**Distribution Agreement**

In presenting this thesis or dissertation as a partial fulfillment of the requirements for an advanced degree from Emory University, I hereby grant to Emory University and its agents the non-exclusive license to archive, make accessible, and display my thesis or dissertation in whole or in part in all forms of media, now or hereafter known, including display on the world wide web. I understand that I may select some access restrictions as part of the online submission of this thesis or dissertation. I retain all ownership rights to the copyright of the thesis or dissertation. I also retain the right to use in future works (such as articles or books) all or part of this thesis or dissertation.

Signature:

_____ _____

Jing Ma Date

Federated Tensor Factorization for Collaborative Health Data Analytics

By

Jing Ma
Doctor of Philosophy

Computer Science and Informatics

---

Joyce C. Ho, Ph.D.
Advisor

---

Li Xiong, Ph.D.
Co-Advisor

---

Vaidy S. Sunderam, Ph.D.
Committee Member

---

Xiaoqian Jiang, Ph.D.
Committee Member

Accepted:

---

Lisa A. Tedesco, Ph.D.
Dean of the James T. Laney School of Graduate Studies

---

Date

# Federated Tensor Factorization for Collaborative Health Data Analytics

By

Jing Ma
B.A., Shanghai University of Finance and Economics, 2012

Advisor: Joyce C. Ho, Ph.D.

An abstract of
A dissertation submitted to the Faculty of the
James T. Laney School of Graduate Studies of Emory University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in Computer Science and Informatics
2021

Abstract

Federated Tensor Factorization for Collaborative Health Data Analytics
By Jing Ma

Modern healthcare systems are collecting a huge volume of healthcare data from a large number of individuals with various medical procedures, medications, diagnosis, lab tests and so on. Tensor factorization has been demonstrated as an efficient approach for computational phenotyping, where massive electronic health records (EHRs) are converted to concise and meaningful clinical concepts. However, the EHR data is also fragmented and is always distributed among independent medical institutions, and they are prohibited from being shared and exchanged. Recently, federated learning offers a paradigm for collaborative learning among different entities, which seemingly provides an ideal potential to further enhance the tensor factorization-based collaborative phenotyping to handle sensitive personal health data. This poses challenges to preserving the privacy of the exchanged intermediary results in order to protect the sensitive patient information. Meanwhile, efforts still need to be made to overcome the limitations of the federated tensor factorization, including the restrictions to the classic tensor factorization, high communication cost and reduced accuracy. Furthermore, it is essential to develop the decentralization techniques for federated tensor factorization to deal with the vulnerability of the central server to malfunction and external attacks. To deal with these challenging problems, we propose 1) a *privacy-preserving* collaborative tensor factorization method for computational phenotyping which is able to deal with heterogeneous data with rigorous privacy guarantee and achieves less communication cost and comparable accuracy; 2) a *communication efficient* federated *generalized* tensor factorization, which is flexible enough to choose from a variate of losses to best suit different types of data in practice; 3) a communication efficient *decentralized* generalized tensor factorization method which enables the absence of the central server and further reduces the communication cost.

# Federated Tensor Factorization for Collaborative Health Data Analytics

By

Jing Ma
B.A., Shanghai University of Finance and Economics, 2012

Advisor: Joyce C. Ho, Ph.D.

A dissertation submitted to the Faculty of the
James T. Laney School of Graduate Studies of Emory University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in Computer Science and Informatics
2021

Acknowledgments

The process to the achievement of this work is long and arduous. Fortunately enough, I have received the help and support from many people, without whom, I could not achieve this single-handedly.

First and foremost, I would like to express my sincerest appreciation to my advisor, Dr. Joyce Ho, who has guided me all the way from the tough beginning and has always acted as my academic role model that taught me to be self-motivated, creative, and hardworking. I would like to thank her for generously devoting her expertise and training me from an academic novice to a better researcher. She has been so supportive and can always offer precious advises and insightful discussions.

The same gratitude goes to my co-advisor, Dr. Li Xiong. I would like to thank her for discovering the potential in me and offering me professional advises and knowledge on my research. She also offers me opportunities for various kinds of collaborations. Her enthusiasm to work, patience and encouragement have always inspired me.

I would also like to extend my gratefulness to my dissertation committee member, Dr. Vaidy Sunderam for valuable and detailed comments to my dissertation, and for helping me during the hard time. I would also like to thank Dr. Xiaoqian Jiang for his impressive discussions on my first work and and inspiring suggestions for this dissertation.

Thanks to the lab colleagues from PRADA-X and AIMS groups. I have learnt a lot from them. My life would not be this colorful without them. At the same time, I would like to thank my collaborator Jian Lou for his contributions. I would also like to thank Dr. Carl Yang for the collaboration and insights on Graph Neural Networks.

Finally, I would like to thank my parents for their unconditional support and encouragement during the time of my depression and self-doubt. Specially, I would like to thank my husband, Qiuchen Zhang for accompanying me through ups and downs in the past years.

# Contents

# List of Figures

# List of Tables

# List of Algorithms

# Chapter 1

# Introduction

## 1.1 Motivations

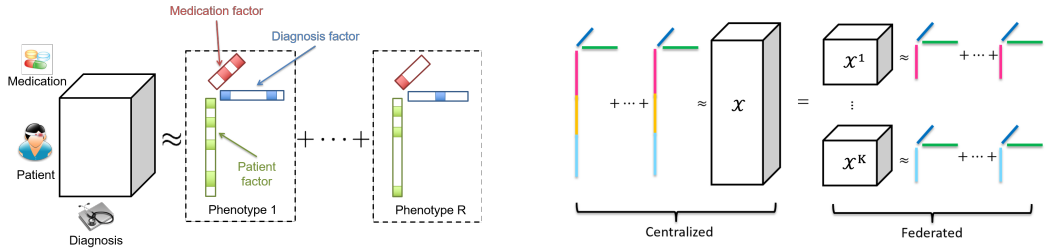
Electronic Health Records (EHRs) have become an important source of comprehensive information for patients' clinical histories. In recent years, the widespread adoption of EHR systems has facilitated the rapid accumulation of patients' clinical data from numerous medical institutions – enabling not only more accurate capturing of patients' medical information and clinical decision support [23], but also improving workflow efficiency and patient care quality [85]. Computational phenotyping is the process of transforming the noisy, massive EHR data into meaningful medical concepts (i.e., phenotypes), which characterize a patient's clinical behavior and corresponding treatments, and not only can be used to assist with in-depth medical decision making (precision medicine, influenza surveillance, drug discovery, etc.), speedup biomedical discovery, improve the quality of healthcare treatment [90, 72], but also can be further applied to multiple downstream tasks such as Genome-wide association studies (GWAS) [53, 61] and risk prediction of a certain disease [27].

Yet, extracting precise and meaningful phenotypes from EHRs can be challenging due to the fact that the observations of EHRs are usually high-dimensional and

heterogeneous, which reduce the interpretability and research quality for scientists [90]. Traditional phenotyping approaches require the involvement of medical domain experts, which is time-consuming and labor-intensive. Recently, unsupervised learning methods have been demonstrated as a more efficient approach for computational phenotyping. Although these methods do not require experts to manually label the data, they require large volumes of EHR data. Meanwhile, the sensitive nature of health data prohibits it from being collected and exchanged among health institutions.

Recently, tensor factorization has risen to be a popular unsupervised computational phenotyping approach [45, 88, 36, 35, 62] as illustrated in Fig. 1.1 (left). Benefited from the capability of succinctly representing the multidimensional data [46], tensors are able to capture the interactions between multiple sources (e.g, specific procedures that are used to treat a disease) and identify patient subgroups and extract concise and potentially more interpretable results by utilizing the multi-way structure of a tensor. Tensors also have a wide range of applications beyond health data analytics, e.g., recommender systems [42], spatio-temporal data analysis [63], computer vision [86], and signal processing [77]. The CANDECOMP/PARAFAC or canonical polyadic (CP) tensor factorization (TF) [11, 31] and its generalization Generalized CANDECOMP/PARAFAC (GTF) [37] are fundamental tools for analyzing the tensors. Despite their effectiveness and wide applications, the scalability is often a major issue preventing it from being applied to larger scale health datasets, which are commonly encountered nowadays. To improve the scalability of TF, distributed tensor factorization (DTF) methods [16, 75, 102, 9, 45, 62, 32] are capable of processing large tensors that cannot be dealt by a single machine.

One existing barrier for high-throughput tensor factorization is that EHRs are fragmented and distributed among independent medical institutions, where healthcare practises are different due to heterogeneous patients populations. One of the reasons is that different hospitals or medical sites differ in the way they manage patients




(a) Illustration of EHR tensor and phenotype extraction via tensor factorization [36].

(b) Illustration of collaborative phenotyping via federated tensor factorization [45].

Figure 1.1: Illustration of Tensor Factorization and Federated Tensor Factorization for Collaborative Computational Phenotyping

[93]. Moreover, effective phenotyping requires a large amount of data to guarantee its reliance and generalizability. Simply analyzing data from single source leads to poor accuracy and bias, which in turn reduces the quality and efficiency of patients' care.

Recent studies have suggested that the integration of health records can provide more benefits [29], which motivated the application of federated tensor learning framework [45] as a better DTF paradigm for decentralized data in terms of privacy protection, while maintaining similar computational and storage scalability. It avoids communicating both the raw tensor and individual mode related variables to the server, which shares the same spirit of the more general federated learning [40], i.e., to learn a joint model across all the clients without communicating individual-level data. By avoiding sharing the raw tensor and the patient mode related variables (e.g., patient factor and partial gradient along the patient mode), Federated Tensor Factorization (FTF) offers better patient privacy protection. Fig. 1.1 (right) illustrates the FTF for collaborative phenotyping.

Recent work [45] proposed a federated tensor factorization model (TRIP) and applied it to the federated phenotyping. This work suffered from six major limitations: 1) There is no rigorous privacy guarantees with the transmission of the intermediate results. 2) There is no mechanism to deal with the non-iid data (i.e., heterogeneous distribution of patients among different medical institutions). 3) TRIP alters the

original objective function by introducing extra terms to enforcing consensus of factors among all clients: it introduces linear constraint and transforms the objective function to Lagrangian dual formulation. These terms can lead the extracted factors to deviate from the centralized solution, thus negatively impacting the phenotyping accuracy. 4) Each communication round of TRIP incurs high communication cost since it requires sending all factors in full precision along with the communication of the Lagrangian dual variables, which doubles the communication cost. 5) TRIP is limited to the CP model and applies least square solver as its client side local updater, which is difficult to be extended to more general losses other than least square loss. 6) The existence of the central server makes the federated learning system vulnerable to malfunctions and malicious adversaries, which will lead to inaccurate model training and leakage of sensitive data. Overall, we summarize the challenges of federated tensor factorization for collaborative phenotyping from 4 perspectives, including privacy, efficiency, and decentralization as 3 major problems, and the utility as the problem that is accompanied with the 3 major problems.

## 1.2 Research Contributions

In this dissertation, we develop federated tensor factorization models which target the 3 major aspects. 1) To deal with the privacy issue, we propose a privacy-preserving efficient federated tensor factorization framework, which offers rigorous privacy guarantee. 2) To deal with the communication efficiency problem, we propose algorithms to aggressively reduce the uplink communication cost with theoretical convergence preservation. 3) To deal with the central server single-point-failure issue, we propose the algorithm that requires no central server for coordination. Overall, we also incorporate multiple techniques to further enhance the utility of the model. Our contributions are briefly summarized as follows:

### 1.2.1 Privacy-preserving Efficient Federated Tensor Factorization

We propose DPFact, a differentially private collaborative tensor factorization framework based on Elastic Averaging Stochastic Gradient Descent (EASGD) for computational phenotyping. DPFact tackles the privacy issue with a well-designed data-sharing strategy, combined with the rigorous zero-concentrated differential privacy (zCDP) technique which is a strictly stronger definition than $(\epsilon, \delta)$-differential privacy considered as the dominant standard for strong privacy protection. We briefly summarize our contributions as:

- Efficiency. DPFact achieves higher accuracy and better convergence than the state-of-the-art federated learning method TRIP. It also reduces the communication cost by the elimination of auxiliary parameters in the ADMM approach.

- Utility. By incorporating a $l_{2,1}$ regularization term, DPFact can jointly decompose local tensors with different distribution patterns and discover both the globally shared and the distinct, site-specific phenotypes.

- Privacy. DPFact provides rigorous privacy protection by applying the zCDP mechanism, which guarantees that there is no inadvertent patient information leakage in the process of intermediary results exchange with high probability which is quantified by privacy parameters.

### 1.2.2 Federated Generalized Tensor Factorization with Improved Communication Efficiency

We study the under explored communication efficiency problem in federated (and more broadly the distributed) generalized tensor factorization for collaborative phenotyping. We propose FedGTF-EF and FedGTF-EF-PC as communication efficient federated generalized tensor factorization algorithms. We summarize our contribution from the perspective of communication and convergence rate:

- Communication reduction. First, we design a two-level per-round communication reduction strategy and propose **FedGTF-EF**. For the block-level, we utilize the block randomization, which enables each client to send only the partial gradient of the sampled block, rather than the full gradient of all blocks. For the element-level, we introduce gradient compression techniques to compress each element of the communicated partial gradient from the floating point representation to low-precision representation. We further reduce the number of communication rounds by introducing periodic communication into and propose **FedGTF-EF-PC**, in which the clients send the update to the server after $\tau > 1$ local iterations.

- Convergence rate. We analyze the convergence of **FedGTF-EF** and obtain the $O(\frac{1}{\sqrt{T}})$ rate after $T$ iterations under common and mild assumptions. The convergence is equivalent to the distributed stochastic gradient descent (SGD) with full precision gradient communication and distributed SGD with gradient compression and error-feedback [103]. In addition, we analyze the convergence of **FedGTF-EF-PC** and obtain the same convergence $O(\frac{1}{\sqrt{T}})$ rate with **FedGTF-EF** under the same set of assumptions. Overall, our proposed **FedGTF-EF-PC** can reduce up to $1 - \frac{1}{32D\tau}$ uplink communication cost if the `Sign` compressor (Def.3.2.1) is used.

### 1.2.3 Decentralized Communication-efficient Generalized Tensor Factorization

We study the communication reduction technique for distributed tensor factorization under the decentralized setting. We propose CiderTF and CiderTF_m, which involves further communication reduction compared with FedGTF-EF-PC, while achieving comparable convergence rate. We summarize our contribution as:

- Communication reduction. Besides the three-level communication introduced in FedGTF-EF-PC, we further introduce an event-triggered communication technique to reduce the communication cost (CiderTF). In this way, besides the periodic communication, each node will only communicate the compressed model updates when the change of its local parameters is beyond a certain threshold.
- We further incorporate the Nesterov's momentum to each client's local gradient updates and propose CiderTF_m to improve the convergence and the generalization.

## 1.3 Organization

The rest of this dissertation is organized as follows. Chapter 2 introduces a privacy-preserving federated tensor factorization framework based on EASGD for optimization and differential privacy for privacy guarantee. Chapter 3 proposes the federated tensor factorization frameworks to further reduce the communication cost from three levels with theoretical convergence guarantee. Chapter 4 extend the work in Chapter 3 to the decentralized settings, and further reduce the communication cost and offers faster convergence by incorporating Nesterov's momentum. Finally, Chapter 5 concludes the dissertation and discusses the future directions of the work.

# Chapter 2

# Privacy-preserving Efficient Federated Tensor Factorization[1]

In this chapter, we target the privacy issue, which is the main challenge for federated tensor factorization. In addition, the proposed algorithm is able to handle the data heterogeneity and reduce the communication cost.

## 2.1 Introduction

Recent studies have suggested that the integration of health records can provide more benefits [29], which motivated the application of federated tensor learning framework [45]. It can mitigate privacy issues under the distributed data setting while achieves high global accuracy and data harmonization via federated computation. But this method has inherent limitations of federated learning: 1) high communication cost; 2) reduced accuracy due to local non-IID data (i.e., patient heterogeneity); and 3) no formal privacy guarantee of the intermediary results shared between local sites and

[1]This chapter is based on the following earlier work [62]: 

the server, which makes patient data at risk of leakage.

In this chapter, we propose DPFact, a differentially private collaborative tensor factorization framework based on Elastic Averaging Stochastic Gradient Descent (EASGD) for computational phenotyping. DPFact assumes all sites share a common model which can be learnt jointly from each site through communication with a central parameter server. Each site performs its own tensor factorization task to discover both common and distinct latent components, while benefiting from the intermediary results generated by other sites. The intermediary results uploaded still contain sensitive information about the patients. Several studies have shown that machine learning models can be used to extract sensitive information used in the input training data through membership inference attacks or model inversion attacks both in the centralized setting [76, 24] and federated setting [34]. Since we assume the central server and participants are honest-but-curious, hence a formal differential privacy guarantee is desired. DPFact tackles the privacy issue with a well-designed data-sharing strategy, combined with the rigorous zero-concentrated differential privacy (zCDP) technique [20, 99] which is a strictly stronger definition than $(\epsilon, \delta)$-differential privacy considered as the dominant standard for strong privacy protection [21, 22, 20]. DPFact offers the following advances to the state-of-the-art federated tensor factorization algorithm.

**1) Efficiency.** DPFact achieves higher accuracy and faster convergence rate than the state-of-the-art federated learning method. It also beats the federated learning method in achieving lower communication cost thanks to the elimination of auxiliary parameters (e.g., in the ADMM approach) and allows each local site to perform most of the computation.

**2) Utility.** DPFact supports phenotype discovery even with a rigorous privacy guarantee. By incorporating a $l_{2,1}$ regularization term, DPFact can jointly decompose local tensors with different distribution patterns and discover both the globally shared

and the distinct, site-specific phenotypes.

**3) Privacy.** DPFact is a privacy-preserving collaborative tensor factorization framework. By applying zCDP mechanisms, it guarantees that there is no inadvertent patient information leakage in the exchange of intermediary results with high probability which is quantified by privacy parameters.

We evaluate DPFact on two publicly-available large EHR datasets and a synthetic dataset. The performance of DPFact is assessed from the following three aspects including efficiency measured by accuracy and communication cost, utility measured by phenotype discovery ability and the evaluation on the effect of privacy.

## 2.2 Preliminaries and Backgrounds

This section describes the preliminaries used in this chapter, including tensor factorization, $(\epsilon, \delta)$-differential privacy, and zCDP.

### 2.2.1 Tensor Factorization

**Definition 2.2.1.** (*Khatri-Rao product*). Khatri-Rao product is the "columnwise" Kronecker product of two matrices $\mathbf{A} \in \mathbb{R}^{I \times R}$ and $\mathbf{B} \in \mathbb{R}^{J \times R}$. The result is a matrix of size $(IJ \times R)$ and defined by

$$\mathbf{A} \odot \mathbf{B} = [\mathbf{a}_1 \otimes \mathbf{b}_1 \cdots \mathbf{a}_R \otimes \mathbf{b}_R]$$

Here, $\otimes$ denotes the *Kronecker product*. The *Kronecker product* of two vectors $\mathbf{a} \in \mathbb{R}^I$, $\mathbf{b} \in \mathbb{R}^J$ is

$$\mathbf{a} \otimes \mathbf{b} = \begin{bmatrix} a_1\mathbf{b} \\ \vdots \\ a_I\mathbf{b} \end{bmatrix}$$

**Definition 2.2.2.** (*CANDECOMP-PARAFAC Decomposition*). The CANDECOMP-PARAFAC (CP) decomposition approximates the original tensor $\mathcal{O}$ by the sum of $R$ rank-one tensors. It can be expressed as

$$\mathcal{O} \approx \mathcal{X} = \sum_{r=1}^{R} \mathbf{a}_{:r}^{(1)} \circ \cdots \circ \mathbf{a}_{:r}^{(N)}, \tag{2.1}$$

where $R$ is the rank of tensor $\mathcal{O}$, $\mathbf{a}_{:r}^{(n)}$ represents the $r^{th}$ column of $A^{(n)}$ for $n = 1, \cdots, N$ and $r = 1, \cdots, R$. $A^{(n)}$ is the $n$-mode factor matrix consisting of $R$ columns representing $R$ latent components which can be represented as

$$A^{(n)} = \left[\mathbf{a}_{:1}^{(n)} \cdots \mathbf{a}_{:R}^{(n)}\right],$$

so that $A^{(n)}$ is of size $I_n \times R$ for $n = 1, \cdots, N$, and the equation of (2.1) can also be represented as

$$[\![A^{(1)}, \cdots, A^{(N)}]\!] = \sum_{r=1}^{R} \mathbf{a}_{:r}^{(1)} \circ \cdots \circ \mathbf{a}_{:r}^{(N)}. \tag{2.2}$$

Note that in this formulation, the scalar weights for each rank-one tensor are assumed to be absorbed into the factors.

In the way of a three-mode tensor $\mathcal{O} \in \mathbb{R}^{I \times J \times K}$, the CP decomposition can be represented as

$$\mathcal{O} \approx \mathcal{X} = \sum_{r=1}^{R} \mathbf{a}_{:r} \circ \mathbf{b}_{:r} \circ \mathbf{c}_{:r}, \tag{2.3}$$

where $\mathbf{a}_{:r} \in \mathbb{R}^I$, $\mathbf{b}_{:r} \in \mathbb{R}^J$, $\mathbf{c}_{:r} \in \mathbb{R}^K$ are the $r$-th column vectors within the three factor matrices $\mathbf{A} \in \mathbb{R}^{I \times R}$, $\mathbf{B} \in \mathbb{R}^{J \times R}$, $\mathbf{C} \in \mathbb{R}^{K \times R}$.

### 2.2.2 Differential Privacy

Differential privacy [22, 20] has been demonstrated as a strong standard to provide privacy guarantees for algorithms on aggregate database analysis, which in our case

is a collaborative tensor factorization algorithm analyzing distributed tensors with differential privacy.

**Definition 2.2.3.** (*($\epsilon$-$\delta$)-Differential Privacy*) [22]. Let $\mathcal{D}$ and $\mathcal{D}'$ be two neighboring datasets that differ in at most one entry. A randomized algorithm $\mathcal{A}$ is ($\epsilon$-$\delta$)-differentially private if for all $\mathcal{S} \subseteq \text{Range}(\mathcal{A})$:

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

where $\mathcal{A}(\mathcal{D})$ represents the output of $\mathcal{A}$ with an input of $\mathcal{D}$.

The above definition suggests that with a small $\epsilon$, an adversary almost cannot distinguish the outputs of an algorithm with two neighboring datasets $\mathcal{D}$ and $\mathcal{D}'$ as its inputs, while $\delta$ allows a small probability of failing to provide this guarantee. Differential privacy is defined using a pair of neighboring databases which in our work are two tensors and differ in only one entry.

