g}_{kt} \perp \mathbf{g}_{kt'}$ for $t \neq t'$.

A3 **gradient iid-ness in a layer:** For each client, the gradient is an i.i.d. sequence within each layer.

A4 **joint Gaussianity across clients:** For the same entry in the same layer, the distribution of gradients at client k and client k' follows a joint Gaussian distribution with mean zero and covariance $\Sigma_{k,k'}$. $\Sigma_{k,k'}$ is the (k,k')-entry of a covariance matrix Σ.

A5 **independence inter-layer:** For simplicity, the gradients across different layers are assumed to be independent in our problem formulation. One may view this as equivalent to considering a single-layer network theoretically. In Sec.2.4.7, Sec.2.5 and Sec.2.6, technical analyses and implementations are done layer-wise for multi-layer DNNs. A multi-layer generalization is discussed in App.A.3.

Remark 1 **[Change of Notation]** *Based on the above assumptions, we model each layer of the gradient at client k as a single random variable with each element in the sequence being a different realization. We change the notation of the gradient sequence of length n at client k at round t to be* $\{G_{kt}\}_{i=1}^{n}$*, abbreviated as* G_k^n *(omitting round t). In App.A.3, a random vector generalization is included.*

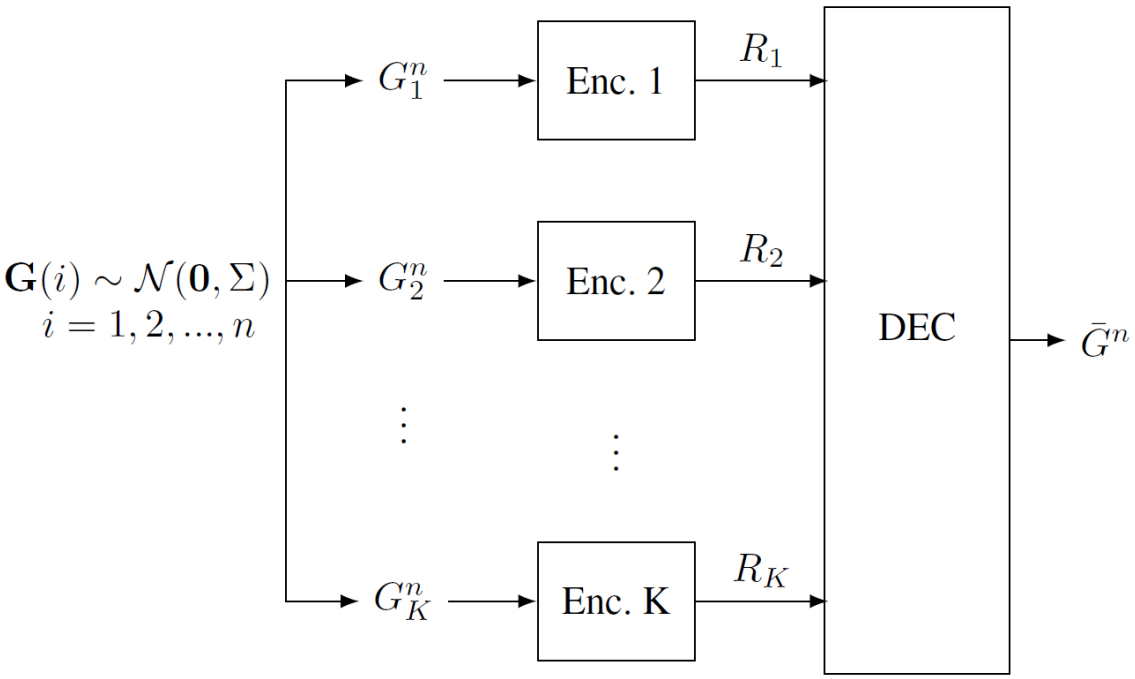

Figure 2.1: The Gaussian multi-terminal source coding problem in FL as introduced in Sec. 2.4.6. See Remark 1 for notations.

2.4.6 Multi-terminal Source Coding Problem in FL

With the above preliminaries, we shall introduce our problem formulation that considers the model update problem in single FL iteration as the direct Gaussian multi-terminal source coding problem.

As depicted in Fig.2.1, at communication round t, client1 through client$K$ each obtains a local gradient sequence of length n, denoted as $G_1^n, G_2^n, ..., G_K^n$. Let the source be a Gaussian source with mean zero and positive definite covariance matrix Σ_G. Following assumptions A3 and A4 that assume i.i.d.-ness within each local gradients sequence and correlation across different clients at the same-position-entry of local gradients, we let $G_k(i), i = 1, ..., n$ denote the entries of local gradient sequence of the k-th client, for $k = 1, ..., K$. Each client encodes the local gradients using

encoding functions f_k: $\mathbb{R}^n \to \{1, 2, ..., M_k\}$, $k = 1, ..., K$ and sends the encoded gradients to PS at respective rates $R_k, k = 1, ..., K$.

Differing from traditional distributed learning problems that focus on reconstructing each individual source, the decoder Φ in FL aims to reconstruct an aggregation function of the sources. Denote the FL reconstruction as $\bar{G}^n$ with the same length as each source. Let d denote the desired distortion level. The distortion constraint in FL is defined as

$$\frac{1}{n}\sum_{i}^{n}\Big(\frac{1}{K}\sum_{k=1}^{K}G_k(i) - \bar{G}(i)\Big)^2 \leq d. \tag{2.4.9}$$

Here, we take FedAvg aggregation with an equal amount of local data as an example for simplicity. Note that our results could be generalized to any linear aggregation function with slight adjustments, as described in App.A.3.

Now we define the model update problem in a single FL iteration as the infimum of all achievable sum-rates for all possible encoders and decoder choices satisfying (2.4.9). The ultimate formulation of the problem we aim to solve is as

$$\begin{aligned} R^*(d) \triangleq \lim_{n\to\infty} \inf_{f_1, f_2, ..., f_K, \Phi} \ & \frac{1}{n}\sum_{k\in[K]}\log M_k \\ \text{s.t.} \quad & \frac{1}{n}\sum_{i}^{n}\Big(\frac{1}{K}\sum_{k=1}^{K}G_k(i) - \bar{G}(i)\Big)^2 \leq d. \end{aligned} \tag{2.4.10}$$

2.4.7 Technical Results

Following the work of [72, 73], we derive the upper and lower bounds to the minimum sum-rate $R^*(d)$. To achieve this, we shall introduce several auxiliary variables: $\Sigma_Z, D, \Sigma_k, \Gamma_k, k = 1, 2, ..., K$ (more details in Appendix A.1).

First we define a positive definite covariance matrix as Σ_Z based on the source

covariance matrix Σ_G, such that $\Sigma_G^{-1}+\Sigma_Z^{-1}$ is a diagonal matrix. We define a diagonal matrix such that $\Sigma = \text{diag}(\Sigma_1, \Sigma_2..., \Sigma_K) = (\Sigma_G^{-1} + \Sigma_Z^{-1})^{-1}$. Note that Σ and Σ_Z uniquely determines each other. In the following, we keep both in the formulation but they should not be considered as two independent variables.

In addition, we define D to be the distortion covariance matrix of the MMSE estimation for $\mathbf{G} \triangleq ((G_1^n)^T, (G_2^n)^T, ..., (G_K^n)^T)^T$ given the encoder outputs $W_{\mathcal{K}} \triangleq (W_1, W_2..., W_K)$ with $W_k = f_k(G_k^n), k = 1, 2, ..., K$, i.e., $D = \Sigma_{\mathbf{G}|W_{\mathcal{K}}}$.

Moreover, we define another diagonal matrix $\Gamma = \text{diag}(\Gamma_1, \Gamma_2..., \Gamma_K)$, where each diagonal element $\Gamma_k, k = 1, 2, ..., K$ is defined as the distortion of the MMSE estimation for each source given corresponding encoder's output and a remote source, i.e., $\Gamma_k = \Sigma_{G_k^n|\mathbf{Y},W_k}, k = 1, 2, ..., K$. Here $\mathbf{Y}$ is determined by Σ_G and Σ_Z, and $G_1^n, G_2^n, ..., G_K^n$ are independent conditioned on $\mathbf{Y}$.

And the relationship between D, Σ_Z and Γ_k can be verified by following the derivation of (10) in [72] that:

$$\text{diag}(\Gamma_1, ..., \Gamma_K) \preceq (D^{-1} + \Sigma_Z^{-1})^{-1}. \tag{2.4.11}$$

Additionally, we define $\mathbb{1} \in \mathbb{R}^K$ to be an all ones vector. With auxiliary variables introduced, we obtain the expressions for the upper bound $\bar{R}^*(d)$ and lower bound $\underline{R}^*(d)$ to the minimum sum-rate.

Upper bound, also referred to as the Berger-Tung upper bound [8, 70], is as:

$$\begin{aligned}
&\bar{R}^*(d) \triangleq \\
&\max_{\Sigma_1,...,\Sigma_K} \min_{D,\Gamma_1,...,\Gamma_K} \frac{1}{2}\log\frac{|\Sigma_G+\Sigma_Z|\,|\mathrm{diag}(\Sigma_1,...,\Sigma_K)|}{|D+\Sigma_Z|\,|\mathrm{diag}(\Gamma_1,...,\Gamma_K)|} \\
&\text{s.t.} \quad \Sigma_Z = (\mathrm{diag}(\Sigma_1,...,\Sigma_K)^{-1} - \Sigma_G^{-1})^{-1}, \\
&\qquad 0 \preceq \mathrm{diag}(\Sigma_1,...,\Sigma_K) \preceq \Sigma_G, \\
&\qquad 0 \preceq D \preceq \Sigma_G, \\
&\qquad \mathbb{1}\cdot D\cdot \mathbb{1}^T \le d, \\
&\qquad 0 \preceq \mathrm{diag}(\Gamma_1,...,\Gamma_K) \preceq \mathrm{diag}(\Sigma_1,...,\Sigma_K), \\
&\qquad \mathrm{diag}(\Gamma_1,...,\Gamma_K) = (D^{-1}+\Sigma_Z^{-1})^{-1}.
\end{aligned} \tag{2.4.12}$$

Lower bound has the same form except for the last constraint, where "$\preceq$" replaces "$=$", as:

$$\begin{aligned}
&\underline{R}^*(d) \triangleq \\
&\max_{\Sigma_1,...,\Sigma_K} \min_{D,\Gamma_1,...,\Gamma_K} \frac{1}{2}\log\frac{|\Sigma_G+\Sigma_Z|\,|\mathrm{diag}(\Sigma_1,...,\Sigma_K)|}{|D+\Sigma_Z|\,|\mathrm{diag}(\Gamma_1,...,\Gamma_K)|} \\
&\text{s.t.} \quad \Sigma_Z = (\mathrm{diag}(\Sigma_1,...,\Sigma_K)^{-1} - \Sigma_G^{-1})^{-1}, \\
&\qquad 0 \preceq \mathrm{diag}(\Sigma_1,...,\Sigma_K) \preceq \Sigma_G, \\
&\qquad 0 \preceq D \preceq \Sigma_G, \\
&\qquad \mathbb{1}\cdot D\cdot \mathbb{1}^T \le d, \\
&\qquad 0 \preceq \mathrm{diag}(\Gamma_1,...,\Gamma_K) \preceq \mathrm{diag}(\Sigma_1,...,\Sigma_K), \\
&\qquad \mathrm{diag}(\Gamma_1,...,\Gamma_K) \preceq (D^{-1}+\Sigma_Z^{-1})^{-1}.
\end{aligned} \tag{2.4.13}$$

It should be emphasized that after this relaxation, $\underline{R}^*(d)$ in (2.4.13) is not on a lower

bound for the Berger-Tung upper bound but also a lower bound for $R^*(d)$. Inspired by [87], we develop an iterative algorithm that solves the max-min problem presented in (2.4.12) and (2.4.13) and the same time. Specifically, we first calculate the lower bound for $\underline{R}^*(d)$; substituting the optimal $\Gamma_1, \ldots, \Gamma_k$ back into (7) gives rise to the upper bound $\bar{R}^*(d)$. Detailed algorithm is presented in Appendix (A.2).

Rate Allocation: in the following, we consider the upper bound of the minimum sum-rate $\bar{R}^*(d)$ for rate allocation, as it is sufficiently achievable, while the lower bound may not. To derive the optimal rate allocation scheme following the sum-rate calculated in (2.4.12), we shall apply *binning* [1]. We use the results from the max-min problem, Σ_k and Γ_k to define M_k, for $k = 1, ..., K$ to be a diagonal matrix with the same dimension as Σ as

$$M_k = \text{diag}(\Sigma_1, ..., \Sigma_k, \Gamma_{k+1}, ...\Gamma_K). \tag{2.4.14}$$

Now we can derive the expression individual rate for each client as:

$$R_k = \frac{1}{2}\log\frac{|\Sigma_Z(\Sigma_Z - M_k)^{-1}\Sigma_Z|}{|\Sigma_Z(\Sigma_Z - M_{k-1})^{-1}\Sigma_Z|} + \frac{1}{2}\log\frac{|\Sigma_k|}{|\Gamma_k|} \quad , \text{for } k = 1, ..., K, \tag{2.4.15}$$

which satisfies that $\sum_{k=1}^{K} R_k = \bar{R}^*(d)$. Note that the result calculated from (2.4.15) is not the unique optimal rate allocation scheme. By changing the estimation order, one may find $K!$ optimal rate allocation schemes, and they, as corner points, define a optimal dominant face. More details are found in Appendix A.5.

[1] also referred to as Cover's random binning [18] or Gel'fand-Pinsker coding [20].

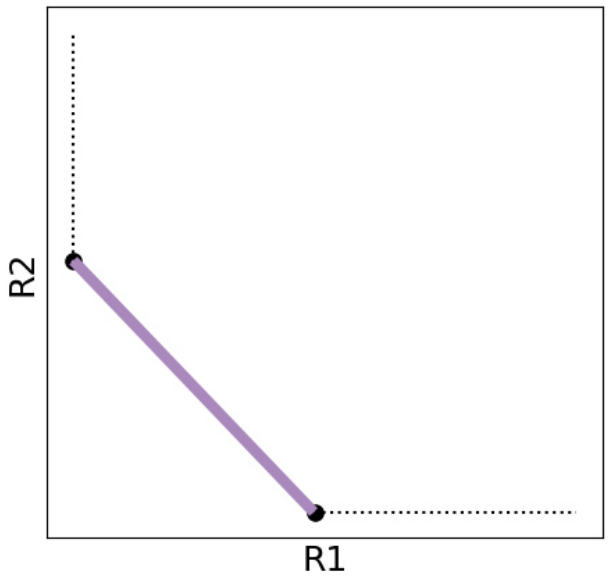

Figure 2.2: Example dominant faces (colored) that describe the optimal rate allocation schemes for $K = 2$.

Contrary to considering binning, one can ignore the correlation and derive a different rate allocation scheme under the same sum-rate constraint, which we refer to as the "no binning" case. Details are included in Appendix A.5.

Another noteworthy feature of our proposed approach is that there indeed exist an infinite number of optimal rate allocation schemes attaining the same optimal sum-rate. The region formed by the equivalent rate allocations is depicted in Fig.2.2 for $K = 2$ (left) and Fig.2.3 for $K = 3$ (right). This region corresponds to the dominant face of a polymatroid arising from the inequalities in (2.4.12), and the corner points are associated with a specific decoding order of the client, giving rise to $K!$ corner points, which can be calculated using variants of (2.4.15). More detailed insights are included in App.A.5.

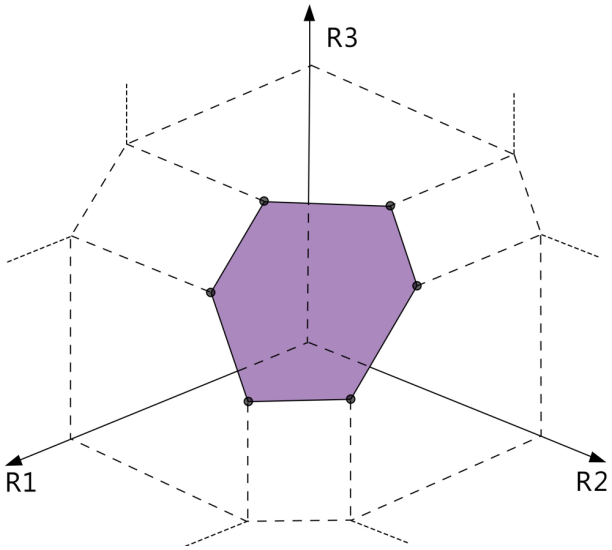

Figure 2.3: Example dominant faces (colored) that describe the optimal rate allocation schemes for $K = 3$.

2.5 Proposed Methods

In this section, we present a detailed exposition of our method, which stems from the technical results. First, we elaborate on the steps for performing correlation estimation, which is not part of the original FL framework, but is essential to the success of our methods. Then we present a novel client selection strategy, which is a natural extension leveraging the correlation estimation.

2.5.1 Correlation Estimation in FL

As presented in the problem formulation and technical results, our formulations profoundly rely on the correlation, which is represented by the covariance matrix of clients gradients, Σ_G. We employ a simple yet effective random sampling procedure to achieve correlation estimation, described as follows. Note, we only consider the

correlation for the same layer across different clients as stated in the assumption section. We leave the investigation on cross-layer correlation to our future work. The specific steps include:

1. Server randomly generates sets of indexes layer-wise and broadcast them along with the global model to the clients at the beginning of each communication iteration.

2. After finishing local updates, clients sample their resulting gradients according to the indexes layer-wise and send them back to the server.

3. As all the clients sample from corresponding indexes in gradients tensors, the server calculates one covariance matrix for each layer.

2.5.2 Correlation-aware Layer-wise Client Selection Strategy

Besides implementing rate allocation based on the estimated correlation, it is so informative that seamlessly allows us to undertake the calculation of the expected distortion, which naturally facilitates our client selection strategy. Given the number of clients needs to be selected, n, and the covariance matrix, $\Sigma_G \in \mathbb{R}^{K \times K}$, we perform client selection as follows:

1. First we construct a "combination matrix", C. Each row in C is a boolean vector, representing one combination of choosing n out of k, e.g., $[1, 0, 1]$ representing choosing the first and the third client out of all three clients.

2. Then we stack an all-ones vector at the bottom of C , and multiplying the covariance matrix Σ_G by C on its left and C^T on its right. We compute a square

matrix, denoted as $Q \in \mathbb{R}^{(m+1)\times(m+1)}$, where $m = \binom{n}{x}$, and $Q \triangleq C\Sigma_G C^T$. We note that the first m diagonal terms of Q represent the variances of corresponding selection, while the last diagonal terms represent the variance of considering all clients, in other words, the variance of the true gradients.

3. Finally, take the difference between the first m diagonal terms with the last diagonal term of Q, we calculate a list of distortion values, where the smallest one indicates the optimal client selection combination.

Note, the resulting client selection strategy yields the smallest expected distortion with respect to the "true" FedAvg result comparing to all other choices from the covariance matrix Σ_G perspectives. Please refer to Appendix (A.4) for more details. We point out that all computation are conducted on the server, which is universally assumed to have substantial computational capability, thereby avoiding any computational burden in FL. Furthermore, we highlight the flexibility of our strategy as it allow us to perform different client selection layer-wise because the correlation estimation is done layer-wise. More importantly, this scheme introduce no extra communication overhead as the amount of total communication cost is the same comparing to traditional client selection strategies. This innovation in unprecedented in the FL literature, and its effectiveness is verified and explained in the Experiment Section.

2.5.3 Overall Framework

We present our method in Algorithm (1), where **Sample**, **CorrelationEstimation**, **ClientSelection** and **RateAllocation** functions have been described above, and **ClientUpdate** is chosen from deep learning optimization method like SGD. Note

that we refer to [84, 83] to approximate the dithered quantization process including **ClientEncode** and **ServerDecode** by passing the source through an additive noise channel. The approximation includes adding a uniformly distributed noise to the source before quantization and subtracting from the quantizer output is proven to be accurate in all levels in dithered quantization. The detailed calculation is included in Appendix (A.5). With acknowledging that incorporating correlation estimation into the FL framework inevitably introduces additional communication and computation costs, we point out that the costs generated by our method are minimal and insignificant when weighed against the benefits.

Algorithm 1 Training procedure of GCFed

Initialize Global Model w_0
for communication iteration $t = 1$ to T **do**
    Server randomly sample index vectors $i\vec{d}x$ for each layer of the model
    Server broadcast current global model w_t and $i\vec{d}x$
    for client $k = 1$ to K in parallel **do**
        $G_{kt}^{n} \leftarrow$ **ClientUpdate**($\mathcal{D}_k$, w_t)
        for layer $l = 1$ to L in parallel **do**
            $G_{ktl}^{(s)} \leftarrow$ **Sample**(G_{ktl}^{n} , $i\vec{d}x$)
        end for
        $G_{kt}^{(s)} = [G_{kt1}^{(s)}, G_{kt2}^{(s)}, ..., G_{ktL}^{(s)}]$
        Client sends $G_{kt}^{(s)}$ back to Server
    end for

    Server performs layerwise:
    $\Sigma_G \leftarrow$ **CorrelationEstimation**($G_{kt}^{(s)}$)
    $\mathcal{K}_t \leftarrow$ **ClientSelection**(Σ_G) , $\mathcal{K}_t \in K$
    $\mathcal{R}_{kt} \leftarrow$ **RateAllocation**(Σ_G, $\mathcal{K}_t$)
    Server broadcast $\mathcal{K}_t$ and $\mathcal{R}_{kt}$

    for client $\tilde{k} = 1$ to $\mathcal{K}_t$ in parallel **do**
        for layer $l = 1$ to L in parallel **do**
            $\hat{G}_{\tilde{k}tl}^{n} \leftarrow$ **ClientEncode**($G_{\tilde{k}tl}^{n}$,$\mathcal{R}_{ktl}$)
        end for
        $\hat{G}_{\tilde{k}t}^{n} = [\hat{G}_{\tilde{k}t1}^{n}, \hat{G}_{\tilde{k}t2}^{n}, ..., \hat{G}_{\tilde{k}tL}^{n}]$
        Client sends $\hat{G}_{\tilde{k}t}^{n}$ back to Server
    end for
    $w_{t+1} \leftarrow$ FedAvg($\mathcal{K}_t$, **ServerDecode**($\hat{G}_{\tilde{k}t}^{n}$, $\mathcal{R}_{kt}$))
end for

2.6 Experiments

In this section, we provide details about our numerical experiment. We first illustrate the experiment setting. Then we present simulation results that validate our technical results in Section III. Following that, we compare our client selection strategy with three baselines and conduct ablation study regarding to its two variants. Finally, we show that the integration our client selection strategy with compression demonstrate promising performance.

2.6.1 Experiment Settings

We conduct experiments on CIFAR-10 [32] and Fashion-MNIST datasets [79] with a VGG16 network [61] and a CNN network, respectively. Note that the application of our approach is not limited to our chosen dataset or network architecture. For the purpose of efficiently analyzing the effectiveness of our methods, we initialize the network using a pre-trained VGG16 weight obtained from ImageNet-1K [16], while initializing the last output layer in order to match the output dimension. During the training process, we freeze the features part of the network except for the last three CONV layers and the subsequent ReLU and pooling layers. Also, we unfreeze the entire classifier part, which consists of 3 Fully-Connected(FC) layers and ReLU and Dropout layers. All the correlation estimation and related applications are deployed on the 3 FC layers because we focus on the spatial correlation between the same position in FC layers. For each layer, the server randomly samples 100 indexes regardless of the size of the layer. More experiment details are included in App.A.7 We experiment six different heterogeneous data partitions on $K = 10$ clients as follows.

(i) **Partition by Shards (PS)**: in this setting, the entire training dataset is divided into $K \cdot S = 10 \cdot S$ shards, where S is a parameter controlling the heterogeneity level. Within each shard, all the data have the same label. Each client will randomly take S shards to consist the local dataset. Thus, a smaller S value indicates a more heterogeneous setting, while a larger S value represents a more uniform case. For example, the extreme case is when $S = 1$, meaning that all the local data of each client is of the same one label. In comparison, when $S = 5$, the local data of each client has at most 5 different labels (we don't exclude the case where client draw two or more shards of the same label). We refer to [68] for this implementation. We experiment with $S = 1, 2$ and 5.

(ii) **Partition by Bias Level (PB)**: $\epsilon \in [0, 1]$ indicates the local dataset distribution: each client is assigned with a bias (favorite) label $n \in [1, ..., 10]$. Among the local dataset of each client, ϵ multiplying the total number of local data points are drawn according to the bias label. The rest $1 - \epsilon$ multiplying the total number of local data points are drawn uniformly from the remaining dataset. A greater ϵ value indicates more heterogeneity. We experiment with $\epsilon = 0.25, 0.5$ and 0.75.

Clients performs 3 rounds of local updates before they communicate with the server. We use SGD as the optimizer with learning rate 0.03 for i) Partition by shards and 0.01 for ii) Partition by bias level, respectively, with momentum equals to 0.5 and learning rate decay equals to 0.995. We apply FedAvg to aggregate the clients updates. We choose the distortion to be $d = 0.1 \cdot (\mathbb{1} \cdot \Sigma_G \cdot \mathbb{1})$ (0.1 times the maximum distortion), which uniquely determines the sum rate. All experiments are repeated with 3 random seeds.

2.6.2 Binning vs. No Binning

Shown in Figure 2.4, on the left, we present the rate gain comparing "binning" with "no binning" under the same distortion level through out the training process. Each curve indicates one heterogeneous data partition. At the top, we show the distortion gain comparing "binning" with "no binning" under the same rate constraints. At the bottom, we show in a reversed way that under the same rate constraint, considering the correlation ("binning") yields much less distortion comparing to ignoring the correlation ("no binning") under all heterogeneity settings. The results validates our technical results, where we see consistent gains through the entire training process under all heterogeneity settings. Especially, the gains are more obvious when the convergence is approaching to later iterations, where we assume the local gradients become more correlated to each other comparing to the initial stage. Furthermore, the results indicate that there is potentially more gain when local data is more heterogeneous comparing to when local data is more uniformly distributed.

As our problem formulation and proposed approach follows the rate-distortion theory, the rate distortion curves yielded by our correlation estimation follow the traditional non-increasing convex shape, which is illustrated in Appendix A.6. Here, we focus on showing the gains attained through our approaches.

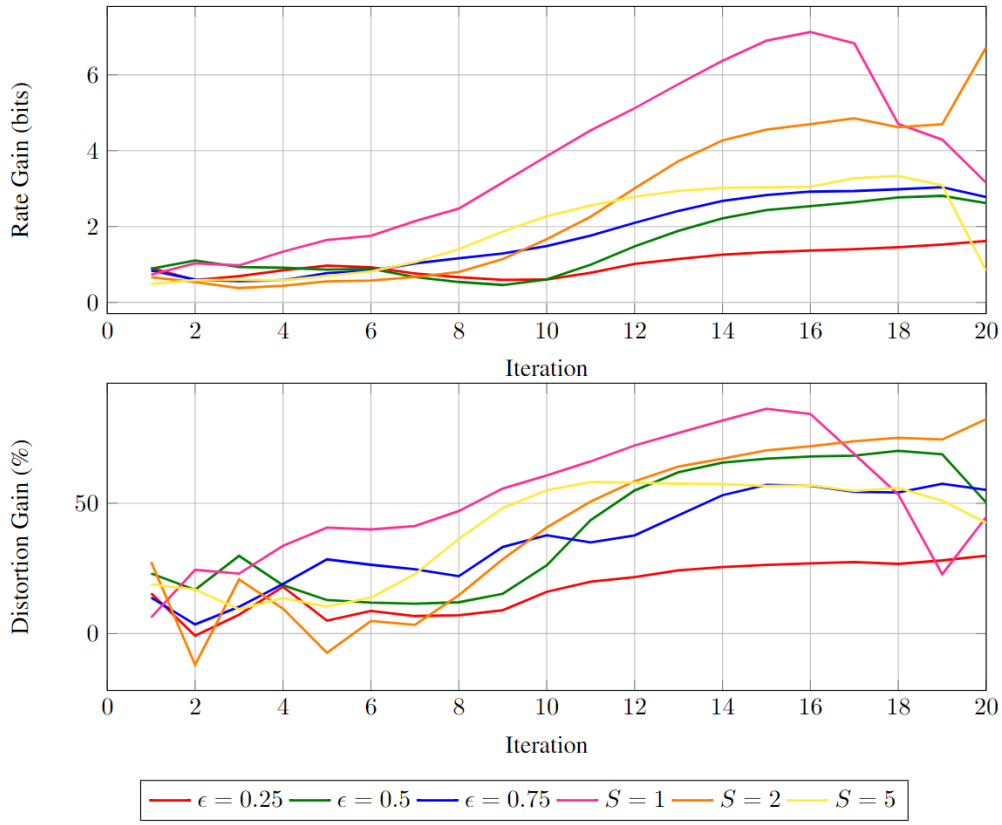

Figure 2.4: Top: under the same distortion constraint, the rate gain in terms of number of bits through the converging process; Bottom: under the same rate constraint, the distortion gain in terms of percentage through the converging process.

2.6.3 Convergence Comparison to SOTA methods

We compare the convergence rate of our method with three baselines: FedCor [68], AFL [23] and Pow-d [29] following the experiment settings described in Sec. 2.6.1.

As shown in Figure 2.5, our method demonstrates the fastest convergence speed and achieves the highest test accuracy in all settings. Although in some cases, the performance gap is less significant between ours and the baselines, our methods obviously outperforms the others under the more heterogeneous settings, especially when

CIFAR-10	Partition by Shards, S		
	$S = 1$	$S = 2$	$S = 5$
FedCor	27.68 ± 1.52	44.14 ± 5.23	59.68 ± 0.67
AFL	38.23 ± 0.87	51.92 ± 0.58	59.65 ± 1.13
POWER-OF-CHOICE	38.83 ± 1.99	52.53 ± 0.71	**60.77** ±0.46
Ours	**40.19** ± 0.84	**54.06** ± 0.44	60.64 ± 1.53

CIFAR-10	Partition by Bias level, ϵ		
	$\epsilon = 0.25$	$\epsilon = 0.5$	$\epsilon = 0.75$
FedCor	**67.55** ± 0.37	**66.46** ± 0.52	62.39 ± 0.69
AFL	**67.55** ± 0.10	66.14 ± 0.36	62.18 ± 0.31
POWER-OF-CHOICE	67.52 ± 0.21	65.99 ± 0.36	62.06 ± 1.42
Ours	67.52 ± 0.08	66.3 ± 0.71	**62.84** ± 0.56

Table 2.1: Test accuracy comparison to SOTA methods in six heterogeneous settings on CIFAR-10.

$S = 1$ and $S = 2$.

In Table 2.1 and Table 2.2, we numerically display the test accuracy (mean $\pm$ `std`) of the converged models. Our method achieves the best test accuracy among 8 out of 12 settings. In 3 out of the losing cases, our method lags behind the best by small margins, no more than 0.16%. While in the winning cases, our method leads the second-best by up to 1.83% on CIFAR-10 and 7.02% on Fashion-MNIST.

Fashion-MNIST	Partition by Shards, S		
	$S = 1$	$S = 2$	$S = 5$
FedCor	20.15 ± 2.82	31.18 ± 4.81	42.11 ± 8.16
AFL	20.27 ± 8.27	37.33 ± 3.22	48.93 ± 4.77
POWER-OF-CHOICE	**24.13** ± 11.49	44.29± 3.26	52.78 ± 5.33
Ours	20.08 ± 2.08	**44.60** ± 6.17	**59.80** ± 4.25

Fashion-MNIST	Partition by Bias level, ϵ		
	$\epsilon = 0.25$	$\epsilon = 0.5$	$\epsilon = 0.75$
FedCor	71.70 ± 0.21	68.92 ± 2.92	64.76 ± 2.37
AFL	71.23 ± 0.40	69.92 ± 0.29	65.27 ± 3.53
POWER-OF-CHOICE	**71.85** ± 0.46	70.33 ± 0.46	64.47 ± 2.34
Ours	**71.85** ± 0.15	**70.64** ± 0.69	**66.98** ± 2.56

Table 2.2: Test accuracy comparison to SOTA methods in six heterogeneous settings on Fashion-MNIST.

2.6.4 Client Selection Ablation Study

To demonstrate the effectiveness of our client selection strategy, we conduct ablation study against two implementations. We refer to "Random" as to randomly selecting three out of total ten clients. And "Top" refers to selecting the clients with the largest three variance values in the covariance matrix, Σ_G. In order to maintain a fair comparison, layer-wise selection is enabled for all three implementations.

As shown in Figure 2.6, Table 2.3 and Table 2.4, our method demonstrates superior performance comparing to the other two implementations in terms of both converging speed and converged test accuracy. We also observe greater performance gains of our method under more heterogeneous settings.

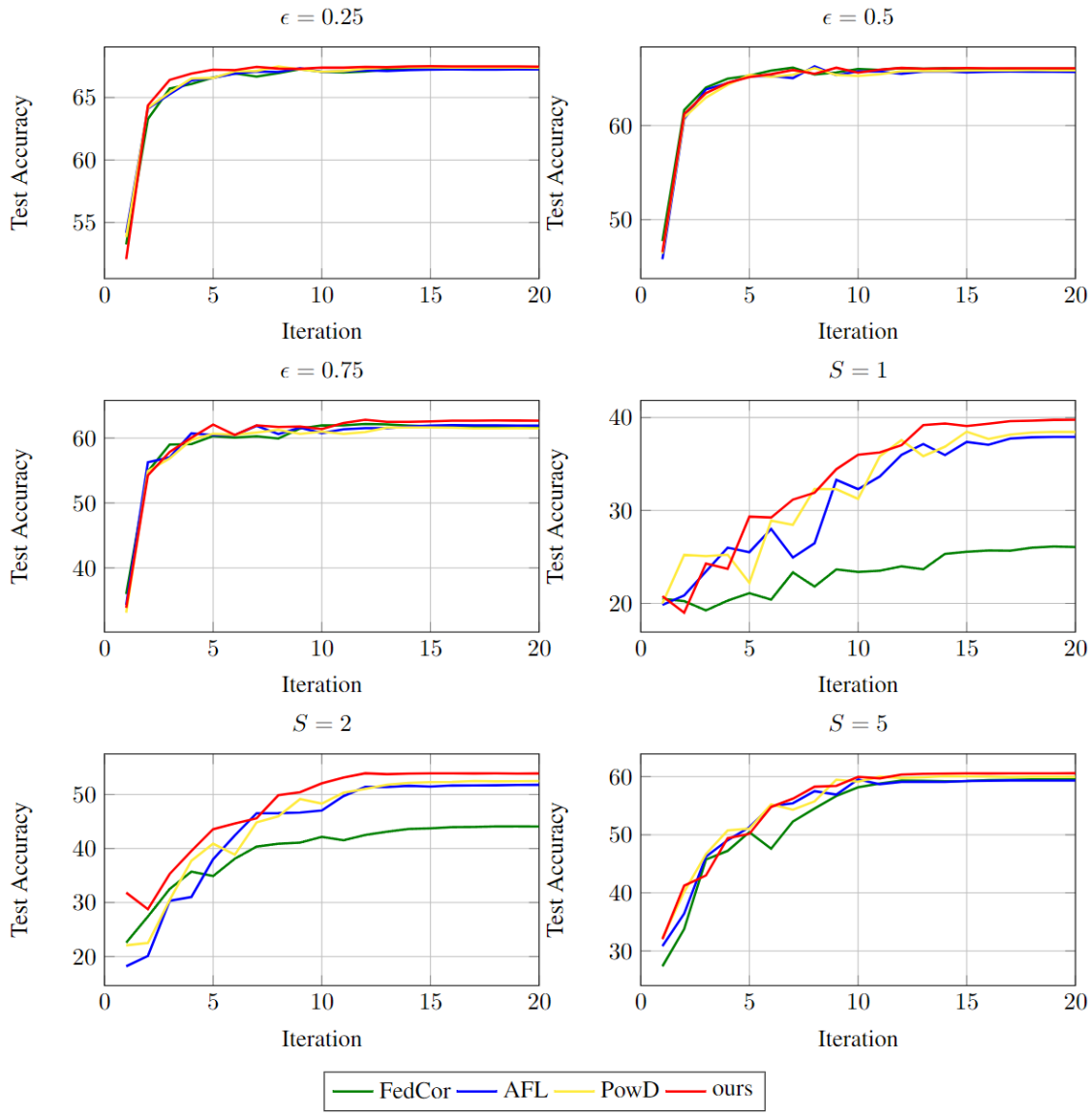

Figure 2.5: Convergence comparison between our client selection strategy and the SOTA. Each figure represents one heterogeneous setting. All experiments in one figure share the same hyperparameters except for the client selection strategy.

CIFAR-10	$S=1$	$S=2$	$S=5$	$\epsilon=0.25$	$\epsilon=0.5$	$\epsilon=0.75$
Random	37.25	52.02	59.95	**67.68**	65.91	62.17
Top	32.48	50.86	59.16	67.40	65.52	61.54
Ours	**40.19**	**54.06**	**60.64**	67.52	**66.30**	**62.84**

Table 2.3: Ablation study of layer-wise client selection on CIFAR-10.

Fashion-MNIST	$S=1$	$S=2$	$S=5$	$\epsilon=0.25$	$\epsilon=0.5$	$\epsilon=0.75$
Random	19.20	36.98	51.26	71.71	70.29	65.60
Top	17.59	40.07	53.43	71.34	68.98	62.85
Ours	**20.08**	**44.60**	**59.80**	**71.85**	**70.64**	**66.98**

Table 2.4: Ablation study of layer-wise client selection on Fashion-MNIST.

2.6.5 Unify Client Selection & Compression

Given the demonstrated effectiveness of our rate allocation and client selection methods when being applied independently, it is natural to inquire about their performance when used jointly. In this experiment, we apply our client selection strategy first to select 3 clients out of 10 in each round. Then we simulate three compression schemes: i) "binning", ii) "no binning" and iii) "centralizing", where binning reflects to our optimal rate allocation, no-binning represents ignoring the correlation, and centralizing simulates the ideal but impractically case where one has the access to all clients original gradients. The details of the simulations are included in Appendix A.5.

In this experiment, all three schemes are under the same total rate constraint in each communication round, and we compare the model convergence speed and final accuracy. The results in Table 2.5 show that rate allocation with "binning" outperforms the "no binning" counterpart in terms of convergence accuracy. Comparing to the ideal but unrealistic "centralizing" case, considering the correlation significantly narrows the convergence gap, and further validate our conjectures that exploiting clients correlation benefits FL under heterogeneous settings.

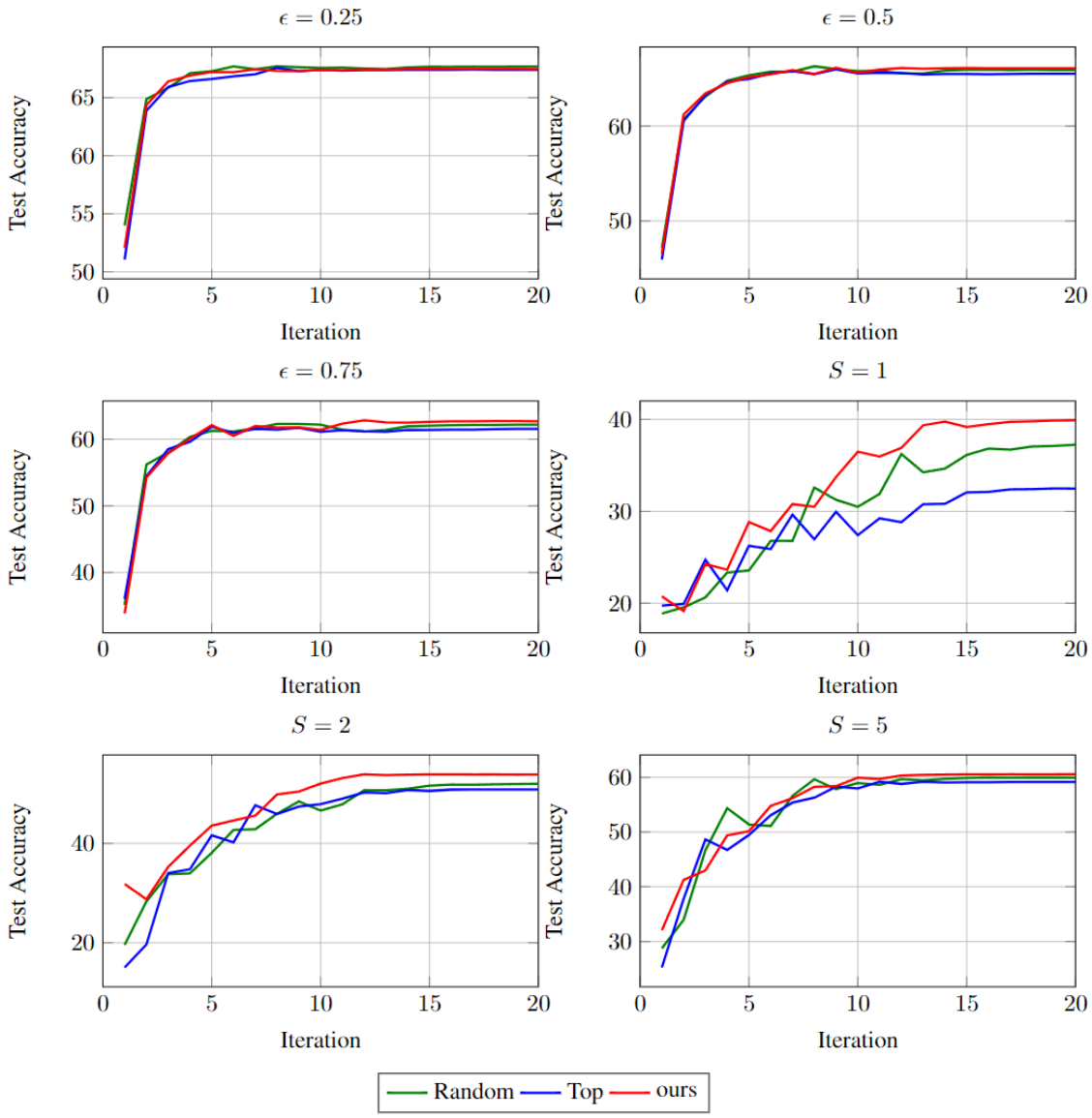

Figure 2.6: Ablation study on layer-wise client selection implementations.

Method	PS, S value			PB, ϵ value		
	1	2	5	0.25	0.5	0.75
centralizing	39.63	52.71	61.06	67.43	65.67	64.11
no-binning	24.55	42.93	56.05	67.63	65.17	63.03
binning	26.77	49.95	60.84	67.77	65.97	63.46

Table 2.5: Test accuracy resulting from integrating our client selection and quantization approaches.

2.6.6 The number of subsamples VS. The Achievable Minimum Sum-rate

Here, we present the experiment results demonstrating the effect of the number of subsamples (taken to estimate the correlation Σ_G) on the resulting achievable minimum sum-rate (the upper bound $\bar{R}^*(d)$ in (2.4.12)). Two layers with different sizes are taken as examples, each respectively has 16000 and 500 elements. For the layer having 16000 elements, we randomly take 8 groups of subsamples, each has the number of 8000, 3200, 1600, 800, 320, 160, 80 and 32. Similarly, for the layer having 500 elements, we randomly take 4 groups of subsamples, each has the number of 250, 100, 50 and 25. For each group of subsamples, we execute the iterative algorithm to compute the the upper bound $\bar{R}^*(d)$ in (2.4.12) and plot the results in Figure 2.7 and Figure 2.8.

We observe that for the layer with larger size, a more aggressive subsampling number can be chosen without sacrificing too much accuracy in terms of the resulting achievable min sum-rate. In Figure 2.7, the subsampling number can be set as $\frac{1}{200}$ of the total number. However, for the layer with fewer elements, the resulting achievable min sum-rate may suffer from a lot accuracy loss even when the subsampling number is set as $\frac{1}{5}$ of the total number, and the accuracy loss keeps increasing as the subsampling

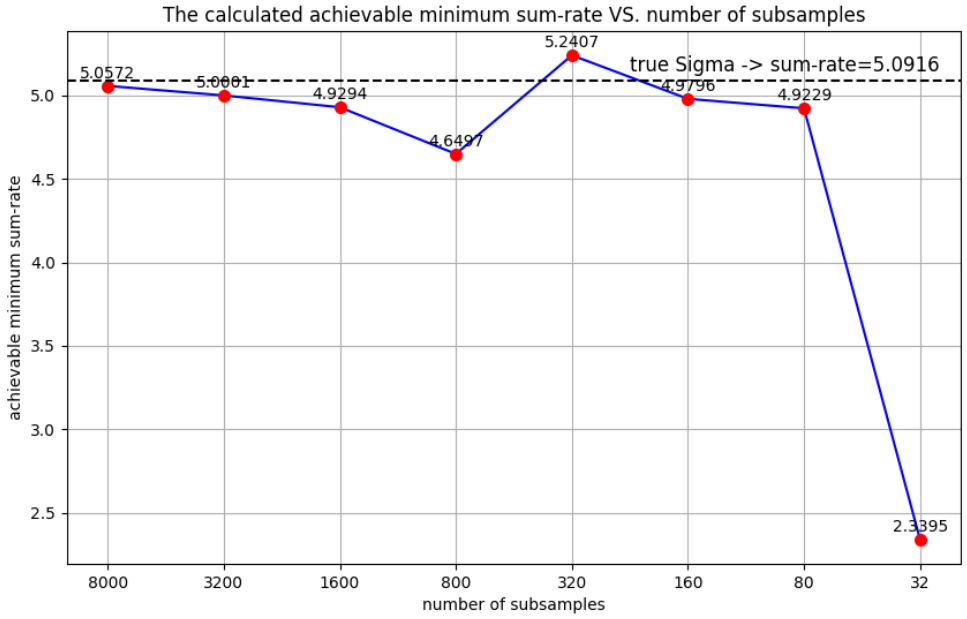

Figure 2.7: The achievable min sum-rate VS. the number of subsamples on the gradient layer with 16000 elements.

elements get fewer. Intuitively, we conjecture this effect as the statistic information of the gradients in large size layers follow closer to our assumptions, while for the small size layers, the assumptions are violated due to too few elements. A more detailed study about the relationship between the number of subsamples and the achievable min sum-rate is one of our future work.

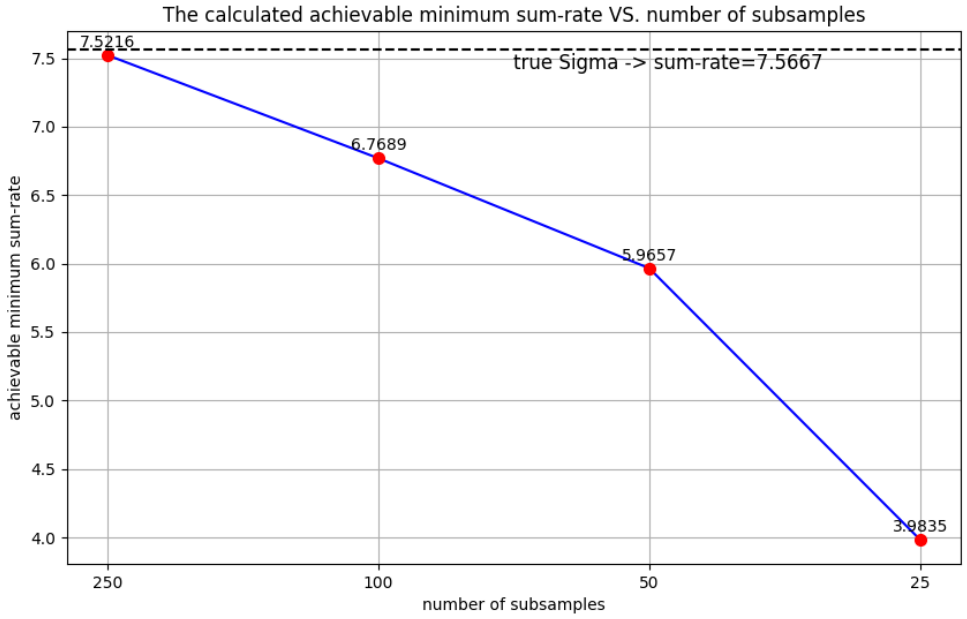

Figure 2.8: The achievable min sum-rate VS. the number of subsamples on the gradient layer with 500 elements.