**Definition 2.2.4.** ($L_2$-*sensitivity*) [22]. For two neighboring datasets $\mathcal{D}$ and $\mathcal{D}'$ differing in at most one entry, the $L_2$-sensitivity of an algorithm $\mathcal{A}$ is the maximum change in the $l_2$-norm of the output value of algorithm $\mathcal{A}$ regarding the two neighboring datasets:

$$\Delta_2(\mathcal{A}) = \sup_{\mathcal{D},\mathcal{D}'} \|\mathcal{A}(\mathcal{D}) - \mathcal{A}(\mathcal{D}')\|_2.$$

**Theorem 2.2.1.** *((Gaussian Mechanism)) [22]. Let $\epsilon \in (0, 1)$ be arbitrary. For $c^2 > 2\ln(1.25/\delta)$, the Gaussian Mechanism with parameter $\sigma \geq c\Delta_2(\mathcal{A})/\epsilon$, adding noise scaled to $\mathcal{N}(0, \sigma^2)$ to each component of the output of algorithm $\mathcal{A}$, is ($\epsilon$-$\delta$)-differentially private.*

### 2.2.3 Concentrated Differential Privacy

Concentrated differential privacy (CDP) is introduced by Dwork and Rothblum [20] as a generalization of differential privacy which provides sharper analysis of many privacy-preserving computations. Bun and Steinke [10] propose an alternative formulation of CDP called "zero-concentrated differential privacy" (zCDP) which utilizes the Rényi divergence between probability distributions to measure the requirement of the privacy loss random variable to be sub-gaussian and provides tighter privacy analysis.

**Definition 2.2.5.** (*Zero-Concentrated Differential Privacy (zCDP) [10]*) A randomized mechanism $\mathcal{A}$ is $\rho$-zero concentrated differentially private if for any two neighboring databases $\mathcal{D}$ and $\mathcal{D}'$ differing in at most one entry and all $\alpha \in (1, \infty)$,

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

where $D_\alpha \left(\mathcal{A}(\mathcal{D}) \| \mathcal{A}\left(\mathcal{D}'\right)\right)$ is called $\alpha$-Rényi divergence between the distributions of $\mathcal{A}(\mathcal{D})$ and $\mathcal{A}\left(\mathcal{D}'\right)$, and $L(o)$ is the privacy loss random variable which is defined as:

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

The following propositions of zCDP will be used in this chapter.

**Proposition 2.2.1.** *[10] The Gaussian mechanism with noise $\mathcal{N}(0, \sigma^2)$ where $\sigma = \sqrt{1/(2\rho)}\Delta_2$ satisfies $\rho$-zCDP.*

**Proposition 2.2.2.** *[10] If a randomized mechanism $\mathcal{A}$ is $\rho$-CDP, then $\mathcal{A}$ is $(\epsilon', \delta)$-DP for any $\delta$ with $\epsilon' = \rho + \sqrt{4\rho \log(1/\delta)}$; For $\mathcal{A}$ to satisfy $(\epsilon, \delta)$-DP, it suffices to satisfy $\rho$-zCDP by setting $\rho \approx \frac{\epsilon^2}{4 \log(1/\delta)}$.*

**Proposition 2.2.3.** *((Serial composition [10])) Let $\mathcal{A} : \mathcal{D}^n \to \mathcal{Y}$ and $\mathcal{A}' : \mathcal{D}^n \to \mathcal{Z}$ be randomized algorithms. Suppose $\mathcal{A}$ is $\rho$-zCDP and $\mathcal{A}'$ is $\rho'$-zCDP. Define $\mathcal{A}'' : \mathcal{D}^n \to \mathcal{Y} \times \mathcal{Z}$ by $\mathcal{A}'' = (\mathcal{A}, \mathcal{A}')$. Then $\mathcal{A}''$ is $(\rho + \rho')$-zCDP.*

**Proposition 2.2.4.** *((Parallel composition [99])) Suppose that a mechanism $\mathcal{A}$ consists of a sequence of $T$ adaptive mechanisms, $\mathcal{A}_1, \ldots, \mathcal{A}_T$, where each $\mathcal{A}_t : \Pi_{j=1}^{iter-1}\mathcal{O}_j \times \mathcal{D}_t \to \mathcal{O}_{iter}$ and $\mathcal{A}_t$ satisfies $\rho_t$-zCDP. Let $\mathcal{D}_1, \ldots, \mathcal{D}_T$ be a randomized partition of the input $\mathcal{D}$. The mechanism $\mathcal{A}(\mathcal{D}) = (\mathcal{A}_1(\mathcal{D}_1), \ldots, \mathcal{A}_T(\mathcal{D}_T))$ satisfies $\frac{1}{T}\sum_{t=1}^{T}\rho_t$-zCDP.*

### 2.2.4 Related Works

#### Tensor Factorization

Tensor analysis is an active research topic and has been widely applied to healthcare data [45, 88, 36], especially for computational phenotyping. Moreover, several algorithms have been developed to scale tensor factorization. GigaTensor [41] used MapReduce for large scale CP tensor decomposition that exploits the sparseness of the real world tensors. DFacTo [16] improves GigaTensor by exploring properties related to the Khatri-Rao Product and achieves faster computation time and better scalability. FlexiFaCT [9] is a scalable MapReduce algorithm for coupled matrix-tensor decomposition using stochastic gradient descent (SGD). ADMM has also been proved to be an efficient algorithm for distributed tensor factorization [45]. However, the above proposed algorithms have the same potential limitation: the distributed data exhibits the same pattern at different local sites. That means each local tensor can be treated as a random sample from the global tensor. Thus, the algorithms are unable to model the scenario where the distribution pattern may be different at each sites. This is common in healthcare as different units (or clinics and hospitals) will have different patient populations, and may not exhibit all the computational phenotypes.

#### Differential Private Factorization

Differential privacy is widely applied to machine learning areas, especially matrix/tensor factorization, as well as on different distributed optimization frameworks

and deep learning problems. Regarding tensor decomposition, there are four ways to enforce differential privacy: input perturbation, output perturbation, objective perturbation and the gradient perturbation. [38] proposed an objective perturbation method for matrix factorization in recommendation systems. [57] proposed a new idea that sampling from the posterior distribution of a Bayesian model can sufficiently guarantee differential privacy. [8] compared the four different perturbation method on matrix factorization and drew the conclusion that input perturbation is the most efficient method that has the least privacy loss on recommendation systems. [89] is the first proposed differentially private tensor decomposition work. It proposed a noise calibrated tensor power method. Our goal in this chapter is to develop a distributed framework where data is stored at different sources, and try to preserve the privacy during knowledge transfer. Nevertheless, these works are based on a centralized framework. [45] developed a federated tensor factorization framework, but it simply preserves privacy by avoiding direct patient information sharing, rather than by applying rigorous differential privacy techniques.

## 2.3 DPFact

In this section, we first provide a general overview and then present detailed formulation of the optimization problem.

### 2.3.1 Overview

DPFact is a distributed tensor factorization model that preserves differential privacy. Our goal is to learn computational phenotypes from horizontally partitioned patient data (e.g., each hospital has its own patient data with the same medical features). Since we assume the central server and participants are honest-but-curious which means they will not deviate from the prescribed protocol but they are curious about others

secrets and try to find out as much as possible about them. Therefore the patient data cannot be collected at a centralized location to construct a global tensor $\mathcal{O}$. Instead, we assume that there are $T$ local sites and a central server that communicates the intermediary results between the local sites. Each site performs tensor factorization on the local data and shares privacy-preserving intermediary results with the centralized server (Fig. 2.1).

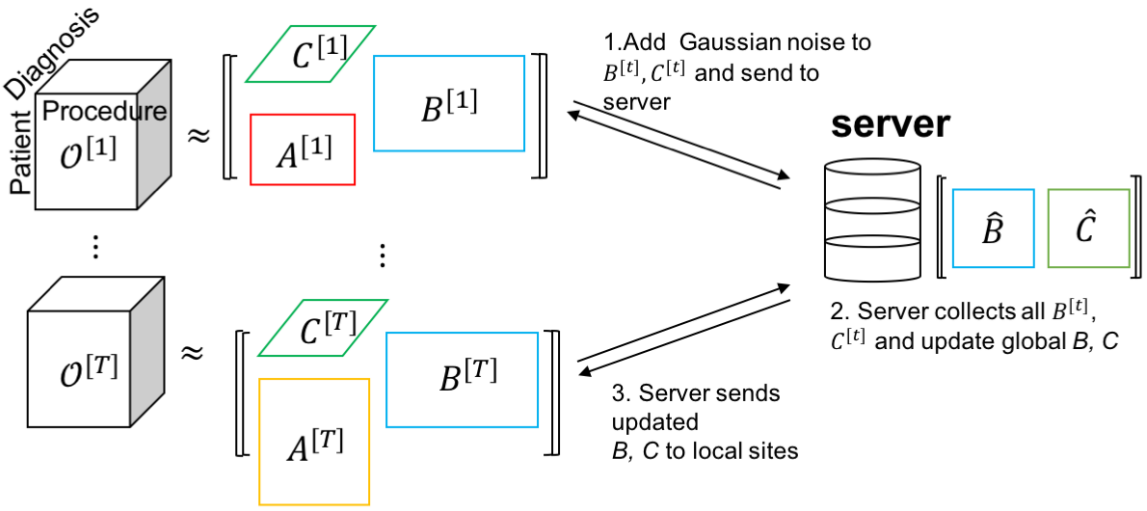



Figure 2.1: Algorithm Overview

The patient data at each site is used to construct a local observed tensor, $\mathcal{O}^{[t]}$. For simplicity and illustration purposes, we discuss a three-mode tensor situation where the modes are patients, procedures, and diagnoses but DPFact generalizes to $N$ modes. The $T$ sites jointly decompose their local tensor into three factor matrices: a patient factor matrix $\mathbf{A}^{[t]}$ and two feature factor matrices $\mathbf{B}^{[t]}$ and $\mathbf{C}^{[t]}$. We assume that the factor matrices on the non-patient modes (i.e., $\mathbf{B}^{[t]}, \mathbf{C}^{[t]}$) are the same across the $T$ sites, thus sharing the same computational phenotypes. To achieve consensus of the shared factor matrices, the non-patient feature factor matrices are shared in a privacy-preserving manner with the central server by adding Gaussian noise to each uploaded factor matrix.

Although the collaborative tensor problem for computational phenotyping has

been previously discussed [45], DPFact provides three important contributions:

**(1) Efficiency**: We adopt a communication-efficient stochastic gradient descent (SGD) algorithm for collaborative learning which allows each site to transmit less information to the centralized server while still achieving an accurate decomposition.

**(2) Heterogeneity**: A traditional global consensus model requires learning the same shared model from multiple sources. However, different data sources may have distinct patterns and properties (e.g., disease prevalence may differ between Georgia and Texas). We propose using the $l_{2,1}$-norm to achieve global consensus among the sites while capturing site-specific factors.

**(3) Differential Privacy Guarantees**: We preserve the privacy of intermediary results by adding Gaussian noise to each non-patient factor matrix prior to sharing with the parameter server. This masks any particular entry in the factor matrices and prevents inadvertent privacy leakage. A rigorous privacy analysis based on zCDP is performed to ensure strong privacy protection for the patients.

### 2.3.2 Formulation

Under a single (centralized) model, CP decomposition of the observed tensor $\mathcal{O}$ results in a factorized tensor $\mathcal{X}$ that contains the $R$ most prevalent computational phenotypes. We represent the centralized tensor as $T$ separate horizontal partitions, $\mathcal{O}^{[1]}, \cdots, \mathcal{O}^{[T]}$. Thus, the global function can be expressed as the sum of $T$ separable functions with respect to each local factorized tensor $\mathcal{X}^{[t]}$ [45]:

$$\min_{\mathcal{X}} \mathcal{L} = \frac{1}{2}||\mathcal{O} - \mathcal{X}||_F^2 = \sum_{t=1}^{T} \frac{1}{2}\left\|\mathcal{O}^{[t]} - \mathcal{X}^{[t]}\right\|_F^2. \tag{2.4}$$

Since the goal is to uncover computational phenotypes that are shared across all sites, we restrict the sites to factorize the observed local tensors $\mathcal{O}^{[t]}$ such that the non-patient factor matrices are the same. Therefore, the global optimization problem

is formulated as:

$$\begin{aligned} \min \ & \sum_{t=1}^{T} \frac{1}{2} \left\| \mathcal{O}^{[t]} - [\![ \mathbf{A}^{[t]}, \mathbf{B}^{[t]}, \mathbf{C}^{[t]} ]\!] \right\|_F^2 \\ \text{s.t. } & \mathbf{B}^{[1]} = \mathbf{B}^{[2]} = \cdots = \mathbf{B}^{[T]} \\ & \mathbf{C}^{[1]} = \mathbf{C}^{[2]} = \cdots = \mathbf{C}^{[T]}. \end{aligned}$$

This can be reformulated as a global consensus optimization, which decomposes the original problem into $T$ local subproblems by introducing two auxiliary variables, $\hat{\mathbf{B}}, \hat{\mathbf{C}}$, to represent the global factor matrices. A quadratic penalty is placed between the local and global factor matrices to achieve global consensus among the $T$ different sites. Thus, the local optimization problem at site $t$ is:

$$\begin{aligned} \min \ & \frac{1}{2} \left\| \mathcal{O}^{[t]} - [\![ \mathbf{A}^{[t]}, \mathbf{B}^{[t]}, \mathbf{C}^{[t]} ]\!] \right\|_F^2 \\ & + \frac{\gamma}{2} \left\| \mathbf{B}^{[t]} - \hat{\mathbf{B}} \right\|_F^2 + \frac{\gamma}{2} \left\| \mathbf{C}^{[t]} - \hat{\mathbf{C}} \right\|_F^2. \end{aligned} \tag{2.5}$$

### 2.3.3 Heterogeneous Patient Populations

The global consensus model assumes that the patient populations are the same across different sites. However, this may be too restrictive as some locations can have distinctive patterns. For example, patients from the cardiac coronary unit may have unique characteristics that are different from the surgical care unit. DPFact utilizes the $l_{2,1}$-norm regularization, to allow flexibility for each site to "turn off" one or more computational phenotypes. For an arbitrary matrix $\mathbf{W} \in \mathbb{R}^{m \times n}$, its $l_{2,1}$-norm is defined as:

$$\|\mathbf{W}\|_{2,1} = \sum_{i=1}^{m} \sqrt{\sum_{j=1}^{n} \mathbf{W}_{ij}^2}. \tag{2.6}$$

From the definition, we can see that the $l_{2,1}$-norm controls the row sparsity of matrix $\mathbf{W}$. As a result, the $l_{2,1}$-norm is commonly used in multi-task feature learning to

perform feature selection as it can induce structural sparsity [30, 55, 96, 67].

DPFact adopts a multi-task perspective, where each local decomposition is viewed as a separate task. Under this approach, each site is not required to be characterized by all $R$ computational phenotypes. To achieve this, we introduce the $l_{2,1}$-norm on the transpose of the patient factor matrices, $\mathbf{A}^{[t]}$, to induce sparsity on the columns. The idea is that if a specific phenotype is barely present in any of the patients (2-norm of the column is close to 0), the regularization will encourage all the column entries to be 0. This can be used to capture the heterogeneity in the patient populations without violating the global consensus assumption. Thus the DPFact optimization problem is:

$$\begin{aligned} \min \ \sum_{t=1}^{T} ( & \frac{1}{2} \left\| \mathcal{O}^{[t]} - [\![ \mathbf{A}^{[t]}, \mathbf{B}^{[t]}, \mathbf{C}^{[t]} ]\!] \right\|_F^2 + \frac{\gamma}{2} \left\| \mathbf{B}^{[t]} - \hat{\mathbf{B}} \right\|_F^2 \\ & + \frac{\gamma}{2} \left\| \mathbf{C}^{[t]} - \hat{\mathbf{C}} \right\|_F^2 + \mu \left\| (\mathbf{A}^{[t]})^\top \right\|_{2,1} ). \end{aligned} \tag{2.7}$$

The quadratic penalty, $\gamma$, provides an elastic force to achieve global consensus between the local factor matrices and the global factor matrices whereas the $l_{2,1}$-norm penalty, $\mu$, encourages sites to share similar sparsity patterns.

## 2.4 DPFact Optimization

DPFact adopts the Elastics Averaging SGD (EASGD) [101] approach to solve the optimization problem (2.7). EASGD is a communication-efficient algorithm for collaborative learning and has been shown to be more stable than the Alternating Direction Method of Multipliers (ADMM) with regard to parameter selection. Moreover, SGD-based approaches scale well to sparse tensors, as the computation is bounded by the number of non-zeros.

Using the EASGD approach, the global consensus optimization problem is solved

alternatively between the local sites and the central server. Each site performs multiple rounds of local tensor decomposition and updates their local factor matrices. The site then only shares the most updated non-patient mode matrices with output perturbation to prevent revealing of sensitive information. The patient factor matrix is never shared with the central server to avoid direct leakage of patient membership information. The server then aggregates the updated local factor matrices to update the global factor matrices and sends the new global factor matrices back to each site. This process is iteratively repeated until there are no changes in the local factor matrices. The entire DPFact decomposition process is summarized in Algorithm 1.

**Algorithm 1:** DPFact

**Input:** $\mathcal{O}$, $\tau$ $\eta$, $\gamma$, $\mu$, $\sigma$, $\rho$.

Randomly initialize the global feature factor matrices $\mathbf{B}$, $\mathbf{C}$ and local feature factor matrices $\mathbf{B}^{[t]}$, $\mathbf{C}^{[t]}$.

**while** $\boldsymbol{B}^{[t]}$, $\boldsymbol{C}^{[t]}$ *not converge* **do**

  **if** ***Hospital*** **then**

    **for** $k = 1, \cdots, \tau$ **do**

      Shuffle tensor elements;

      **for** *observation* $i$ **do**

        Update $\mathbf{A}^{[t]}$ using (2.13);

        Update $\mathbf{B}^{[t]}$, $\mathbf{C}^{[t]}$ using (2.17);

      **end**

      Proximal update for ${}_{new}\mathbf{A}^{[t]}$ using (2.14);

    **end**

    Calibrate *Gaussian* noise matrix $\mathcal{M}_B^{[t]}$ and $\mathcal{M}_C^{[t]}$ as $\mathcal{N}\left(0, \Delta_2^2/(2\rho)\right)$ for each factor matrix;

    Update factor matrices ${}_{priv}\mathbf{B}^{[t]}$ and ${}_{priv}\mathbf{C}^{[t]}$ using (2.18);

    Send ${}_{priv}\mathbf{B}^{[t]}$, ${}_{priv}\mathbf{C}^{[t]}$ to Server.

  **end**

  **if** ***Server*** **then**

    Collect ${}_{priv}\mathbf{B}^{[t]}$, ${}_{priv}\mathbf{C}^{[t]}$ from each hospital;

    Update $\widehat{\mathbf{B}}$, $\widehat{\mathbf{C}}$ using (2.19);

    Send $\widehat{\mathbf{B}}$, $\widehat{\mathbf{C}}$ back to hospitals.

  **end**

**end**

### 2.4.1 Local Factors Update

Each site updates the local factors by solving the following subproblem:

$$\begin{aligned}\min\ & \frac{1}{2}\left\|\mathcal{O}^{[t]} - [\![\mathbf{A}^{[t]}, \mathbf{B}^{[t]}, \mathbf{C}^{[t]}]\!]\right\|_F^2 + \frac{\gamma}{2}\left\|\mathbf{B}^{[t]} - \hat{\mathbf{B}}\right\|_F^2 \\ & + \frac{\gamma}{2}\left\|\mathbf{C}^{[t]} - \hat{\mathbf{C}}\right\|_F^2 + \mu\left\|(\mathbf{A}^{[t]})^\top\right\|_{2,1}.\end{aligned} \tag{2.8}$$

EASGD helps reduce the communication cost by allowing sites to perform multiple iterations (each iteration is one pass of the local data) before sending the updated factor matrices. We further extend the local optimization updates using permutation-based SGD (P-SGD), a practical form of SGD [91]. In P-SGD, instead of randomly sampling one instance from the tensor at a time, the non-zero elements are first shuffled within the tensor. The algorithm then cycles through these elements to update the latent factors. At each local site, the shuffling and cycling process is repeated $\tau$ times, hereby referred to as a $\tau$-pass P-SGD. There are two benefits of adopting the P-SGD approach: 1) the resulting algorithm is more computationally effective as it eliminates some of the randomness of the basic SGD algorithm. 2) it provides a mechanism to properly estimate the total privacy budget (see Section 2.4.2).

#### Patient Factor Matrix

For site $t$, the patient factor matrix $\mathbf{A}^{[t]}$ is updated by minimizing the objective function using the local factorized tensor, $\mathcal{X}^{[t]}$ and the $l_{2,1}$-norm:

$$\min_{\mathbf{A}^{[t]}} \underbrace{\frac{1}{2}\left\|\mathcal{O}^{[t]} - [\![\mathbf{A}^{[t]}, \mathbf{B}^{[t]}, \mathbf{C}^{[t]}]\!]\right\|_F^2}_{\mathcal{F}} + \underbrace{\mu\left\|(\mathbf{A}^{[t]})^\top\right\|_{2,1}}_{\mathcal{H}}. \tag{2.9}$$

While the $l_{2,1}$-norm is desirable from a modeling perspective, it also results in a non-differentiable optimization problem. The local optimization problem (2.9) can be seen as a combination of a differentiable function $\mathcal{F}$ and a non-differentiable function

$\mathcal{H}$. Thus, we propose using the proximal gradient descent method to solve local optimization problem for the patient mode. Proximal gradient method can be applied in our case since the gradient of the differentiable function $\mathcal{F}$ is *Lipschitz* continuous with a *Lipschitz* constant $L$ (see Appendix for details).

Using the proximal gradient method, the factor matrix $\mathbf{A}^{[t]}$ is iteratively updated via the proximal operator:

$$_{new}\mathbf{A}^{[t]} = \mathbf{prox}_{\eta\mathcal{H}}\left(\mathbf{A}^{[t]} - \eta\nabla\mathcal{F}(\mathbf{A}^{[t]})\right), \tag{2.10}$$

where $\eta > 0$ is the step size at each local iteration. The proximal operator is computed by solving the following equation:

$$\mathbf{prox}_{\eta\mathcal{H}}(\Theta) = \arg\min_{\Theta}\left(\frac{1}{2\eta}\|\Theta - \hat{\Theta}\| + \mathcal{H}(\Theta)\right), \tag{2.11}$$

where $\hat{\Theta} = \mathbf{A}^{[t]} - \eta\nabla\mathcal{F}(\mathbf{A}^{[t]})$ is the updated matrix. It has been shown that if $\nabla\mathcal{F}$ is *Lipschitz* continuous with constant $L$, the proximal gradient descent method will converge for step size $\eta < 2/L$ [17]. For the $l_{2,1}$-norm, the closed form solution can be computed using the soft-thresholding operator:

$$\mathbf{prox}_{\eta\mathcal{H}}(\widehat{\Theta}) = \widehat{\Theta_{r:}}\left(1 - \frac{\mu}{\|\widehat{\Theta_{r:}}\|_2}\right)_+, \tag{2.12}$$

where $r \in (0, R]$ and $r$ represents the $r$-th column of the factor matrix $\widehat{\Theta}$, and $(z)_+$ denotes the maximum of 0 and $z$. Thus, if the norm of the $r$-th column of the patient matrix is small, the proximal operator will "turn off" that column.