2.7 Conclusion

This work studies the communication efficiency and convergence problem in FL by exploiting the clients' gradient correlation. From a rate-distortion perspective, we formulate the model update in single FL iteration as a multi-terminal source coding problem, and convert it to a convex optimization problem. With the derived convex optimization problem and upper and lower bounds expressions for the minimum sum-rate-distortion functions, we propose GCFed, an FL framework that achieves optimal rate allocation and client selection. Experimental results on different levels of heterogeneity verify that considering the gradient correlation improves communication efficiency and convergence in FL.

Chapter 3

M22: A Communication-Efficient Algorithm for Federated Learning Inspired by Rate-Distortion

3.1 Abstract

In federated learning (FL), the communication constraint between the remote clients and the Parameter Server (PS) is a crucial bottleneck. For this reason, model updates must be compressed so as to minimize the loss in accuracy resulting from the communication constraint. This paper proposes "**M**-*magnitude weighted* $L_{\mathbf{2}}$ *distortion* + **2** *degrees of freedom*" (M22) algorithm, a rate-distortion inspired approach to gradient compression for federated training of deep neural networks (DNNs). In particular, we propose a family of distortion measures between the original gradient and the reconstruction we refer to as "M-magnitude weighted L_2" distortion, and we assume that gradient updates follow an i.i.d. distribution – generalized normal or Weibull,

which have two degrees of freedom. In both the distortion measure and the gradient distribution, there is one free parameter for each that can be fitted as a function of the iteration number. Given a choice of gradient distribution and distortion measure, we design the quantizer to minimize the expected distortion in gradient reconstruction. To measure the gradient compression performance under a communication constraint, we define the *per-bit accuracy* as the optimal improvement in accuracy that one bit of communication brings to the centralized model over the training period. Using this performance measure, we systematically benchmark the choice of gradient distribution and distortion measure. We provide substantial insights on the role of these choices and argue that significant performance improvements can be attained using such a rate-distortion inspired compressor.

3.2 Introduction

Federated learning (FL) holds the promise of enabling the distributed training of large models over massive datasets while preserving data locality, guarantying scalability, and also preserving data privacy. Despite the great advantages promised by FL, the communication overhead of distributed training poses a challenge to contemporary networks. As the size of the trained models and the number of devices participating to the training is ever increasing, the transmission from remote clients to the parameter server (PS) orchestrating the training process becomes the critical performance bottleneck [31, 36]. In order to address this issue, the design of an effective gradient compression algorithm is of paramount importance. In this paper, we propose "**M**-*magnitude weighted* $L_{\mathbf{2}}$ *distortion +* **2** *degrees of freedom*" (M22) – a gradient compression algorithm inspired by rate-distortion principles. More specifically, M22

adopts model-wise sparsification for preliminary gradients dimension reduction and then performs gradient quantization for further compression using (i) a family of distortion metrics which provides higher precision for gradients of larger magnitudes and (ii) under the assumption that the gradients as i.i.d. samples from a distribution. We show that, by leveraging these two modelling choices, one can design an efficient gradient compressor in FL scenarios. More specifically, we measure the compression performance in terms of the *per-bit accuracy*, that is the largest improvement in accuracy that can be attained, on average, by one-bit communication between the remote clients and the PS. Through this performance measure, we are able to show that the superiority of M22 over other approaches in the literature.

3.2.1 Literature Review

In recent years, distributed learning has received considerable attention in the literature [11]. In the following, we shall briefly review the contributions dealing with communication aspects of FL, which are most relevant to the development of the paper. From a general perspective, FL consists of a central model which is trained locally at the remote clients by applying Stochastic Gradient Descent (SGD) over a dataset present at the clients. The local gradients are then communicated to the central PS for aggregation into a global model. Since this aggregation model does not require data centralization, it provides substantial advantages in terms of scalability, robustness, and security [36]. For these reasons, there has been a significant interest in developing effective and efficient FL algorithms [59, 71, 5].

When considering a concrete deployment of FL algorithms, the performance bottleneck is often found in the communication rates between the remote clients and the

PS. This scenario is often referred to as the *rate-limited FL* scenario [60]. In this case, some researchers put efforts on finding compression solutions in both communication directions [54, 12, 58], while most of the researchers dedicate to determine the relationship between the accuracy and the uplink communication rate [60, 53]. In order to limit the communication overhead in the rate-limited FL settings, compressing the local gradient at the remote clients has been proven to be effective [64]. Generally speaking, gradient compression algorithms can be divided in two classes: gradient sparsification [59, 71, 5, 69, 3, 38, 31, 25] and (ii) gradient quantization [54, 12, 58, 57, 31, 19, 10, 2, 35, 28, 64, 52, 4, 76].

Let us discuss these two approaches next.

Gradient sparsification: Gradient sparsification consists in selecting only a subset of the gradient entries for transmission. The sparsity of the resulting vector can be leveraged to reduce the communication load. Generally, sparsification is based on the magnitude of the gradients – top$K$ -and various strategies in selecting the sparsification threshold have been investigated in the literature [69, 3, 38]. In [31], the authors proposed an algorithm that applies a sparsification mask which is learned locally, using the dataset available at the remote clients. Another approach in obtaining a sparse gradient is by updating a subset of the network weights at each iterations: this approach was investigated in [25]. Gradient sparsity might be identified in some linear subset of the original space. For this reason, dimensionality reduction techniques followed by sparsification offer an effective approach. The authors of [28] considered a scheme in which each client performs local compression to the local stochastic gradient by count sketch via a common sketching operator. Random rotations were also considered in [25].

Gradient quantization: Gradient compression generally refers to the procedure of quantizing the gradient vector, that is mapping to a restricted support representation. Scalar quantization was considered in [4]. The authors of [76] proposed TernGrad: this algorithm uses ternary gradients to reduce communication cost. In other works in literature, the gradient compression is performed on the whole gradient vector, rather than entry-wise. For instance, the authors of [19] introduced vector quantization for SGD. Generally speaking, when gradient compression is employed, the training performance can be improved through error feedback [64, 52]. Partial model updates, together with sampling and quantization, were considered in [12]. To investigate the loss accumulation issue being introduced by the quantization-based approaches, the authors in [58] considered the interplay of model compression at both the remote clients and the central server. Sparse ternary compression for non-i.i.d. data was studied in [54] for the compression both uplink and downlink communications.

More naively, constraints in the communication capabilities between the remote clients and the PS can also be addressed by restricting the number of communication iterations between gradient updates [43, 26]. From an implementation-oriented perspective, [65] studied the effect of gradient quantization when constrained to a *sign-exponent-mantissa* representation. The design of good gradient quantizers sometimes relies on the assumption on the gradient distribution. Assuming that DNN gradient entries are i.i.d. distributed according to some distributions is a powerful approximation which is adopted in various contexts, from network pruning to inference modelling. Some authors assumed that gradients have i.i.d Gaussian [30] or Laplace [27] entries. Other authors considered distributions with two degrees of freedom, such as generalized Normal [13] or two-sided Weibull [17].

3.2.2 Contributions

In this paper, we consider the FL scenario where the communication constraint is applied on the transmission from the remote clients to the PS, while the transmission from the PS to the remote clients is unconstrained. In this scenario, we propose M22, a novel gradient compression algorithm inspired by rate-distortion principles. More specifically,M22 relies on two rate-distortion principles to design a scalar quantizer that meets the communication constraint:

• **M2**: **M**-*magnitude weighted* L_2 distortion measure– a choice of a distortion measure between the original and the compressed gradient: the "M-magnitude weighted L_2" distortion. This distortion promotes higher fidelity for higher gradient entries and reflects the practitioner's intuition that larger gradients have greater impact on the model updates.

• **2**: **2** *degrees of freedom* distribution fitting – the fitting of the gradient distribution uses a distribution having two degrees of freedom: the generalized normal distribution (GenNorm) distribution [13], or the two-sided Weibull distribution [17]. Such choice of distributions allows one to match the variance of the gradient sample distribution, as well as the tail decay as it evolves through the iteration number.

To measure the compression performance in the rate-limited FL setting, we introduce the concept of *per-bit accuracy* as the relevant performance measure for distributed training under communication constraints. The per-bit accuracy corresponds to the improvement in accuracy that a gradient compressed within R bits can provide to a given model. For the proposed algorithm – M22 – we provide a convergence proof under the assumption that the gradient entries are i.i.d. draws from the same distribution.

Also, we provide extensive numerical simulation results to show that M22 outperforms other approaches in the literature.

Notation: In the following, lowercase boldface letters (eg. $\mathbf{z}$) are used for column vectors and uppercase boldface letters (e.g., $\mathbf{M}$) designate matrices. The all-zero vector of dimension d is indicated as $\mathbf{0}_d$. We also adopt the shorthands $[m:n] \triangleq \{m, ..., n\}$ and $[n] \triangleq \{1, ..., n\}$. The p-norm of the vector $\mathbf{x}$ is indicated as $\|\mathbf{x}\|_p$.

Calligraphic scripts are used to denote sets (e.g., $\mathcal{A}$) and $|\mathcal{A}|$ is used to denote its cardinality.

The code for the numerical evaluations of this paper is provided online at `https://github.com/yangyiliu21/FL_RD`.

3.3 System Model

In the following, we consider the distributed training of a machine learning (ML) model across N devices where the communication between the remote device and the PS is limited to R bits per learner. We introduce the per-bit accuracy as the performance measure that allows for the comparison across gradient compression algorithms. Finally, the problem is specialized to the federated DNN training scenario.

3.3.1 Distributed Optimization Setting

Consider the setting with N clients, each possessing a local dataset $\mathcal{D}_n = \{\mathbf{d}_{nk}\}_{k \in [|\mathcal{D}_n|]}$ for $n \in [N]$ and wishing to minimize the *loss function* $\mathcal{L}$ as evaluated across all the

local datasets and over the choice of model $\mathbf{w} \in \mathbb{R}^d$, that is

$$\mathcal{L}(\mathbf{w}) = \frac{1}{\sum_{n\in[N]} |\mathcal{D}_n|} \sum_{n\in[N]} \sum_{\mathbf{d}_{nk}\in[\mathcal{D}_n]} \mathcal{L}(\mathbf{d}_{nk}, \mathbf{w}), \tag{3.3.1}$$

with $\mathcal{D}(\mathbf{w})$ positive defined. For the loss function in (3.3.1), we assume that there exists a unique minimizer $\mathbf{w}^*$ of (3.3.1), that is

$$\mathbf{w}^* = \underset{\mathbf{w}\in\mathbb{R}^d}{\operatorname{argmin}} \ \mathcal{L}(\mathbf{w}). \tag{3.3.2}$$

A common approach for numerically determining the optimal value in (3.3.2) in the centralized scenario is through the iterative application of (synchronous) stochastic gradient descent (SGD). In the (centralized) SGD algorithm, the learner maintains an estimate of the minimizer in (3.3.2), $\mathbf{w}_t$, for each time $t \in [T]$. The final estimate of (3.3.2) is $\mathbf{w}_T$. At each time $t \in [T]$, the estimate $\mathbf{w}_t$ is updated as

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \mathbf{g}_t, \tag{3.3.3}$$

for $\mathbf{w}_0 = \mathbf{0}_d$, where η_t is an iteration-dependent step size η_t called *learning rate*, and where $\mathbf{g}_t$ is the stochastic gradient of $\mathcal{L}$ evaluated at $\mathbf{w}_t$, that is

$$\mathbb{E}\left[\mathbf{g}_t\right] = \sum_{\mathbf{d}_k\in\mathcal{D}} \mathbb{E}\left[\nabla\mathcal{L}(\mathbf{d}_k, \mathbf{w}_t)\right]. \tag{3.3.4}$$

In (3.3.4), $\nabla\mathcal{L}(\mathbf{d}_k, \mathbf{w}_t)$ denotes the gradient of $\mathcal{L}(\mathbf{d}_k, \mathbf{w}_t)$ at $\mathbf{w}_t$ as evaluated over the dataset $\mathcal{D} = \bigcup_{n\in[N]} \mathcal{D}_n$; the expectation is over the randomness in the evaluation of the gradient – e.g., minibatch effects.

In the FL setting, given that the datasets $\mathcal{D}_n$ are distributed at multiple remote clients, the SGD algorithm as in (3.3.3) has to be adapted as follows. First (i) the PS transmits the current model estimate, $\mathbf{w}_t$, to each client $n \in [N]$, then (ii) each client $n \in [N]$ accesses its local dataset $\mathcal{D}_n = \{(\mathbf{d}_n(k), v_n(k))\}_{k \in [[|\mathcal{D}_n|]}$ and computes the stochastic gradient, $\mathbf{g}_{nt}$, as in (3.3.4) and communicates it to the PS. Finally (iii) the PS updates the model estimate as in (3.3.3) but where $\mathbf{g}_t$ is obtained as

$$\mathbf{g}_t = \frac{1}{N} \sum_{n \in [N]} \mathbf{g}_{nt}. \tag{3.3.5}$$

The distributed version of SGD for the FL setting is referred to as *federated averaging* (FedAvg) [42].

3.3.2 Federated Learning with Communication Constraints

Customarily, in the FL setting, the communication is assumed to take place over some noiseless, infinity capacity link connecting the PS and the remote clients and vice-versa. In a practical scenario, the clients model wireless mobiles, IoT devices, or sensors which have significant limitations in the available power and computational capabilities. In these scenarios, we can still assume that clients rely on some physical and MAC layers' protocols that are capable of reliably delivering a certain payload from the clients to the PS.

For this reason, in the following, we assume that the communication between each of the remote clients and the PS takes place over a rate-limited channel of capacity $d\mathsf{R}$, where d is the dimension of the model in Section 3.3.1. In other words, each client can communicate up to $d\mathsf{R}$ bits for each iteration $t \in [T]$.

In the following, we refer to the operation of converting the d-dimensional gradient vector $\mathbf{g}_{nt}$ to a $d\mathsf{R}$ binary vector as *compression*. Mathematically, compression is indicated though the operator

$$\text{comp}_{\mathsf{R}}: \quad \mathbb{R}^d \to [2^{d\mathsf{R}}]. \tag{3.3.6}$$

Similarly, the reconstruction of the gradient is denoted by $\text{comp}_{\mathsf{R}}^{-1}$. Note that in (3.3.6), R indicates the number of bits per model dimension.

Assumption 1 Homogeneous remote clients. *We assume that (i) all clients are subject to the same communication constraint and (ii) all clients employ the same compressor during the whole iteration process.*

Generalizing the results in the paper to the more general scenario in which remote clients have different communications constraints and use different compressors is rather straightforward. More specifically, each user would use a compressor matched to its communication constraint and the PS would reconstruct the gradient accordingly. After all gradients have been reconstructed, they can be accumulated as in (3.3.5). We adopt Assumption 1 for clarity of analysis. From an implementation perspective, the results could be generalized to more complicated scenarios.

Under the assumptions in Assumption 1, the model update in (3.3.3) can be reformulated as

$$\begin{aligned}\widehat{\mathbf{w}}_{t+1} &= \widehat{\mathbf{w}}_t - \eta_t \widehat{\mathbf{g}}_t \\ \widehat{\mathbf{g}}_t &= \frac{1}{n} \sum_{n\in[N]} \text{comp}_{\mathsf{R}}^{-1}(\text{comp}_{\mathsf{R}}(\mathbf{g}_{tn})),\end{aligned}$$

with $\widehat{\mathbf{w}}_0 = \mathbf{0}$. For the model update rule in (3.3.2), it is possible to derive similar convergence guarantees to that for the unconstrained problem in (3.3.3) – see [64].

3.3.3 Compression Performance Evaluation

In the following, we are interested in characterizing the compression performance in terms of the loss of accuracy as a function of the communication rate. More formally, given the model estimate $\widehat{\mathbf{w}}_t$ and the gradient estimate $\widehat{\mathbf{g}}_t$, we wish to determine

$$\mathsf{G}_{\mathsf{R}}(\widehat{\mathbf{w}}_{t+1}) = \min_{\mathrm{comp}_{\mathsf{R}}, \mathrm{comp}_{\mathsf{R}}^{-1}} \mathcal{L}(\widehat{\mathbf{w}}_{t+1}), \tag{3.3.7}$$

where $\widehat{\mathbf{w}}_{t+1}$ is obtained as in (3.3.2).

In general, we are interested in determining the effect of compression through the SGD iterations: to this we define

$$\Delta(T, \mathsf{R}) = \frac{1}{d\mathsf{R}} \frac{\mathbb{E}[\mathcal{L}(\mathbf{w}_0) - \mathsf{G}_{\mathsf{R}}(\widehat{\mathbf{w}}_T)]}{T}, \tag{3.3.8}$$

as the *per-bit accuracy*. In (3.3.8), the expectation is over all sources of randomness in the compression scheme, as well as the model initialization, denoted as $\mathbf{w}_0$. This definition (3.3.8) corresponds to the overall loss of accuracy due to compression of the gradient to R-bits per dimension at the training horizon, T. By comparing $\Delta(T, \mathsf{R})$ for different values of T and R on the same optimization problem and a number of remote clients, one can gauge the impact of the communication constraint over the training process at hand.

Generally speaking, the minimization in (3.3.7) is too complex, as the loss function $\mathcal{L}$ is generally non-convex in the model $\mathbf{w}$. Additionally, lacking a statistical

description of the SGD process, it is impossible to resort to classical compression techniques from information theory. To address these difficulties, later in Section 3.4, we consider a rate-distortion approach in which we simplify the minimization in (3.3.7) for the case of DNN training by considering a family of distortion measures which captures the loss of accuracy as a function of the gradient magnitude, and assume that the DNN gradients are i.i.d. draws from the GenNorm distribution or Weibull distribution. These two simplifications yield a compressor design which shows improved performance over other compressors considered in the literature.

3.3.4 DNN Training

While the problem formulation in Section 3.3.3 is rather general, in the remainder of the paper, we shall only consider the scenario of DNN training. More specifically, we consider a simple convolutional neural network (CNN) and two widely-used architectures – ResNet18 and VGG16. The three networks above are trained for image classification and other computer vision tasks over the CIFAR-10 dataset. In Table 3.1, we list the parameter information of our three models: CNN, ResNet18, and VGG16. Detailed layer-wise parameter information could be found in Appendix B. During our training, the CNN model in is trained using SGD with learning rate 0.0001 and cross-entropy loss. The ResNet18 and VGG16 models are trained using Adam with learning rate 0.001 and 0.00005, respectively. Other training hyper-parameters, including mini-batch size and etc. could be found in Table 3.2.

In our FL setting, we randomly split the CIFAR-10 dataset, which has $50,000$ samples in the training set and $10,000$ samples in the test set, and allocate to two or five remote clients with equal quantity, thus the data distributions of the local datasets

are approximately the same. Additionally, since we focus on validating the algorithm effectiveness, we do not consider clients drop-out, meaning the server requires all the remote clients to report their local updates once they finish three local training epoches. This combination of hyper-parameters ensures that the training framework fits into the FL scheme, and facilitates the convergence of the global model.

Remark 1 *There are numerous FL training settings of practical relevance in which the choice of hyper-parameters is drastically different from the one we consider here. We believe that the three networks above represent a simple and yet meaningful benchmark for the proposed approach, M22. We leave the testing on more sophisticated ones for future work.*

Architectures	Layers	Total Params	conv_layer	dense_layer
CNN	44	552,874	549,280	0
ResNet18	98	11,184,068	11,171,008	0
VGG16	32	33,638,218	14,714,688	18,882,560

Table 3.1: A summary of the parameter information of the models in Section 3.3.4.

Model	CNN	ResNet18	VGG16
Dataset	CIFAR-10		
Optimizer	SGD	Adam	Adam
Learning Rate	0.01	0.001	0.0005
Momentum	0	0	0
Loss	Categorical Cross Entropy		
Mini-Batch Sizes	64	64	32

Table 3.2: Parameters and hyperparameters used for the training of the DNN models.

3.3.5 Further Comments

Before delving further in the paper, let us clarify what aspects of the problem setting of Section 3.3.2 will not be considered in the remainder of the paper.

• **Lossless universal compression.** In the following, we assume that gradients are transmitted after compression. Note that one could further apply some lossless universal compression algorithm to further reduce the communication load. Such algorithm are readily available and very efficiently exploit the redundancy in the data to further reduce the transmission payload [45]. For simplicity, we do not consider this further compression opportunity as it will complicate the evaluation of the effective dimension of the compressed gradient.

• **Varying the number of remote client.** In the remainder of the paper, we consider the case of two or five remote clients and do not consider the effect of client scheduling (having a varying number of remote clients). As the number of remote clients varies, the overall compression error increases and thus the choice of learning rate, mini-batch size and other hyper-parameters need to be adjusted accordingly. Although providing insight on this aspect of the hyper-parameter choice is valuable, this is outside the scope of the paper.

• **Layer-wise dependency.** In the following, we compress the gradients by assuming that they are independent across iterations, although dependencies of the gradients across layers exists. For instance, in [6], it is shown that the per-layer distribution is conditionally dependent only on the weights in the previous layer. For simplicity, in the following, we do not consider this dependency. The design of a version of the proposed approach taking advantage of this correlation is left for future research.

The following aspects are briefly touched upon in Section 3.6.3, although we do

not provide a full treatment of these matters.

• **Heterogeneous communication constraints – Figure 3.5:** In our theoretical analysis as well as in our numerical experiments, we consider the case in which remote users are subject to the same communication constraint – R. In Figure 3.5 we discuss the effect of different communication constraints for different users.

• **Heterogeneous local datasets – Figure 3.6:** In our analysis and simulations, we assume that all remote clients have the same data distribution. In practical scenarios, the heterogeneity of the local dataset is a problem of concern. In Figure 3.6 we consider the scenario in which users have heterogeneous data: each client has a favorite label which is disproportionately represented in the local dataset.

3.4 A Rate-distortion Approach to DNN Gradient Compression

In this section, we introduce the main ingredients of the proposed approach M22. These ingredients further clarify the rate-distortion principles employed in the design of the optimal compressor for DNN gradients.

Since (3.3.7) maximizing the per-bit accuracy in (3.3.8) as in Section 3.3.3 is generally intractable, we instead simplify the problem as follows. We choose (i) a distribution to approximate the gradient entries, and (ii) a distortion that correlates with the loss in accuracy in (3.3.7) when compressing the original weights. Once these two elements have been selected – that a gradient distribution and a gradient distortion measure – the compressor in (3.3.6) is chosen as the quantizer which minimizes the chosen distortion for the given gradient distribution, as in the classic [39].

More details about the choice of gradient distribution, distortion measure, and quantizer design are discussed in this section.

3.4.1 Gradient Distribution

Let us assume that gradient entries are well-approximated as i.i.d. random variables from a certain distribution $P_{G,t}$, which varies with $t \in [T]$. We notice that mean field theory has provided a partial validation of this assumption. In series of papers [44, 6, 46], it has been shown that DNN weights in a given layer become indistinguishable as the number of SGD steps grow large in various regimes. In practice, meaningful statistics can be obtained from the gradient realization, which justifies the adoption of this assumption from a practical perspective.

Some efforts have been made to characterize the gradient distribution such as Laplace distribution [27] and Gaussian distribution[33, 41]. We refer to these distributions as one-parameter distribution due to the fact the mean of the gradients are universally assumed to be zero, leaving the only degree of freedom to be the scale of the distribution. Consequently, we believe that they do not provide sufficient modeling capability to approximate the gradient distribution as it evolves through the iteration process. For this reason, we follow two approaches that approximate the gradients using either the generalized normal distribution (GenNorm) distribution in [13], or using a two-sided Weibull (double-Weibull or d-Weibull) distribution in [17].