The gradient of the smooth part can be derived with respect to each row in the patient mode factor matrix, $\mathbf{A}^{[t]}$. The update rule for each row is:

$$\mathbf{a}^{[t]}_{i:} \leftarrow \mathbf{a}^{[t]}_{i:} - \eta\left[\left(\mathbf{a}^{[t]}_{i:}(\mathbf{b}^{[t]}_{j:} * \mathbf{c}^{[t]}_{k:})^\top - \mathcal{O}^{[t]}_{ijk}\right)\left(\mathbf{b}^{[t]}_{j:} * \mathbf{c}^{[t]}_{k:}\right)\right] \tag{2.13}$$

After one pass through all entries in a local tensor to update the patient factor matrix, the second step is to use proximal operator (2.12) to update the patient factor matrix $\mathbf{A}^{[t]}$:

$$_{new}\mathbf{A}^{[t]} = \mathbf{prox}_{\eta\mathcal{H}}(\mathbf{A}^{[t]}). \tag{2.14}$$

**Feature Factor Matrices**

The local feature factor matrices, $\mathbf{B}^{[t]}$ and $\mathbf{C}^{[t]}$, are updated based on the following objective functions:

$$\begin{aligned}
\min_{\mathbf{B}^{[t]}} f_b &= \frac{1}{2}\left\|\mathcal{O}^{[t]} - [\![\mathbf{A}^{[t]}, \mathbf{B}^{[t]}, \mathbf{C}^{[t]}]\!]\right\|_F^2 + \frac{\gamma}{2}\left\|\mathbf{B}^{[t]} - \hat{\mathbf{B}}\right\|_F^2, \\
\min_{\mathbf{C}^{[t]}} f_c &= \frac{1}{2}\left\|\mathcal{O}^{[t]} - [\![\mathbf{A}^{[t]}, \mathbf{B}^{[t]}, \mathbf{C}^{[t]}]\!]\right\|_F^2 + \frac{\gamma}{2}\left\|\mathbf{C}^{[t]} - \hat{\mathbf{C}}\right\|_F^2.
\end{aligned} \tag{2.15}$$

The partial derivatives of $f_b, f_c$ with respect to $\mathbf{b}_{j:}^{[t]}$ and $\mathbf{c}_{k:}^{[t]}$, the $j$-th and $k$-th row of the $\mathbf{B}^{[t]}$ and $\mathbf{C}^{[t]}$ factor matrices, respectively, are computed.

$$\begin{aligned}
\frac{\partial f_b}{\partial \mathbf{b}_{j:}^{[t]}} &= \left[\left(\mathbf{a}_{i:}^{[t]}(\mathbf{b}_{j:}^{[t]} * \mathbf{c}_{k:}^{[t]})^\top - \mathcal{O}_{ijk}^{[t]}\right)\left(\mathbf{a}_{i:}^{[t]} * \mathbf{c}_{k:}^{[t]}\right)\right] \\
\frac{\partial f_c}{\partial \mathbf{c}_{k:}^{[t]}} &= \left[\left(\mathbf{a}_{i:}^{[t]}(\mathbf{b}_{j:}^{[t]} * \mathbf{c}_{k:}^{[t]})^\top - \mathcal{O}_{ijk}^{[t]}\right)\left(\mathbf{a}_{i:}^{[t]} * \mathbf{b}_{j:}^{[t]}\right)\right].
\end{aligned} \tag{2.16}$$

$\mathbf{B}^{[t]}$ and $\mathbf{C}^{[t]}$ are then updated row by row by adding up the partial derivative of the quadratic penalty term and the partial derivative with respect to $\mathbf{b}_{j:}^{[t]}$ and $\mathbf{c}_{k:}^{[t]}$ shown in (2.16).

$$\begin{aligned}
\mathbf{b}_{j:}^{[t]} &\leftarrow \mathbf{b}_{j:}^{[t]} - \eta\left[\frac{\partial f_n}{\partial \mathbf{b}_{j:}^{[t]}} + \gamma\left(\mathbf{b}_{j:}^{[t]} - \widehat{\mathbf{b}_{j:}}\right)\right]; \\
\mathbf{c}_{k:}^{[t]} &\leftarrow \mathbf{c}_{k:}^{[t]} - \eta\left[\frac{\partial f_n}{\partial \mathbf{c}_{k:}^{[t]}} + \gamma\left(\mathbf{c}_{k:}^{[t]} - \widehat{\mathbf{c}_{k:}}\right)\right].
\end{aligned} \tag{2.17}$$

Each site simultaneously does several rounds ($\tau$) of the local factor updates. After $\tau$ rounds are completed, the feature factor matrices will be perturbed with *Gaussian*

noise and sent to central server.

**Privacy-Preserving Output Perturbation**

Although the feature factor matrices do not directly contain patient information, it may inadvertently violate patient privacy (e.g., a rare disease that is only present in a small number of patients). To protect the patient information from being speculated by semi-honest server, we perturb the feature mode factor matrices using the Gaussian mechanism, a common building block to perturb the output and achieve rigorous differential privacy guarantee.

The Gaussian mechanism adds zero-mean Gaussian noise with standard deviation $\sigma = \Delta_2^2/(2\rho)$ to each element of the output [10]. Thus, the noise matrix $\mathcal{M}$ can be calibrated for each factor matrices $\mathbf{B}^{[t]}$ and $\mathbf{C}^{[t]}$ based on their $L_2$-sensitivity to construct privacy-preserving feature factor matrices:

$$\begin{aligned} {}_{priv}\mathbf{B}^{[t]} &\leftarrow \mathbf{B}^{[t]} + \mathcal{M}_B^{[t]}, \\ {}_{priv}\mathbf{C}^{[t]} &\leftarrow \mathbf{C}^{[t]} + \mathcal{M}_C^{[t]}, \end{aligned} \tag{2.18}$$

As a result, each factor matrix that is shared with the central server satisfies $\rho$-zCDP by Proposition 2.7. A detailed privacy analysis for the overall privacy guarentee is provided in the next subsection.

### 2.4.2 Privacy Analysis

In this section we analyze the overall privacy guarantee of Algorithm 1. The analysis is based on the following knowledge of the optimization problem: 1) each local site performs a $\tau$-pass P-SGD update per epoch; 2) for the local objective function $f$ in (2.15), when fixing two of the factor matrices, the objective function becomes a convex optimization problem for the other factor matrix.

**$L_2$-sensitivity**

The objective function (2.15) satisfies $L - Lipschitz$, with $Lipschitz$ constant $L$ the tight upper bound of the gradient. For a $\tau$-pass P-SGD, having constant learning rate $\eta = \eta_k \leq \frac{2}{\beta}$ ($k = 1, ..., \tau$, $\beta$ is the $Lipschitz$ constant of the gradient of (2.15) regarding $\mathbf{B}^{[t]}$ or $\mathbf{C}^{[t]}$, see Appendix for $\beta$ calculation), the $L_2$-sensitivity of this optimization problem in (2.15) is calculated as $\Delta_2(f) = 2\tau L\eta$ [91].

**Overall Privacy Guarantee**

The overall privacy guarantee of Algorithm 1 is analyzed under the zCDP definition which provides tighter privacy bound than strong composition theorem [21] for multiple folds Gaussian mechanism [10, 99]. The total $\rho$-zCDP will be transferred to $(\epsilon, \delta)$-DP in the end using Proposition 2.8.

**Theorem 2.4.1.** *Algorithm 1 is $(\epsilon, \delta)$-differentially private if we choose the input privacy budget for each factor matrix per epoch as*

$$\rho = \frac{\epsilon^2}{8E \log(1/\delta)}$$

*where $E$ is the number of epochs when the algorithm is converged.*

*Proof.* Let the "base" zCDP parameter be $\rho_b$, $\mathbf{B}^{[t]}$ and $\mathbf{C}^{[t]}$ together cost $2E\rho_b$ after $E$ epochs by Proposition 2.9. All $T$ user nodes cost $\frac{1}{T}\sum_{t=1}^{T} 2E\rho_b = 2E\rho_b$ by the *parallel composition theorem* in Proposition 2.10. By the connection of zCDP and $(\epsilon, \delta)$-DP in Proposition 2.8, we get $\rho_b = \frac{\epsilon^2}{8E \log(1/\delta)}$, which concludes our proof. □

### 2.4.3 Global Variables Update

The server receives $T$ local feature matrix updates, and then updates the global feature matrices according to the same objective function in (2.5). The gradient for the global

feature matrices $\widehat{\mathbf{B}}$ and $\widehat{\mathbf{C}}$ are:

$$\begin{aligned}\widehat{\mathbf{B}} &\leftarrow \widehat{\mathbf{B}} + \eta \sum_{t=1}^{T} \gamma \left( {}_{priv}\mathbf{B}^{[t]} - \widehat{\mathbf{B}} \right) \\ \widehat{\mathbf{C}} &\leftarrow \widehat{\mathbf{C}} + \eta \sum_{t=1}^{T} \gamma \left( {}_{priv}\mathbf{C}^{[t]} - \widehat{\mathbf{C}} \right).\end{aligned} \tag{2.19}$$

The update makes the global phenotypes similar to the local phenotypes at the $T$ local sites. The server then sends the global information, $\widehat{\mathbf{B}}, \widehat{\mathbf{C}}$ to each site for the next epoch.

## 2.5 Experimental Evaluation

We evaluate DPFact on three aspects: 1) efficiency based on accuracy and communication cost; 2) utility of the phenotype discovery; and 3) impact of privacy. The evaluation is performed on both real-world datasets and synthetic datasets.

### 2.5.1 Experimental Settings

#### Dataset

We evaluated DPFact on one synthetic dataset and two real-world datasets, MIMIC-III [39] and the CMS DE-SynPUF dataset. Each of the dataset has different sizes, sparsity (i.e., % of non-zero elements), and skewness in distribution (i.e., some sites have more patients).

**MIMIC-III.** This is a publicly-available intensive care unit (ICU) dataset collected from 2001 to 2012. We construct 6 local tensors with different sizes representing patients from different ICUs. Each tensor element represents the number of co-occurrence of diagnoses and procedures from the same patient within a 30-day time window. For better interpretability, we adopt the rule in [44] and select 202 procedures ICD-9 codes and 316 diagnoses codes that have the highest frequency. The resulting

tensor is $40,662$ patients $\times$ 202 procedures $\times$ 316 diagnoses with a non-zero ratio of $4.0382 \times 10^{-6}$.

**CMS.** This is a publicly-available Data Entrepreneurs' Synthetic Public Use File (DE-SynPUF) from 2008 to 2010. We randomly choose 5 samples out of the 20 samples of the outpatient data to construct 5 local tensors with patients, procedures and diagnoses. Different from MIMIC-III, we make each local tensor the same size. There are 82,307 patients with 2,532 procedures and 10,983 diagnoses within a 30-day time window. We apply the same rule in selecting ICD-9 codes. By concatenating the 5 local tensors, we obtain a big tensor with $3.1678 \times 10^{-7}$ non-zero ratio.

**Synthetic Dataset.** We also construct tensors from synthetic data. In order to test different dimensions and sparsities, we construct a tensor of size $5000 \times 300 \times 800$ with a sparsity rate of $10^{-5}$ and then horizontally partition it into 5 equal parts.

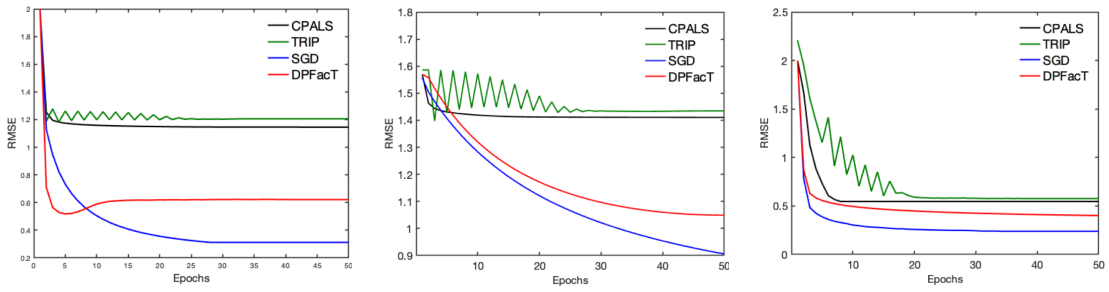
### Baselines

We compare our DPFact framework with two centralized baseline methods and an existing state-of-the-art federated tensor factorization method as described below.

**CP-ALS**: A widely used, centralized model that solves tensor decomposition using an alternating least squares approach. Data from multiple sources are combined to construct the global tensor.

**SGD**: A centralized method that solves the tensor decomposition use the stochastic gradient descent-based approach. This is equivalent to DPFact with a single site and no regularization ($T = 1, \gamma = 0, \mu = 0$). We consider this a counterpart to the CP-ALS method.

**TRIP** [45]: A federated tensor factorization framework that enforces a shared global model and does not offer any differential privacy guarantee. TRIP utilizes the consensus ADMM approach to decompose the problem into local subproblems.




(a) MIMIC-III: ICU (b) CMS (c) Synthetic

Figure 2.2: Average RMSE on (a) MIMIC-III, (b) CMS, (c) Synthetic datasets using 5 random initializations.

### Implementation Details

DPFact is implemented in MatlabR2018b with the Tensor Toolbox Version 2.6 [4] for tensor computing and the Parallel Computing Toolbox of Matlab. The experiments were conducted on m5.4xlarge instances of AWS EC2 with 8 workers. For prediction task, we build the logistic regression model with Scikit-learn library of Python 2.7. For reproducibility purpose, we made our code publicly available[2].

### Parameter Configuration

Hyper-parameter settings include quadratic penalty parameter $\gamma$, $l_{2,1}$ regularization term $\mu$, learning rate $\eta$, and the input per-epoch, per-factor matrix privacy budget $\rho$. The rank $R$ is set to 50 to allow some site-specific phenotypes to be captured.

**Quadratic penalty parameter $\gamma$.** The quadratic penalty term can be viewed as an elastic force between the local factor matrices and the global factor matrices. Smaller $\gamma$ allows more exploration of the local factors but will result in slower convergence. To balance the trade-off between convergence and stability, we choose $\gamma = 5$ after grid search through $\gamma = \{2, 5, 8, 10\}$.

**$l_{2,1}$-regularization term $\mu$.** We evaluate the performance of DPFact with different $\mu$ for different ICU types as they differ in the *Lipschitz* constants. Smaller $\mu$ has minimal

[2]https://github.com/jma78/DPFact.

| # of Sites | MIMIC-III | CMS | Synthetic |
|---|---|---|---|
| 1 | 18.73 | 22.89 | 1.55 |
| 5 | 93.62 | 114.42 | 7.75 |
| 10 | 189.83 | 228.83 | 15.50 |

Table 2.1: Communication cost of DPFact for different number of sites (Seconds)

effect on the column sparsity, as there are no columns that are set to 0, while higher $\mu$ will "turn off" a large portion of the factors and prevent DPFact from generating useful phenotypes. Based on Fig. A.1, we choose $\mu = \{1, 1.8, 3.2, 1.8, 1.5, 0.6\}$ for TSICU, SICU, MICU, CSRU, CCU, NICU respectively for MIMIC-III to maintain noticeable differences in the column magnitude and the flexibility to have at least one unshared column (see Appendix for details). Similarly, we choose $\mu = 2$ equally for each site for CMS and $\mu = 0.5$ equally for each site for the synthetic dataset.

**Learning rate $\eta$.** The learning rate $\eta$ must be the same for local sites and the parameter server. The optimal $\eta$ was found after grid searching in the range $[10^{-5}, 10^{-1}]$. We choose $10^{-2}, 10^{-3}$, and $10^{-2}$ for MIMIC-III, CMS, and synthetic data respectively.

**Privacy budget $\rho$.** We choose the per-epoch privacy budget under the zCDP definition for each factor matrix as $\rho = 10^{-3}$ for MIMIC-III, CMS, and synthetic dataset. By Theorem 4.1, the total privacy guarantee is $(1.2, 10^{-4})$, $(1.9, 10^{-4})$, and $(1.7, 10^{-4})$ under the $(\epsilon, \delta)$-DP definition for MIMIC-III, CMS, and synthetic dataset respectively when DPFact converges (we choose $\delta$ to be $10^{-4}$).

**Number of sites $T$.** To gain more knowledge on how communication cost would be reduced regarding the number of sites, we evaluate the communication cost when the number of sites ($T$) are increased. To simulate a larger number of sites, we randomly partition the global observed tensor into 1, 5, and 10 sites for the three datasets. Table A.2 shows that the communication cost of DPFact scales proportionally with the number of sites.

| Algorithm | MIMIC-III | CMS | Synthetic |
|---|---|---|---|
| TRIP | 175.26 | 183.72 | 9.77 |
| DPFact | 93.62 | 114.42 | 7.75 |

Table 2.2: Communication Cost of DPFact and TRIP (Seconds)

### 2.5.2 Result Analysis

#### Efficiency

**Accuracy.** Accuracy is evaluated using the root mean square error (RMSE) between the global observed tensor and a horizontal concatenation of each factorized local tensor. Fig. 2.2 illustrates the RMSE as a function of the number of epochs. We observe that DPFact converges to a smaller RMSE than CP-ALS and TRIP. SGD achieves the lowest RMSE as DPFact suffers some utility loss by sharing differentially private intermediary results.

**Communication Cost.** The communication cost is measured based on the total number of communicated bytes divided by the data transfer rate (assumed as 15 MB/second). As CP-ALS and SGD are both centralized models, only TRIP and DPFact are compared.

Table 2.2 summarizes the communication cost on all the datasets. DPFact reduces the cost by 46.6%, 37.7%, and 20.7% on MIMIC-III, CMS, and synthetic data, respectively. This is achieved by allowing more local exploration at each site (multiple passes of the data) and transmitting fewer auxiliary variables. Moreover, the reduced communication cost does not result in higher RMSE (see Fig. 2.2).

#### Utility

The utility of DPFact is measured by the predictive power of the discovered phenotypes. A logistic regression model is fit using the patients' membership values (i.e., $\mathbf{A}_{i:}^{[t]}$, $\widehat{\mathbf{A}_{i:}}$ of size $1 \times R$) as features to predict in-hospital mortality. We use a 60-40 train-test

| Rank | CP-ALS | TRIP | DPFact | | |
|---|---|---|---|---|---|
| | | | DPFact | w/o $l_{2,1}$ | w/o DP |
| 10 | 0.7516 | 0.7130 | 0.7319 | 0.5189 | 0.7401 |
| 20 | 0.7573 | 0.7596 | 0.7751 | 0.6886 | 0.7763 |
| 30 | 0.7488 | <u>0.7644</u> | 0.7679 | 0.6977 | 0.7705 |
| 40 | 0.7603 | 0.7574 | 0.7737 | 0.7137 | 0.7756 |
| 50 | 0.7643 | 0.7633 | <u>0.7759</u> | 0.7212 | <u>0.7790</u> |
| 60 | <u>0.7648</u> | 0.7588 | 0.7758 | <u>0.7312</u> | 0.7763 |

Table 2.3: Predictive performance (AUC) comparison for (1) CP-ALS, (2) TRIP, (3) DPFact, (4) DPFact without $l_{2,1}$-norm (w/o $l_{2,1}$), (5) non-private DPFact (w/o DP).

split and evaluated the model using area under the receiver operating characteristic curve (AUC).

**Global Patterns.** Table 2.3 shows the AUC for DPFact, CP-ALS (centralized), and TRIP (distributed) as a function of the rank ($R$). From the results, we observe that DPFact outperforms both baseline methods for achieving the highest AUC. This suggests that DPFact captures similar global phenotypes as the other two methods. We note that DPFact has a slightly lower AUC than CP-ALS for a rank of 10, as the $l_{2,1}$-regularization effect is not prominent.

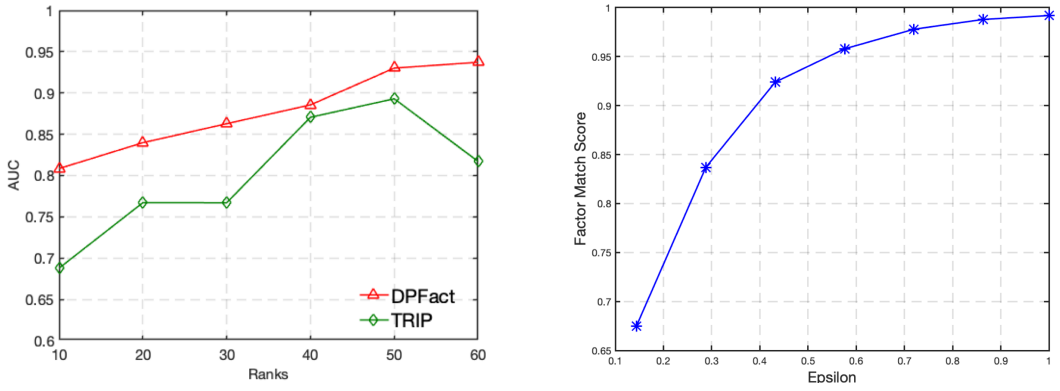
**Site-Specific Patterns.** Besides achieving the highest predictive performance, DPFact also can be used to discover site-specific patterns. As an example, we focus on the neonatal ICU (NICU) which has a drastically different population than the other 5 ICUs. The ability to capture NICU-specific phenotypes can be seen in the AUC comparison with TRIP (Fig. 2.3(a)). DPFact consistently achieves higher AUC for NICU patients. The importance of the $l_{2,1}$-regularization term is also illustrated in Table 2.3. DPFact with the $l_{2,1}$-regularization is more stable and achieves higher AUC compared without the regularization term ($\mu = 0$).

Table 2.4 illustrates the top 5 phenotypes with respect to the magnitude of the logistic regression coefficient (mortality risk related to the phenotype) for NICU. The phenotypes are named according to the non-zero procedures and diagnoses. A high $\lambda$

| Phenotypes | Coef | p-value | $\lambda$ | Prevalence |
|---|---|---|---|---|
| 25: Congenital heart defect | -2.1865 | 0.005 | 198 | 34.32 |
| 29: Anemia | 3.5047 | <0.001 | 77 | 13.22 |
| 30: Acute kidney injury | **5.8806** | <0.001 | 68 | 23.38 |
| 34: Pneumonia | -5.1050 | <0.001 | 37 | 37.58 |
| 35: Respiratory failure | -0.9141 | <0.001 | 85 | 24.40 |

Table 2.4: Top 5 representative phenotypes from NICU based on the factor weights, $\lambda_r = \|\mathbf{A}_{:r}\|_F \|\mathbf{B}_{:r}\|_F \|\mathbf{C}_{:r}\|_F$. Prevalence is the proportion of patients who have non-zero membership to the phenotype.

and prevalence means this phenotype is common. From the results, we observe that heart disease, respiratory failure, and pneumonia are more common but less associated with mortality risk (negative coefficient). However, acute kidney injury (AKI) and anemia are less prevalent and highly associated with death. In particular, AKI has the highest risk of in-hospital death, which is consistent with other reported results [98]. Table 2.5(a) shows an NICU-specific phenotype, which differs slightly from the corresponding global phenotype showing in table 2.5(b).




(a) AUC comparison (b) Factor Match Score (FMS)

Figure 2.3: (a) Predictive performance (AUC) comparison for NICU between (1) TRIP, (2) DPFact. (b) Factor Match Score (FMS) under different privacy budget ($\epsilon$).

### Privacy

We investigated the impact of differential privacy by comparing DPFact with its non-private version. The main difference is that non-private DPFact does not perturb

Table 2.5: Example of the representative phenotypes. (a) NICU-specific phenotype of Congenital heart defect; (b) and (c) are the globally shared phenotype of Heart failure, showing the difference of DPFact and non-private DPFact.