Compared with the conventional one-parameter distributions, in addition to satisfying the symmetric property, both GenNorm and d-Weibull distributions have an extra degree of freedom: the shape parameter, which substantially strengthens the layer-wise approximation capability of gradients in different models throughout the

training iterations. Note that the generalized normal distribution encompasses the Laplace distribution when the shape parameter $\beta = 1$ and normal distribution when the shape parameter $\beta = 2$ as special cases. The power density function (PDF) of the GenNorm distribution is described as

$$f(x, \mu, s, \beta) = \frac{\beta}{2s\Gamma(1/\beta)}\, e^{-(|x-\mu|/s)^{\beta}}, \tag{3.4.1}$$

where μ and s denote the mean and scale respectively, $\beta > 0$, and Γ represents the gamma function: one commonly used extension of the factorial function to complex numbers. When the shape parameter $1 < \beta < 2$, the distribution is leptokurtic and has fatter tail than the normal distribution. Besides the tails, the shape parameter also controls the peakedness, as it converges point-wise to a uniform distribution on $(\mu - \alpha, \mu + \alpha)$ when $\beta \to \infty$. Despite the fact that the unimodal and symmetric properties of the GenNorm distribution coincide with the nature of DNN gradient values [10], Fu et al. [17] proposed to approximate the gradients with a d-Weibull distribution because of its central tendency and long-tails characteristics. The PDF of the *d-Weibull* distribution is described as

$$f(x, \mu, s, k) = \frac{k}{2s}\left(\frac{|x-\mu|}{s}\right)^{(k-1)} e^{-(|x-\mu|/s)^{k}}, \tag{3.4.2}$$

where k is the shape parameter, and the restriction $k \in (0, 1]$ guarantees the monotony of Weibull family distributions. Comparing with the Gaussian distribution and Laplace distribution, d-Weibull distribution allows the approximated distributions to be more centralized and longer-tailed, same as the GenNorm distribution. This effect corresponds to the empirical evaluation on tested models by introducing the

shape parameter. An example of the fitting the distributions to the gradients empirical histogram is provided in Table 3.3, where the right columns represents the simulation when more aggressive top$K$ sparsification is applied comparing to the top panel. From an empirical validation perspective, the GenNorm distribution fits the gradient histogram better than the one-parameter distributions when most of the gradients are preserved, while the d-Weibull is more suitable as the approximation distribution when more aggressive sparsification is applied.

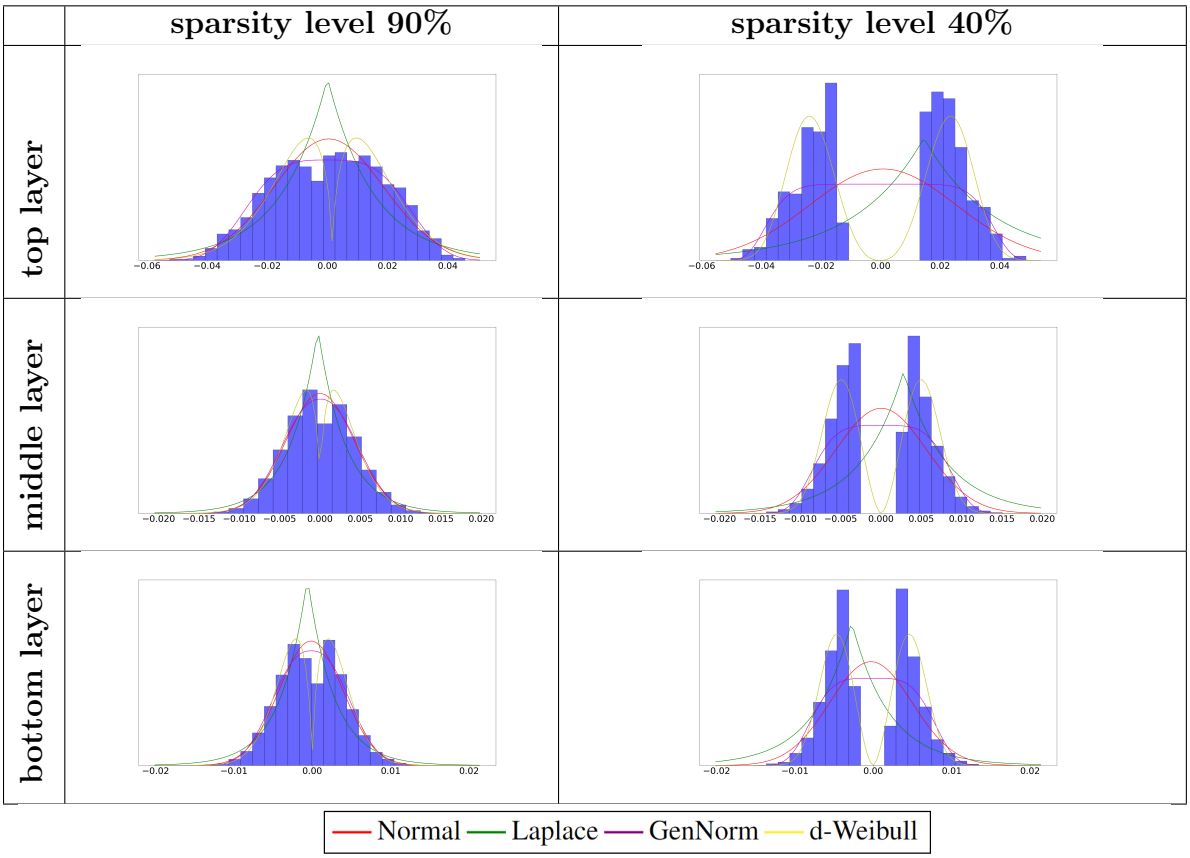

Table 3.3: Examples of the sample gradient distribution and its fitting as discussed in Section 3.4.1 for CNN in Appendix B.

To better illustrate the fitting performance, let us consider Table 3.3: in each row

correspond to one layer of the CNN model in Appendix B – 42,32, and 18– and in each column a different sparsification level – 90% and 40%. In each table entry, we plot the sample histogram of the sparsified gradient, the fitting using the normal, Laplace, GenNorm, and d-Weibull fitting. The legend of the figures is provided at the bottom of Table 3.3. From this table, two observations emerge: (i) the gradient distribution in higher layers have higher tails than Gaussian which are well fitted through a generalized normal, and (ii) high sparsification levels result in a gap in the sample distribution of the sparsified gradient which is well modeled through a d-Weibull distribution.

Benefiting from the extra shape parameter, GenNorm distribution fits these gradients' histograms better with a flatter peak and heavier tails, compared to Laplace and normal distributions. When the sparsification level is 90%, only a small number of small-magnitude gradients are set to zero while 90% of the gradients are preserved – this is shown in the left column of Table 3.3. The tail fitting advantage of GenNorm distribution is even more obvious as shown in the right column of Table 3.3, where the sparsification level is at 40%. The d-Weibull distribution also fits the tails perfectly, while its bi-modal characteristic makes it potentially a better candidate in these low sparsification level cases. Overall, we believe that the choice of these two distributions covers all sparsification levels in different layers of all the models we have tested.

3.4.2 Distortion Measure

The second ingredient of M22 is a judiciously chosen distortion measure between the original gradient and its reconstruction. We believe that this ingredient is indeed the most innovative contribution of the paper. Note that the existing works mostly adopt

L_2 loss when measuring the distortion between the original gradients and perturbed gradients. Some exceptions [27] consider an L_1 distortion measure in the context of network pruning.

In devising a distortion metric for our quantizer design, two aspects are taken into consideration: (i) the practitioner perspective on effective sparsification and (ii) existing bounds on the accuracy loss from gradients perturbation. In the literature, top$K$ sparsification consists in setting all but the K largest weights of the gradient entries to zero at each iteration. It is well known among the ML practitioners that top$K$ sparsification can be effectively used to reduce the dimensionality of the gradient updates while resulting in only a small loss in accuracy [64, 5]. Another approach for gradient compression is uniform quantization with non-subtractive dithering [4, 76, 67, 19]. This approach finds its theoretical foundations in works such as [34, eq. (8)] which provides a bound on the loss in accuracy as a function of the L_2 perturbation of the DNN weights.

Under extreme communication constraint circumstances, using L_2 loss performs exceptionally well. However, we found that the model convergence speed slows down when the constraint is relaxed using the same L_2 loss. Some intuitions behind this phenomenon could be gained as when the constraint is relaxed, we should penalize the loss harder in order to speed up the converging process. Therefore, in addition to the conventional L_2 term, we equip the loss with another term with a hyper-parameter that could adjust the penalizing intensity of perturbed gradients. Details and the function of this term revealed by the simulation results are discussed in Section 3.6.2.

Given the intuition of magnitude and two compression approaches in which either only the magnitude of the gradients is considered or the L_2 distortion of the

gradients is used for quantizer design, we propose a new class of distortion measures combining these approaches. That is, the "M-magnitude weighted L_2" distortion, mathematically defined as

$$d_{M-L_2}(\mathbf{g}, \widehat{\mathbf{g}}) = \frac{1}{d} \sum_{j \in [d]} |g_j|^M \|g_j - \hat{g}_j\|_2, \tag{3.4.3}$$

where g_j and $\hat{g}_j$ denote the j-th elements of $\mathbf{g}$ and $\widehat{\mathbf{g}}$ (gradient and quantized vectors), respectively. Note that the parameter M is again a hyper-parameter that can be used to tune the distortion to the particular iteration and instance of training. Similarly to the choice of gradient distribution, the choice of distortion measure provides us with a degree of freedom that can be adaptively adjusted to improve the training performance.

The distortion in (3.4.3) can be re-interpreted as an assumption on the sensitivity of the loss function in (3.3.1) in the neighborhood of the gradient

Assumption 2 *[$M - L_2$ sensitivity] Consider a differentiable loss function $\mathcal{L}(\mathbf{w})$, then the $M - L_2$ sensitivity of $\mathcal{L}(\mathbf{w})$ is defined as*

$$\mathcal{L}(\mathbf{m}) - \mathcal{L}(\mathbf{w}) \geq d_{M-L_2}(\nabla \mathcal{L}(\mathbf{w}), \mathbf{m} - \mathbf{w}) \tag{3.4.4}$$

In plain words, Assumption 2 can be seen as a variation of the definition of convexity where the RHS considers a specific form of divergence to measure the distance between the gradient direction and the difference vector.

3.4.3 Quantizer design

Given our choice of gradient distribution and the gradient distortion measure, as in Section 3.4.2 and Section 3.4.1, one can then construct a quantizer using the classic LGB algorithm [39]. In M22, we actually apply sparsification before quantization as it is more computationally efficient to code the zero values using a run-length encoding. The K-means/LGB algorithm for the proposed class of distortions in (3.4.3) takes a surprisingly simple form.

$$c_{k+1}(i+1) = \frac{\int_{t_k(i)}^{t_k(i+1)} g^{M+1} \,\mathrm{pdf}\,(g)dg}{\int_{t(i)}^{t(i+1)} g^{M} \,\mathrm{pdf}\,(g)dg}, \tag{3.4.5}$$

$$t_{k+1}(i+1) = \frac{c_k(i+1) + c_k(i)}{2}, \tag{3.4.6}$$

for $i \in [1, 2^{\mathsf{R}}]$ where 2^{R} is the number of quantization levels, $\mathrm{pdf}(g)$ denotes the distribution fitted to the gradient vector, $c(.)$ and $t(.)$ represent the quantization centers and thresholds, respectively.

A plot of the change of quantization centers and thresholds regions versus the change of M values modeled by GenNorm distribution is presented in Figure 3.1. Due to the symmetry in the quantizer design, only the positive regions are shown. Note that the first quantization region to the left corresponds to the positive half of the sparsification region. Note that larger choices of M result in more sparsified quantization regions, i.e., diverging from the center quantization bin. This corresponds to the objectives of M22: as M increases, the quantization preserves larger gradient magnitudes better.

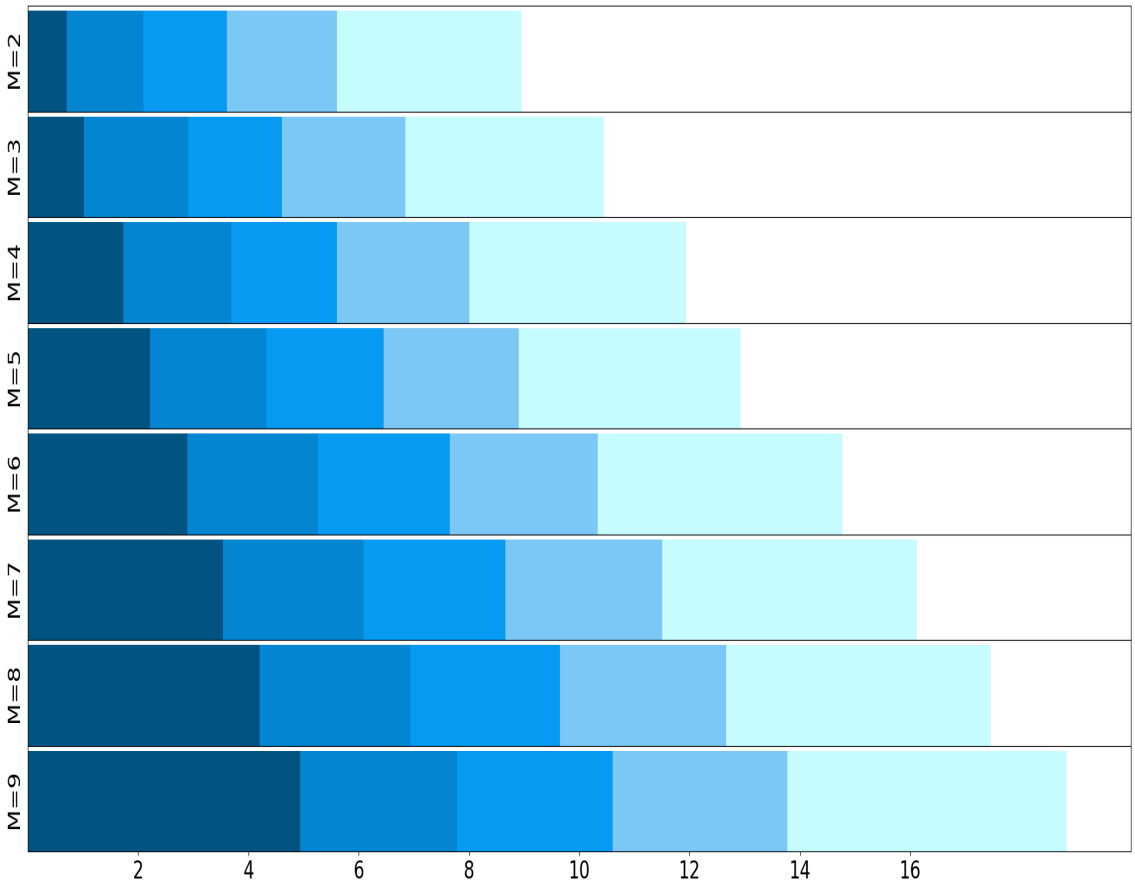

Figure 3.1: The positive region of the scalar quantizer described in Section 3.4.3 for the GenNorm distribution. Each row corresponds to a different value of M with M increasing with the row index. Note that the quantizer is symmetric around zero. Also, the first quantization region corresponds to the positive half of the sparsification region.

3.5 M22: the Proposed Approach

With the demonstration of our guiding rate-distortion principles in Section 3.4, we are finally ready to introduce M22 – our approach inspired by rate-distortion for the design of the optimal compressor for DNN gradients.

3.5.1 M22 Pseudo-code

We start by reminding the reader that the model update in M22 follows that of Federated Averaging (FedAVG) [42] as in (3.3.2). The novelty of M22 is in the design of the compressors in (3.3.6). See Algorithm 2.

3.5.2 Further Comments

In the workflow pipeline of M22, various elements could affect the performance. The hyper-parameters of FL setting, including the number of remote clients, the mini-batch size, the number of local training epoch and the model optimizer choice. As described in Section 3.3.4, we chose a naive setting for the major part of our simulations, merely to show the effectiveness of M22. It has been tested that M22 could be adapted in more complicated FL settings, including where partial clients are selected in each round, multiple training epochs are performed locally and the local datasets are heterogeneous. The choice of M is also considered as a hyper-parameter of the compression setting.

Inspired by the [64], where memory is equipped with SGD under compression, we implemented memory with M22. Each local client keeps a copy of the difference between its trained local model and the compressed version. This client-wise distinct

Algorithm 2 M22 algorithm

Input: weight initialization: $\mathbf{w}_0$, Compression rate R, sparsification level $K = \kappa$, set of learning rates $\{\eta_t\}_{t\in T}$
Output: optimal weight estimate: $\mathbf{w}_T$
for $t \in [T]$ **do**
    PS broadcasts $\mathbf{w}_t$ to N remote clients *// T iterations, t iteration index*
    for $n \in [N]$ **do**
        client n evaluates local gradient $\mathbf{g}_{nt}$ over the local dataset $\mathcal{D}_n$ as in (3.3.5)
        // N clients, n client index

$$\mathbf{g}_{nt} = \nabla\mathcal{L}(\mathcal{D}_n, \mathbf{g}_t) \tag{3.5.1}$$

        client n applies top$K$ sparsification for $K = \kappa$ to $\mathbf{g}_{nt}$

$$\mathbf{g}'_{nt} \leftarrow \text{top}K(\kappa, \mathbf{g}_{nt}) \tag{3.5.2}$$

        for $l \in [L]$ **do**
            client n selects the gradient entries in $\mathbf{g}'_{nt}$ from layer l, call it $\mathbf{g}'_{nt}(l)$
        // L layers, l layer index

            client n fits distribution: GenNorm as in (3.4.1) or d-Weibull as in (3.4.2)
            client n compresses using the scalar quantizer of rate R in Sec. 3.4.3

$$\mathbf{g}''_{tn}(l) = \text{comp}_{\mathsf{R}}(\mathbf{g}'_{tn}(l)) \tag{3.5.3}$$

        end for
        client n transmits $\mathbf{g}''_{tn}$ to the server

$$\mathbf{g}''_{tn} = [\mathbf{g}''_{tn}(1) \dots \mathbf{g}''_{tn}(L)] \tag{3.5.4}$$

    together with the position of sparse entries and fitting parameters in each layer l
    end for
    PS reconstructs the gradient vector of all N clients as in (3.3.2), accumulates and produces a model update

$$\widehat{\mathbf{g}}_t = \frac{1}{n}\sum_{n\in[N]} \text{comp}^{-1}_{\mathsf{R}}(\mathbf{g}''_{tn}) \tag{3.5.5}$$

$$\widehat{\mathbf{w}}_{t+1} = \widehat{\mathbf{w}}_t - \eta_t\widehat{\mathbf{g}}_t$$

end for

difference is added back to downloaded global model by each client before the local training. The problems with this mechanism in the FL setting are memory accumulation and local optima. The memory stored by the clients would be accumulated throughout the training process, which could cause memory explosion. Additionally, the clients could converge to local optima in different directions. The server aggregation serves helps rectify the progressing directions to a global one, while an uncalibrated memory implementation would cause each client heading to different directions again. In our simulations, a fine-tuned memory weights could help the model convergence but much less significantly comparing with tuning the M value.

3.5.3 Algorithm Convergence Analysis

In this section we study the convergence of the M22 under some rather standard assumptions on the optimization problem. The proof relies in showing that the specific choice of the scalar quantizer in Section 3.4.3 provides an unbiased estimate of the true gradient. The proof is not particularly novel but it leads to a precise formulation of the assumption that the gradient entries have an i.i.d. distribution.

Assumption 3 (Regularity of $\mathcal{L}(\mathbf{v})$) *The function $\mathcal{L}(\mathbf{v})$ in (3.3.1) is assumed:*

- *convex, that is, for all $\lambda \in [0, 1]$, it holds that*

$$\mathcal{L}(\lambda \mathbf{w}(1-\lambda)\mathbf{v}) \leq \lambda \mathcal{L}(\mathbf{w}) + (1-\lambda)\mathcal{L}(\mathbf{v}). \tag{3.5.6}$$

- *L-Lipschitz, that is*

$$\|\nabla \mathcal{L}(\mathbf{w}) - \nabla \mathcal{L}(\mathbf{v})\|_2 \leq L\|\mathbf{w} - \mathbf{v}\|_2. \tag{3.5.7}$$

With respect to the local stochastic gradients, we assume that all d gradient have the same distribution with zero mean and fixed variance.

Assumption 4 (i.i.d. gradient entry assumptions) *At iteration t and for all remote clients $n \in [N]$ observes a local gradient $\mathbf{g}_{tn}$ is comprised of i.i.d. entries with zero mean and variance σ_t^2, that is*

$$\mathbb{E}[\mathbf{g}_{tn}(i)] = 0 \tag{3.5.8}$$

$$\mathbb{E}[\mathbf{g}_{tn}(i)^2] = \sigma_t^2 \tag{3.5.9}$$

Additionally, we assume that the gradient distribution is symmetric.

Note that the Assumption 4 should be read as follows: conditioned on the network initialization, it still holds that $\mathbf{g}_{tn} = \nabla\mathcal{L}(\mathcal{D}, \mathbf{w}_t)$, but averaged over all initializations, the distribution of each gradient entry is i.i.d..

Theorem 1 *Consider the optimization problem in* (3.3.2) *under Assumption 3 and Assumption 4. Further, assume that the variance of each gradient entry is σ^2. For this setting, consider applying the algorithm M22 as detailed in Section 3.5.1 with*

$$\nu \leq \frac{1}{L}. \tag{3.5.10}$$

Then it holds that

$$\mathbb{E}[\mathcal{L}(\bar{\mathbf{w}}_T)] - \mathcal{L}(\mathbf{w}^*) \leq \frac{\|\mathbf{w}_0 - \mathbf{w}^*\|_2^2}{2\nu T} + \frac{\nu(d-1)^{M/2}}{T2^{2\mathsf{R}}}(\sum_{t\in[T]} \gamma_t + \epsilon), \tag{3.5.11}$$

where $\bar{\mathbf{w}}_T = 1/T \sum_{t\in[T]} \mathbf{w}_t$, $\epsilon \in \mathbb{R}^+$, *and for* $\gamma_t = 7.404(d-1)^{M/2}\sigma_t^3$. *In particular, if we take the*

$$\gamma_t = (d-1)^{M/2} \frac{\Gamma(\frac{3+\beta_t}{\beta_t})^3}{3\ s\ \Gamma(1/\beta_t)} \sigma_t^2 \tag{3.5.12}$$

$$\gamma_t = (d-1)^{M/2} \frac{9}{3\ s_t^4} \sigma_t^2 \tag{3.5.13}$$

Further, let the fitting parameter for at GenNorm and the d-Weibull be β_t *and* s_t, *while variance is* σ_t^2.

for the GenNorm distribution in (3.4.1) *and the d-Weibull distribution in* (3.4.2), *respectively.*

Proof: By construction of the scalar quantizer Section 3.4.3– which is a centroid quantizer – and the symmetry of the gradient distribution, we have that the sparsified and quantized gradient and the quantization error are uncorrelated. This means that the approximate gradient $\mathbf{g}_t$ in line 15 of Algorithm 2 is an unbiased version of the true gradient, that is

$$\mathbb{E}[\widehat{\mathbf{g}}_t] = \nabla \mathcal{L}(\mathcal{D}, \mathbf{w}_t). \tag{3.5.14}$$

Next, we wish to upper bound the error variance after sparsification and quantization. We begin by noting that in (3.4.3) we have

$$d_{M-L_2}(\mathbf{g}, \widehat{\mathbf{g}}) \leq (\max_i \mathbf{g}_i)^M \|\mathbf{g} - \widehat{\mathbf{g}}\|_2, \tag{3.5.15}$$

so that the $M - L_2$ distortion in (3.4.3) can be upper bounded by the MSE distortion multiplied by the largest element in the vector. From this it follows that the minimum expected $M - L_2$ distortion is upper bounded by the minimum MSE distortion

multiplied by the expected value of the largest gradient entry to the power of M. Since the expected value of the largest gradient entry only depends on the gradient distribution, we have that this outer bound is minimized by the scaler quantizer that minimized the MSE distortion. For this setting, we can use the Shannon lower bound to obtain an approximation of the quantization error performance as

$$\mathbb{E}[\|\mathbf{g} - \widehat{\mathbf{g}}\|_2^2] \leq 7.404\sigma_t^2 2^{-2\mathsf{R}} + \epsilon \tag{3.5.16}$$

for some ϵ that decreases as R increases [22].

For the result in (3.5.13) we use the approximation in [49] for the high rate regime

$$\mathbb{E}[\|\mathbf{g} - \widehat{\mathbf{g}}\|_2^2] \approx \frac{1}{12 \cdot 2^{2\mathsf{R}}}(\int_{\mathbb{R}} \sqrt[3]{F_g(g)}\mathrm{g})^3. \tag{3.5.17}$$

The remainder of the derivation is focus on the GenNorm and d-Weibull distribution.