(a) NICU-specific Phenotypes discovered by DPFact

| **Procedures** | **Diagnoses** |
|---|---|
| Cardiac catheterization | Ventricular fibrillation |
| Insertion of non-drug-eluting coronary artery stent(s) | Unspecified congenital anomaly of heart |
| Prophylactic administration of vaccine against other disease | Benign essential hypertension |

(b) Globally shared phenotype discovered by DPFact

| **Procedures** | **Diagnoses** |
|---|---|
| Attachment of pedicle or flap graft | Rheumatic heart failure |
| Right heart cardiac catheterization | Ventricular fibrillation |
| Procedure on two vessels | Benign essential hypertension |
| Other endovascular procedures on other vessels | Paroxysmal ventricular tachycardia |
| Insertion of non-drug-eluting coronary artery stent(s) | Nephritis and nephropathy |

(c) Globally shared phenotype discovered by non-private DPFact

| **Procedures** | **Diagnoses** |
|---|---|
| Right heart cardiac catheterization | Hypopotassemia |
| Attachment of pedicle or flap graft | Rheumatic heart failure |
| Excision or destruction of other lesion or tissue | Benign essential hypertension |
| of heart, open approach | Paroxysmal ventricular tachycardia |
| | Systolic heart failure |

the local feature factor matrices that are transferred to the server. We use the factor match score (FMS) [15] to compare the similarity between the phenotype discovered using DPFact and non-private DPFact. FMS defined as:

$$score(\bar{\mathcal{X}}) = \frac{1}{R}\sum_{r}\left(1 - \frac{\left|\xi_r - \bar{\xi_r}\right|}{max\left\{\xi_r, \bar{\xi_r}\right\}}\right)\prod_{\mathbf{x=a,b,c}}\frac{\mathbf{x}_r^T\bar{\mathbf{x}}_r}{\|\mathbf{x}_r\|\|\bar{\mathbf{x}}_r\|},$$

$$\xi_r = \prod_{\mathbf{x=a,b,c}}\|\mathbf{x}_r\|, \bar{\xi_r} = \prod_{\mathbf{x=a,b,c}}\|\bar{\mathbf{x}}_r\|$$

where $\bar{\mathcal{X}} = [\![\bar{\mathbf{A}}, \bar{\mathbf{B}}, \bar{\mathbf{C}}]\!]$ is the estimated factors and $\mathcal{X} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$ is the true factors. $\mathbf{x}_r$ is the $r^{th}$ column of factor matrices.

We treat the non-private version DPFact factors as the benchmark for DPFact factors. Fig. 2.3(b) shows how the FMS changes with an increase of the privacy budget. As the privacy budget becomes larger, the FMS increases accordingly and will gradually approximate 1, which means the discovered phenotypes between the two methods are equivalent. This result indicates that when a stricter privacy constraint is enforced, it may negatively impact the quality of the phenotypes. Thus, there is a practical need to balance the trade-off between privacy and phenotype quality.

Table 2.5 presents a comparison between the top 1 (highest factor weight $\lambda_r$) phenotype DPFact-derived phenotype and the closest phenotype derived by its non-private version. We observe that DPFact contains several additional noisy procedure and diagnosis elements than the non-private DPFact version. These extra elements are the results of adding noise to the feature factor matrices. This is also supported in Table 2.3 as the non-private DPFact has better predictive performance than DPFact. Thus, the output perturbation process may interfere with the interpretability and meaningfulness of the derived phenotype. However, there is still some utility from the DPFact-derived phenotype as experts can still distinguish this phenotype to be a heart failure phenotype. Therefore, DPFact still retains the ability to perform phenotype discovery.

# Chapter 3

# Federated Generalized Tensor Factorization with Improved Communication Efficiency[1]

In this chapter, we further increase the communication efficiency compared with DPFact (Chapter 2) and support the flexibility of choosing different loss functions according to data distributions. In addition, we also incorporate computational efficient techniques and provide theoretical guarantees for convergence.

## 3.1 Introduction

Tensor factorization incurs high computational complexity and storage complexity due to the computation of the MTTKRP operation and the formation of the matricized tensor. Besides computational complexity and alleviating storage usage which are the focus of most existing DTF methods, the communication overhead can be a third

[1]This chapter is based on earlier work [64] published in Proceedings of the Web Conference 2021 (WWW '21). Association for Computing Machinery, New York, NY, USA, 171–182. DOI:https://doi.org/10.1145/3442381.3449832 

important bottleneck, especially for the federated setting, where the participating institutions do not have a dedicated communication network for communication purposes, e.g., hospitals, clinics. Considering the asymmetric bandwidths, the uplink communication (i.e. the communication from the client to the server) can quickly become the bottleneck preventing these clients from participating in the FTF. In federated computational phenotyping, due to the great variety of the attributes (e.g., types of medication can be thousands), the high dimensional tensor incurs high communications cost to communicate the intermediate variables during each communication cycle.

In this chapter, we investigate how to reduce the uplink communication cost of the federated tensor factorization-based collaborative phenotyping with guaranteed convergence and quality preservation. It is a challenging task, especially considering the communication efficiency issue is under studied in the broader distributed tensor factorization literature. To be more flexible and suitable for a variety of applications, we consider the federated generalized tensor factorization (FGTF), which greatly extends the existing federated classic TF [45, 62].

First, we aim to reduce the uplink communication cost in each communication round. We design a two-level per-round communication reduction strategy: block-level and element-level, which reduce $(1 - \frac{1}{D})$ and over 96.8% of the uplink communication, correspondingly, where $D$ is the number of blocks. For the block-level, we exploit the multi-factor structure of TF/GTF by utilizing the randomized block update. It enables each client to send only the partial gradient of the sampled block, rather than the full gradient of all blocks. For the element-level, we introduce gradient compression techniques, which have found success in deep learning training [43, 103, 5, 92, 3], to compress each element of the communicated partial gradient from the floating point representation to low-precision representation. Since there exists error between the true partial gradient and the compressed one, the convergence can be slower and the

output quality can be lower. We further introduce the error-feedback mechanism [43] which records such error and feeds it back to restore the shift.

With both levels of per-round communication reduction, we propose the federated GTF with communication compression and error-feedback **(FedGTF-EF)**. We analyze the convergence of **FedGTF-EF** and obtain the $O(\frac{1}{\sqrt{T}})$ rate after $T$ iterations (Thm.3.4.1) under common and mild assumptions (Assumptions 3.4.1–3.4.5). The convergence is equivalent to the distributed stochastic gradient descent (SGD) with full precision gradient communication and distributed SGD with gradient compression and error-feedback [103]. In addition, since constraints and nonsmooth regularizations are common in GTF, we further extend the convergence result to the proximal setting (3.4.2) where the additional "simple regularizer" in Assumption 3.4.6 is satisfied. Compared to the existing analysis with gradient compression and error-feedback, our convergence analysis accounts for both the block randomized update strategy and the proximal operation.

Second, we reduce the number of communication rounds to further reduce the uplink communication. To do so, we introduce periodic communication [80, 54, 5] into **FedGTF-EF** and denote this algorithm as **FedGTF-EF-PC**, in which the clients send the update to the server after $\tau > 1$ local iterations instead of communicating after every iteration. A key question is whether the periodic communication will slow down the convergence. If so, the number of iterations will increase and the overall number of communications may not reduce. We analyze the convergence of **FedGTF-EF-PC** in Thm. 3.4.3 and obtain the same convergence $O(\frac{1}{\sqrt{T}})$ rate with **FedGTF-EF** under the same set of assumptions. This indicates that **FedGTF-EF-PC** can indeed further reduce the uplink communication cost by $1 - \frac{1}{\tau}$. As a result, our proposed **FedGTF-EF-PC** can reduce up to $1 - \frac{1}{32D\tau}$ uplink communication cost if the `Sign` compressor (Def.3.2.1) is used.

Third, we evaluate **FedGTF-EF** and **FedGTF-EF-PC** in the federated collabo-

rative phenotyping task. We conduct experiments on two real-world EHR datasets, which show that the proposed method can effectively reduce uplink communication cost (99.90% reduction), without compromising convergence and factorization quality.

## 3.2 Preliminaries and Background

### 3.2.1 Notation

The frequently used notation in this chapter is summarized in Table 4.1. We denote an order $D$ tensor by $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times ... \times I_D}$, its $(i_1, ..., i_D)$-th element by MATLAB representation $\boldsymbol{\mathcal{X}}(i_1, ..., i_D)$. Let $\mathcal{I}$ denote the index set of all tensor entries, $|\mathcal{I}| = I_\Pi = \prod_{d=1}^{D} I_d$. The mode-$d$ unfolding (also called matricization) is denoted by $\mathbf{X}_{<d>} \in \mathbb{R}^{I_d \times I_\Pi / I_d}$, where $(\mathbf{X}_{<d>})(i_d, j)$ and $\boldsymbol{\mathcal{X}}(i_1, i_2, ..., i_D)$ has the **index mapping:** $j = 1 + \sum_{\substack{k=1, \\ k \neq d}}^{D} (i_k - 1) J_k$, $J_k = \prod_{\substack{q=1, \\ q \neq d}}^{k-1} I_q$. Each column $\mathbf{X}_{<d>}(:, j)$ is called a mode-$d$ fiber of $\boldsymbol{\mathcal{X}}$.

### 3.2.2 Generalized Tensor Factorization

As illustrated in Fig. 1.1 (left), let us consider the EHR tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times, ..., \times I_D}$, which consists of patient mode $(I_1)$, diagnosis mode $(I_2)$, medication mode $(I_3)$, and so on. The regularized Generalized CANDECOMP-PARAFAC (GTF) [37] extracts the phenotypes by decomposing the EHR tensor into $R$ phenotyps, where each consists of a patient factor, diagnosis factor, and a medication factor. GTF has the following objective function:

$$\begin{aligned} \arg\min_{\boldsymbol{\mathcal{A}}} F(\boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{X}}) &= \sum_{i \in \mathcal{I}} f(\boldsymbol{\mathcal{A}}(i), \boldsymbol{\mathcal{X}}(i)) + \sum_{d=1}^{D} r_d(\mathbf{A}_{(d)}), \\ s.t.\ \boldsymbol{\mathcal{A}} &= \sum_{i=1}^{R} \mathbf{A}_{(1)}(:, i) \circ ... \circ \mathbf{A}_{(D)}(:, i), \end{aligned} \tag{3.1}$$

which breaks down into three parts:

1. Factorization constraint: The constraint of $\boldsymbol{\mathcal{A}} = \sum_{i=1}^{R} \mathbf{A}_{(1)}(:, i) \circ ... \circ \mathbf{A}_{(D)}(:, i)$ approximates the low-rank CP tensor $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{I_1 \times, ..., \times I_D}$ as the sum of $R$ rank-one tensors, where $\mathbf{A}_{(d)} \in \mathbb{R}^{I_d \times R}$ is the $d$-th factor matrix and $\mathbf{A}_{(d)}(:, i)$ is its $i$-th column. For phenotyping, $\mathbf{A}_{(1)}, \mathbf{A}_{(2)}, \mathbf{A}_{(3)}$ correspond to the patient factor, diagnosis factor, and medication factor, correspondingly.

2. Element-wise loss function: $f(\boldsymbol{\mathcal{A}}(i), \boldsymbol{\mathcal{X}}(i))$ is the element-wise loss between the low-rank CP tensor $\boldsymbol{\mathcal{A}}$ and the input tensor $\boldsymbol{\mathcal{X}}$. For the classic CP [11, 31], $f(\boldsymbol{\mathcal{A}}(i), \boldsymbol{\mathcal{X}}(i)) := \frac{1}{2}(\boldsymbol{\mathcal{A}}(i) - \boldsymbol{\mathcal{X}}(i))^2$, which is the least square loss. GCP is more generalized in the sense that the loss function can take other forms to best suit the property of the input tensor. For example, $f(\cdot)$ can be chosen based on the distribution of the tensor entries, e.g. logit loss for binary data: $f_{\text{logit}} = \log(1 + \boldsymbol{\mathcal{A}}(i)) - \boldsymbol{\mathcal{X}}(i)\boldsymbol{\mathcal{A}}(i)$, for all $i \in \mathcal{I}$, or $f(\cdot)$ can be the Huber loss for robustness purpose.

3. Regularization: $r_d(\cdot)$ is the regularization applied to the factor $\mathbf{A}_d$, which can be the smooth $\|\mathbf{A}_{(d)}\|_F^2$ norm or the nonsmooth $\|\mathbf{A}_{(d)}\|_1$ norm. In practice, the regularization can improve the interpretability of the phenotypes.

**Existing federated computational phenotyping.** Two recent papers [45] and [62] consider Federated Tensor Factorization and apply it to the federated phenotyping. They have the following limitations. 1) Both are limited to the CP model and [45] applies least square solver as its client side local updater, which is difficult to be extended to more general losses other than least square loss. 2) Although extensible to using infrequent communication, each communication round still incurs high communication cost since both requires sending all factors in full precision. In addition, [45] also requires communication of the Lagrangian dual variables which doubles the communication cost. 3) Both alter the original objective function by introducing extra terms to enforcing consensus of factors among all clients: [45]

introduces linear constraint and transforms it to Lagrangian dual formulation while [62] introduces elastic penalty terms. These terms can lead the extracted factors to deviate from the centralized solution, thus negatively impacting the phenotyping accuracy.

### 3.2.3 SGD with Gradient Compression, Error-Feedback and Periodic Communication

**Gradient Compression.** Recently, one of the most successful approaches to mitigating the communication overhead is via gradient compression, which compresses the gradient to be communicated from the full precision representation (e.g. float or double number representation) to a much lower precision representation (e.g. aggressively compressed to 1-bit). The following definition introduces one of the most popular compressors:

**Definition 3.2.1.** (Sign Compressor) For an input tensor $\mathbf{x} \in \mathbb{R}^d$, its compression via $\texttt{Sign}(\cdot)$ is $\texttt{Sign}(\mathbf{x}) = \|\mathbf{x}\|_1/d \cdot sign(\mathbf{x})$, where $sign$ takes the sign of each element of $\mathbf{x}$.

**Error-Feedback.** Due to aggressive compression, the algorithm can converge slower (or even diverge) compared to the full precision counterpart. The main cause is the error between the full precision gradient and the compressed one. Error-feedback [43, 103, 81] is a technique that memorizes this error in the current iteration and feeds it back to the gradient of the next iteration. By doing so, it can rigorously guarantee uncompromised convergence compared to the full-precision SGD.

**Periodic Communication.** Instead of reducing the communication cost per-communication round, periodic communication or local SGD [80, 54] reduces it by decreasing the communication frequency in hope that the total number of communications rounds can be reduced. Each clients will execute $\tau > 1$ local updates before communicating to the server. [5] shows that it is possible to combine communication compression and

periodic communication together. [81] provides a unified framework by error-feedback to analyze the convergence of gradient compression and local SGD.

## 3.3 Proposed Methods

Under the federated setting as illustrated in Fig. 1.1 (right), the EHR tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times,\ldots,\times I_D}$ will be collectively held by $K$ institutions. The $k$-th client's local tensor is denoted by $\boldsymbol{\mathcal{X}}^k \in \mathbb{R}^{I_{1^k} \times I_2 \times \ldots \times I_D}$, which contains information about $I_{1^k}$ individuals, such that $\sum_{k=1}^{K} I_{1^k} = I_1$. That is, we consider the horizontally partitioned setting where different hospitals share the same feature space. We also note that there are related works addressing other settings like vertically partitioned settings [95, 59, 13, 58] which are complementary to our work. The aim of the federated computational phenotying is to collaboratively compute the phenotyes from EHR tensor across $K$ institutions without sharing the raw tensor and patient mode variables. The objective function of the federated GTF is as follows

$$\begin{aligned} \underset{(\mathbf{A}_{(1)},\ldots,\mathbf{A}_{(D)})}{\operatorname{argmin}} & \sum_{k=1}^{K} F(\boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{X}}^k) + \sum_{d=1}^{D} r_d(\mathbf{A}_{(d)}), \\ & s.t.\ \boldsymbol{\mathcal{A}} = \mathbf{A}_{(1)} \circ \ldots \circ \mathbf{A}_{(D)}. \end{aligned} \tag{3.2}$$

In fact, the above formulation can be extended to general multi-block problems as well. Thus, our algorithms are not limited to federated GTF problems but also to other nonconvex problems possessing a multi-block decision variable structure, e.g. [100]. In the following, we propose the federated generalized tensor factorization with communication efficiency improvements via block randomization, gradient compression, error feedback and periodic communication. The execution of the proposed algorithm is illustrated in Fig. 3.1.

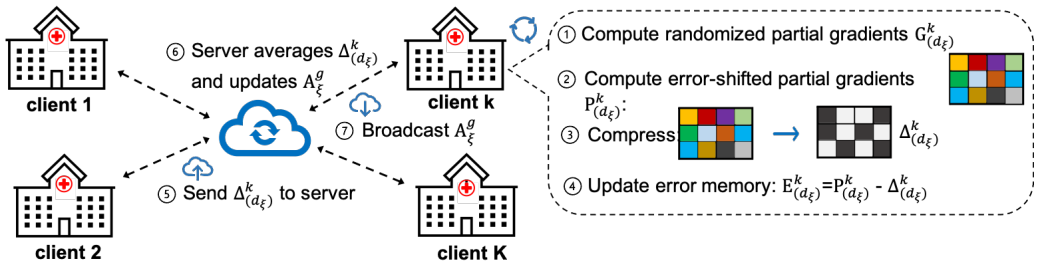



Figure 3.1: Illustration of the execution of FedGTF-EF and FedGTF-EF-PC.

### 3.3.1 FedGTF-EF: Communication Efficient GTF with Block Randomization, Gradient Compression and Error-Feedback

**Algorithm 2:** FedGTF-EF: Communication Efficient GTF with Block Randomization, Gradient Compression and Error-Feedback

**Input:** $\mathcal{X}, \gamma[t], \mathbf{A}[0]$, randomized block sampling sequence $d_\xi[0], ..., d_\xi[T]$;

**for** $t = 0, ..., T$ **do**
  **On Each Client Nodes** $k \in 1, ..., K$**:**
  **if** $d = d_{(\xi)}[t]$ **then**
    Compute stochastic gradient $\mathbf{G}^k_{(d)}[t]$ by eq.(4.6);
    $\mathbf{P}^k_{(d)}[t] = \gamma[t]\mathbf{G}^k_{(d)}[t] + \mathbf{E}^k_{(d)}[t]$; %% error feedback
    $\mathbf{\Delta}^k_{(d)}[t] = \texttt{Compress}(\mathbf{P}^k_{(d)}[t])$, Send $\mathbf{\Delta}^k_{(d)}[t]$ (i.e. $\mathbf{\Delta}^k_{(d_\xi[t])}[t]$) to the server; %% compression
    Receive $\frac{1}{K}\sum_{k=1}^K \mathbf{\Delta}^k_{(d)}[t]$ (i.e. $\frac{1}{K}\sum_{k=1}^K \mathbf{\Delta}^k_{(d_\xi[t])}[t]$) from the server;
    Smooth regularization case: $\mathbf{A}_{(d)}[t+1] = \mathbf{A}_{(d)}[t] - \frac{1}{K}\sum_{k=1}^K \mathbf{\Delta}^k_{(d)}[t]$; %% update factor
    Nonsmooth regularization case: $\mathbf{A}_{(d)}[t+1] = \texttt{Prox}_{r_d}(\mathbf{A}_{(d)}[t] - \frac{1}{K}\sum_{k=1}^K \mathbf{\Delta}^k_{(d)}[t])$;
    $\mathbf{E}^k_{(d)}[t+1] = \mathbf{P}^k_{(d)}[t] - \mathbf{\Delta}^k_{(d)}[t]$; %% update error memory
  **else if** $d \neq d_\xi[t]$ **then**
    $\mathbf{A}_{(d)}[t+1] = \mathbf{A}_{(d)}[t]$, $\mathbf{E}^k_{(d)}[t+1] = \mathbf{E}^k_{(d)}[t]$; %% unselected blocks are kept unchanged
  **end if**
  **On Server Node:**
  Receive $\mathbf{\Delta}^k_{(d_\xi[t])}[t]$ from all client nodes; Broadcast $\frac{1}{K}\sum_{k=1}^K \mathbf{\Delta}^k_{(d_\xi[t])}[t]$ to all client nodes;
**end for**

We reduce the uplink communication in each communication round at two levels: block-level and element-level. The detailed algorithm is displayed in Algorithm 2 with functionalities of key steps annotated. At the block-level, to avoid sending all factors,

we use a randomized block (i.e., randomized factor) update, which only requires the communication of the partial gradient of the factor being sampled (the computation of the partial gradient will be detailed in Sec.3.3.3). At the element-level, we compress each element of the communication to a low-precision representation before sending to the server (Line 6). Each client $k$ keeps $D$ local pairs of $\mathbf{P}_{(d)}^k$ (the error-shifted full-precision partial gradient), $\Delta_{(d)}^k$ (the compressed gradient to be communicated), $\mathbf{E}_{(d)}^k$ (error record between the full precision gradient and the compressed gradient), for all $d = 1, ..., D$ factors. Depending on whether the regularizer is smooth or not, either simple gradient descent (Line 8) or proximal gradient descent (Line 9) can be chosen to update the sampled factor, respectively.

**Algorithm 3:** FedGTF-EF-PC: Further Reducing Communication Cost by Periodic Communication

**Input:** $\boldsymbol{\mathcal{X}}, \gamma[t], \mathbf{A}[0], \mathbf{A}^k[0] = \mathbf{A}[0], \forall k = 1, ..., K$, randomized block sampling sequence $d_\xi[0], ..., d_\xi[T]$;

1: **for** $t = 0, ..., T$ **do**
2: **On Each Client Nodes** $k \in 1, ..., K$**:**
3: **if** $d = d_{(\xi)}[t]$ **then**
4: Compute stochastic gradient $\mathbf{G}_{(d)}^k[t]$ by eq.(4.6);
5: $\mathbf{A}_{(d)}^k[t+\frac{1}{2}] = \mathbf{A}_{(d)}^k[t] - \gamma[t]\mathbf{G}_{(d)}^k[t]$; %% local update by stochastic gradient descent
6: **if** $(t \mod \tau) \neq 0$ **then**
7: $\mathbf{E}_{(d)}^k[t+1] = \mathbf{E}_{(d)}^k[t]$, $\mathbf{A}_{(d)}^k[t+1] = \mathbf{A}_{(d)}^k[t+\frac{1}{2}]$, $\mathbf{A}_{(d)}^g[t+1] = \mathbf{A}_{(d)}^g[t]$; %% no communication
8: **else**
9: $\mathbf{P}_{(d)}^k[t] = (\mathbf{A}_{(d)}^g[t] - \mathbf{A}_{(d)}^k[t+\frac{1}{2}]) + \mathbf{E}_{(d)}^k[t]$; %% error feedback to accumulated update
10: $\boldsymbol{\Delta}_{(d)}^k[t] = \texttt{Compress}(\mathbf{P}_{(d)}^k[t])$, Send $\boldsymbol{\Delta}_{(d)}^k[t]$ (i.e. $\boldsymbol{\Delta}_{(d_\xi[t])}^k[t]$) to the server;
11: Receive $\mathbf{A}_{(d)}^g[t+1]$ from the server, $\mathbf{A}_{(d)}^k[t+1] = \mathbf{A}_{(d)}^g[t+1]$; %% compression
12: **end if**
13: $\mathbf{E}_{(d)}^k[t+1] = \mathbf{P}_{(d)}^k[t] - \boldsymbol{\Delta}_{(d)}^k[t]$; %% update error memory
14: **else if** $d \neq d_\xi[t]$ **then**
15: $\mathbf{A}_{(d)}^k[t+1] = \mathbf{A}_{(d)}^k[t]$, $\mathbf{E}_{(d)}^k[t+1] = \mathbf{E}_{(d)}^k[t]$;
16: **end if**
17: **On Server Node:**
18: Receive $\boldsymbol{\Delta}_{(d_\xi[t])}^k[t]$ from all client nodes; Broadcast $\mathbf{A}_{(d_\xi[t])}^g[t+1] = \mathbf{A}_{(d_\xi[t])}^g[t] - \frac{1}{K}\sum_{k=1}^K \boldsymbol{\Delta}_{(d_\xi[t])}^k[t]$ to all client nodes;
19: **end for**

### 3.3.2 FedGTF-EF-PC: Further Communication Reduction by Periodic Communication

We further reduce the uplink communication cost by introducing a third communication compression level: round level. That is, we decrease the communication frequency from one iteration per-communication to $\tau > 1$ iterations per-communication, which manifests a periodic communication behaviour [80, 54, 5]. The detailed algorithm is provided in Algorithm 3. The major difference with Algorithm 2 is that each client compresses and sends the collective updates across $\tau$ iterations (Line 9-10), instead of the partial gradient in a single iteration. The error feedback (Line 9) and error memory (Line 7, 13) are adjusted accordingly.