For the GenNorm we have use the fact that

$$\int_0^\infty e^{-x^n}\, dx = \frac{1}{n}\int_0^\infty u^{1/n-1}e^{-u}\, du = \frac{1}{n}\Gamma(1/n) = \Gamma(1+1/n). \tag{3.5.18}$$

and for the double-Weibull we use

$$\int_0^\infty \sqrt[3]{k}x^{\frac{k-1}{3}}e^{-\frac{13}{x^k}}\, dx \leq \int_0^\infty kx^{k-1}e^{-\frac{13}{x^k}}\, dx = 3 \tag{3.5.19}$$

The last step of the proof is concerned with bounding the largest element of the

gradient vector. To this end, we consider the classic bound

$$\mathbb{E}(\max_i X_i) \leq \mu + \sigma\sqrt{n-1}, \tag{3.5.20}$$

which, combined with Jensens' inequality, yields

$$\mathbb{E}[(\max_i \mathbf{g}_i)^M] \leq \sigma_t (d-1)^{M/2}. \tag{3.5.21}$$

From the expression in (3.5.11), we see that the sub-optimality gap decreases as $1/T$ and $1/2^{2\mathsf{R}}$ as one would expect from quantized SGD. The dependency on M does not appear in the analysis in Theorem 1 as this dependency is not captured in the modeling of the loss function. This remains a topic for future research.

3.6 Numerical Evaluations

In this section, we compare the performance of our proposed M22 algorithm with other ML compression techniques including conventional floating point conversion, top$K$ sparsification, sketching, and non-uniform quantization using the L_2 norm.

To ensure the fairness of the comparison, the algorithms are implemented under the same FL setting and communication overhead constraint. The training parameters are kept consistently across numerical experiments.

3.6.1 Compression Strategies Benchmark

In our simulations, we consider the following gradient compression techniques for benchmarking.

• top$K$ *sparsification + floating point (fp) representation*: In the context of (3.3.6), top$K$ can be applied to meet the rate constraint only once a certain format for representing the gradient entries has been established. Therefore, we consider a fp representation of the entries with 8 and 4 bits. Accordingly, the relationship between the sparsification parameter K, the digit precision p (in bits), and the rate constraint R in (3.3.7) is

$$d\mathsf{R} = \log \binom{d}{K_{\text{fp}}} + K_{\text{fp}} p. \tag{3.6.1}$$

• top$K$ *sparsification + scalar uniform quantization*: For the uniform quantizer with a given quantizer rate R_{u}, the $2^{R_{\text{u}}}$ quantization centers are uniformly distributed between the minimum and maximum values of the samples in each layer and each iteration. The sparsification level K_{u} is accordingly chosen such that

$$d\mathsf{R} = \log \binom{d}{K_{\text{u}}} + K_{\text{u}} R_{\text{u}}. \tag{3.6.2}$$

• *Count Sketch* – [28]: Inspired by [28], where the count sketch method is associated with SGD in the distributed learning scenario, we integrate it into our gradient compression workflow to make another comparison to our M22 algorithm. Following a top$K$ sparsification of level K_{sk}, each client performs local compression to its local stochastic gradient by count sketch via a common sketching operator; and the server recovers the indices and the values of large entries of the aggregated stochastic gradient from the gradient sketches. By choosing sketching ratio $r_{\text{sk}} \in (0, 1]$ of the common count sketch operator, the communication overhead is calculated as the following

$$d\mathsf{R} = \log \binom{d}{K_{\text{sk}}} + r_{\text{sk}} K_{\text{sk}}. \tag{3.6.3}$$

• *TINYSCRIPT* – [17]: In [17], the authors introduced a non-uniform quantization algorithm, TINYSCRIPT to compress the activations and gradients of a DNN. In gradient compression, this scheme does not consider sparsification and applies K-means clustering to each layer before scalar quantization. Finally, the scalar quantizer hat minimizes the L_2 loss is employed. We note that the layer-wise clustering is rather computationally expensive and the execution time suffers dramatically. For this reason, in our benchmark, we removed the clustering step so that the execution time of all scheme is comparable. After removing the step to cluster the gradients and applying the same sparsification strategy as our proposed M22, the workflow of TINYSCRIPT is adapted to be comparable to our approach. The difference between our M22 and TINYSCRIPT lies in the quantizer design shown in (3.4.5), where for TINYSCRIPT, the M value is always 0. Thus its communication overhead calculation could be categorized into the M22 and its variants. We wish to point out that we have tried to adapt the schemes in [28] and [17] to the setting of Section 3.3 so as to yield a fair comparison. To comprehend the mechanism of TINTSCRIPT, we refer the readers to the original schemes [17].

• *M22 variants*: M22 has various incarnation, depending on the choice of M values in (3.4.3) and the choice of fitting distribution – GenNorm or Weibull. The workflow of our proposed M22 algorithm includes top$K$ sparsification and fitting each layer of the gradients with a pre-defined distribution. Fitting with GenNorm or Weibull distribution, the quantization centers will be distributed in a non-uniform manner. Changing from one distribution to another produces variants of the M22 algorithm, as well as adapting different choices of M value when computing gradient distortion. Because all the variants share a similar compression strategy, the calculation of $d\mathsf{R}$

for them is the same, which means, for our proposed quantizer with a given rate R_{mw}, a number of $2^{R_{\text{mw}}}$ quantization centers at each layer and iteration are found by the K-means algorithm described in Section 3.4.3. We use a top$K$ sparsification before our compressor where the sparsification level K_{mw} satisfies the following

$$d\mathsf{R} = \log \binom{d}{K_{\text{mw}}} + K_{\text{mw}} R_{\text{mw}}, \tag{3.6.4}$$

where R_{mw} is the quantizer rate and K_{mw} is the sparsification level.

3.6.2 Simulation Results

The results we provide in this section will show the superiority of our M22 in various aspects including: (i) improving the global model convergence comprehensively comparing to scalar quantization methods, uniform quantization methods and other adaptive quantization algorithms; (ii) expediting the convergence by flexibly tune the M value and (iii) exhibiting universal effectiveness for different DNN model architectures. In all simulations, we consider the case in which three rounds of local SGD are performed for each communication between the remote user and the PS. Other experiment settings could be reviewed in Section 3.3.4.

- *M22 vs. all*: In Figure 3.2, we plot the accuracy vs. iteration number under communication constraints of $d\mathsf{R}$ = 332k bits and $d\mathsf{R}$ = 996k bits for the CNN network introduced in Table 3.1. Such communication constraints are equivalent to allowing each non-zero gradient to be represented using 1 bit and 3 bits, respectively. In the following figures, each curve represents one of the compression strategies as detailed in Section 3.6.1. We consider the setting with (i) different number of remote users - two or five, and (ii) different total communication rate $d\mathsf{R}$ = 332k bits or

Methods	$d\mathsf{R} = 332$k bits	$d\mathsf{R} = 996$k bits
Uniform	$R_{\mathrm{u}} = 1$, $K_{\mathrm{u}} = 331724$	$R_{\mathrm{u}} = 3$, $K_{\mathrm{u}} = 331724$
8fp	$p = 8$, $K_{\mathrm{fp}} = 41466$	$p = 8$, $K_{\mathrm{fp}} = 124396$
4fp	$p = 4$, $K_{\mathrm{fp}} = 82931$	$p = 4$, $K_{\mathrm{fp}} = 248793$
M22+GenNorm	$M = 3$, $R_{\mathrm{mw}} = 1$	$M = 9$, $R_{\mathrm{mw}} = 3$
TINYSCRIPT	$M = 0$, $R_{\mathrm{mw}} = 1$	$M = 0$, $R_{\mathrm{mw}} = 3$
M22+dWeibull	$M = 4$, $R_{\mathrm{mw}} = 1$	$M = 7$, $R_{\mathrm{mw}} = 3$
count sketch	$r_{\mathrm{sk}} = 1$, $K_{\mathrm{sk}} = 331724$	$r_{\mathrm{sk}} = 3$, $K_{\mathrm{sk}} = 331724$

Table 3.4: Experiment parameters information of methods in Section 3.6.1 with two remote clients.

$d\mathsf{R} = 996$k. All accuracy result is the average of five different initializations. To match the rate constraint $d\mathsf{R} = 332$k bits or $d\mathsf{R} = 996$k bits, we choose the parameters following Table 3.4 in the experiments of two remote clients. Note, for M22-related and TINTSCRIPT methods, K_{mw} is chosen to be 33172 to satisfy the rate constraints. For the experiments of five remote clients, the parameters are the same except for the choice of M value, which is considered as a hyper-parameter related to the distortion function (3.4.3).

Each approach described above is represented by one curve of a particular color as shown in the figure legend. Note that we use the letter "G" or "W" to indicate either GenNorm or d-Weibull distribution is applied in gradients distribution approximation.

We point out that non-uniform quantization algorithms generally have better performance. In the top left panel of Figure 3.2, when the communication overhead constraint is set to $d\mathsf{R} = 332$k bits for CNN training, the averaging bit per non-zero parameter is 1 bit, and the actual bit rate for the quantizer follows the calculations from (3.6.1) to (3.6.4). Under such a restricted condition, where most of the traditional approaches converge reluctantly, our M22 algorithm with a fine-tuned M value not only outperforms other approaches but also maintains a similar converging speed

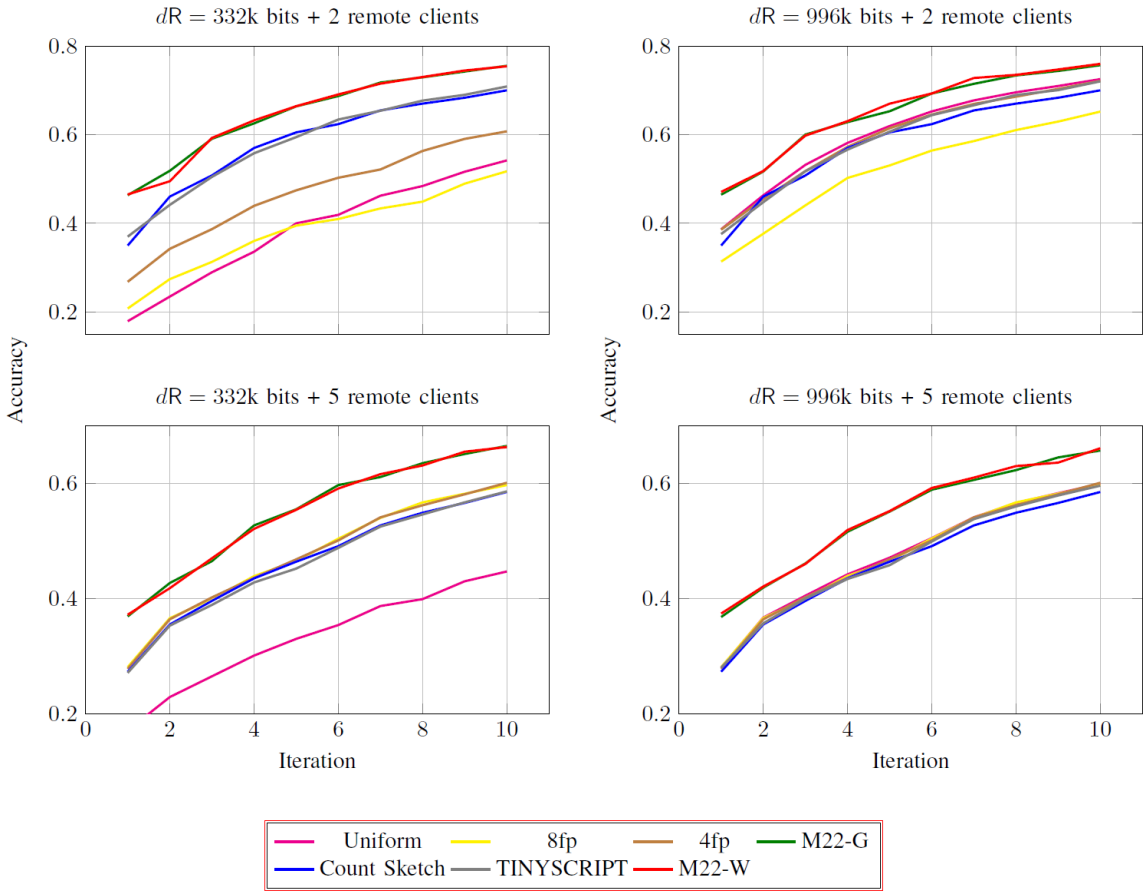

Figure 3.2: Comparison across different gradient compression approaches under four different settings: (i) row-wise: 2 or 5 remote users, atd (ii) column-wise: $d\mathsf{R} = 332$k bits or $d\mathsf{R} = 996$k bits of total communication rate.

compared with a loose-constraint regime. In the top right panel of Figure 3.2, when the communication overhead constraint is relaxed to $d\mathsf{R} = 996$k bits, the performance of traditional and scalar approaches start to be competitive with count sketch and TINYSCRIPT. However, the M22 curves stay on top by an obvious margin compared to all others. Note that the quantizer for M22 and TINYSCRIPT is chosen adaptively according to Section 3.4.3, as a function of the empirical gradient distribution at each

iteration. For the case of M22 GenNorm, this is attained by pre-calculating the quantization centers for different values of shape parameter β of GenNorm distribution (3.4.1). At each iteration, the gradient vector is normalized to obtain a zero-mean unit-variance vector which is then quantized using the pre-calculated quantizer. The bottom two panels of Figure 3.2 demonstrate the results of five clients' settings. Although introducing more remote clients results in reduced local data quantity for each client, thus affecting the global model convergence, the M22 curves with fine-tuned M values (both GenNorm and d-Weibull) still outperform the others by a significant margin. These advantages are observed under both $d\mathsf{R} = 332$k bits and $d\mathsf{R} = 996$k bits communication constraints, which further validates the superiority of the proposed algorithm.

• *M's effect*: By conducting numerous ablation studies, the M value (3.4.3) has been verified to play a vital role in model convergence speed and final accuracy. Recall that we enable the flexibility of choosing the M value when designing the gradient distortion measurement in Section 3.4.2. In Figure 3.3, we employ a list of M choices to demonstrate how the value impacts the convergence substantially when the communication overhead is fixed. If zooming in to the first couple of iterations (two panels on the right), we can observe that increasing the M value would result in more sparse quantization centers and thresholds, consistent with the fact that gradients are long-tailed, and therefore achieve better performance. The same effect could be found when comparing TINYSCRIPT with M22 using d-Weibull distribution. TINYSCRIPT approach can be regarded as a degenerate version of our M22 with $M = 0$ when calculating the distortion, while a fined-tuned positive M value is applied in our M22, which always yields a better result.

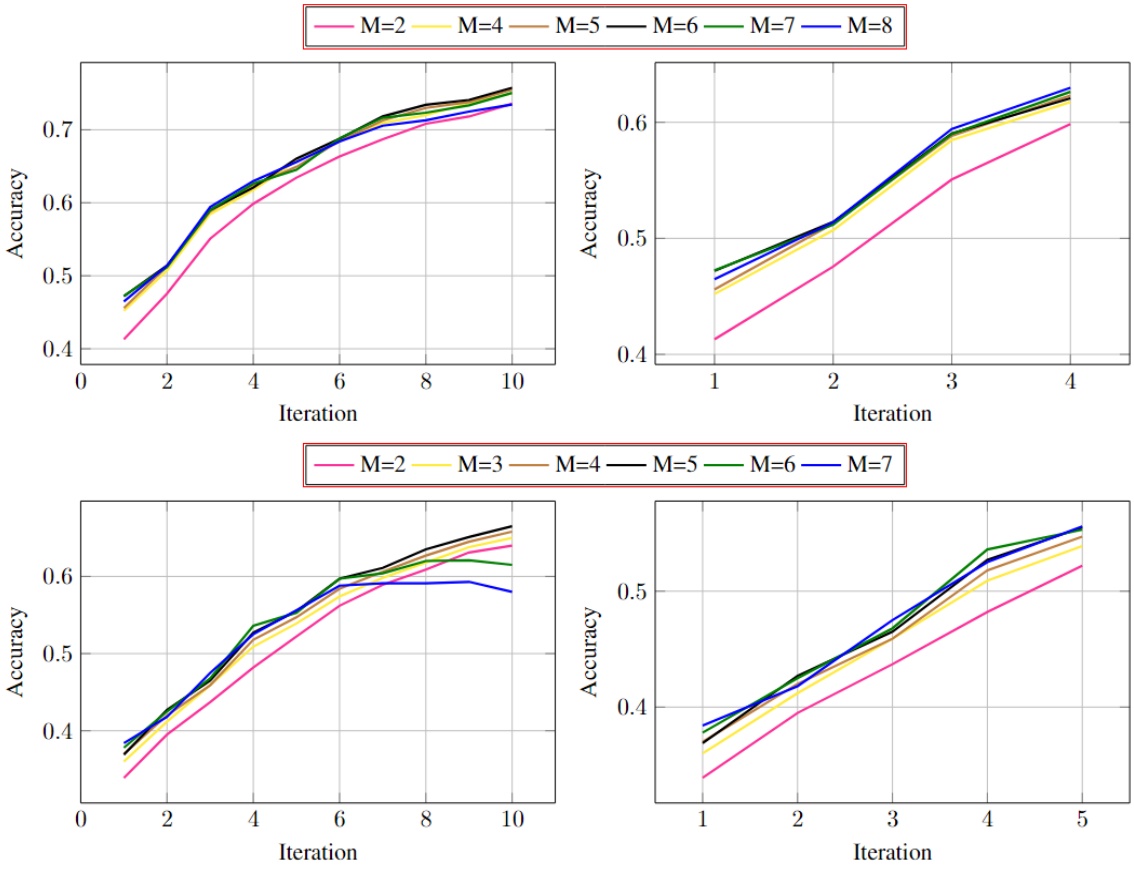

Figure 3.3: Ablation study of the effect of M value. 10-round accuracy comparison in 2 remote clients setting (top left); first 4-round accuracy comparison in 2 remote clients setting (top right); 10-round accuracy comparison in 5 remote clients setting (bottom left); first 5-round accuracy comparison in 5 remote clients setting (bottom right).

The top two panels of Figure 3.3 illustrate the results obtained from 2 remote client setting. Fitting the gradients with GenNorm distribution and limiting the communication overhead to $d\mathsf{R} = 664$k bits, the performance of $M = 6$ is the best in terms of final accuracy. However, in terms of the first four rounds, $M = 8$ achieves the highest accuracy. According to our extensive simulation results, large values of M could boost the model convergence in the initial training phase. In the five remote

client setting shown in the bottom two panels, similar results are found. An M value of five yields the best accuracy after 10 iterations (bottom left), but for the first four or five rounds, $M = 6$ or $M = 7$ outperform $M = 5$ in terms of accuracy (bottom right). Therefore, we conclude that large values of M can boost model convergence in the initial training phase, but excessively large values can hinder convergence after a certain stage. By adapting our M22 algorithm, a fine-tuned M value would not only result in fast improvement at the early training process but also ensure the final accuracy. The same effect of M value could be found in the cases where the Weibull distribution is adapted to approximate the gradient distribution and other model structures are used.

• *ResNet18 & VGG16*: Besides the CNN network, our M22 algorithm guarantees fast and reliable global model convergence when applied to other conventional network architectures, including ResNet and VGG networks. In the left panel of Figure 3.4, under the same communication constraints, we compare the 3 non-uniform compression algorithms: count sketch, TINYSCRIPT and M22 using ResNet18 model. Speaking of the converging speed and accuracy, our M22 algorithm adapting with GenNorm distribution fitting performs very close to TINYSCRIPT, which estimates the gradient updates with a Weibull distribution, while count sketch approach falls behind by an obvious margin. In the right panel of Figure 3.4, we compare the scheme where no quantization is applied with our M22 approach under 4 different communication overhead constraints, including $d\mathsf{R} = 332$k bits, $d\mathsf{R} = 664$k bits, $d\mathsf{R} = 996$k bits and $d\mathsf{R} = 1.33$m bits. On one hand, when the communication constraint is loose, M22 algorithm performs almost as good as the case, where there is no quantization applied. On the other hand, with one-eighth of the communication overhead, although

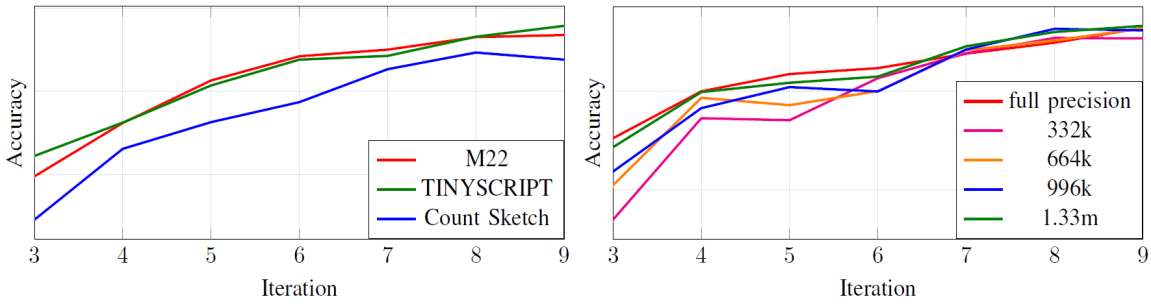

Figure 3.4: Compare the 3 non-uniform compression algorithms on ResNet18 (left); Compare the scheme where no quantization is applied with our M22 algorithm under different communication overhead constraints on VGG16 (right).

the convergence process inevitably becomes less smooth, our M22 is still competitive in terms of progressing speed, and it does not take too long until it achieves similar final accuracy comparing to more relaxed-constraint cases.

3.6.3 Further Research Direction

In this paper, we were focus on the setting in which the remote users are homogeneous in their communication and computation capabilities, as in Assumption 1. In Section 3.3.5 we hint at some relevant variations of the homogeneous setting that, although interesting, are not considered in the further paper. These and other issues – such as client dropout and scheduling, straggling, privacy, and anonymity – are concrete obstacles to the deployment of FL algorithms in real-world networks. Although the M22 is developed upon straightforward assumptions and validated in relatively ideal settings, we recognize the importance of examining its capability in more complex settings. As more flexibility is allowed in the M22 benefiting from more choices of the distortions and distributions, we believe that our algorithm would be competitive in such scenarios. We briefly argue for such a belief by considering two heterogeneous

settings originally mentioned in Section 3.3.5.

• *Heterogeneous communication constraints:* In Figure 3.5, we deploy the M22 in an experiment setting where we accommodate varying communication constraints for different clients and employ distinct compressors for different clients. The **baseline setting** is equivalent to that of 10 individual clients each using 1 bit to represent one gradient, which we name as the "uniform case" as each client equally divides up the total overhead. The **second setting** is that one client uses 2 bits to represent one gradient while the rest eight clients use 1 bit. By this design, the total overhead in this case is kept the same to the uniform case while the client who uses 2 bits will use a different compressor than the clients who use 1 bit. The **third setting** and the **fourth setting** represent one client using 3 bits while the rest seven clients use 1 bit, and one client using 4 bits while the rest six clients use 1 bit, respectively. In this fashion, the communication overhead is fixed for all four settings. The experiment results indicate that the uniform case has the best performance. As the total communication overhead is divided up more non-uniformly, the converging speed generally slows down. In conclusion, the M22 well adapts in the setting of varying communication constraints and employing distinct compressors for different clients. However, if the total communication overhead is fixed, each client uniformly dividing up the overhead and employing the same compressor will yield the best results.