### 3.3.3 Efficient Partial Stochastic Gradient Computation for FedGTF

After presenting the overall algorithms, we now present an efficient partial stochastic gradient computation subroutine to compute $\mathbf{G}_{(d)}^{k}[t]$ in Step 1 of Fig. 3.1 and Line 4 of Algorithm 2 and 3. The first mode (i.e., $I_1$) is the individual mode (e.g., patient mode) which can be kept local to each client. Thus, when $d_\xi[t] = 1$, we skip the communication, which not only further reduces the communication cost, but also is beneficial to the privacy since the individual-level information is not shared.

Next, we specify the computation of the partial stochastic gradient $\mathbf{G}_{(d)}^{k}[t]$ based on the efficient fiber sampling technique [6, 25]. The deterministic partial gradient is $\nabla_{\mathbf{A}_{(d)}} F(\mathbf{A}) = \mathbf{Y}_{<d>}\mathbf{H}_d$ [37], where $\mathbf{H}_d \in \mathbb{R}^{I_\Pi/I_d \times R}$ is the mode-$d$ Khatri-Rao product of the all factors except the $d$-th, i.e. $\mathbf{H}_d = \mathbf{A}_{(D)} \odot ... \odot \mathbf{A}_{(d+1)} \odot \mathbf{A}_{(d-1)} ... \odot \mathbf{A}_{(1)}$; and $\mathbf{Y}_{<d>}$ is the $d$-unfolding of the element-wise partial gradient $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{I_1 \times ... \times I_D}$, where $\boldsymbol{\mathcal{Y}}(i) = \frac{\partial f(\boldsymbol{\mathcal{A}}(i), \boldsymbol{\mathcal{X}}(i))}{\partial \boldsymbol{\mathcal{A}}(i)}$, for all $i \in \mathcal{I}$. We approximate $\nabla_{\mathbf{A}_{(d)}} F(\mathbf{A})$ by sampling $|\mathcal{S}|$ fibers (i.e. $|\mathcal{S}|$ columns of $\mathbf{Y}_{(d)}$) and the corresponding $|\mathcal{S}|$ rows of $\mathbf{H}_d$, where $\mathcal{S}$ denotes the

index of the sampled fibers. The stochastic partial gradient is then

$$\mathbf{G}_{(d)}[t] = \mathbf{Y}_{<d>}(:, \mathcal{S})\mathbf{H}_d(\mathcal{S}, :), \tag{3.3}$$

where both $\mathbf{Y}_{<d>}(:, \mathcal{S})$ and $\mathbf{H}_d(\mathcal{S}, :)$ can be efficiently computed, because: 1) the computation of $\mathbf{Y}_{<d>}(:, \mathcal{S})$ only involves $I_d \times |\mathcal{S}|$ element-wise partial gradient computation [47] and 2) the computation of $\mathbf{H}_d(\mathcal{S}, :)$ can be obtained without forming the full Khatri-Rao product of $\mathbf{H}_d$ [77]. For the $s$-th row of $\mathbf{H}_d$, its index $(i_1^s, ..., i_D^s)$ can be obtained by the **index mapping** in Section 3.2.1. Then, $\mathbf{H}(s, :) = \mathbf{A}_{(1)}(i_1^s, :) \circledast ... \circledast \mathbf{A}_{(d-1)}(i_{d-1}^s, :) \circledast \mathbf{A}_{(d+1)}(i_{d+1}^s, :) \circledast ... \circledast \mathbf{A}_{(D)}(i_D^s, :)$, where $\circledast$ is the Hadamard product. Finally, the local stochastic gradient $\mathbf{G}_{(d)}^k[t]$ can be efficiently computed by substituting its local tensor partition $\mathbf{Y}^k$ and local factors $\mathbf{A}_{(d)}^k$ into eq.(3.3), which gives

$$\mathbf{G}_{(d)}^k[t] = \mathbf{Y}_{<d>}^k(:, \mathcal{S})\mathbf{H}_d^k(\mathcal{S}, :), \tag{3.4}$$

where $\mathbf{H}^k(s, :) = \mathbf{A}_{(1)}^k(i_1^s, :) \circledast ... \circledast \mathbf{A}_{(d-1)}^k(i_{d-1}^s, :) \circledast \mathbf{A}_{(d+1)}^k(i_{d+1}^s, :) \circledast ... \circledast \mathbf{A}_{(D)}^k(i_D^s, :)$. According to the complexity analysis, our gradient computation in eq.(4.6) matches the state-of-the-art efficiency of GTF computation, e.g., [25].

## 3.4 Algorithm Analysis

This section presents the convergence analysis and complexity analysis of FedGTF-EF and FedGTF-EF-PC. A proof sketch of the convergence analysis is provided in the Appendix.

### 3.4.1 Convergence Analysis

**Assumptions.** In order to analyze the convergence, we make the following assumptions which are common to many machine learning problems [5, 25, 103, 81].

Let the randomness of computing stochastic gradient of $\mathbf{G}_{(d_{\xi[t]})}[t]$ be $\zeta[t]$, the randomness of sampling the block be $\xi[t]$, the filtration upon iteration $t$ be $\mathcal{F}[t] = \{\zeta[0], \xi[0], ..., \zeta[t-1], \xi[t-1]\}$.

**Assumption 3.4.1.** *(Block-wise Smoothness of the Loss Function) $F(\cdot)$ is $L_{(d)}$-block-wise smooth, for $d = 1, ..., D$, i.e. for all $\mathbf{A}, \mathbf{B}$, $F(\mathbf{B}) \leq F(\mathbf{A}) + \langle \nabla_{\mathbf{A}_{(d)}}, \mathbf{B}_{(d)} - \mathbf{A}_{(d)} \rangle + \frac{L_{(d)}}{2} \|\mathbf{B}_{(d)} - \mathbf{A}_{(d)}\|_F^2$.*

**Assumption 3.4.2.** *(Unbiased Gradient Estimation) The stochastic gradient is unbiased: $\mathbb{E}_{\zeta[t]}\Big[\mathbf{G}^k_{d_{\xi[t]}}[t]\Big|\mathcal{F}[t], \xi[t]\Big] = \nabla_{\mathbf{A}_{(d_{\xi[t]})}} F(\mathbf{A}[t])$.*

**Assumption 3.4.3.** *(Bounded Variance) The stochastic gradient has bounded variance:*
$\mathbb{E}_{\zeta[t]}\Big[\big\|\mathbf{G}^k_{(d_{\xi[t]})}[t] - \nabla_{\mathbf{A}_{(d_{\xi[t]})}} F(\mathbf{A}[t])\big\|_F^2 \Big| \mathcal{F}[t], \xi[t]\Big] \leq \sigma_d^2$.

**Assumption 3.4.4.** *(Bounded Gradient) $\|\nabla_{\mathbf{A}_{(d)}} F(\mathbf{A}[t])\|_F^2 \leq \omega_d^2$.*

**Assumption 3.4.5.** *($\delta$-approximated Compression [43]) An operator* `Compress` *: $\mathbb{R}^d \to \mathbb{R}^d$ is an $\delta$-approximate compressor for $\delta \in (0, 1]$ if $\|\texttt{Compress}(\mathbf{x}) - \mathbf{x}\|_2^2 \leq (1-\delta)\|\mathbf{x}\|_2^2$.*

Many compressors satisfy the above condition [5]: top-k or random k-sparsification, stochastic k-level quantization, stochastic rotated quantization and the `Sign` compressor in Definition 2.3.

**Assumption 3.4.6.** *(Simple Regularization Function) The regularization functions $r_d(\cdot)$, $d = 1, ..., D$, are convex, lower semi-continuous and admit closed-form proximal operator:*
$\texttt{Prox}_{r_d}(\mathbf{B}_d) = \text{argmin}_{\mathbf{A}_{(d)}} \frac{1}{2}\|\mathbf{A}_{(d)} - \mathbf{B}_{(d)}\|_F^2 + r_d(\mathbf{A}_{(d)})$.

Many common regularizations satisfy this assumption, for example, the $\ell_1$-norm for inducing sparsity which has the soft-thresholding operator as its proximal operator.

**Convergence Analysis of Algorithm 2**

**Smooth regularization case.** To prove the convergence, we extend the delayed gradient perspective in [43] to our block randomized setting by introducing the following virtual variables only for the proof: $\widetilde{\mathbf{A}}_{(d)}[t] := \mathbf{A}_{(d)}[t] - \frac{1}{K}\sum_{k=1}^{K}\mathbf{E}_{(d)}^{k}[t]$. Then, we have the following virtual recurrence: if $d = d_{\xi}[t]$, $\widetilde{\mathbf{A}}_{(d)}[t+1] = \mathbf{A}_{(d)}[t+1] - \frac{1}{K}\sum_{k=1}^{K}\mathbf{E}_{(d)}[t+1] = \widetilde{\mathbf{A}}_{(d)}[t] - \gamma[t]\frac{1}{K}\sum_{k=1}^{K}\mathbf{G}_{(d)}^{k}[t]$; else if $d \neq d_{\xi[t]}$, $\widetilde{\mathbf{A}}_{(d)}[t+1] = \widetilde{\mathbf{A}}_{(d)}[t]$. Thus, the recurrence can be viewed as the block randomized SGD with the variable $\widetilde{\mathbf{A}}_{(d)}[t]$ which corresponds to $\mathbf{A}_{(d)}[t]$ with delayed information $\frac{1}{K}\sum_{k=1}^{K}\mathbf{E}_{(d)}^{k}[t]$ added. The convergence of Algorithm 2 applied to the smooth smooth regularization is as follows.

**Theorem 3.4.1.** *Suppose that Assumptions 3.4.1-3.4.5 hold. Let* $(\mathbf{A}_{(1)}[t], ..., \mathbf{A}_{(D)}[t])$ *be the iterates of Algorithm 2 with Line 8. Let* $\gamma = \min\{\frac{1}{2L}, \frac{\varrho}{\sqrt{T+1}/\sqrt{K} + \frac{(1-\delta)^{1/3}}{\delta^{2/3}}T^{1/3}}\}$*, for some* $\varrho > 0$*. We have*

$$\begin{aligned}
&\mathbb{E}[\frac{1}{D}\sum_{d=1}^{D}\|\nabla_{\mathbf{A}_{(d)}}F(\mathbf{A}[\texttt{Output}])\|_F^2] \\
&\leq \frac{8L}{T+1}(F(\mathbf{A}[0]) - F^*) + \Big[\frac{4}{\varrho}(F(\mathbf{A}[0]) - F^*) + \frac{2L\sigma^2\varrho}{D}\Big]\frac{1}{\sqrt{M(T+1)}} \\
&+ \Big[\frac{4}{\varrho}(F(\mathbf{A}[0]) - F^*) + \frac{8L^2\varrho^2(\sigma^2+\omega^2)}{D}\Big]\frac{(1-\delta)^{1/3}}{\delta^{2/3}(T+1)^{2/3}},
\end{aligned}$$

*where* $\mathbf{A}[\texttt{Output}] = (\mathbf{A}_{(1)}[\texttt{Output}], ..., \mathbf{A}_{(D)}[\texttt{Output}])$ *is sampled from* $\mathbf{A}[0]$ *to* $\mathbf{A}[T]$ *with uniform distribution,* $F^*$ *is the optimal value,* $\sigma^2 = \sum_{d=1}^{D}\sigma_d^2$ *and* $\omega^2 = \sum_{d=1}^{D}\omega_d^2$*.*

**Remark 1.** Under the similar assumptions, our convergence rate matches the rates of the distributed synchronize SGD and the distributed SGD with gradient compression and error-feedback [103]. Thus, we can further reduce computation and uplink communication from a full-length gradient update and communication [103, 5] to a single randomized block of the partial gradient update and communication without

slowing down the convergence rate.

**Nonsmooth regularization case.** This case corresponds to the execution of Line 9 in Algorithm 2. An appropriate optimally condition is based on the generalized gradient measure [70, 94, 71, 25]: $\widetilde{\mathbf{G}}_{(d)}[t] = \frac{1}{\gamma[t]}(\mathbf{A}_{(d)} - \texttt{Prox}_{\gamma[t],r_d}(\boldsymbol{\mathcal{A}}_{(d)}[t] - \gamma[t]\nabla_{\mathbf{A}_{(d)}} F(\mathbf{A}[t])))$. The following theorem shows the convergence of Algorithm 2 for the nonsmooth regularization case.

**Theorem 3.4.2.** *Suppose that Assumptions 3.4.1-3.4.6 hold.*
*Let* $(\mathbf{A}_{(1)}[t], ..., \mathbf{A}_{(D)}[t])$ *be the iterates of Algorithm 2 with proximal operator (Line 9). Assume* $\gamma[t] = \frac{1}{4L}$. *We have*

$$\begin{aligned}\mathbb{E}\Big[\sum_{d=1}^{D}\frac{1}{D}\|\widetilde{\mathbf{G}}_{(d)}[\texttt{Output}]\|_F^2\Big] \leq \frac{16L}{T+1}(\Phi(\mathbf{A}[0]) - \Phi^*) \\ + \frac{4\sigma^2}{DK} + \frac{32(1-\delta)}{D\delta^2}(\sigma^2+\omega^2),\end{aligned} \tag{3.5}$$

*where* $\mathbf{A}[\texttt{Output}]$ *is sampled from* $\mathbf{A}[0]$ *to* $\mathbf{A}[T]$ *with uniform distribution,* $\Phi(\mathbf{A}[0]) = F(\mathbf{A}[0]) + \sum_{d=1}^{D} r_d(\mathbf{A}[0])$ *and* $\Phi^*$ *is the optimal value.*

**Remark 2.** In the nonsmooth regularization case, the above convergence result is weaker than the previous smooth case in that we only ensure the difference between the initial loss and the optimal value will get smaller, but the generalized gradient is not guaranteed to approach 0 given that the variance and gradient norm related terms will dominate with increasing $T$. However, our empirical results show that the algorithm is able to converge to small losses.

### Convergence Analysis of Algorithm 3

Now, we provide the convergence rate of Algorithm 3 by extending the proof in [5] to the block randomized setting, which is obtained under the same assumptions with Theorem 3.4.1. The main idea for the analysis is to introduce the virtual sequence of

$\widetilde{\mathbf{A}}_{(d)}^{avg}[t+1] = \widetilde{\mathbf{A}}_{(d)}^{avg}[t] - \gamma[t]\frac{1}{K}\sum_{k=1}^{R} G_{(d)}^{k}[t]$ and build an iterative descent relation for it. Meanwhile, we keep track of the error between the true and virtual averages of $\mathbf{A}_{(d)}^{avg}[t] - \widetilde{\mathbf{A}}_{(d)}^{avg}[t]$, and the deviation between the local variables and the true average of $\mathbf{A}^{avg}[t] - \mathbf{A}^{k}[t]$. Since both deviations are well-bounded, it means $\mathbf{A}^{k}[t], \mathbf{A}^{avg}[t], \widetilde{\mathbf{A}}_{(d)}^{avg}[t]$ are close to each other. Finally, we can obtain the convergence result for the true sequence $\mathbf{A}^{k}[t]$ by substituting the deviations into the descent relation obtained for $\widetilde{\mathbf{A}}_{(d)}^{avg}[t]$.

**Theorem 3.4.3.** *Suppose that Assumptions 3.4.1-3.4.5 hold. Let* $(\mathbf{A}_{(1)}^{k}[t], ..., \mathbf{A}_{(D)}^{k}[t])$ *be the iterates of Algorithm 2, for* $k = 1, ..., K$ *and* $t = 0, ..., T$. *Let* $\gamma[t] = \frac{C}{\sqrt{T+1}}$ *with* $0 < C \leq \frac{1}{L}$. *We have*

$$\mathbb{E}[\sum_{d=1}^{D}\frac{1}{D}\|\nabla_{\mathbf{A}_{(d)}}F(\mathbf{A}[\texttt{Output}])\|_F^2] \leq (4C[F(\mathbf{A}[0]) - F^*] + 2CL\sigma^2)\frac{1}{\sqrt{T+1}}$$
$$+ \Big(\frac{32C^2L^2(1-\delta^2)(\sigma^2+\omega^2)}{D\delta^2} + \frac{8C^2L^2(\sigma^2+\omega^2)}{DK}\Big)\frac{\tau^2}{T+1},$$

*where* $\mathbf{A}[\texttt{Output}] = (\mathbf{A}_{(1)}[\texttt{Output}], ..., \mathbf{A}_{(D)}[\texttt{Output}])$ *is sampled from* $\mathbf{A}^{k}[0]$ *to* $\mathbf{A}^{k}[T]$, *for all* $k = 1, ..., K$, *with uniform distribution,* $F^*$ *is the optimal value,* $\sigma^2 = \sum_{d=1}^{D}\sigma_d^2$ *and* $\omega^2 = \sum_{d=1}^{D}\omega_d^2$.

**Remark 3.** Algorithm 3 maintains the same convergence rate of $O(\frac{1}{\sqrt{T+1}})$ as Algorithm 2, despite the periodic communication. The communication gap $\tau$ only affects the term with order $O(\frac{1}{T+1})$, which is insignificant compared to the $O(\frac{1}{\sqrt{T+1}})$ overall convergence rate. Thus, without increasing the iteration complexity, the periodic communication can further reduce communication cost.

### 3.4.2 Complexity Analysis

We provide the computation, storage and communication complexities for FedGTF-EF and FedGTF-EF-PC given $|\mathcal{S}|$ fibers being sampled by each client and the rank of the GTF being $R$.

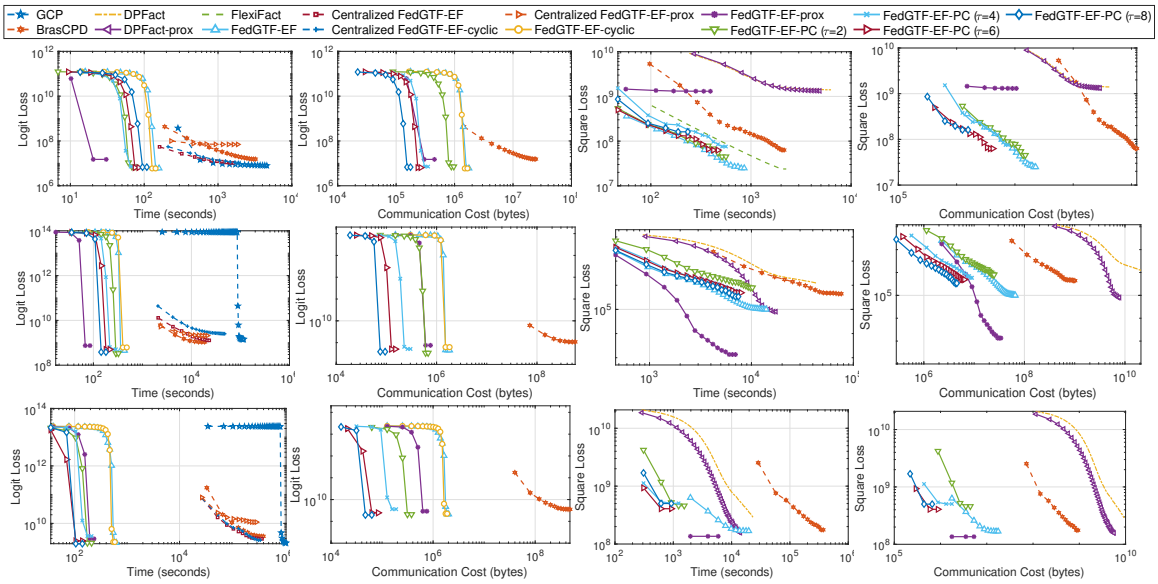



Figure 3.2: Loss decrease with respect to 1) computation time measured by seconds (column 1, 3 for Bernoulli Logit Loss and Least Square Loss respectively); 2) uplink communication cost measured by number of bytes (column 2, 4 for Bernoulli Logit Loss and Least Square Loss respectively). Top: 3-rd order CMS; Middle: 4-th order CMS; Bottom: MIMIC-III.

**Computational Complexity.** Our method is very efficient when compared to the following methods: 1) the classic CP-ALS and the full gradient descent-based GTF, which cost $O(DR\prod_{d=1}^{D} I_d)$; 2) the sampled randomized CP-ALS in [6] and SGD-based GTF in [37] with the same number of elements sampled, which cost $O(R\mathcal{S}|\sum_{d=1}^{D} I_d)$; and 3) the same complexity as the full precision block randomized SGD-based TF [25].

**Theorem 3.4.4.** *The per-iteration computational complexity of Algorithm* ***FedGTF-EF*** *and* ***FedGTF-EF-PC*** *for each client is*
$O(\frac{1}{D}(\sum_{d=1}^{D} I_d)R|\mathcal{S}|)$.

**Communication Complexity.** Assume we are using the `Sign` compressor and comparing with full precision distributed SGD with all blocks communicated. Let $D = 4, \tau = 8$, **FedGTF-EF** and **FedGTF-EF-PC** reduces up to 99.22% and 99.90% uplink communications. In general, we have:

**Theorem 3.4.5.** ***FedGTF-EF*** *reduces up to* $1 - \frac{1}{32D}$ *uplink communication and*

***FedGTF-EF-PC*** *reduces up to* $1 - \frac{1}{32D\tau}$ *uplink communication.*

**Storage Complexity.** The fiber sampling based stochastic partial gradient avoids forming the whole element-wise partial gradient tensor $\mathcal{Y}$, which reduces the storage for this step from $O(\prod_{d=1}^{D} I_d)$ to $O(|\mathcal{S}|\frac{1}{D}\sum_{d=1}^{D} I_d)$, thus achieving the same cost efficiency with sampling-based random CP-ALS [6], full precision SGD [37] and block randomized full precision SGD [25].

### 3.4.3 Privacy Analysis

FedGTF-EF offers privacy preservation by keeping the patient information securely at each local site, and letting each client hold its own data. Privacy loss is reduced through the use of block and element level communication, where less information is disclosed through the data exchange process. FedGTF-EF-PC further reduces the privacy loss through the use of the periodic communication strategy, which reduce the communication frequency, and in turn results in less information disclosure.

## 3.5 Experiment

### 3.5.1 Experimental Setup

**Datasets.** We consider two real-world EHR datasets[2], as well as a synthetic dataset, which are introduced below,

**i). CMS [1]** : A publicly available healthcare dataset with patients' information protected. We adopt the rules in [45] to select the top 500 frequently observed diagnoses, procedures, and medications to form a 4th order tensor of size $125,961 \times 500 \times 500 \times 500$ and a 3rd order tensor of size $91999 \times 500 \times 500$ (with medication mode omitted).

**ii). MIMIC-III [39]** : It is a publicly available relational dataset that describes the

[2]Code available at: https://github.com/jma78/FedGTF-EF

Table 3.1: Comparison of algorithms in ablation study.

| Algorithm | Element-level Reduction | Block-level Reduction | Round-level Reduction | Convergence Guarantee | Compression Ratio |
|---|---|---|---|---|---|
| DistBrasCPD | ✗ | ✓ | ✗ | ✗ | $1 - 1/D$ |
| DistBrasCPD-comp | ✓ | ✓ | ✗ | ✗ | $1 - 1/32D$ |
| DistSGD-EF | ✗ | ✗ | ✗ | ✗ | 0 |
| DistSGD-EF-comp | ✓ | ✗ | ✗ | ✗ | $1 - 1/32$ |
| FedGTF-EF | ✓ | ✓ | ✗ | ✓ | $1 - 1/32D$ |
| FedGTF-EF-PC | ✓ | ✓ | ✓ | ✓ | $1 - 1/32D\tau$ |

patients information of the Intensive Care Units (ICUs). Similar to CMS dataset, we form a 4 mode tensor representing patients-diagnoses-procedures-medications with size $34,272 \times 500 \times 500 \times 500$.

**iii). Synthetic data** : Synthetic data with size $4000 \times 500 \times 500 \times 500$ is generated as follows: for the nonzero entries, their values are sampled from uniform distribution for the least square loss setting and from binomial distribution for the logit loss setting, while positions of the non-zero entries are the same for both loss settings which are uniformly sampled from all entries with $10^{-4}$ non-zero ratio.

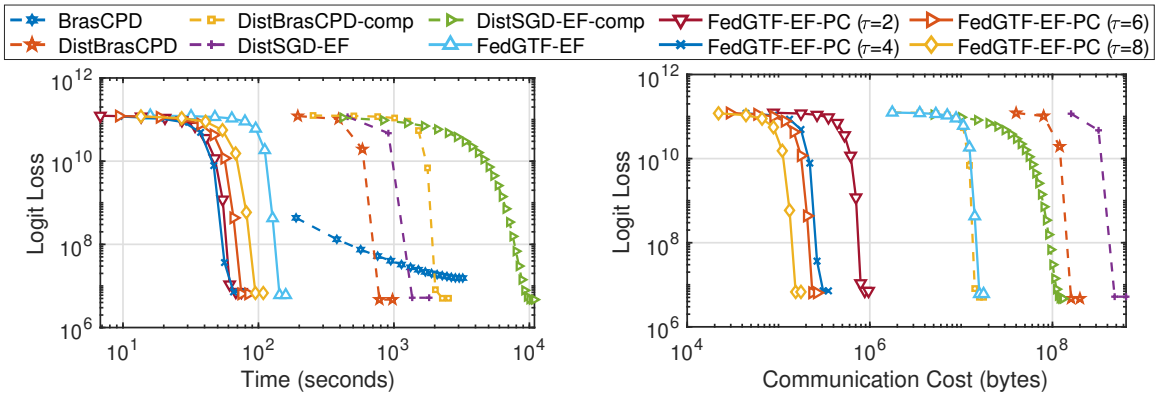



Figure 3.3: Ablation Study on 3-rd order CMS for Bernoulli Logit Loss.

**Algorithms for comparison.** We consider two different loss functions: the Bernoulli logit loss $f_{logit}$ and the least square loss. For the Bernoulli logit loss, we compare with: i) **GCP** (centralized) [47]; ii) **BrasCPD** (centralized) [25]; iii) **Centralized versions of FedGTF-EF**, iv) **FedGTF-EF-cyclic** and v) **FedGTF-EF-prox**. For the least square loss, we compare with: i) **BrasCPD** (centralized) [25]; ii) **FlexiFact** [9, 32]: a distributed tensor factorization algorithm; iii) **TRIP** [45]: a federated tensor

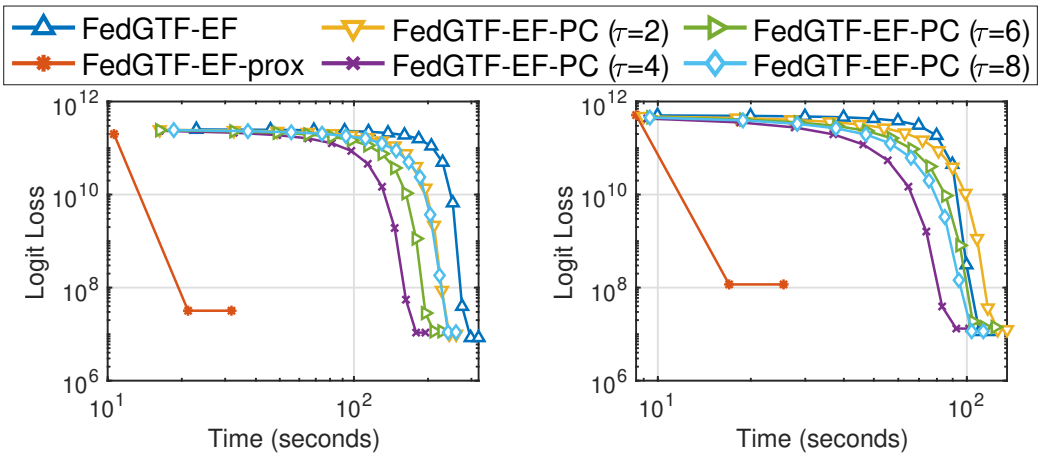



Figure 3.4: Comparison of different number of workers on 3-rd order CMS for Bernoulli Logit Loss.

factorization algorithm optimized with ADMM, which has deterministic per-iteration update solved in closed-form; iv) **DPFact** [62]: a federated SGD algorithm designed for collaborative tensor factorization. For fair comparison, we remove the differential privacy part of DPFact and substitute the $l_{2,1}$ regularization with the $l_1$ regularization as a new variant, DPFact-prox.

**Ablation study.** We conduct ablation studies to illustrate the contribution of each communication reduction mechanism to the overall communication efficiency, which includes i) DistBrasCPD: the distributed version of BrasCPD [25] or FGTF with only the block-randomized technique; ii) DistBrasCPD-comp: FGTF with both block-randomized and gradient compression techniques; iii) DistSGD-EF: distributed SGD with error-feedback that communicates full gradients and all blocks; iv) DistSGD-EF-comp: DistSGD-EF with gradient compression. Table 4.2 summarize the comparison with the proposed algorithms.

For our proposed algorithms, in addition to FedGTF-EF and FedGTF-EF-PC, we consider two variants: FedGTF-EF-cyclic (a variant of FedGTF-EF with cyclic mode updates), FedGTF-EF-prox (FedGTF-EF with $l_1$ regularization). We vary the value of $\tau$ in $\{2, 4, 6, 8\}$ for FedGTF-EF-PC.

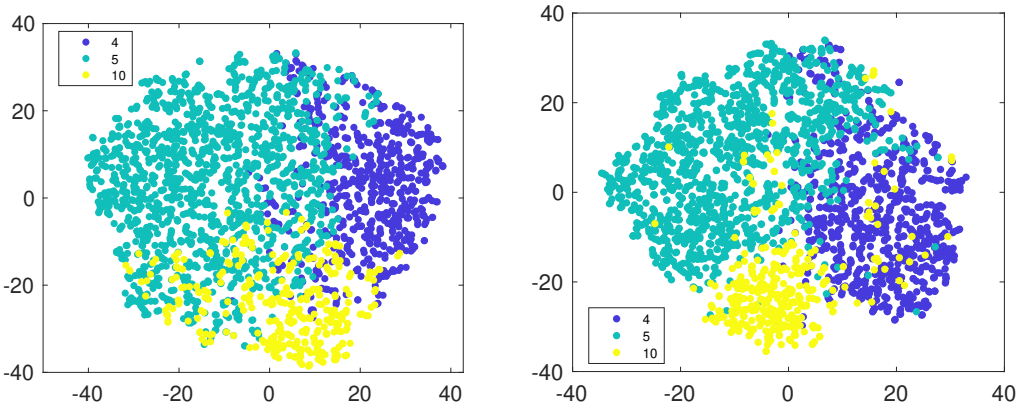



Figure 3.5: tSNE visualization of the patient representation learned by BrasCPD (left) and FedGTF-EF-PC($\tau = 8$) (right). Each point represents a patient which is colored according to the highest-valued coordinate in the patient representation vector among the top 3 phenotypes extracted based on the factor weights $\lambda_r = \left\|\mathbf{A}_{(1)}(:,r)\right\|_F \left\|\mathbf{A}_{(2)}(:,r)\right\|_F \cdots \left\|\mathbf{A}_{(D)}(:,r)\right\|_F$.

### 3.5.2 Experiment results

Our experiments show that FedGTF-EF and FedGTF-EF-PC are able to greatly improve the communication efficiency without slowing down the convergence and deteriorating the factorization quality. In detail, we have the following four observations: **i)** FedGTF-EF and its variants reduce loss faster with much less communication cost, for both the Bernoulli Logit Loss (Fig. 3.2 first two columns) and the Least Square loss (Fig. 3.2 last two columns) compared to the baseline methods. The communication cost per communication round is further reduced by increasing the local update iterations $\tau$ from 2 to 8 without hurting the performance of the Bernoulli logit loss and with a slightly worse loss for the least square loss. **ii)** FedGTF-EF, FedGTF-EF-PC and their variants are computationally efficient due to the fiber-sampling technique, i.e., they use lower computation cost compared to the baselines. By Fig. 3.2, for both objective functions, FedGTF-EF-PC, FedGTF-EF and its variants converges to similar losses as their centralized counterparts, while cost less time because more workers are involved in the updating process for the federated setting. Note that although TRIP converges faster in terms of time, but it tends to be trapped into

bad local minima caused potentially by its deterministic per-iteration update. **iii)** FedGTF-EF, FedGTF-EF-PC and their variants converge to similar losses as the centralized counterpart, which indicates communication efficiency can be improved without sacrificing the factorization quality. **iv)** FedGTF-EF and FedGTF-EF-PC converge faster in terms of running time with more workers. As shown in Fig. 3.2 upper left and Fig. 4.4, with the number of workers increased from 8 to 16, the time for FedGTF-EF to converge reduces 65.58%.

From the ablation study (Fig. 4.5), we can see: **i)** Block-randomized update and gradient compression can greatly reduce the communication cost by 75.00% and 96.88%, respectively. Therefore, gradient compression plays a more important role in communication reduction. **ii)** With both block-randomized and gradient compression, FedGTF-EF achieves a gradient reduction of 98.90% over FGTF. **iii)** Periodic communication further reduces the communication cost over FGTF by 99.94%, 99.97, 99.98%, and 99.99% with $\{2, 4, 6, 8\}$ rounds of local communications respectively.

Finally, we evaluate the quality of the federated factorization factors by considering the patient subgroup identification following [68], as illustrated in Fig. 3.5. We use tSNE to map the $R$ dimensional vectors into the 2 dimensional space. We first identify the top 3 phenotypes that have the largest factor weights, which are the phenotypes $\#4, \#5, \#10$ in Fig. 3.5 (phenotype details are shown in Table 3.2). Then, we color the patients by assigning each patient to one of the top 3 phenotypes using the largest patient weight among the top 3 along the representation vector. Fig. 3.5 shows FedGTF-EF-PC with $\tau = 8$ local updates has comparable performance to the centralized baseline BrasCPD in clustering the patients with the same phenotype together. This demonstrates that our method can achieve communication compression without sacrificing the factorization quality.

Table 3.2: Top 3 phenptypes extracted by FedGTF-EF-PC($\tau = 8$) on MIMIC-III data. Red, blue, and green indicate diagnoses, procedures, and medication, respectively.

| **P10: Diabetic Heart Failure** |
|---|
| Diabetes mellitus without mention of complication |
| Background diabetic retinopathy |
| Acute systolic heart failure |
| Acute on chronic systolic heart failure |
| Chronic diastolic heart failure |
| Acute on chronic combined systolic and diastolic heart failure |
| Insertion of one vascular stent |
| Open heart valvuloplasty of tricuspid valve without replacement |
| Operations on other structures adjacent to valves of heart |
| (Aorto)coronary bypass of three coronary arteries |
| Captopril (ACE inhibitor), Insulin, Pyridostigmine Bromide, Isosorbide Dinitrate |
| **P5: Hypertensive Heart Failure** |
| Pure hypercholesterolemia |
| Cardiac tamponade |
| Ventricular fibrillation |
| Cardiac arrest |
| Acute systolic heart failure |
| Percutaneous insertion of carotid artery stent(s) |
| Pericardiocentesis |
| Extracorporeal circulation auxiliary to open heart surgery |
| Other endovascular procedures on other vessels |
| Rosuvastatin Calcium, Isosorbide Dinitrate, Hydrochlorothiazide, Digoxin, Clonidine HCl |
| **P4: Peripheral Arterial Disease** |
| Congestive heart failure |
| Atherosclerosis of native arteries of the extremities |
| – with intermittent claudication |
| Acute venous embolism and thrombosis of |
| –superficial veins of upper extremity |
| Insertion of drug-eluting coronary artery stent(s) |
| (Aorto)coronary bypass of two coronary arteries |
| Interruption of the vena cava |
| Suture of artery |
| Angioplasty of other non-coronary vessel(s) |
| Carvedilol, Metoprolol succinate, Amiodarone HCl, Nitroglycerin, Calcium Chloride |

# Chapter 4

# Decentralized Communication-efficient Generalized Tensor Factorization

In this chapter, we extend the work in Chapter 3 to the decentralized setting, and further reduce the communication cost at the round level via an event-driven communication strategy. Moreover, we also incorporate Nesterov's momentum to accelerate the training.

## 4.1 Introduction

Under the federated learning settings, the central server is the most important computation resource as it is in charge of coordinating the clients (i.e., picking random clients to communicate at each iteration), aggregating the clients' intermediate results, and updating the global model. This shows that federated learning systems heavily rely on the central server. However, a single server might have several shortcomings: 1) limited connectivity and bandwidth, which restrict the server from collecting data from as many clients as possible; 2) vulnerability to malfunctions, which will cause inaccurate

model updates, or even learning failures; and 3) exposure to external attacks and malicious adversaries, which will lead to sensitive information leakage (thus, under the federated settings, a server is usually assumed to be honest-but-curious [45, 62]). Due to the above limitations, it is obvious that traditional Federated Tensor Factorization suffers from the bottleneck of the central server regarding the communication latency and bandwidth, and is exposed to high risk of single-point-failure. To avoid relying on the server as the only source of computation, decentralization has been proposed as a solution to this single-point-failure issue [50, 51]. Decentralized federated learning is designed without the participation of the central server, while each client will rely on its own computation resources and communicate only with its neighbors in a peer-to-peer manner. Besides the necessities of a decentralized communication topology, it is also worth noting that the network capacity between clients are usually much smaller than the datacenter in many real-world applications [87]. Therefore it is necessary that the clients communicate the model updates efficiently with limited cost.

In this chapter, we study the decentralized optimization of tensor factorization under the horizontal data partition setting, and propose `CiderTF`, a Communication-effIcient DEcentralized geneRalized Tensor Factorization algorithm for collaborative analysis over a communication network. To enable more flexibility on choosing different loss functions under various scenarios, we extend the classic Federated Tensor Factorization into a more Generalized Tensor Factorization. To the best of our knowledge, this chapter is the first one proposing a decentralized generalized tensor factorization, let alone considering the decentralized setting with communication efficiency. Our contributions are briefly summarized as follows.

First, we develop a decentralized tensor factorization framework which employs four levels of communication reduction strategies to the decentralized optimization of tensor factorization to reduce the communication cost over the communication network. At the element-level, we utilize gradient compression techniques [43, 103, 5, 3, 73]

to reduce the number of bytes transmitted between clients by converting the partial gradient from the floating point representation to low-precision representation. At the block-level, we apply the randomized block coordinate descent [7, 26, 66] for the factor updates, which only requires sampling one mode from all modes of a tensor for the update per round and communicating only one mode factor updates with the neighbors. At the round-level, we adopt a periodic communication strategy [80, 54, 5] to reduce the communication frequency by allowing each client to perform $\tau > 1$ local update rounds before communicating with its neighbors. In addition, at the communication event-level, we apply an event-triggered communication strategy [56, 19, 78] to boost the communication reduction at the round level.

Second, we further incorporate Nesterov's momentum into the local updates of `CiderTF` and propose `CiderTF_m`, in order to achieve better generalization and faster convergence.

Third, we conduct comprehensive experiments on both real-world and synthetic datasets to corroborate the theoretical communication reduction and the convergence of `CiderTF`. Experiment results demonstrate that `CiderTF` achieves comparable convergence performance with the communication reduction of 99.99%. Furthermore, we conduct an extensive case study on MIMIC-III data with regard to the factorization quality from both quantitative and qualitative aspects. The resulting factor matrices highly resembles the factors extracted by the centralized tensor factorization baseline with significantly less communication cost. The `CiderTF` extracted phenotypes are shown to be highly interpretable according to a clinician.

## 4.2 Preliminaries and Background

In this section, we summarize the frequently used definitions and notations, and introduce the background knowledge of tensor factorization, the related communication

Table 4.1: Symbols and notations used in this chapter

| Symbol | Definition |
|---|---|
| $\mathbf{x}, \mathbf{X}, \boldsymbol{\mathcal{X}}$ | Vector, Matrix, Tensor |
| $\boldsymbol{\mathcal{X}}_{<d>}$ | Mode-$d$ matricization of $\boldsymbol{\mathcal{X}}$ |
| $\|\cdot\|_1$ | $\ell_1$-norm |
| $\|\cdot\|_F$ | Frobenius norm |
| $\circledast$ | Hadamard (element-wise) multiplication |
| $\odot$ | Khatri Rao product |
| $\circ$ | Outer product |
| $\langle\cdot,\cdot\rangle$ | Inner product |

reduction techniques, and decentralized optimization.

### 4.2.1 Notations and Operators

For a $D$-th order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times ... \times I_D}$, the tensor entry indexed by $(i_1, ..., i_D)$ is denoted by the MATLAB representation $\boldsymbol{\mathcal{X}}(i_1, ..., i_D)$. Let $\mathcal{I}$ denote the index set of all tensor entries, $|\mathcal{I}| = I_\Pi = \prod_{d=1}^{D} I_d$. The mode-$d$ unfolding (also called matricization) is denoted by $\mathbf{X}_{<d>} \in \mathbb{R}^{I_d \times I_\Pi / I_d}$.

**Definition 4.2.1.** (*MTTKRP*). The MTTKRP operation stands for the *matricized tensor times Khatri-Rao product*. Given a tensor $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{I_1 \times ... \times I_D}$, its mode-$d$ matricization is $\mathbf{Y}_{<d>}$, $[\mathbf{A}_{(1)}, ..., \mathbf{A}_{(D)}]$ is the set of CP factor matrices. $\mathbf{H}_d \in \mathbb{R}^{I_\Pi / I_d \times R}$ is defined as

$$\mathbf{H}_d = \mathbf{A}_{(D)} \odot ... \odot \mathbf{A}_{(d+1)} \odot \mathbf{A}_{(d-1)} ... \odot \mathbf{A}_{(1)},$$

where $\odot$ is the Khatri-Rao product. The MTTKRP operation can thus be defined as the matrix product between $\mathbf{Y}_{<d>}$ and $\mathbf{H}_d$ as $\mathbf{Y}_{<d>} \cdot \mathbf{H}_d$.

### 4.2.2 Communication Reduction

**Gradient compression.** Communication can be a primary bottleneck of the efficiency of the distributed training, especially in federated learning since the connection between clients and the server usually operates at low speeds ($\sim$1 Mbps) ([73]), and

the uplink bandwidth is generally slower than the downlink bandwidth. Gradient compression based methods can compress the communicated information that are transmitted from clients and the server by reducing the number of bits.

**Periodic communication.** Periodic communication, which is also known as local SGD [80, 54], has been developed in order to overcome the communication bottleneck in distributed training. Instead of keeping different clients in frequent synchronization, periodic communication allows clients to perform $\tau > 1$ local updates before communicating, which reduces the communication frequency. Most recently, [5] explored the combination of periodic communication with various gradient compression strategies, [81] introduced the error-feedback to analyze the convergence rate for the biased gradient compression and local SGD.

**Event-triggered communication.** The event-triggering mechanism was first proposed in the control community [18, 33, 74], and then was extended to be used in distributed optimization [14, 19]. Most recently, [78, 79] were proposed which study the combination of the event-driven lazy communication with gradient compression under the decentralized settings.

### 4.2.3 Decentralized Optimization

Decentralized optimization algorithms have been widely studied from multiple domains considering the limited bandwidth, communication latency, and data privacy of the distributed networks. In the machine learning domain, D-PSGD [51] was first proposed for the non-convex setting as a decentralized version of SGD with linear speedup, where $K$ is the number of workers, and $T$ is the total number of iterations. [84] extended D-PSGD to the data heterogeneity settings. [49] provided a unified framework for the gossip-based decentralized SGD with theoretical convergence analysis for

In addition, there are multiple works investigating the communication reduction in decentralized optimization. [83] proposed DCD/ECD-PSGD which quantize the model

updates with high precision quantizers. [48] proposed CHOCO-SGD, which is the first work that supports arbitrary compressors. [60] expanded the scope of the applicable quantizers and supported 1-bit quantizer with no additional memory required. [78] further reduced the communication cost with an event-driven communication strategy.

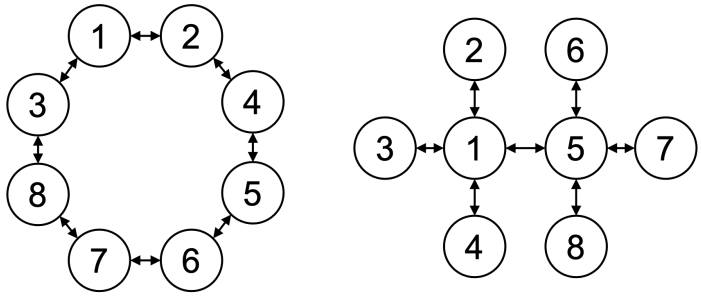



Figure 4.1: Ring topology (left) and star topology (right).

## 4.3 Proposed Method

We study the decentralized generalized tensor factorization, where the EHR tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times,...,\times I_D}$ is horizontally partitioned along the patient mode into $K$ small local tensors $\boldsymbol{\mathcal{X}}^k \in \mathbb{R}^{I_{1^k} \times I_2 \times ... \times I_D}, k = 1, \cdots, K$, which are distributed among $K$ institutions. The aim is to collaboratively learn the phenotypes through communicating with the neighboring clients in the decentralized network without the coordination of the central server. We propose to solve the decentralized optimization using the gossip-based algorithm with multiple communication reduction techniques, including block randomization, gradient compression, periodic communication, and event-triggered communication.