• *Heterogeneous local datasets*: In Figure 3.6, we test the M22 in the heterogeneous local data setting. We consider the scenario in which 10 remote clients jointly train a classification network for the Fashion MNIST dataset [79]. This setting considers the problem of classifying images in 10 categories of clothing. The DNN used for this task is a rather simple CNN network with two convolutional layers followed by two

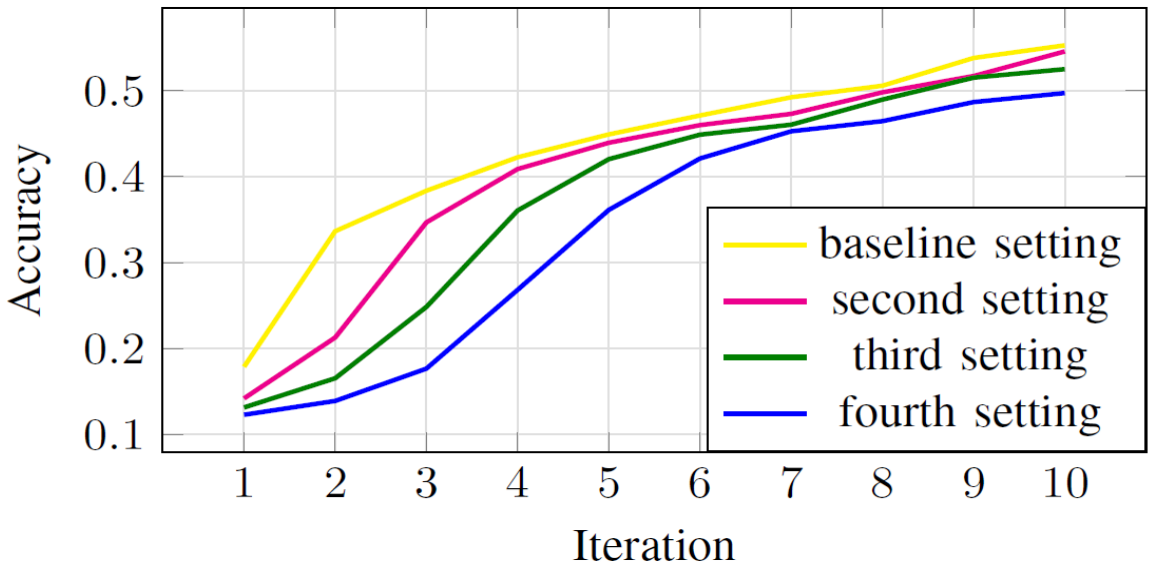

Figure 3.5: Test the M22 algorithm in the case where the total communication overhead is fixed, while individual users are under different communication constraints with distinct compressors employed.

fully connected layers. In these experiments, we consider the following heterogeneity model in the local dataset distribution: a dataset entry in the dataset of client $n \in [10]$ (i) belongs to the class n with probability ϵ, and (ii) it belongs to the class $m \neq n$, $m \in [10]$ with probability $(1-\epsilon)/9$. In other words, client n has as its favorite class the class n and this class represents a portion ϵ of the dataset. The other classes are uniformly distributed in the remaining $1-\epsilon$ portion of the dataset. In the following, we refer to ϵ has the *heterogeneity level*: (i) $\epsilon = 1$ represent the extreme scenario in which client n only has images from the class n while (ii) $\epsilon = 1/10$ represents the homogeneous case. In our simulations, we consider (i) three heterogeneity values: $\epsilon \in \{0.8, 0.5, 0.2\}$, and, for each bias value, we test our M22 in (ii) 1 bit, 2 bits, 3 bits and 4 bits compression regime. We also consider the full precision regime serving as the baseline.

In line with our expectation, the results indicate that as the local dataset becomes more heterogeneous, the converging speed slows down, while if more bits are used, the results are better. In conclusion, the ability of M22 to adapt in the heterogeneous local dataset setting is verified.

We acknowledge that there are many aspects of deploying FL in practice that we have not taken into consideration. However, the above simulation results could demonstrate the effectiveness of M22 when settings become more complicated. Integrating it in more practical FL scheme is the focus of our future work.

3.7 Conclusion

In this paper, the problem of efficient gradient compression for federated learning has been considered. For this problem, we propose M22 as an efficient gradient compression algorithm developed from a rate-distortion perspective. More specifically, M22 tackles gradient compression by designing a quantizer under (i) an assumption on the distribution of the model gradient updates and (ii) a choice of the distortion measure which minimizes the loss in accuracy. Regarding the gradients distribution, we assume that the gradient updates in each model layer and at each iteration follow a 2-degree of freedom distribution, either generalized normal distribution or double-sided Weibull distribution. This assumption is verified by numerous simulation results we produced, and therefore makes us confident to conclude that the extra degree of freedom is essential for approximating the model gradients, especially under the federated learning setting. In terms of the distortion measure, we assume that the distortion capturing the relationship between gradient perturbation and loss in accuracy is the M-magnitude weighted L_2 distortion, that is the L_2 loss between the compressed and original gradient updates multiplied by the magnitude of the original gradient to the power of M. We argue that this choice of distortion naturally bridges between two classical gradient sparsification approaches: $M = 0$ recovers uniform quantization, while $M \to \infty$ recovers top$K$ sparsification. Our simulations show that the choice of M plays an important role in the federated learning scheme, where small values are preferred when we need to compress the models aggressively, while larger values are more appropriate under loose communication overhead constraints. In this work, both of these assumptions are validated through numerical evaluations under different federated learning settings and different model structures. The performance of M22

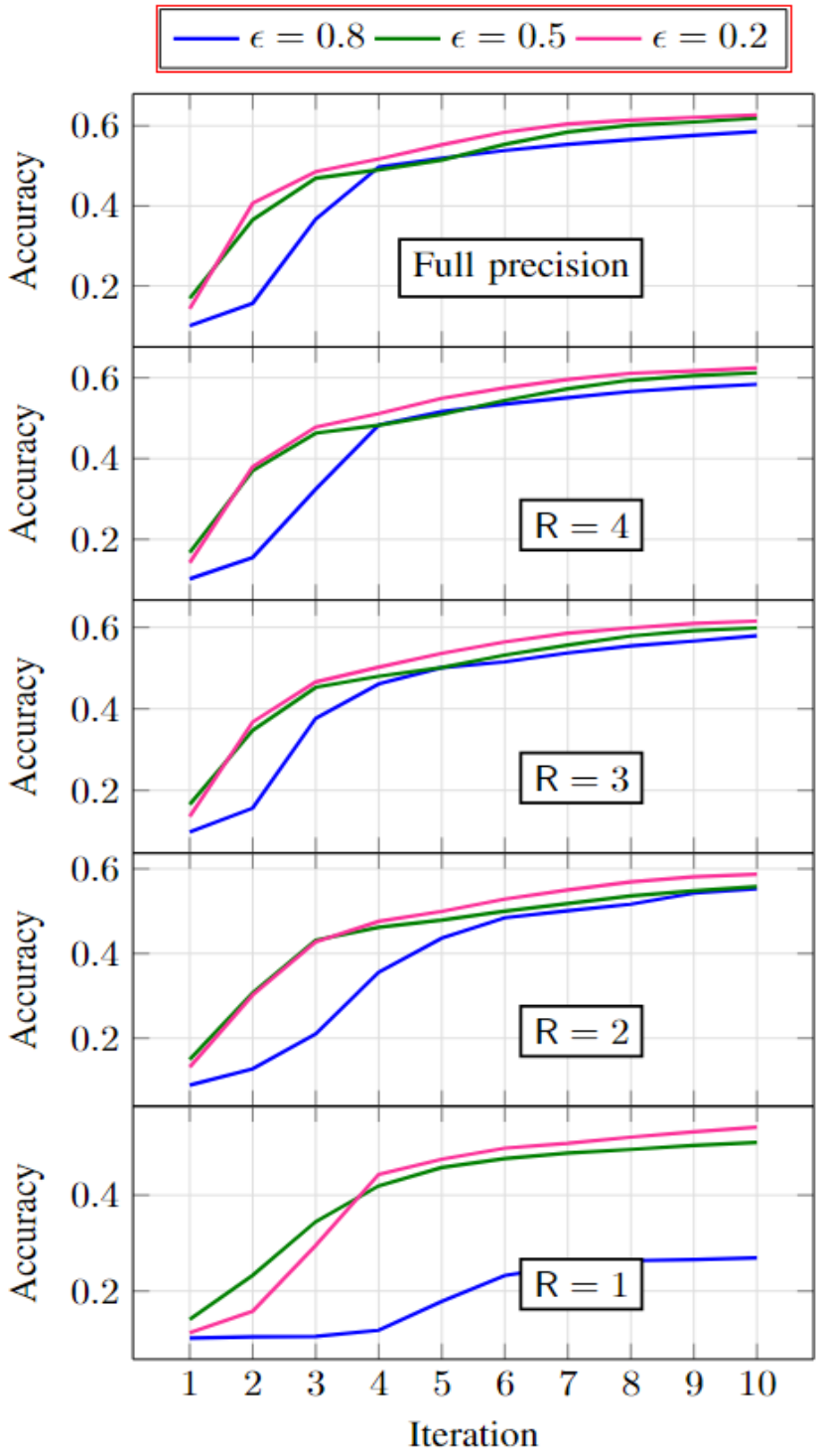

Figure 3.6: In a heterogeneous local data setting using FMNIST dataset for various compression rates. Each line corresponds to a heterogeneity level: $\epsilon = \{0.8, 0.5, 0.2\}$.

is competitive comparing with other state-of-the-art methods in most cases and surpasses them in a few. Simulations under more complicated federated learning settings definitely require more computation power. Finally, more theoretical justification for these two assumptions will be investigated in our future research.

Chapter 4

Conclusion and Future Work

4.1 Conclusion

In this thesis, we comprehensively study the communication efficiency and convergence problem occurring in Federated Learning. From problem reformulation to practical compression approach designs including client selection strategies and quantizers, we provide theoretical analysis and experimental simulations showing that our methodologies are dedicated to effectively reduce the communication cost while maintaining satisfactory convergence speed and final test accuracy.

In Chapter 2, we begin by formulating the model update problem in a single FL iteration as a multi-terminal source coding problem from an information theory perspective. We characterize the problem with sum-rate-distortion functions and derive convex optimization formulations that solve for the upper bound and lower bound to the minimum sum-rate. Under the derived sum-rate constraints, we provide the expressions for realizing optimal rate allocation. With all the technical results, we propose GCFed, an FL framework that consists of optimal rate allocation and client

selection. We conduct extensive experiments under different heterogeneity settings to demonstrate the effectiveness of the proposed approaches when applied independently or jointly, and the results validate the superior performance of our framework compared to the state-of-the-art methods.

In Chapter 3, from empirical analysis of the gradients being communicated throughout the FL training process, we propose a family of distortion measures between the original gradient and the reconstruction we refer to as "M-magnitude weighted L_2" distortion, and we assume that gradient updates follow one of the two distributions that have two degrees of freedom: "generalized normal distribution" or "double-side Weibull distribution". We design the quantizer to minimize the expected distortion in gradient reconstruction, and define the *per-bit accuracy* as the optimal improvement in accuracy that one bit of communication brings to the centralized model over the training period, in order to measure the gradient compression performance under a communication constraint. Our experimental results validate the effectiveness of our compressor designs. Additionally, we provide a thorough study regarding the effectiveness of the proposed hyperparameter M. We found that the choice of M plays an important role in the federated learning scheme, where small values are preferred when we need to compress the models aggressively, while larger values are more appropriate under loose communication overhead constraints.

4.2 Future Work

There are abundant opportunities to explore in the context of FL communication efficiency and convergence. In this thesis, we propose to study and leverage the clients' gradient correlations, but we must acknowledge that the clients gradient correlations

have not been fully exploited yet. We adopt the assumption that the gradients are i.i.d. Gaussian distributed within one local gradient sequence for each layer and that the correlations only exist at the same indices across different clients' gradient sequences. This assumption facilitates numerical derivations but could be violated as we empirically find the local gradients do not follow a strict Gaussian distribution in some cases. In addition, this assumption neglects the potential correlation existing within one local gradient sequence or across different clients at different indices. We also adopt the assumption that ignores the cross-layer correlation within each client or across different clients. In some literature, the cross-layer correlation plays an important role in model compression. Our work is a starting point for incorporating gradient correlations analysis in FL, while there are numerous opportunities to improve the problem formulation.

Our problem formulation poses demands for source correlation estimation. In our proposed method, we tackle this by asking each client to randomly sample a small subset of local gradients and send it to the server. Although this mechanism is simple and proved effective, it inevitably introduces some extra communication overhead into the FL process. A future direction is to study a more low-cost and precise way to calculate the source covariance matrix. In our implementation, we treat each layer using a scalar variable for simplicity. However, our problem formulation could be generalized to the vector Gaussian sources case, as well as the implementations. Furthermore, we only experiment and demonstrate the effectiveness of our proposed layer-wise rate allocation and client selection on fully connected (FC) layers. Interesting avenues for future research also include exploiting the gradients correlations in other types of DNN layers.

Regarding the practical compressor designs, we adopt a model-wise sparsification and layer-wise quantization combination in M22. In fact, with more comprehensive knowledge about cross-layer correlations, one may achieve improvements by performing both sparsification and quantization layer-wise. In GCFed, we simulate the dithered quantization process by passing the sources through an additive quantization noise channel. An implementation of the corresponding quantizers is among the future work.

Appendix A

Supplement of GCFed: Exploiting Clients Correlation in Federated Learning to Improve Communication Efficiency and Convergence

A.1 Detailed Introduction About the Variables in Upper and Lower Bounds

In this section, we present detailed descriptions of the variables introduced in upper bound (2.4.12) and lower bound (2.4.13).

Given the source sequences $G_1^n, G_2^n, ..., G_K^n$ and the unique encoder at each client$k$,

$f_k : \mathbb{R}^n \to \{1, 2, ..., M_k\}, k = 1, 2, ..., K$, we denote the output of the encoders as $W_{\mathcal{K}} = (W_1, W_2 ..., W_K)$ with $W_k = f_k(G_k^n), k = 1, 2, ..., K$. The assumptions stated in 2.4.5 indicate that for each source sequence G_k^n, there are n i.i.d. copies, i.e., $G_k(i), i = 1, 2, ..., n$ and $G_k(i) \perp G_k(i')$ for $i \neq i'$. We assume that the correlations exist between the same indices across different local gradient sequences, meaning $G_k(i)$ and $G_{k'}(i)$ for $k \neq k'$ and $i = 1, 2, ..., n$ are correlated.

Based on the covariance matrix of the source Σ_G, we define a positive definite covariance matrix as Σ_Z such that $\Sigma_G^{-1} + \Sigma_Z^{-1}$ is a diagonal matrix. The existence of Σ_Z is obvious as any positive definite diagonal matrix Δ with sufficiently large diagonal entries makes $(\Delta - \Sigma_G^{-1})^{-1}$ a feasible solution to Σ_Z. And we denote $\mathbb{N}(\Sigma_G)$ to be the set of feasible Σ_Z given Σ_G.

Following the definition of Σ_Z, we let $Z_1^n, Z_2^n, ..., Z_K^n$ be a Gaussian random vector with zero-mean and covariance matrix $\Sigma_Z \in \mathbb{N}(\Sigma_G)$. We assume $(Z_1^n, Z_2^n, ..., Z_K^n)$ is independent of $(G_1^n, G_2^n, ..., G_K^n)$ and define $(Y_1^n, Y_2^n, ..., Y_K^n) = (G_1^n + Z_1^n, G_2^n + Z_2^n, ..., G_K^n + Z_K^n)$. One may view as $(Y_1^n, Y_2^n, ..., Y_K^n)$ a remote source. Because of the Gaussianity of $(Z_1^n, Z_2^n, ..., Z_K^n)$ and the independence with $(G_1^n, G_2^n, ..., G_K^n)$, we know that $(Y_1^n, Y_2^n, ..., Y_K^n)$ is jointly Gaussian distributed with $(G_1^n, G_2^n, ..., G_K^n)$. And as we define $\Sigma_G^{-1} + \Sigma_Z^{-1}$, which is the distortion covariance matrix of the linear MMSE estimation for $(G_1^n, G_2^n, ..., G_K^n)$ given $(Y_1^n, Y_2^n, ..., Y_K^n)$, to be a diagonal matrix, we say that $G_1^n, G_2^n, ..., G_K^n$ are independent conditioned on $(Y_1^n, Y_2^n, ..., Y_K^n)$.

To facilitate representations, we shall define $((G_1^n)^T, (G_2^n)^T, ..., (G_K^n)^T)^T$, $((Y_1^n)^T, (Y_2^n)^T, ..., (Y_K^n)^T)^T$, $((Z_1^n)^T, (Z_2^n)^T, ..., (Z_K^n)^T)^T$ as $\mathbf{G}, \mathbf{Y}, \mathbf{Z}$, respectively.

With the above "source"-related variables being define, we define couple of more in terms of MMSE estimation.

First, we define a diagonal matrix $\Sigma = \text{diag}(\Sigma_1, \Sigma_2..., \Sigma_K)$, with $\Sigma_k, k = 1, 2..., K$ being the diagonal entries of $(\Sigma_G^{-1} + \Sigma_Z^{-1})^{-1}$. And we assume that $0 \preceq \text{diag}(\Sigma_1, ..., \Sigma_K) \preceq \Sigma_G$. In fact, Σ_k represents the distortion of the MMSE estimation for G_k^n given $\mathbf{Y}$, which is denoted as $\Sigma_k = \Sigma_{G_k^n|\mathbf{Y}}$.

Then we define D to be the the distortion covariance matrix of the MMSE estimation for $\mathbf{G}$ given $W_{\mathcal{K}}$, i.e., $D = \Sigma_{\mathbf{G}|W_{\mathcal{K}}}$.

Moreover, we define another diagonal matrix $\Gamma = \text{diag}(\Gamma_1, \Gamma_2..., \Gamma_K)$, with diagonal elements Γ_k representing the distortion of the MMSE estimation for G_k^n given $\mathbf{Y}$ and W_k, i.e., $\Gamma_k = \Sigma_{G_k^n|\mathbf{Y},W_k}$.

With the meaning behind all the variables introduced, one can follow the derivation of (10) in [72] to prove that between D and Γ_k, the following holds:

$$\text{diag}(\Gamma_1, ..., \Gamma_K) \preceq (D^{-1} + \Sigma_Z^{-1})^{-1}. \tag{A.1.1}$$

Following the Theorem 1 and Theorem 2 in [72], one can derive the lower bound $\underline{R}^*(d)$ and upper bound $\bar{R}^*(d)$ and prove the two sharing the same form except for one constraint, which is (A.1.1). The upper bound requires $\text{diag}(\Gamma_1, ..., \Gamma_K) = (D^{-1} + \Sigma_Z^{-1})^{-1}$, while the lower bound only requires the inequality $\text{diag}(\Gamma_1, ..., \Gamma_K) \preceq (D^{-1} + \Sigma_Z^{-1})^{-1}$.

A.2 Algorithm to Solve the Optimization Problem

In order to solve (2.4.13) using convex optimization solvers in MATLAB or Python, we need all constraints to be linear matrix constraints. Therefore, we need to re-write

the last constraint of it. By applying matrix inversion lemma, we get:

$$(D^{-1} + \Sigma_Z^{-1})^{-1} = D - D(\Sigma_Z + D)^{-1}D \tag{A.2.1}$$

And the last constraint becomes:

$$0 \preceq D - \text{diag}(\Gamma_1, ..., \Gamma_K) - D(\Sigma_Z + D)^{-1}D \tag{A.2.2}$$

By Schur complement, this is equivalent to:

$$0 \preceq \begin{pmatrix} \Sigma_Z + D & D \\ D & D - \text{diag}(\Gamma_1, ..., \Gamma_K) \end{pmatrix} \tag{A.2.3}$$

The convex optimization problem is converted to:

$$\begin{aligned} \max_{\Sigma_1,...,\Sigma_K} \min_{D,\Gamma_1,...,\Gamma_K} & \frac{1}{2} \log \frac{|\Sigma_G + \Sigma_Z| \, |\text{diag}(\Sigma_1, ..., \Sigma_K)|}{|D + \Sigma_Z| \, |\text{diag}(\Gamma_1, ..., \Gamma_K)|} \\ \text{s.t.} \quad & \Sigma_Z = (\text{diag}(\Sigma_1, ..., \Sigma_K)^{-1} - \Sigma_G^{-1})^{-1}, \\ & 0 \preceq \text{diag}(\Sigma_1, ..., \Sigma_K) \preceq \Sigma_G, \\ & 0 \preceq D \preceq \Sigma_G, \\ & \mathbb{1} \cdot D \cdot \mathbb{1} \leq d, \\ & 0 \preceq \text{diag}(\Gamma_1, ..., \Gamma_K), \\ & 0 \preceq \begin{pmatrix} \Sigma_Z + D & D \\ D & D - \text{diag}(\Gamma_1, ..., \Gamma_K) \end{pmatrix} \end{aligned} \tag{A.2.4}$$

Note, as stated in [73, 87], if denote $\Sigma^* = \text{diag}(\Sigma_1^{*-1}, ..., \Sigma_K^{*-1})$ as the optimal

solution to the max part of (A.2.4), it can be verified that $(\text{diag}(\Sigma_1^{*-1}, ..., \Sigma_K^{*-1}) - \Sigma_G^{-1})$ must be rank-deficient, which numerically prevents us doing the inversion of it to calculate Σ_Z. Therefore, we need to handle it by projection. Let $\alpha_1, ...\alpha_p$ denote the positive eigenvalues of $(\text{diag}(\Sigma_1^{*-1}, ..., \Sigma_K^{*-1}) - \Sigma_G^{-1})$, and let $\pi_1, ..., \pi_p$ denote the corresponding eigenvectors. We define:

$$\Lambda \triangleq \text{diag}(\frac{1}{\alpha_1}, ..., \frac{1}{\alpha_p}) \qquad \text{and} \qquad \Pi \triangleq (\pi_1, ..., \pi_p), \tag{A.2.5}$$

which satisfies that:

$$\Sigma_Z^{-1} = \Pi^T \Lambda^{-1} \Pi \qquad \text{and} \qquad \Sigma_Z = \Pi \Lambda \Pi^T. \tag{A.2.6}$$

Perform projection on the objective function follows:

$$\frac{1}{2} \log \frac{|\Sigma_G + \Sigma_Z| \, |\text{diag}(\Sigma_1, ..., \Sigma_K)|}{|D + \Sigma_Z| \, |\text{diag}(\Gamma_1, ..., \Gamma_K)|} = \frac{1}{2} \log \frac{|\Pi^T \Sigma_G \Pi + \Lambda| \, |\text{diag}(\Sigma_1, ..., \Sigma_K)|}{|\Pi^T D \Pi + \Lambda| \, |\text{diag}(\Gamma_1, ..., \Gamma_K)|}. \tag{A.2.7}$$

And projection on the last constraint is as:

$$0 \preceq \begin{pmatrix} \Pi^T D \Pi + \Lambda & \Pi^T D \\ D\Pi & D - \text{diag}(\Gamma_1, ..., \Gamma_K). \end{pmatrix} \tag{A.2.8}$$

The final convex optimization problem that is solvable for the convex optimization

solvers in MATLAB or Python is as:

$$\begin{aligned}
\max_{\Sigma_1,...,\Sigma_K} \min_{D,\Gamma_1,...,\Gamma_K} \; & \frac{1}{2}\log\frac{|\Pi^T\Sigma_G\Pi+\Lambda|\,|\mathrm{diag}(\Sigma_1,...,\Sigma_K)|}{|\Pi^T D\Pi+\Lambda|\,|\mathrm{diag}(\Gamma_1,...,\Gamma_K)|} \\
\text{s.t.} \quad & 0 \preceq \mathrm{diag}(\Sigma_1,...,\Sigma_K) \preceq \Sigma_G, \\
& 0 \preceq D \preceq \Sigma_G, \\
& \mathbb{1}\cdot D\cdot \mathbb{1} \le d, \\
& 0 \preceq \mathrm{diag}(\Gamma_1,...,\Gamma_K), \\
& 0 \preceq \begin{pmatrix} \Pi^T D\Pi+\Lambda & \Pi^T D \\ D\Pi & D-\mathrm{diag}(\Gamma_1,...,\Gamma_K) \end{pmatrix}.
\end{aligned} \tag{A.2.9}$$

Now, we introduce our algorithm in detail. As we realize that there is a saddle point in the optimization problem in (A.2.9), we develop an iterative algorithm to solve the minimization and maximization part individually, and gradually reach to the saddle point, which is also referred to as the optimal solution.