### 4.3.1 Problem Formulation

In the decentralized tensor factorization setting, the communication topology is represented by an undirected graph $\boldsymbol{\mathcal{G}} = (\boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{E}})$, where $\boldsymbol{\mathcal{V}} := \{1, 2, ..., K\}$ denotes the

set of clients participating in the communication network. Each node $k$ in the graph represents a client. The neighbors of client $k$ is denoted as ${}_k := \{(k,j) : (k,j) \in \boldsymbol{\mathcal{E}}\}$. There is a connectivity matrix $W \in \mathbb{R}^{K\times K}$, the $(k,j)$-th entry $w_{kj} \in [0,1], \forall (k,j) \in \boldsymbol{\mathcal{E}}$ in which denotes the weights of edge $(k,j) \in \boldsymbol{\mathcal{E}}$ and measures how much the client $k$ is impacted by client $j$. If the graph is symmetric, then $w_{kj} = w_{jk}$, while if there is no connection between client $k$ and client $j$, then $w_{kj} = 0$. We assume the connectivity matrix $W$ is symmetric and doubly stochastic where each row and column sums to one ($\sum_k w_{kj} = \sum_j w_{jk} = 1$).

Each client in the decentralized communication graph will hold a local tensor $\boldsymbol{\mathcal{X}}^k$, which can be seen as the horizontal partition of a global tensor $\boldsymbol{\mathcal{X}}$. The aim for the decentralized federated learning is to jointly factorize the local tensors $\boldsymbol{\mathcal{X}}^k$ to get the globally shared feature factor matrices $\mathbf{A}_{(2)}, ..., \mathbf{A}_{(D)}$, and the individual mode factor matrices $\mathbf{A}^k_{(1)}$ from all clients

$$\boldsymbol{\mathcal{X}} = \begin{bmatrix} \boldsymbol{\mathcal{X}}^1 \\ \vdots \\ \boldsymbol{\mathcal{X}}^K \end{bmatrix} \approx \begin{bmatrix} \boldsymbol{\mathcal{A}}^1 = \mathbf{A}^1_{(1)} \circ \mathbf{A}_{(2)} \circ ... \circ \mathbf{A}_{(D)} \\ \vdots \\ \boldsymbol{\mathcal{A}}^K = \mathbf{A}^K_{(1)} \circ \mathbf{A}_{(2)} \circ ... \circ \mathbf{A}_{(D)} \end{bmatrix}. \tag{4.1}$$

The objective function for the decentralized generalized tensor factorization is shown as

$$\begin{aligned} &\underset{(\mathbf{A}_{(1)},...,\mathbf{A}_{(D)})}{\operatorname{argmin}} \sum_{k=1}^{K} F(\boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{X}}^k), \\ &s.t.\ \boldsymbol{\mathcal{A}} = \mathbf{A}_{(1)} \circ ... \circ \mathbf{A}_{(D)}, \end{aligned} \tag{4.2}$$

which can be further extended to other multiblock optimization problems which are not limited to tensor factorization [100].

**Algorithm 4:** `CiderTF`: Decentralized Generalized Tensor Factorization with Compressed, Block-randomized, Periodic, and Event-triggered Communication

**Input:** Input tensor $\boldsymbol{\mathcal{X}}$, constant learning rate $\gamma[t]$, $\mathbf{A}[0], \mathbf{A}^k[0] = \mathbf{A}[0], \forall k = 1, ..., K$, randomized block sampling sequence $d_\xi[0], ..., d_\xi[T]$, event-triggering threshold $\lambda[t]$;

**for** $t = 0, ..., T$ **do**
  **On Each Client Nodes** $k \in 1, ..., K$**:**
  **if** $d = d_{(\xi)}[t]$ **then**
    Compute stochastic gradient $\mathbf{G}^k_{(d)}[t]$ by eq. (4.6);
    $\mathbf{A}^k_{(d)}[t + \frac{1}{2}] = \mathbf{A}^k_{(d)}[t] - \gamma[t]\mathbf{G}^k_{(d)}[t]$;
    **if** $(t \mod \tau) \neq 0$ **then**
      No communication:
      $\mathbf{A}^k_{(d)}[t+1] = \mathbf{A}^k_{(d)}[t + \frac{1}{2}]$, $\widehat{\mathbf{A}}^k_{(d)}[t+1] = \widehat{\mathbf{A}}^k_{(d)}[t]$;
    **else**
      **for** $j \in \boldsymbol{\mathcal{N}}_k \cup k$ **do**
        **if** $\|\mathbf{A}^k_{(d)}[t + \frac{1}{2}] - \widehat{\mathbf{A}}^k_{(d)}[t]\|^2_F \geq \lambda[t](\gamma[t])^2$ **then**
          $\boldsymbol{\Delta}^k_{(d)}[t] = \texttt{Compress}(\mathbf{A}^k_{(d)}[t + \frac{1}{2}] - \widehat{\mathbf{A}}^k_{(d)}[t])$;
        **else**
          $\boldsymbol{\Delta}^k_{(d)}[t] = \mathbf{0}_{I^k \times R}$;
        **end if**
        Send $\boldsymbol{\Delta}^k_{(d)}[t]$ to all $j$ and receive $\boldsymbol{\Delta}^j_{(d)}[t]$ from all $j$, where $j \in \boldsymbol{\mathcal{N}}_k$;
        $\widehat{\mathbf{A}}^j_{(d)}[t+1] = \widehat{\mathbf{A}}^j_{(d)}[t] + \boldsymbol{\Delta}^j_{(d)}[t]$;
      **end for**
      $\mathbf{A}^k_{(d)}[t+1] = \mathbf{A}^k_{(d)}[t + \frac{1}{2}] + \varrho \sum_{j \in \boldsymbol{\mathcal{N}}^k} w_{kj}(\widehat{\mathbf{A}}^j_{(d)}[t+1] - \widehat{\mathbf{A}}^k_{(d)}[t+1])$;
    **end if**
  **else if** $d \neq d_\xi[t]$ **then**
    $\mathbf{A}^k_{(d)}[t+1] = \mathbf{A}^k_{(d)}[t]$, $\widehat{\mathbf{A}}^k_{(d)}[t+1] = \widehat{\mathbf{A}}^k_{(d)}[t]$;
  **end if**
**end for**

### 4.3.2 CiderTF: Decentralized Generalized Tensor Factorization with Compressed, Block-randomized, Periodic, and Event-triggered Communication

**Overview**

We propose CiderTF, a decentralized tensor factorization framework which achieves communication efficiency through four levels of communication reduction. We utilize the widely used sign compressor [81, 82] (Def. 3.2.1) for gradient compression.

Moreover, we apply the block randomization for the factor updates, which can not only reduce the computational complexity per iteration, but also eliminate the need of sending full factor matrices. Furthermore, we reduce the communication frequency via the combination of the periodic communication and the event-triggered communication strategies. Each client only need to check for the triggering condition every $\tau$ iterations, and will only need to communicate the compressed updates when the triggering condition is satisfied.

The detailed algorithm is shown in Algorithm 4 with the key steps annotated. In CiderTF, each client $k \in [K]$ maintains the local factor matrices $\mathbf{A}^k_{(d)}$ from each mode $d = 1, ..., D$. The goal is to achieve consensus on the feature mode factor matrices $\mathbf{A}^k_{(d)}, \forall d = 2, ..., D$. Therefore, besides the local factor matrices, each client also need to maintain the estimation of the local factor matrices $\mathbf{A}^j_{(d)}$ from both itself $k$ and its neighbors $\boldsymbol{\mathcal{N}}_k$ ($j \in \boldsymbol{\mathcal{N}}_k \cup k$). The sequence of the randomized sampling blocks for every round $t = 1, ..., T$ is denoted as $d_\xi[0], ..., d_\xi[T]$. At every round for the sampled block $d_\xi[t]$, each client checks for the triggering condition for every $\tau$ iterations at the communication round (line 10). The triggering threshold is set to be $\lambda[t]$. When the difference between the updated factor and the local estimation is larger than the threshold, each client will send and receive the compressed updates to its neighbors. While if the triggering condition is not satisfied, then the clients will just communicate

a matrix of zero instead (line 10-14). After receiving the compressed updates from all its neighbors, each client will first update the local estimation of the factor matrices $\widehat{\mathbf{A}}^j_{(d)}[t+1], j \in \boldsymbol{\mathcal{N}}_k \cup k$ (line 16), and conduct the consensus step and update the local factors $\mathbf{A}^k_{(d)}[t+1]$ through the decentralized consensus step (line 18). At the non-communication round, each client will just keep updating the local factor matrices (line 6-7). For the rest of the blocks not selected, they will remain the same at the last round (line 20-22).

**Optimization**

At each iteration, each client $k$ first needs to compute the GCP gradient as the partial derivative with regard to the factor matrix $\mathbf{A}^k_{(d)}$ using the MTTKRP operator

$$\frac{\partial F(\boldsymbol{\mathcal{A}}^k, \boldsymbol{\mathcal{X}}^k)}{\partial \mathbf{A}^k_{(d)}[t]} = \mathbf{Y}^k_{<d>}\mathbf{H}^k_d, \tag{4.3}$$

where each element of the matrix $\mathbf{Y}_{<d>}$ is defined as

$$\boldsymbol{\mathcal{Y}}^k(i) = \frac{\partial f(\boldsymbol{\mathcal{A}}^k(i), \boldsymbol{\mathcal{X}}^k(i))}{\partial \boldsymbol{\mathcal{A}}^k(i)}, \forall i \in \mathcal{I}, \tag{4.4}$$

and $\mathbf{H}^k_d$ denotes the Khatri-Rao product of mode $d$ of the factor matrices as is shown in definition 4.2.1. Then the local factor matrix is updated with the gradient descent step

$$\mathbf{A}^k_{(d)}[t+\frac{1}{2}] = \mathbf{A}^k_{(d)}[t] - \gamma[t]\frac{\partial F(\boldsymbol{\mathcal{A}}^k, \boldsymbol{\mathcal{X}}^k)}{\partial \mathbf{A}^k_{(d)}[t]}. \tag{4.5}$$

**Fiber Sampling.** Note that the computational complexity of the full gradient $\frac{\partial F(\boldsymbol{\mathcal{A}}^k, \boldsymbol{\mathcal{X}}^k)}{\partial \mathbf{A}^k_{(d)}[t]}$ is $O(R\prod_{d=1}^{D} I_d)$, which is the bottleneck of the traditional gradient based optimization for tensor factorization, especially for EHR tensors where each dimension can be very large. In order to tackle the time complexity of computing the gradient, we propose to utilize an efficient fiber sampling technique [6, 26], which randomly

samples $|\mathcal{S}_d|$ fibers from mode $d$. The fiber sampling technique provides efficient computation of $\mathbf{H}_d^k(\mathcal{S}_d, :)$, where only fibers are sampled, and $\mathbf{H}_d^k(\mathcal{S}_d, :)$ is computed without forming $\mathbf{H}_d^k$ explicitly, but only the $s$-th rows of $\mathbf{H}_d^k$ are required to be computed as the Hadamard product ($\circledast$) of the certain rows of the factor matrices at time $t$ as $\mathbf{H}_d^k(s, :) = \mathbf{A}_{(1)}^k(i_1^s, :) \circledast ... \circledast \mathbf{A}_{(d-1)}^k(i_{d-1}^s, :) \circledast \mathbf{A}_{(d+1)}^k(i_{d+1}^s, :) \circledast ... \circledast \mathbf{A}_{(D)}^k(i_D^s, :)$, where the indices of rows of the factor matrices are obtained from the index mapping $\{i_1^s, ..., i_D^s\}, s \in \mathcal{S}_d$. Fiber sampling also allows us to avoid forming the full matricization of $\mathbf{Y}_{<d>}$, but only need to form $\mathbf{Y}_{<d>}^k(:, \mathcal{S}_d)$ with size $I_d \times |\mathcal{S}|$ [47]. Therefore, the local partial stochastic gradient, which is denoted as $\mathbf{G}_{(d)}^k[t]$, is considered as an unbiased estimation of the gradient $\frac{\partial F(\boldsymbol{\mathcal{A}}^k, \boldsymbol{\mathcal{X}}^k)}{\partial \mathbf{A}_{(d)}^k[t]}$, and can be efficiently computed with the fiber sampling technique as

$$\mathbf{G}_{(d)}^k[t] = \mathbf{Y}_{<d>}^k(:, \mathcal{S}_d)\mathbf{H}_d^k(\mathcal{S}_d, :), \tag{4.6}$$

**Block randomization.** Besides fiber sampling, we utilize the block randomization [26] to further improve the computation efficiency. Under block randomization, we randomly select a mode to update at each round, instead of updating all modes. In other words, for every epoch, there is a random variable $d_\xi[t] \in \{1, ..., D\}$ representing the selected mode, and the probability of each mode to be updated at each round is

$$Pr(d_\xi[t] = d) = \frac{1}{D}. \tag{4.7}$$

Specially for `CiderTF`, we always keep the patient mode (i.e. the 1-st mode) securely at local to avoid directly sharing patient related information, thus when $d_\xi[t] = 1$, we skip the communication of this round, and only update the local patient mode factors. This not only improves the computation efficiency of the optimization, but also reduces the communication cost at the block level.

### 4.3.3 `CiderTF_m`: `CiderTF` with Nesterov's momentum

We further propose `CiderTF_m` with Nesterov's momentum incorporated in the local SGD update step to speedup the convergence and achieve less total communication bits. After computing the partial stochastic gradient $\mathbf{G}^k_{(d)}[t]$ (line 4), we update the momentum velocity component as

$$\mathbf{M}^k_{(d)}[t] = \mathbf{G}^k_{(d)}[t] + \beta \frac{\eta[t-1]}{\eta[t]} \mathbf{M}^k_{(d)}[t-1] \tag{4.8}$$

where $\beta$ is the momentum parameter. The intermediate factor matrix will be updated as

$$\mathbf{A}^k_{(d)}[t+\frac{1}{2}] = \mathbf{A}^k_{(d)}[t] - \gamma[t](\mathbf{G}^k_{(d)}[t] + \beta \mathbf{M}^k_{(d)}[t]) \tag{4.9}$$

### 4.3.4 Complexity Analysis

We analyze the complexity from the perspective of computation, communication, and memory cost.

#### Computational Complexity

**Theorem 4.3.1.** *The per-iteration computational complexity of* `CiderTF` *for each client is* $O(\frac{1}{D}(\sum_{d=1}^{D} I_d)R|\mathcal{S}|)$.

For each client, assume $|\mathcal{S}_d|$ are the same for all $d$ as $|\mathcal{S}|$, the computational complexity per-iteration of the partial stochastic gradient $\mathbf{G}^k_{(d)}[t] = \mathbf{Y}^k_{<d>}(:, \mathcal{S}_d)\mathbf{H}^k_d(\mathcal{S}_d, :)$ consists of three parts: 1) computing the matricization with fiber sampling $\mathbf{Y}^k_{<d>}(:, \mathcal{S}_d)$ takes $O(I_d|\mathcal{S}|)$; 2) computing the Khatri-Rao product with fiber sampled factor matrices $\mathbf{H}^k_d(\mathcal{S}_d, :)$ takes $O(R|\mathcal{S}|(D-1))$; 3) the matrix multiplication between $\mathbf{Y}^k_{<d>}(:, \mathcal{S}_d)$ and $\mathbf{H}^k_d(\mathcal{S}_d, :)$ takes $O(|\mathcal{S}|RI_d)$. In addition, the factor matrix updates is conducted per iteration with the size of $O(I_d R)$, where $I_d$ is the number of element of mode

$d$, and $R$ is the rank of the decomposed tensor. Overall, the total computational complexity is summarized as $O(\frac{1}{D}(\sum_{d=1}^{D} I_d)R|\mathcal{S}|)$.

### Communication Complexity

**Theorem 4.3.2.** `CiderTF` *reduces a lower bound of* $1 - \frac{1}{32D\tau}$ *communication.*

The use of `Sign` compressor will require each client $k$ sending $\frac{1}{D}R\sum_{d=2}^{D} I_d$ bits to each neighbor in $\boldsymbol{\mathcal{N}}_k$ with the block randomization in reducing the communication by a factor of $D$. The periodic communication strategy helps reduce the communication cost by $\frac{1}{\tau}$ and results in a total cost of $\frac{1}{D\tau}R\sum_{d=2}^{D} I_d$. Then without the event-triggering mechanism, there is a lower bound of communication reduction of $1 - \frac{1}{32D\tau}$, compared with the decentralized SGD with full precision gradients which has the per-iteration cost of $32(\sum_{d=1}^{D} I_d)$. The event-triggering mechanism helps further reduce the communication cost by an upper bound of $36\times$ per epoch (an epoch contains 500 iterations for our experiments). The total communication reduction is 99.99% compared with the full precision decentralized SGD based on experimental results.

### Memory Complexity

**Theorem 4.3.3.** `CiderTF` *has the memory complexity of* $O(|\mathcal{S}|\frac{1}{D}\sum_{d=1}^{D} I_d)$.

The memory complexity savings of `CiderTF` comes from the fiber sampling technique, which eliminates the need for each client to form the whole mode-$d$ matricized tensor $\mathbf{Y}^k_{<d>}$ with the size of $\prod_{d=1}^{D} I_d$. Instead, each client only needs to form a "sketched version" of $\mathbf{Y}^k_{<d>}(:, \mathcal{S}_d)$ with size $I_d \times |\mathcal{S}|$. Thus the memory complexity is reduced to $O(|\mathcal{S}|\frac{1}{D}\sum_{d=1}^{D} I_d)$.

### Privacy Enhancement

`CiderTF` offers privacy enhancement through the elimination of the central server, which reduces the risk of having ther server exposed to the external attacks and being

the single failure point. It further provides the privacy preservation through making the communication more efficient. By employing the block level, round level, and event level communication reduction techniques, `CiderTF` will require less information to be transmitted between clients. The application of the Nesterov's momentum, which helps increase the convergence speed, also helps reduce the information to be exchanged. In addition, with the element level communication reduction technique, since clients only need to transmit the low-precision version of the model updates instead of the true values, it reveals less information, and thus enhances the privacy of the algorithm.

## 4.4 Experiment

### 4.4.1 Experimental Settings

#### Datasets

We conduct experiments on two real-world datasets, including MIMIC-III [39] and CMS [1], which are large volume, publicly available and de-identified. MIMIC-III data contains more than fifty thousand of intensive care unit (ICU) stays from 2001-2012. The CMS (DE-SynPUF) dataset is a realistic set of claim data with the highest degree of protection on the patient information and has similar data structure as the real CMS data. It contains more than six billion beneficiary records from 2008-2010. We also generate a synthetic dataset with similar sparsity as the real-world datasets to further testify the generalizability of our algorithm. To reduce the sparsity, we follow the rules in [45] and select the top 500 diagnoses, procedures, and medications of the most frequently observed records to form the tensors with patient mode 34,272, 125,961, and 4000 for MIMIC-III, CMS, and Synthetic data, respectively.

### Baselines

We consider the centralized tensor factorization baselines. i) **GCP** [47] as the centralized baseline of the generalized tensor factorization; ii) **BrasCPD** [26] as the computation efficient centralized tensor factorization baseline; iii) **Centralized** `CiderTF`, `CiderTF` with $K = 1$ and uses error-feedback to adjust the compression error.

In addition, we implement the decentralized version SGD under the non-convex settings as the decentralized baselines, since there is no existing decentralized tensor factorization framework. i) **D-PSGD** [51, 49] as a pure decentralized version of stochastic gradient descent (SGD); ii) **SPARQ-SGD** [78] as a decentralized communication-efficient stochastic gradient descent framework which employs the gradient compression, local SGD, and the event-triggered communication to reduce the communication cost; iii) **D-PSGDbras** can be considered as D-PSGD with block randomization.

### Parameter Settings

Experiments are performed on two kinds of objective functions including Bernoulli-logit loss to fit the binary data and Least Square Loss to fit the data with Gaussian distribution, which is also considered as standard CP decomposition.

We set the number of iterations per epoch as 500. We use a fixed learning rate $\gamma[t]$, which is determined through searching the grid of powers of 2. We follow the rule in [78] to set the triggering threshold $\lambda[t]$. The triggering threshold is initialized as $\lambda[0]$ at the beginning, and will be increased by a constant factor $\alpha_\lambda$ every $m$ epochs until convergence in order to prevent the clients satisfying the triggering condition every epoch. We set $\lambda[0]$ as $\frac{1}{\gamma[t]}$ according to [78], and set $\alpha_\lambda$ and $m$ through grid search within $[1, 2]$ and $[1, 5]$. For `CiderTF_m`, we set the momentum factor $\beta$ as 0.9.

### 4.4.2 Result Analysis

We form a decentralized communication topology as a ring, and have a default of eight workers with data horizontally partitioned and distributed evenly across all the eight clients.

#### Comparison to the Baselines

From Fig. 4.2, we have four major observations.

i) `CiderTF` **converges to comparable losses as the centralized tensor factorization baselines.** `CiderTF`, with various number of local update rounds ($\tau = \{2, 4, 6, 8\}$), achieves similar losses at convergence compared with the centralized tensor factorization algorithms. These results empirically validate the convergence of `CiderTF`.

ii) `CiderTF` **has less communication cost compared with the decentralized baselines.** To achieve the same loss, `CiderTF` takes 99.99% less communication cost than D-PSGD, 75% less communication cost than SPARQ-SGD and 99.92% less than D-PSGDbras. This communication reduction is achieved without sacrificing the convergence rate compared with the decentralized SGD with full precision gradients.

iii) `CiderTF` **is computationally efficient.** From the 1, 3 columns of Fig. 4.2, we observe that `CiderTF` is computationally efficient compared with GCP and D-PSGD. This is because the fiber sampling technique and the block randomization helps reduce the computational complexity, which also verifies the computational complexity analysis in Sec. 4.3.4. `CiderTF` is also slightly more efficient than BrasCPD mainly due to the scalability of the decentralized data distribution which helps parallelize the local tensor factorization.

iv) **Nesterov's momentum can offer** `CiderTF_m` **faster convergence, thus will lead to less overall communication cost**. From Fig. 4.2, we observe that `CiderTF_m` requires less epochs to converge, which in turn helps reduce the total

communication bytes with little sacrifice of the accuracy.