Solving the minimization part requires an initial guess of $\Sigma_1,...,\Sigma_K$. In numerical simulations, we found that their values are of vital importance for the algorithm to converge. Thus, instead of randomly initializing their values, we connect to Wyner's common information [77] to obtain the initial values for $\Sigma_1,...,\Sigma_K$. First, we introduce a new variable Θ as:

$$(\Sigma_G^{-1}+\Theta^{-1})^{-1} = D \tag{A.2.10}$$

Then the max part of the problem becomes:

$$\begin{aligned}\max_{\Sigma_1,...,\Sigma_K} \frac{1}{2}\log\frac{|\Sigma_G+\Theta|\,|\mathrm{diag}(\Sigma_1,...,\Sigma_K)|}{|\mathrm{diag}(\Sigma_1,...,\Sigma_K)+\Theta|\,|\mathrm{diag}(\Gamma_1,...,\Gamma_K)|}\\ \text{s.t.}\quad 0\preceq \mathrm{diag}(\Sigma_1,...,\Sigma_K)\preceq\Sigma_G,\\ (\mathrm{diag}(\Gamma_1^{-1},...,\Gamma_K^{-1})-\Theta^{-1})^{-1}\preceq\mathrm{diag}(\Sigma_1,...,\Sigma_K)\preceq\Sigma_G.\end{aligned} \tag{A.2.11}$$

If we consider when $d\approx \mathbb{1}\cdot\Sigma_G\cdot\mathbb{1}$, the diagonal entries of Θ will go to infinity because of the definition $\Theta=(D^{-1}+\Sigma_G^{-1})^{-1}$, causing the result of $|\mathrm{diag}(\Sigma_1,...,\Sigma_K)+\Theta|$ not depending on $(\Sigma_1,...,\Sigma_K)$. Furthermore, if we neglect the second constraint for now, we arrive at a simple formulation:

$$\begin{aligned}\max_{\Sigma_1,...,\Sigma_K} \frac{1}{2}\log|\mathrm{diag}(\Sigma_1,...,\Sigma_K)|\\ \text{s.t.}\quad 0\preceq\mathrm{diag}(\Sigma_1,...,\Sigma_K)\preceq\Sigma_G,\end{aligned} \tag{A.2.12}$$

which we find to be exactly the optimization problem to determine the Wyner's common information if $K=2$, i.e., if there are only two random variables ($K=2$). This is equivalent to:

$$\begin{aligned}\min_{P_{W|G_1,G_2}} I(G_1,G_2;W)\\ \text{s.t.}\quad G_1\longleftrightarrow W\longleftrightarrow G_2\quad\text{form a Markov Chain.}\end{aligned} \tag{A.2.13}$$

With the initialization process, we are ready to introduce the full algorithm that solves the optimization problem.

Step 1: We start with the optimization in (A.2.12) to generate intial guesses for $\Sigma_1,...,\Sigma_K$. We denote the obtained values as $\Sigma_1^{int},...,\Sigma_K^{int}$.

Step 2: We will fix the value for $\Sigma_1^{int},...,\Sigma_K^{int}$ and solve the minimization part of

(A.2.9):

$$\begin{aligned}
\min_{D,\Gamma_1,...,\Gamma_K} \; & \frac{1}{2}\log\frac{|\Pi^T\Sigma_G\Pi+\Lambda|\,|\mathrm{diag}(\Sigma_1^{int},...,\Sigma_K^{int})|}{|\Pi^T D\Pi+\Lambda|\,|\mathrm{diag}(\Gamma_1,...,\Gamma_K)|} \\
\text{s.t.} \quad & 0 \preceq D \preceq \Sigma_G, \\
& \mathbb{1}\cdot D\cdot\mathbb{1} \le d, \\
& 0 \preceq \mathrm{diag}(\Gamma_1,...,\Gamma_K), \\
& 0 \preceq \begin{pmatrix} \Pi^T D\Pi+\Lambda & \Pi^T D \\ D\Pi & D-\mathrm{diag}(\Gamma_1,...,\Gamma_K) \end{pmatrix},
\end{aligned} \tag{A.2.14}$$

of which we will obtain the values for $D^t, \Gamma_1^t, ..., \Gamma_K^t$, where t denotes the number of iterations this algorithm has been applied for. Before solving the maximization, we calculate $\Theta^t = (D^{t^{-1}} + \Sigma_G^{-1})^{-1}$.

Step 3: We fix the values for $D^t, \Gamma_1^t, ..., \Gamma_K^t$ and try to solve the maximization part of (A.2.9):

$$\begin{aligned}
\max_{\Sigma_1,...,\Sigma_K} \; & \frac{1}{2}\log\frac{|\Sigma_G+\Theta^t|\,|\mathrm{diag}(\Sigma_1,...,\Sigma_K)|}{|\mathrm{diag}(\Sigma_1,...,\Sigma_K)+\Theta^t|\,|\mathrm{diag}(\Gamma_1,...,\Gamma_K)|} \\
\text{s.t.} \quad & 0 \preceq \mathrm{diag}(\Sigma_1,...,\Sigma_K) \preceq \Sigma_G, \\
& (\mathrm{diag}(\Gamma_1^{-1},...,\Gamma_K^{-1}) - \Theta^{t^{-1}})^{-1} \preceq \mathrm{diag}(\Sigma_1,...,\Sigma_K) \preceq \Sigma_G.
\end{aligned} \tag{A.2.15}$$

As the solvers in MATLAB and Python require all the formulations to be linear, which is against by the objective function in (A.2.15), we use Gradient Descent to perform the linearization. By taking the derivative to the objective function in

(A.2.15) with respect to $\Sigma_1, ..., \Sigma_K$, the problem becomes:

$$\max_{\Sigma_1,...,\Sigma_K} \frac{1}{2}\log|\text{diag}(\Sigma_1, ..., \Sigma_K)| - \frac{1}{2}\text{trace}((\text{diag}(\Sigma_1, ..., \Sigma_K) + \Theta^t)^{-1} \cdot \text{diag}(\Sigma_1, ..., \Sigma_K))$$
$$\text{s.t.} \quad 0 \preceq \text{diag}(\Sigma_1, ..., \Sigma_K) \preceq \Sigma_G,$$
$$(\text{diag}(\Gamma_1^{-1}, ..., \Gamma_K^{-1}) - \Theta^{t^{-1}})^{-1} \preceq \text{diag}(\Sigma_1, ..., \Sigma_K) \preceq \Sigma_G. \tag{A.2.16}$$

By solving this problem, we can have a new set of $\Sigma_1^{new}, ..., \Sigma_K^{new}$. We define the gradients to be $g_1, ..., g_K = (\Sigma_1^{new} - \Sigma_1), ..., (\Sigma_K^{new} - \Sigma_K)$. Multiplying the gradients with a pre-determined step-size and subtracting from the original $\Sigma_1, ..., \Sigma_K$, we perform one step of gradient descent. Keep updating $\Sigma_1, ..., \Sigma_K$ by this method will reach the final values of $\Sigma_1, ..., \Sigma_K$ that maximize the problem (A.2.15) with current $D^t, \Gamma_1^t, ..., \Gamma_K^t, \Theta^t$ values. Therefore, by adopting "Successive Convex Optimization" and "Gradient Descent", Step 3 could be implemented as follows:

Step 3.1: Start with a feasible point, which is $\Sigma_1^{int}, ..., \Sigma_K^{int}$.

Step 3.2: Take derivative of the objective function of (A.2.15) around the current $\Sigma_1, ..., \Sigma_K$ values, resulting in the objective function of (A.2.16).

Step 3.3: Solve the linearized optimization problem, and calculate the gradients $g_1, ..., g_K$.

Step 3.4: Update $\Sigma_1, ..., \Sigma_K$ with gradients multiplying the step-size to form the new $\Sigma_1, ..., \Sigma_K$.

Step 3.5: Repeat until convergence. Since we find that there is a saddle point effect in our min-max problem. The convergence condition is either i) the objective function values of minimization and maximization being very close or ii) the gap between them held at a constant. Until one of this condition is met, we conclude that

we finish the maximization part (Step 3).

Step 4: $t = t+1$, and repeat Step 2 and Step 3 until convergence. The convergence condition for Step 4 is defined as either i) the upper bound and the lower bound of the sum-rate coincided or ii) the gap between them held at a constant.

The upper bound of the sum-rate is calculated as:

$$\bar{R}^*(d) = \frac{1}{2}\log\frac{|\Sigma_G + \Sigma_Z|\,|\mathrm{diag}(\Sigma_1, ..., \Sigma_K)|}{|D^* + \Sigma_z|\,|\mathrm{diag}(\Gamma_1, ..., \Gamma_K)|} \quad \text{, where } D^* = ((\mathrm{diag}(\Gamma_1, ..., \Gamma_K)^{-1}) - \Sigma_Z^{-1})^{-1}. \tag{A.2.17}$$

The lower bound of the sum-rate is:

$$\underline{R}^*(d) = \frac{1}{2}\log\frac{|\Sigma_G + \Sigma_Z|\,|\mathrm{diag}(\Sigma_1, ..., \Sigma_K)|}{|D + \Sigma_z|\,|\mathrm{diag}(\Gamma_1, ..., \Gamma_K)|} \quad \text{, where } D \text{ is the solution to (A.2.14).} \tag{A.2.18}$$

And the gap between them is simply:

$$\bar{R}^*(d) - \underline{R}^*(d) = \frac{1}{2}\log\frac{|\Sigma_G + \Sigma_Z|\,|\mathrm{diag}(\Sigma_1, ..., \Sigma_K)|}{|D^* + \Sigma_z|\,|\mathrm{diag}(\Gamma_1, ..., \Gamma_K)|} - \frac{1}{2}\log\frac{|\Sigma_G + \Sigma_Z|\,|\mathrm{diag}(\Sigma_1, ..., \Sigma_K)|}{|D + \Sigma_z|\,|\mathrm{diag}(\Gamma_1, ..., \Gamma_K)|}. \tag{A.2.19}$$

A.3 Generalizations of the Problem Formulation and Results

In the main context, we employ layer-wise rate-distortion analysis, which means we use our iterative algorithm to solve for corresponding (2.4.13) under a specific d distortion level for each layer.

Generalization 1: our problem formulation could be seamlessly generalized to multi-layer rate-distortion analysis. Without the loss of generality, let us consider

a two-layer regime. Assuming two covariance matrices $\Sigma_G^{(1)}$ and $\Sigma_G^{(2)}$ correspond to the correlation estimation on two layers and each layer has $N^{(1)}$ and $N^{(2)}$ number of weights, respectively. Define the total distortion constraint d_{total}. Our problem formulation is generalized to a joint of two optimization problems as:

$$\begin{aligned}
&\max_{\substack{\Sigma_1^{(1)},...,\Sigma_K^{(1)}\\ \Sigma_1^{(2)},...,\Sigma_K^{(2)}}} \min_{\substack{D^{(1)},\Gamma_1^{(1)},...,\Gamma_K^{(1)}\\ D^{(2)},\Gamma_1^{(2)},...,\Gamma_K^{(2)}}} \frac{1}{2}\log\frac{|\Sigma_G^{(1)}+\Sigma_Z^{(1)}|\,|\mathrm{diag}(\Sigma_1^{(1)},...,\Sigma_K^{(1)})|}{|D^{(1)}+\Sigma_Z^{(1)}|\,|\mathrm{diag}(\Gamma_1^{(1)},...,\Gamma_K^{(1)})|} \\
&\qquad\qquad + \frac{1}{2}\log\frac{|\Sigma_G^{(2)}+\Sigma_Z^{(2)}|\,|\mathrm{diag}(\Sigma_1^{(2)},...,\Sigma_K^{(2)})|}{|D^{(2)}+\Sigma_Z^{(2)}|\,|\mathrm{diag}(\Gamma_1^{(2)},...,\Gamma_K^{(2)})|} \\
\text{s.t.}\quad & \Sigma_Z^{(1)} = (\mathrm{diag}(\Sigma_1^{(1)},...,\Sigma_K^{(1)})^{-1} - {\Sigma_G^{(1)}}^{-1})^{-1}, \Sigma_Z^{(2)} = (\mathrm{diag}(\Sigma_1^{(2)},...,\Sigma_K^{(2)})^{-1} - {\Sigma_G^{(2)}}^{-1})^{-1}, \\
& 0 \preceq \mathrm{diag}(\Sigma_1^{(1)},...,\Sigma_K^{(1)}) \preceq \Sigma_G^{(1)}, \quad 0 \preceq \mathrm{diag}(\Sigma_1^{(2)},...,\Sigma_K^{(2)}) \preceq \Sigma_G^{(2)}, \\
& 0 \preceq D^{(1)} \preceq \Sigma_G^{(1)}, \quad 0 \preceq D^{(2)} \preceq \Sigma_G^{(2)}, \\
& \frac{N^{(1)}}{N^{(1)}+N^{(2)}}\mathbb{1}\cdot D^{(1)}\cdot\mathbb{1} + \frac{N^{(2)}}{N^{(1)}+N^{(2)}}\mathbb{1}\cdot D^{(2)}\cdot\mathbb{1}^T \le d_{total}, \\
& 0 \preceq \mathrm{diag}(\Gamma_1^{(1)},...,\Gamma_K^{(1)}) \preceq \mathrm{diag}(\Sigma_1^{(1)},...,\Sigma_K^{(1)}), \\
& 0 \preceq \mathrm{diag}(\Gamma_1^{(2)},...,\Gamma_K^{(2)}) \preceq \mathrm{diag}(\Sigma_1^{(2)},...,\Sigma_K^{(2)}), \\
& \mathrm{diag}(\Gamma_1^{(1)},...,\Gamma_K^{(1)}) \preceq ({D^{(1)}}^{-1} + {\Sigma_Z^{(1)}}^{-1})^{-1}, \\
& \mathrm{diag}(\Gamma_1^{(2)},...,\Gamma_K^{(2)}) \preceq ({D^{(2)}}^{-1} + {\Sigma_Z^{(2)}}^{-1})^{-1}
\end{aligned} \tag{A.3.1}$$

Note that our iterative algorithm is applicable to solve this joint optimization problem.

Generalization 2: another generalization is about the aggregation function. In the main context and in (A.3.1), we taking FedAvg with equal amount of local data as an example. Under this setting, we need to left and right multiply the defined distortion

matrix D by all-ones vector $\mathbb{1}$ and $\mathbb{1}^T$. We highlight that our formulations could be generalized to all linear aggregation functions. The adjustments lie in the all-ones vector.

Here, we take FedAvg with different amount of local data for instance. Since each client's contribution to the total distortion is proportional to the size of the local datasets, we define $\mathbf{d} = (\sqrt{\mathbf{d}_1/\sum \mathbf{d}_k}, \sqrt{\mathbf{d}_2/\sum \mathbf{d}_k}, ..., \sqrt{\mathbf{d}_K/\sum \mathbf{d}_k}) \in \mathbb{R}^K$, where $\mathbf{d}_k$ represents the size of client$k$'s local dataset and $\sum \mathbf{d}_k$ represents the total size of all clients local data. Replacing $\mathbb{1}$ with $\mathbf{d}$, i.e., $\mathbf{d} \cdot D \cdot \mathbf{d}^T$, yields the distortion constraint for FedAvg with different amount of local data.

Generalization 3: As stated in Sec. 2.4.5, we model each layer of the gradient at each client as a random variable, and each element/position of the gradient sequence is i.i.d. distributed. Our methodologies can indeed be generalized to random vector cases. This section briefly describes the changes in the rate allocation formulations.

When modeling each layer of the gradient with random vectors, the notation of diag$(\cdot)$ no longer represents a diagonal matrix. Instead, it represents a symmetric matrix in block-diagonal form. In addition, the newly introduced variables Σ_k and Γ_k, for $k = 1, ..., K$ each represents a PSD matrix. Thus, diag$(\Sigma_1, ..., \Sigma_K)$ and diag$(\Gamma_1, ..., \Gamma_K)$ are block-diagonal matrices with Σ_k and Γ_k aligning along the diagonal from $k = 1$ to $k = K$. Note that the same algorithm we proposed could be used to numerically solve this generalized formulation.

We highlight that the above generalizations can be easily combined.

A.4 Client Selection Strategy Details

We describe our client selection strategy in matrix formulation.

Without the loss of generality, we take "choosing two out three" as an example. In this case, we begin with a covariance matrix $\Sigma_G \in \mathbb{R}^{3\times 3}$ as:

$$\Sigma_G = \begin{bmatrix} \mathrm{var}(G_1) & \mathrm{cov}(G_1, G_2) & \mathrm{cov}(G_1, G_3) \\ \mathrm{cov}(G_2, G_1) & \mathrm{var}(G_2) & \mathrm{cov}(G_2, G_3) \\ \mathrm{cov}(G_3, G_1) & \mathrm{cov}(G_3, G_2) & \mathrm{var}(G_3) \end{bmatrix} \tag{A.4.1}$$

As described in SEC. 2.5.2, first we construct a "combination matrix", C, of which each row in C is a boolean vector, representing one combination of choosing 2 out of 3. Then stack an all-ones vector at the bottom of it as:

$$C = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{A.4.2}$$

Left and right multiplying Σ_G with C and C^T, we define a matrix $Q \in \mathbb{R}^{4\times 4}$ as:

$$Q \triangleq C\Sigma_G C^T = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \mathrm{var}(G_1) & \mathrm{cov}(G_1, G_2) & \mathrm{cov}(G_1, G_3) \\ \mathrm{cov}(G_2, G_1) & \mathrm{var}(G_2) & \mathrm{cov}(G_2, G_3) \\ \mathrm{cov}(G_3, G_1) & \mathrm{cov}(G_3, G_2) & \mathrm{var}(G_3) \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix}$$

We specify that the diagonal elements of Q represent the variance of sums as:

$$\text{diag}(Q) = (\text{var}(G_1 + G_2), \text{var}(G_1 + G_3), \text{var}(G_2 + G_3), \text{var}(G_1 + G_2 + G_3)) \quad \text{(A.4.3)}$$

Then we construct another "subtracting matrix" $S \in \mathbb{R}^{3\times 4}$ as:

$$S = \begin{bmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \end{bmatrix} \quad \text{(A.4.4)}$$

Left and right multiplying Q with S and S^T, we result in a matrix $P \in \mathbb{R}^{3\times 3}$ with diagonal elements being the expected distortion between "choosing two out of three" and having all three clients gradients as:

$$P = SQS^T \quad \text{(A.4.5)}$$

$$\begin{aligned} \text{diag}(P) = (&\text{var}(G_1 + G_2) - \text{var}(G_1 + G_2 + G_3), \\ &\text{var}(G_1 + G_3) - \text{var}(G_1 + G_2 + G_3), \\ &\text{var}(G_2 + G_3) - \text{var}(G_1 + G_2 + G_3)) \end{aligned} \quad \text{(A.4.6)}$$

Finally, the diagonal element with the smallest absolute value indicates the optimal client selection.

A.5 Rate Allocation and Quantization Approximation

In (2.4.15), we show the general expression for "binning" case. In this section, we show more details. First, we present the two extreme cases of $M_k, k = 1, ..., K$ as:

$$M_0 = \text{diag}(\Gamma_1, ...\Gamma_K), \text{ and} \tag{A.5.1}$$

$$M_K = \text{diag}(\Sigma_1, ..., \Sigma_K). \tag{A.5.2}$$

Note, above expressions are two extreme cases for (2.4.14). In later expressions, we use the definitions of $W_k, k = 1, ..., K$ and $\mathbf{Y}$ in Appendix A.1. Then with defining that $\text{diag}(\Sigma_1, ..., \Sigma_K) = (\Sigma_G^{-1} + \Sigma_Z^{-1})^{-1}$ and $\text{diag}(\Gamma_1, ..., \Gamma_K) = (D^{-1} + \Sigma_Z^{-1})^{-1}$, which

is satisfied by the upper bound $\bar{R}^*(d)$, the optimal rate allocation is as:

$$\begin{aligned}
R_K &= I(G_K; W_K) \\
&= H(\mathbf{Y}) - H(\mathbf{Y}|W_K) + I(G_K; W_K|\mathbf{Y}) \\
&= \frac{1}{2}\log\frac{|\Sigma_Z(\Sigma_Z - M_K)^{-1}\Sigma_Z|}{|\Sigma_Z(\Sigma_Z - M_{K-1})^{-1}\Sigma_Z|} + \frac{1}{2}\log\frac{|\Sigma_K|}{|\Gamma_K|} \\
\text{, and } R_k &= I(G_k; W_k|W_{k+1}, ..., W_K) \\
&= H(\mathbf{Y}|W_{k+1}, ..., W_K) - H(\mathbf{Y}|W_1, ..., W_K) + I(G_k; W_k|\mathbf{Y}) \qquad \text{(A.5.3)} \\
&= \frac{1}{2}\log\frac{|\Sigma_Z(\Sigma_Z - M_k)^{-1}\Sigma_Z|}{|\Sigma_Z(\Sigma_Z - M_{k-1})^{-1}\Sigma_Z|} + \frac{1}{2}\log\frac{|\Sigma_k|}{|\Gamma_k|} \\
\text{, and } R_1 &= I(G_1; W_1|W_2, ..., W_K) \\
&= H(\mathbf{Y}|W_2, ..., W_K) - H(\mathbf{Y}|W_1, ..., W_K) + I(G_1; W_1|\mathbf{Y}) \\
&= \frac{1}{2}\log\frac{|\Sigma_Z(\Sigma_Z - M_1)^{-1}\Sigma_Z|}{|\Sigma_Z(\Sigma_Z - M_0)^{-1}\Sigma_Z|} + \frac{1}{2}\log\frac{|\Sigma_1|}{|\Gamma_1|}
\end{aligned}$$

As mentioned in Section 2.4.7, (A.5.3) and (2.4.15) represents one of the K! orders, which represents that we determine client$_K$'s rate by itself first. Then using client$_K$'s information to determine client$_{K-1}$'s rate, and so on until client$_1$. One can easily show that there are $K!$ permutations regarding to the determining order. We note that these $K!$ orders, as corner points, define a dominant face. Any point lying in this dominant face describes an optimal rate allocation scheme.

When projection is applied because of the rank deficiency of Σ_Z, we can re-write as follows to calculate the rate allocation:

$$\frac{1}{2}\log\frac{|\Sigma_Z(\Sigma_Z - M_K)^{-1}\Sigma_Z|}{|\Sigma_Z(\Sigma_Z - M_{K-1})^{-1}\Sigma_Z|} = \frac{1}{2}\log\frac{|\Lambda(\Lambda - \Pi^T M_K \Pi)^{-1}\Lambda|}{|\Lambda(\Lambda - \Pi^T M_{K-1}\Pi)^{-1}\Lambda|} \qquad \text{(A.5.4)}$$

Following the expressions of rate allocation in "binning" case in (2.4.15), we present,

under the same sum-rate constraint, what is the individual rate for each client for the "no binning" case, meaning that we don't take correlations into consideration when designing the quantizers (each client quantizes based on itself only).

$$\begin{aligned} R_k = I(G_k; U_k) &= H(G_k) - H(G_k|W_k) \\ &= \frac{1}{2}\log(|\Sigma_{Gk}||\Sigma_{G_k}^{-1} + \Gamma_k^{-1} - \Sigma_k^{-1}|) \qquad \text{, for } k = 1, ..., K, \end{aligned} \tag{A.5.5}$$

where Σ_{Gk} is the k-th diagonal element of Σ_G. Note that this "no binning" rate allocation still satisfies that $\sum_{k=1}^{K} I(G_k; W_k) = \bar{R}^*(d)$.

With the rate allocation for both binning and no-binning cases introduced, we are ready to introduce how do we simulate the quantization under those rates according to [84]. As proved in the reference, the simulation process for dithered quantization includes 1) adding a uniformly distributed noise to the source; 2) reconstructing the sources by linear MMSE estimation.

Step 1): for both binning and no-binning cases, once the rate R_k is determined, because we model each layer of gradients as a single variable, the variance of the noise need to be added onto the source can be calculated as (degenerated to a scalar case):

$$R_k = \frac{1}{2}\log\frac{\sigma_{g_k}^2 + \sigma_{Nk}^2}{\sigma_{Nk}^2}, \quad k = 1, ..., K. \tag{A.5.6}$$

After obtaining the noise variance σ_{Nk}^2, $k = 1, ..., K$, we uniformly sample a tensor of noise following $N(0, \sigma_N^2)$ and add onto the corresponding layer of gradients, resulting in $\hat{G}_1, ..., \hat{G}_K$, which denotes the noisy version of the sources with additive Gaussian noise.

Step 2): for reconstructing the source with linear MMSE estimation, we need to

first define:

$$\Psi \triangleq (D^{-1} - \Sigma_g^{-1})^{-1} \tag{A.5.7}$$

And then define:

$$A \triangleq \Sigma_G \cdot (\Sigma_g + \Psi)^{-1} \tag{A.5.8}$$

To get the reconstructions $\tilde{G}_1, ..., \tilde{G}_K$, we do the follows:.

$$[\tilde{g}_1, ..., \tilde{g}_K]^T = A \cdot [\hat{G}_1, ..., \hat{G}_K]^T \tag{A.5.9}$$

Note, this process [84]: from the original source $(G_1, ..., G_K)$ to noisy version $(\hat{G}_1, ..., \hat{G}_K)$ and finally to the denoised version $\tilde{G}_1, ..., \tilde{G}_K$ well-approximates the encode-decode process using a dithered lattice quantizer following the rate allocation R_k, for $k = 1..., K$.

For the "centralizing case" mentioned in Section 2.6.5, we hypothesis that there is one external client with access to all local clients gradients, meaning that external client has direct access to the true FedAvg result. Considering a communication rate constraint between this external client and the server, we still must use the above-mentioned steps to simulate the compression and decompression process. The only difference is that all the sum-rate is assigned to the external client without the need for rate allocation under this centralizing setting.

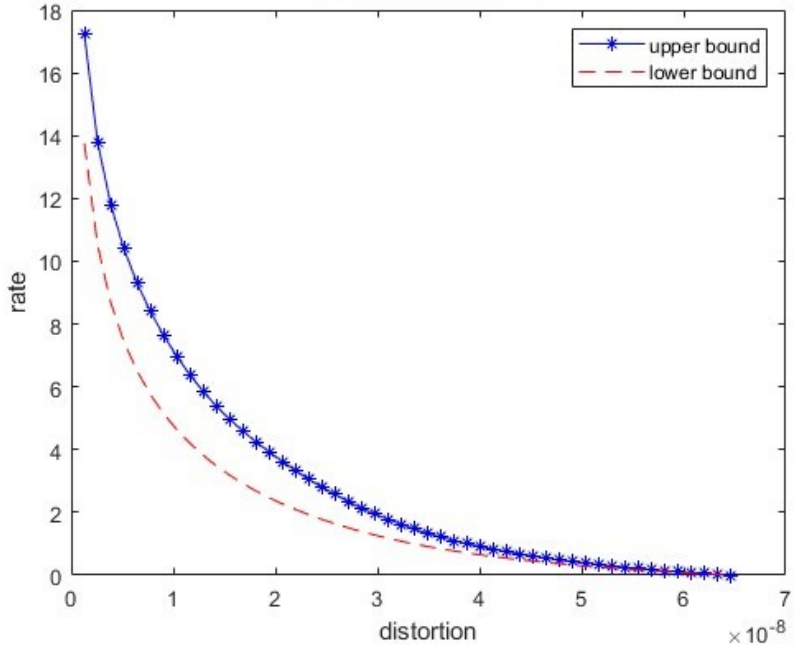

Figure A.1: Example Rate Distortion curve. Upper bound is described in (2.4.12) and lower bound is described in (2.4.13) .

A.6 An Example of the Rate-Distortion Curve from Our Correlation Estimation

By applying the iterative algorithm to solve (2.4.13), we get two R-D curves respectively representing the upper bound and lower bound to sum-rate versus different distortion level, d. An example is presented in Figure A.1.

A.7 Experiment Setting Details

When conducting experiments on CIFAR-10 with VGG16 network, we initialize the network using a pre-trained weights obtained from ImageNet-1K [16] except for the

last output layer for the purpose of efficiently analyzing the effectiveness of our methods. During the training process, we freeze the feature part of the network except for the last three CONV layers and the subsequent ReLU and pooling layers. Also, we unfreeze the entire classifier part, which consists of 3 Fully-Connected(FC) layers, as well as and Dropout layers. When experimenting on Fashion-MNIST, we adopt a simple CNN network with 2 CONV layers and 2 FC layers. All the correlation estimation and related applications are deployed on the 3 FC layers because we focus on the spatial correlation between the same positions in FC layers. For each layer, the server randomly samples 100 indices regardless of the size of the layer. Note that the application of our approach is not limited to our chosen dataset or network architecture.

In terms of FL settings, each client performs 3 rounds of local updates before they communicate with the server, and the total number of communication iterations is set to 20. We use SGD as the optimizer with learning rate 0.03 for i) Partition by shards and 0.01 for ii) Partition by bias level, respectively, with momentum equal to 0.5 and a learning rate decay equal to 0.995. We apply FedAvg to aggregate the clients' updates. We choose the distortion to be $d = 0.1 \cdot (\mathbb{1} \cdot \Sigma_G \cdot \mathbb{1})$ (0.1 times the maximum distortion), which uniquely determines the sum rate.

The Table A.1 demonstrates the detailed parameter information of the DNN we use to experiment with Fashion-MNIST dataset.

Layer	Output Shape	Param No.
conv2d_1	(10, 24, 24)	260
conv2d_2	(20, 8, 8)	5020
dropout_2d	(20, 20, 8)	0
FC_1	(50)	16050
FC_2	(10)	510

Table A.1: CNN model information for Fashion-MNIST dataset.

Appendix B

Supplement of M22: A Communication-Efficient Algorithm for Federated Learning Inspired by Rate-Distortion

The Table B.1, Table B.2, Table B.3 demonstrate the detailed parameter information of the three models we have tested.

Layer	Output Shape	Param No.
conv2d	(32, 32, 32)	896
BN	(32, 32, 32)	128
conv2d_1	(32, 32, 32)	9248
BN_1	(32, 32, 32)	128
max_pooling2d	(16, 16, 32)	0
dropout	(16, 16, 32)	0
conv2d_2	(16, 16, 64)	18496
BN_2	(16, 16, 64)	256
conv2d_3	(16, 16, 64)	36928
BN_3	(16, 16, 64)	256
max_pool2d_1	(8, 8, 64)	0
dropout_1	(8, 8, 64)	0
conv2d_4	(8, 8, 128)	73856
BN_4	(8, 8, 128)	512
conv2d_5	(8, 8, 128)	147584
BN_5	(8, 8, 128)	512
max_pool2d_2	(4, 4, 128)	0
dropout_2	(4, 4, 128)	0
flatten	(2048)	0
dense	(128)	262272
BN_6	(128)	512
dropout_3	(128)	0
dense_1	(10)	1290

Table B.1: CNN model information.

Layer	Output Shape	Param No.
conv2d_1	(16, 16, 64)	9408
BN_1	(16, 16, 64)	128
relu_1	(16, 16, 64)	0
max_pooling2d_1	(8, 8, 64)	0
conv2d_2	(8, 8, 64)	36864
BN_2	(8, 8, 64)	128
relu_2	(8, 8, 64)	0
con2d_3	(8, 8, 64)	36864
BN_3	(8, 8, 64)	128
relu_3	(8, 8, 64)	0
basic_block_2	(8, 8, 64)	73984
basic_block_3	(4, 4, 128)	230144
basic_block_4	(4, 4, 128)	295424
basic_block_5	(2, 2, 256)	919040
basic_block_6	(2, 2, 256)	1180672
basic_block_7	(1, 1, 512)	3673088
basic_block_8	(1, 1, 512)	4720640
adaptive_avg_pool2d	(1, 1, 512)	0
dense_1	(10)	5130

Table B.2: ResNet18 model information.

Layer	Output Shape	Param No.
conv2d_1	(32, 32, 64)	1792
conv2d_2	(32, 32, 64)	36928
max_pooling2d_1	(16, 16, 64)	0
conv2d_3	(16, 16, 128)	73856
conv2d_4	(16, 16, 128)	147584
max_pool2d_2	(8, 8, 128)	0
conv2d_5	(8, 8, 256)	295168
conv2d_6	(8, 8, 256)	590080
conv2d_7	(8, 8, 256)	590080
max_pool2d_3	(4, 4, 256)	0
conv2d_8	(4, 4, 512)	1180160
conv2d_9	(4, 4, 512)	2359808
conv2d_10	(4, 4, 512)	2359808
max_pool2d_4	(2, 2, 512)	0
conv2d_11	(2, 2, 512)	2359808
conv2d_12	(2, 2, 512)	2359808
conv2d_13	(2, 2, 512)	2359808
max_pool2d_5	(1, 1, 512)	0
flatten	(512)	0
dense_1	(4096)	2101248
dense_2	(4096)	16781312
dense_3	(10)	40970

Table B.3: VGG16 model information.

Bibliography

[1] A. Abdi and F. Fekri. Reducing communication overhead via ceo in distributed training. In *2019 IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pages 1–5, 2019. doi: 10.1109/SPAWC.2019.8815453.

[2] N. Agarwal, A. T. Suresh, F. X. Yu, S. Kumar, and B. McMahan. cpSGD: Communication-efficient and differentially-private distributed SGD. In *32nd Advances in Neural Information Processing Systems (NIPS)*, pages 7564–7575, Montréal, Canada, Dec. 2018.

[3] A. F. Aji and K. Heafield. Sparse communication for distributed gradient descent. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark, Sept. 2017. Association for Computational Linguistics.

[4] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic. QSGD: Communication-efficient sgd via gradient quantization and encoding. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[5] D. Alistarh, T. Hoefler, M. Johansson, N. Konstantinov, S. Khirirat, and C. Renggli. The convergence of sparsified gradient methods. In *Advances in Neural Information Processing Systems*, pages 5973–5983, 2018.

[6] D. Araújo, R. I. Oliveira, and D. Yukimura. A mean-field limit for certain deep neural networks. *arXiv preprint arXiv:1906.00193*, 2019.

[7] L. P. Barnes, A. Dytso, and H. V. Poor. Improved information theoretic generalization bounds for distributed and federated learning. In *2022 IEEE International Symposium on Information Theory (ISIT)*, pages 1465–1470. IEEE, 2022.

[8] T. Berger. Multiterminal source coding. In *The Information Theory Approach to Communications,*, volume 229, pages 171–231. G. Longo, Ed. New York: Springer-Verlag, 1978.

[9] T. Berger, Z. Zhang, and H. Viswanathan. The ceo problem [multiterminal source coding]. *IEEE Transactions on Information Theory*, 42(3):887–902, 1996.

[10] J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar. signSGD: Compressed optimization for non-convex problems. In *Advances in Neural Information Processing Systems*, pages 560–569, 2018.

[11] D. Bertsekas and J. Tsitsiklis. *Parallel and distributed computation: numerical methods*. Athena Scientific, 2015.

[12] S. Caldas, J. Konečny, H. B. McMahan, and A. Talwalkar. Expanding the reach of federated learning by reducing client resource requirements, 2019.

[13] Z.-J. Chen, E. E. Hernandez, Y.-C. Huang, and S. Rini. DNN gradient lossless compression: Can GenNorm be the answer? In *ICC 2022 - IEEE International Conference on Communications*, pages 407–412, 2022. doi: 10.1109/ICC45855.2022.9838754.

[14] R. Chor, M. Sefidgaran, and A. Zaidi. More communication does not result in smaller generalization error in federated learning. In *2023 IEEE International Symposium on Information Theory (ISIT)*, pages 48–53. IEEE, 2023.

[15] T. M. Cover and J. A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, USA, 2006. ISBN 0471241954.

[16] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

[17] F. Fu, Y. Hu, Y. He, J. Jiang, Y. Shao, C. Zhang, and B. Cui. Don't waste your bits! Squeeze activations and gradients for deep neural networks via TinyScript. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3304–3314. PMLR, 13–18 Jul 2020.

[18] A. E. Gamal and Y.-H. Kim. Lecture notes on network information theory, 2011. URL `https://arxiv.org/abs/1001.3404`.

[19] V. Gandikota, D. Kane, R. K. Maity, and A. Mazumdar. vqsgd: Vector quantized

stochastic gradient descent. In *International Conference on Artificial Intelligence and Statistics*, pages 2197–2205. PMLR, 2021.

[20] S. Gel'fand and M. Pinsker. Coding for channel with random parameters. *Problems of control and information theory*, 9(1):19–31, 1980.

[21] P. Gholami and H. Seferoglu. Communication efficient federated learning with differentiated aggregation. In *2nd Workshop on Advancing Neural Network Training: Computational Efficiency, Scalability, and Resource Optimization (WANT@ ICML 2024)*, 2024.

[22] H. Gish and J. Pierce. Asymptotically efficient quantizing. *IEEE Transactions on Information Theory*, 14(5):676–683, 1968.

[23] J. Goetz, K. Malik, D. Bui, S. Moon, H. Liu, and A. Kumar. Active federated learning, 2019. URL `https://arxiv.org/abs/1909.12641`.

[24] F. Haddadpour, M. M. Kamani, M. Mahdavi, and V. Cadambe. Local sgd with periodic averaging: Tighter analysis and adaptive synchronization. *Advances in Neural Information Processing Systems*, 32, 2019.

[25] J. Hamer, M. Mohri, and A. T. Suresh. FedBoost: A communication-efficient algorithm for federated learning. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3973–3983. PMLR, 13–18 Jul 2020.

[26] R. Hu, Y. Gong, and Y. Guo. CPFed: Communication-efficient and privacy-preserving federated learning. *arXiv preprint arXiv:2003.13761*, 2020.

[27] B. Isik, A. No, and T. Weissman. Successive pruning for model compression via rate distortion theory. *arXiv preprint arXiv:2102.08329*, 2021.

[28] N. Ivkin, D. Rothchild, E. Ullah, V. Braverman, I. Stoica, and R. Arora. Communication-efficient distributed SGD with sketching. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[29] Y. Jee Cho, J. Wang, and G. Joshi. Towards understanding biased client selection in federated learning. In G. Camps-Valls, F. J. R. Ruiz, and I. Valera, editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 10351–10375. PMLR, 28–30 Mar 2022. URL `https://proceedings.mlr.press/v151/jee-cho22a.html`.

[30] M. E. E. Khan, A. Immer, E. Abedi, and M. Korzepa. Approximate inference turns deep networks into Gaussian processes. *Advances in neural information processing systems*, 32, 2019.

[31] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[32] A. Krizhevsky, V. Nair, and G. Hinton. Cifar-10 (canadian institute for advanced research). 2009. URL `http://www.cs.toronto.edu/~kriz/cifar.html`.

[33] J. Lee, Y. Bahri, R. Novak, S. S. Schoenholz, J. Pennington, and J. Sohl-Dickstein. Deep neural networks as Gaussian processes. *arXiv preprint arXiv:1711.00165*, 2017.

[34] J. Lee, S. Park, S. Mo, S. Ahn, and J. Shin. Layer-adaptive sparsity for the magnitude-based pruning. In *International Conference on Learning Representations*, 2020.

[35] T. Li, Z. Liu, V. Sekar, and V. Smith. Privacy for free: Communication efficient learning with differential privacy using sketches. *Available: https://arxiv.org/abs/1911.00972*, 2019.

[36] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020. doi: 10.1109/MSP.2020.2975749.

[37] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang. On the convergence of fedavg on non-iid data, 2020. URL `https://arxiv.org/abs/1907.02189`.

[38] Y. Lin, S. Han, H. Mao, Y. Wang, and W. J. Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training, 2020.

[39] Y. Linde, A. Buzo, and R. Gray. An algorithm for vector quantizer design. *IEEE Transactions on communications*, 28(1):84–95, 1980.

[40] Y. Liu, S. Salehkalaibar, S. Rini, and J. Chen. M22: Rate-distortion inspired gradient compression. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2023. doi: 10.1109/ICASSP49357.2023.10097231.

[41] A. G. d. G. Matthews, M. Rowland, J. Hron, R. E. Turner, and Z. Ghahramani. Gaussian process behaviour in wide deep neural networks. *arXiv preprint arXiv:1804.11271*, 2018.

[42] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In A. Singh and J. Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR, 20–22 Apr 2017.

[43] H. B. McMahan, D. R. K. Talwar, and L. Zhang. Learning differentially private recurrent language models. In *Int. Conf. Learning Representations (ICRL)*, 2018.

[44] S. Mei, A. Montanari, and P.-M. Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33):E7665–E7671, 2018.

[45] M. Nelson and J.-L. Gailly. The data compression book 2nd edition. *M & T Books, New York, NY*, 1995.

[46] P.-M. Nguyen and H. T. Pham. A rigorous framework for the mean field limit of multilayer neural networks. *arXiv preprint arXiv:2001.11443*, 2020.

[47] Y. Oohama. Gaussian multiterminal source coding. *IEEE Transactions on Information Theory*, 43(6):1912–1923, 1997. doi: 10.1109/18.641555.

[48] Y. Oohama. The rate-distortion function for the quadratic gaussian ceo problem. *IEEE Transactions on Information Theory*, 44(3):1057–1070, 1998.

[49] P. Panter and W. Dite. Quantization distortion in pulse-count modulation with nonuniform spacing of levels. *Proceedings of the IRE*, 39(1):44–48, 1951.

[50] M. Ribero, H. Vikalo, and G. de Veciana. Federated learning under intermittent client availability and time-varying communication constraints. *IEEE Journal of Selected Topics in Signal Processing*, 17(1):98–111, 2023. doi: 10.1109/JSTSP.2022.3224590.

[51] A. Rodio, F. Faticanti, O. Marfoq, G. Neglia, and E. Leonardi. Federated learning under heterogeneous and correlated client availability, 2023. URL `https://arxiv.org/abs/2301.04632`.

[52] D. Rothchild, A. Panda, E. Ullah, N. Ivkin, I. Stoica, V. Braverman, J. Gonzalez, and R. Arora. FetchSGD: Communication-efficient federated learning with sketching. In *International Conference on Machine Learning*, pages 8253–8265. PMLR, 2020.

[53] R. Saha, S. Rini, M. Rao, and A. J. Goldsmith. Decentralized optimization over noisy, rate-constrained networks: Achieving consensus by communicating differences. *IEEE Journal on Selected Areas in Communications*, 40(2):449–467, 2021.

[54] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek. Robust and communication-efficient federated learning from non-i.i.d. data. *IEEE Transactions on Neural Networks and Learning Systems*, 31(9):3400–3413, 2020. doi: 10.1109/TNNLS.2019.2944481.

[55] M. Sefidgaran, R. Chor, and A. Zaidi. Rate-distortion theoretic bounds on generalization error for distributed learning. *Advances in Neural Information Processing Systems*, 35:19687–19702, 2022.

[56] M. Sefidgaran, R. Chor, A. Zaidi, and Y. Wan. Lessons from generalization error analysis of federated learning: You may communicate less often! In *Forty-first International Conference on Machine Learning*, 2024.

[57] F. Seide, H. Fu, J. Droppo, G. Li, and D. Yu. 1-bit stochastic gradient descent and its application to data-parallel distributed training of speech DNNs. In *INTERSPEECH*, pages 9850–9861, 2014.

[58] S. M. Shah and V. K. N. Lau. Model compression for communication efficient federated learning. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–15, 2021. doi: 10.1109/TNNLS.2021.3131614.

[59] N. S. Shai Shalev-Shwartz and T. Zhang. Trading accuracy for sparsity in optimization problems with sparsity constraints. *SIAM J. Optimization*, 2010.

[60] N. Shlezinger, S. Rini, and Y. C. Eldar. The communication-aware clustered federated learning problem. In *2020 IEEE International Symposium on Information Theory (ISIT)*, pages 2610–2615. IEEE, 2020.

[61] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition, 2015. URL `https://arxiv.org/abs/1409.1556`.

[62] D. Slepian and J. Wolf. Noiseless coding of correlated information sources. *IEEE Transactions on Information Theory*, 19(4):471–480, 1973. doi: 10.1109/TIT.1973.1055037.

[63] S. U. Stich. Local sgd converges fast and communicates little, 2019. URL `https://arxiv.org/abs/1805.09767`.

[64] S. U. Stich, J.-B. Cordonnier, and M. Jaggi. Sparsified SGD with memory. In *Advances in Neural Information Processing Systems 31*, pages 4448–4459. 2018.

[65] X. Sun, J. Choi, C.-Y. Chen, N. Wang, S. Venkataramani, V. V. Srinivasan, X. Cui, W. Zhang, and K. Gopalakrishnan. Hybrid 8-bit floating point (HFP8) training and inference for deep neural networks. *Advances in Neural Information Processing Systems*, 32:4900–4909, 2019.

[66] L. Tan, X. Zhang, Y. Zhou, X. Che, M. Hu, X. Chen, and D. Wu. Adafed: Optimizing participation-aware federated learning with adaptive aggregation weights. *IEEE Transactions on Network Science and Engineering*, 9(4):2708–2720, 2022. doi: 10.1109/TNSE.2022.3168969.

[67] H. Tang, S. Gan, C. Zhang, T. Zhang, and J. Liu. Communication compression for decentralized training. In *Advances in Neural Information Processing Systems 31*, pages 7662–7672. Curran Associates, Inc., 2018.

[68] M. Tang, X. Ning, Y. Wang, J. Sun, Y. Wang, H. Li, and Y. Chen. Fedcor: Correlation-based active client selection strategy for heterogeneous federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10102–10111, 2022.

[69] Y. Tsuzuku, H. Imachi, and T. Akiba. Variance-based gradient compression for efficient distributed deep learning, 2018.

[70] S.-Y. Tung. *Multiterminal Source Coding*. Doctoral dissertation,, Cornell University, School of Electrical Engineering, Ithaca, NY, 1978.

[71] H. Wang, S. Sievert, S. Liu, Z. Charles, D. Papailiopoulos, and S. Wright. Atomo: Communication-efficient learning via atomic sparsification. In *Advances in Neural Information Processing Systems*, pages 9850–9861, 2018.

[72] J. Wang and J. Chen. On the sum rate of vector gaussian multiterminal source coding. In *2010 IEEE International Symposium on Information Theory*, pages 46–50, 2010. doi: 10.1109/ISIT.2010.5513295.

[73] J. Wang and J. Chen. Vector gaussian multiterminal source coding. *IEEE Transactions on Information Theory*, 60(9):5533–5552, 2014. doi: 10.1109/TIT.2014.2333473.

[74] J. Wang and G. Joshi. Cooperative sgd: A unified framework for the design and analysis of communication-efficient sgd algorithms, 2019. URL `https://arxiv.org/abs/1808.07576`.

[75] J. Wang, J. Chen, and X. Wu. On the sum rate of gaussian multiterminal source coding: New proofs and results. *IEEE Transactions on Information Theory*, 56 (8):3946–3960, 2010. doi: 10.1109/TIT.2010.2050960.

[76] W. Wen, C. Xu, F. Yan, C. Wu, Y. Wang, Y. Chen, and H. Li. Terngrad: Ternary gradients to reduce communication in distributed deep learning, 2017. URL `https://arxiv.org/abs/1705.07878`.

[77] A. Wyner. The common information of two dependent random variables. *IEEE*

Transactions on Information Theory, 21(2):163–179, 1975. doi: 10.1109/TIT.1975.1055346.

[78] A. Wyner and J. Ziv. The rate-distortion function for source coding with side information at the decoder. *IEEE Transactions on Information Theory*, 22(1): 1–10, 1976. doi: 10.1109/TIT.1976.1055508.

[79] H. Xiao, K. Rasul, and R. Vollgraf. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms, 2017.

[80] Y. Xu, Z. Jiang, H. Xu, Z. Wang, C. Qian, and C. Qiao. Federated learning with client selection and gradient compression in heterogeneous edge systems. *IEEE Transactions on Mobile Computing*, 23(5):5446–5461, 2024. doi: 10.1109/TMC.2023.3309497.

[81] S. Yagli, A. Dytso, and H. V. Poor. Information-theoretic bounds on the generalization error and privacy leakage in federated learning. In *2020 IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pages 1–5. IEEE, 2020.

[82] H. Yang, T. Ding, and X. Yuan. Federated learning with lossy distributed source coding: Analysis and optimization. *IEEE Transactions on Communications*, 71 (8):4561–4576, 2023. doi: 10.1109/TCOMM.2023.3277882.

[83] Y. Yang, V. Stankovic, Z. Xiong, and W. Zhao. On multiterminal source code design. *IEEE Transactions on Information Theory*, 54(5):2278–2302, 2008.

[84] R. Zamir and M. Feder. On lattice quantization noise. *IEEE Transactions on Information Theory*, 42(4):1152–1159, 1996. doi: 10.1109/18.508838.

[85] N. Zhang, M. Tao, and J. Wang. Sum-rate-distortion function for indirect multiterminal source coding in federated learning, 2021. URL `https://arxiv.org/abs/2101.08696`.

[86] N. Zhang, M. Tao, J. Wang, and F. Xu. Fundamental limits of communication efficiency for model aggregation in distributed learning: A rate-distortion approach. *IEEE Transactions on Communications*, 71(1):173–186, 2023. doi: 10.1109/TCOMM.2022.3224977.

[87] X. Zhao. *Determining the Distributed Karhunen-Loève Transform via Convex Semidefinite Relaxation*. M.a.sc. thesis, McMaster University, Hamilton, ON, August 2018. Available at `http://hdl.handle.net/11375/24119`.

[88] F. Zhou and G. Cong. On the convergence properties of a k-step averaging stochastic gradient descent algorithm for nonconvex optimization. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 3219–3227. International Joint Conferences on Artificial Intelligence Organization, 7 2018. doi: 10.24963/ijcai.2018/447. URL `https://doi.org/10.24963/ijcai.2018/447`.