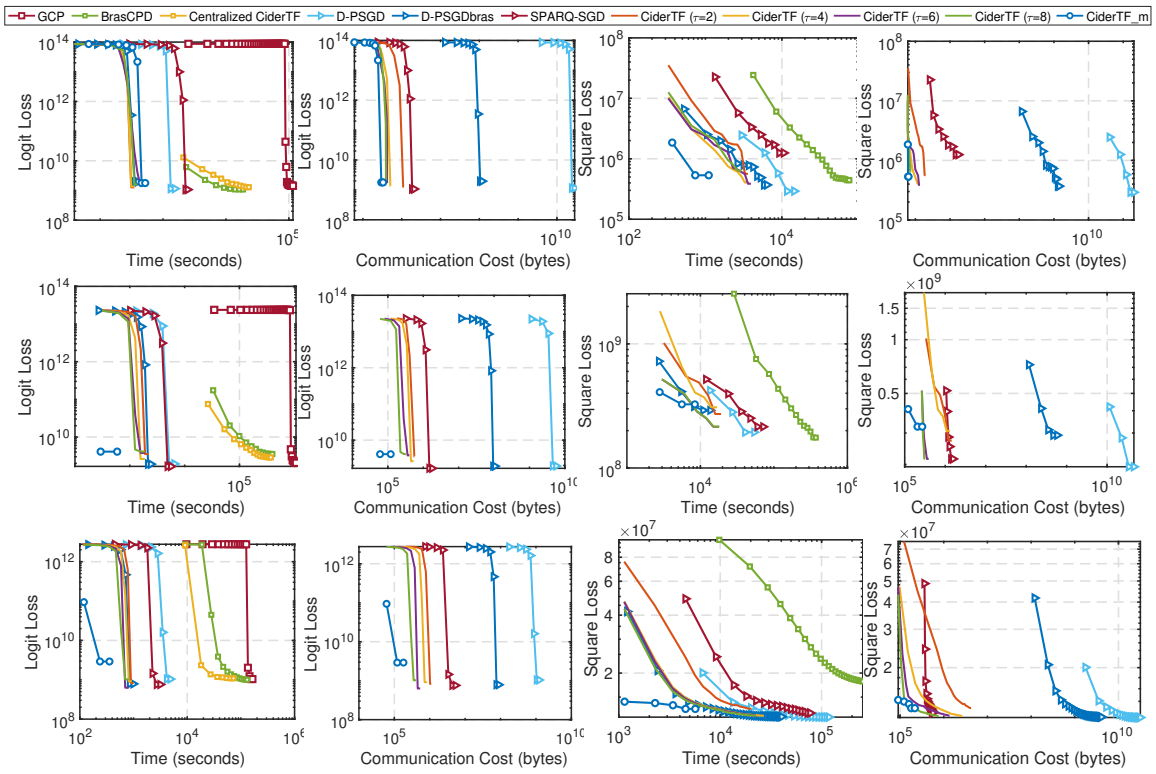



Figure 4.2: Bernoulli-logit Loss (1-2 columns) and Least Square Loss (3-4 columns) for ring topology with respect to time and communication. Centralized approaches are marked with square, decentralized baselines are marked in triangle, `CiderTF` with different number of local rounds are marked in solid lines. From top to bottom: CMS, MIMIC-III, Synthetic dataset.

### Impact of Topology

We also test `CiderTF` on two different topologies, including ring topology and the star topology for the same number of workers (shown in Fig. 4.1). From the results in Fig. 4.3, we observe that different topologies do not affect the convergence performance and both of them converge to similar losses for both Bernoulli-logit Loss and Least Square Loss, which means that `CiderTF` can generalize to different kinds of communication network topologies. Fig. 4.3 (left) illustrates that two topologies do not vary much in the computation time, since the number of workers are the same. However, star topology enjoys less communication cost because the total degree of the star topology

is less than the ring topology (Fig. 4.3 right).

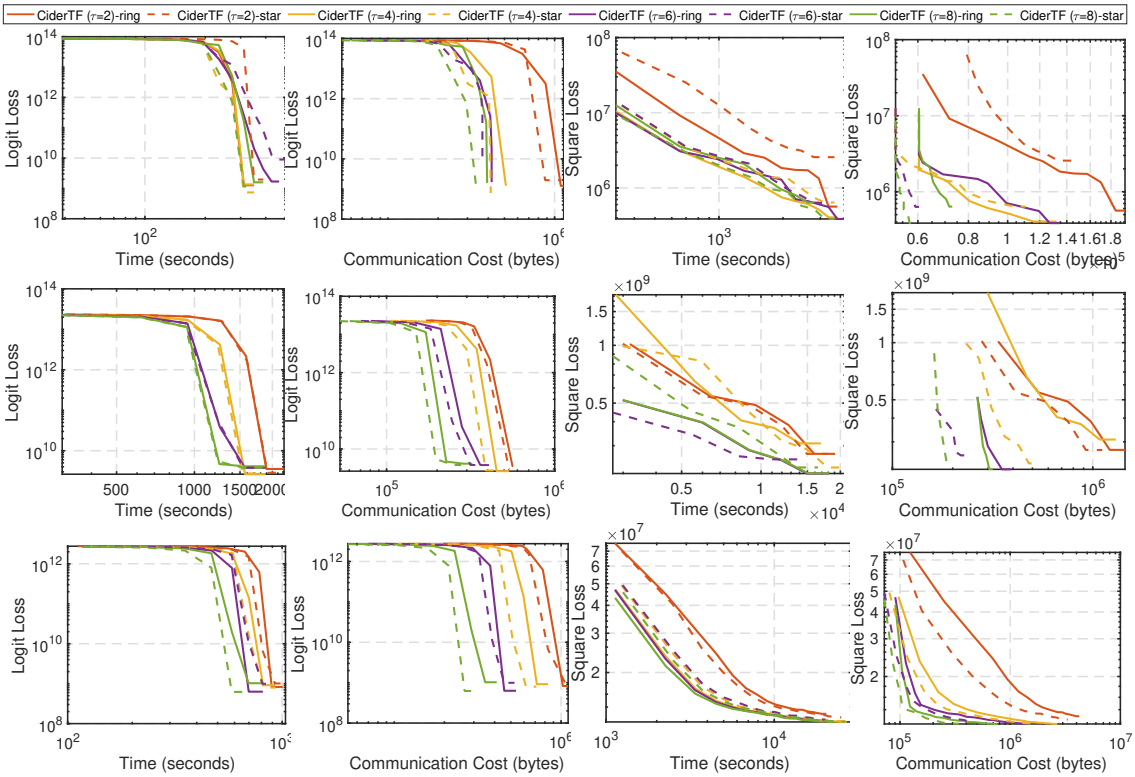



Figure 4.3: Bernoulli-logit Loss (1-2 columns) and Least Square Loss (3-4 columns) for ring topology (solid lines) and star topology (dashed lines) with respect to time and communication. From top to bottom: CMS, MIMIC-III, Synthetic dataset.

### Scalability

Moreover, we test the scalability of `CiderTF`. By increasing the number of clients from $K = 8$ to $K = 16$ and $K = 32$ involved in the computation, we can observe linear scalability in the computation time (Fig. 4.4 left) without sacrificing the accuracy. However, as the number of clients increases, the communication cost will increase accordingly (Fig. 4.4 right). Therefore, there exists a computation-communication trade-off when increasing the number of clients involved in the decentralized tensor factorization framework.

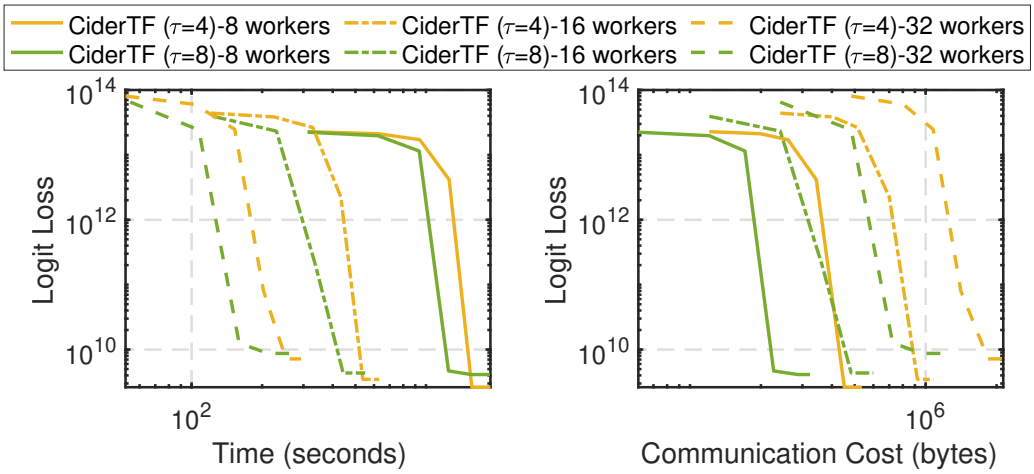



Figure 4.4: Bernoulli-logit loss with respect to time and communication for MIMIC-III data with 8, 16, and 32 workers for local update rounds $\tau = 4, 8$.

Table 4.2: Comparison of the decentralized optimization algorithms in ablation study.

| **Algorithm** | **Element-level** | **Block-level** | **Round-level** | **Event-driven** | **Compression Ratio** |
|---|---|---|---|---|---|
| D-PSGD | ✗ | ✗ | ✗ | ✗ | 0 |
| D-PSGDbras | ✗ | ✓ | ✗ | ✗ | $1 - 1/D$ |
| D-PSGD+signSGD | ✓ | ✗ | ✗ | ✗ | $1 - 1/32$ |
| D-PSGDbras+signSGD | ✓ | ✓ | ✗ | ✗ | $1 - 1/32D$ |
| SPARQ-SGD | ✓ | ✗ | ✓ | ✓ | $1 - 1/32\tau$ |
| `CiderTF` | ✓ | ✓ | ✓ | ✓ | $1 - 1/32D\tau$ |

### Ablation Study

We conduct an ablation study to clarify the reduction of the communication cost. We compare `CiderTF` with the following algorithms: 1) D-PSGD: decentralized SGD with full gradient and full block and single round communication; 2) D-PSGDbras: D-PSGD with block randomization; 3) D-PSGD+signSGD: D-PSGD with gradient compression using sign compressor; 4) D-PSGDbras+signSGD: D-PSGD with block randomization and gradient compression. The algorithm comparisons are summarized in Table 4.2.

From Fig. 4.5, we observe that D-PSGD which communicates the full gradient and blocks has the highest communication cost. Gradient compression plays the most important role in reducing the communication cost, which reduces the actual communication cost of MIMIC-III data of 96.88%. The block randomization further reduces the communication cost to 75.00%. The application of the event-driven and periodic communication helps reduce the communication with a lower bound of 87.5% and up to an upper bound to 97.22%.

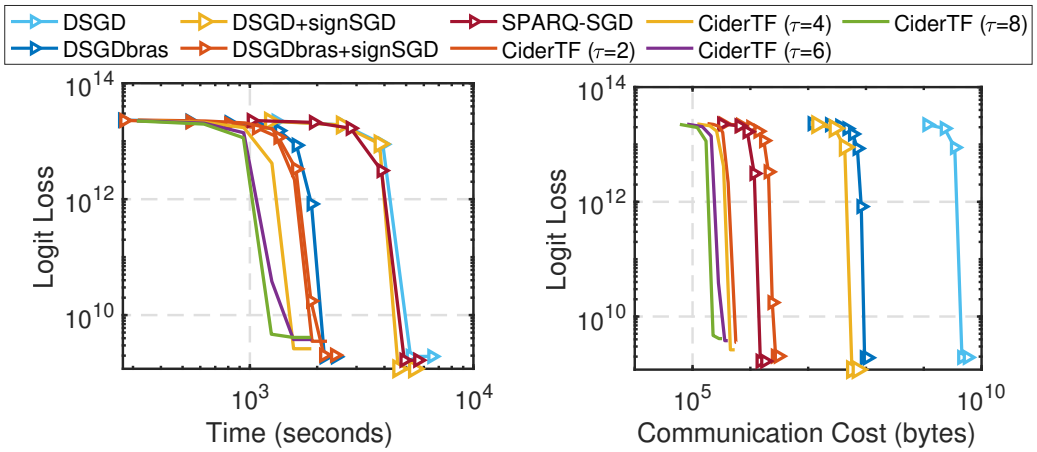



Figure 4.5: Ablation study for decentralized optimization algorithms with different levels of communication reduction.

**Comparison to Federated Tensor Factorization with Central Server**

In order to compare the decentralized method to the federated algorithm with the central server, we further conduct experiments to compare `CiderTF` under different topologies with the federated tensor factorization algorithm FedGTF-EF-PC with similar communication reduction techniques proposed in Chapter 3, while `CiderTF` employs the event-driven technique to further reduce the communication cost. Fig. 4.6 shows the training curve with respect to the number of epochs and communication cost. We can observe that `CiderTF` requires similar epochs (columns 1, 3) to converge compared with FedGTF-EF-PC. For Bernoulli-logit loss, FedGTF-EF-PC has less communication cost than `CiderTF` (column 2) because all the communication is doubled for the decentralized optimization networks. However, we can also observe that FedGTF-EF-PC has more communication cost for the square loss (column 4). This indicates that the event-driven method can reduce the communication cost, which offsets the communication caused by the decentralization.

### 4.4.3 Case Study on MIMIC-III

In order to validate the ability of extracting effective phenotypes using `CiderTF`, we conduct a case study on MIMIC-III dataset. We evaluate the extracted phenotypes from both quantitative and qualitative perspectives. From the quantitative aspect, we use the Factor Match Score (FMS) [2] to measure the similarity of the factor matrices of `CiderTF` with the centralized baseline BrasCPD. FMS ranges from 0 to 1 with the best possible value of 1.

Fig. 4.7 indicates that `CiderTF` achieves the highest FMS of the decentralized methods as it gradually approaches 1. This means that `CiderTF` can extract the factor matrices similar to its centralized counterpart BrasCPD. We can also observe that `CiderTF` approaches the centralized factors with the least time and communication cost.

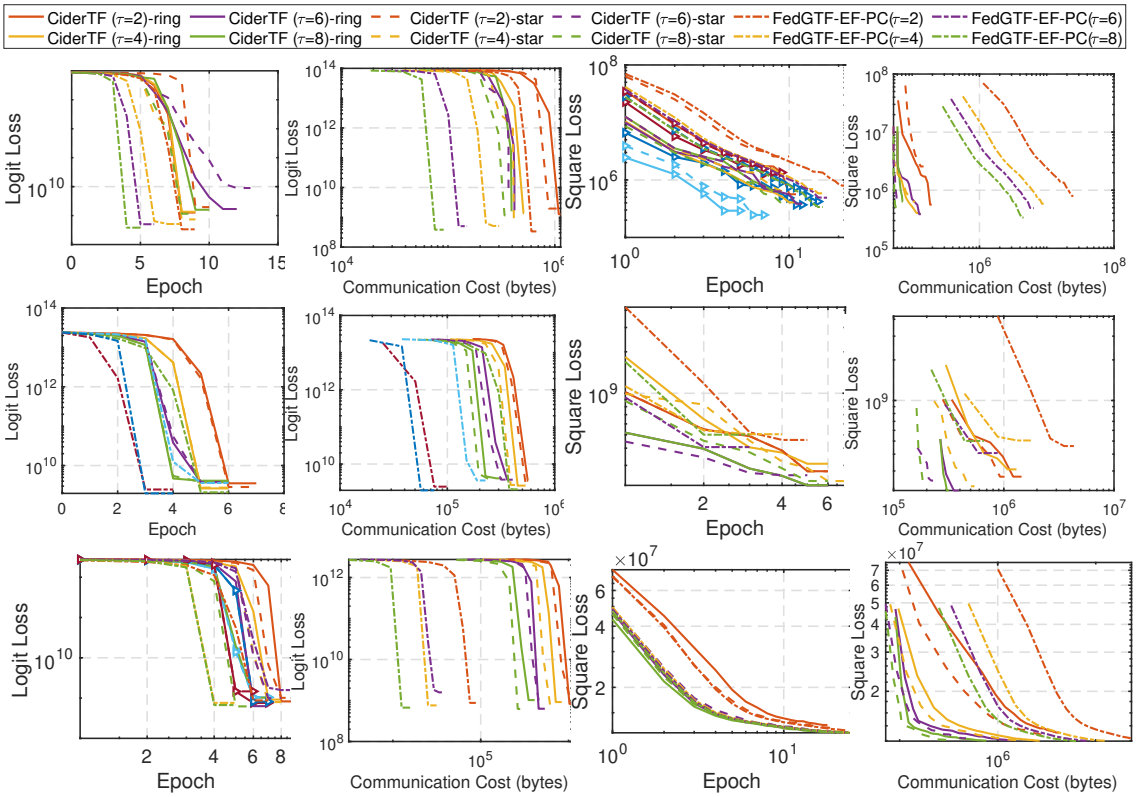



Figure 4.6: Bernoulli-logit Loss (1-2 columns) and Least Square Loss (3-4 columns) for CiderTF with ring topology (solid lines) and star topology (dashed lines), and FedGTF-EF-PC (dash-dotted lines) with respect to number of epochs and communication cost. From top to bottom: CMS, MIMIC-III, Synthetic dataset.

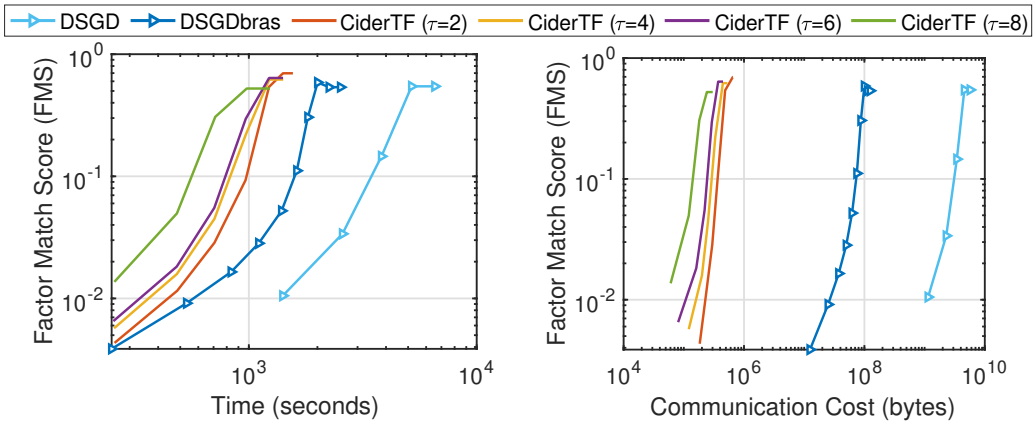



Figure 4.7: Factor Match Scores (FMS) with respect to time and communication.

Furthermore, from the the qualitative perspective, we evaluate the quality of the phenotypes by patient subgroup identification ability. Following the precedent set in [68], we first identify the top three phenotypes according to the phenotype importance factor $\lambda_r = \left\|\mathbf{A}_{(1)}(:,r)\right\|_F \left\|\mathbf{A}_{(2)}(:,r)\right\|_F \cdots \left\|\mathbf{A}_{(D)}(:,r)\right\|_F$. We then group the patients by assigning each according to the largest value among the top 3 along the patient representation vector, and use tSNE to map the patient representation into two-dimensional space. Fig. 4.3 shows that `CiderTF` ($\tau = 8$) achieves comparable patient subgroup identification ability as the centralized baseline BrasCPD. When compared to the decentralized baselines with the same communication cost (since 1 epoch of D-PSGD and D-PSGDbras already incurs more communication cost, we show the result after 1 epoch), `CiderTF` achieves better clustered subgroups, demonstrating that `CiderTF` is able to better identify patient subgroups. In addition, the top 3 phenotypes extracted by `CiderTF`, shown in Table 4.4, are clinically meaningful and interpretable as annotated by a pulmonary and critical care physician.

Table 4.3: tSNE visualization of the patient subgroup identification with the extracted phenotypes. Each point represents a patient which is colored according to the highest-valued coordinate in the patient representation vector among the top 3 phenotypes extracted based on the factor weights $\lambda_r = \left\|\mathbf{A}_{(1)}(:,r)\right\|_F \left\|\mathbf{A}_{(2)}(:,r)\right\|_F \cdots \left\|\mathbf{A}_{(D)}(:,r)\right\|_F$.

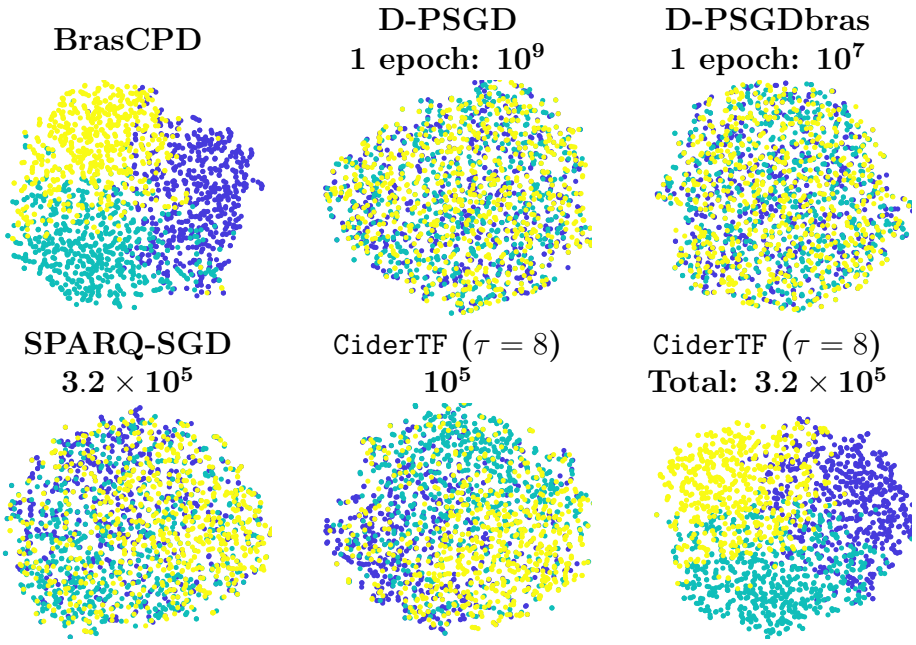


Table 4.4: phenptypes extracted by `CiderTF` ($\tau = 8$) on MIMIC-III data. Dx, Px, and Med indicate diagnoses, procedures, and medication.

| | |
|---|---|
| **P1:** | **Acute myocardial infarction** |
| Dx | Other and unspecified angina pectoris<br>Coronary atherosclerosis of autologous vein bypass graft<br>Old myocardial infarction |
| Px | (Aorto)coronary bypass of two coronary arteries<br>(Aorto)coronary bypass of three coronary arteries<br>Implant of pulsation balloon |
| Med | Diltiazem Hydrochloride Extended-Release<br>Metoprolol succinate, Rosuvastatin Calcium<br>Valsartan/hydrochlorothiazide, Losartan Potassium |
| **P2:** | **Respiratory failure** |
| Dx | Acute respiratory failure, Hypoxemia,<br>Contusion of lung without mention of<br>open wound into thorax<br>Disruption of internal operation (surgical) wound |
| Px | Non-invasive mechanical ventilation<br>Continuous invasive mechanical ventilation for less than<br>96 consecutive hours |
| Med | Dextrose, Albuminar-25, Plasmanate |
| **P3:** | **Intracranial hemorrhage or cerebral infarction** |
| Dx | Pure hypercholesterolemia, Subdural hemorrhage<br>Cerebral artery occlusion |
| Px | Injection or infusion of thrombolytic agent<br>Control of hemorrhage |
| Med | Ticagrelor, Atorvastatin Calcium |

# Chapter 5

# Conclusion and Future Work

## 5.1 Conclusion

In this dissertation, we investigate the problem of Federated Tensor Factorization for collaborative heath data analytics from multiple aspects.

From the aspect of privacy issue under the federated learning settings, we propose **DPFact**, which is able to successfully tackle the privacy issue under the distributed setting with limited privacy loss by the application of zCDP and the parallel composition theorem. Moreover, our model recognizes that the learned global latent factors need not be present at all sites, allowing the discovery of both shared and site-specific computational phenotypes. Furthermore, by adopting a communication-efficient EASGD algorithm, DPFact greatly reduces the communication overhead. Experiments on real-world and synthetic datasets demonstrate that our model outperforms other state-of-the-art methods in terms of communication cost, accuracy, and phenotype discovery ability.

From the aspect of communication and computation efficiency, we proposed **FedGTF-EF/FedGTF-EF-PC** with the flexibility of choosing different loss functions according to the data distribution. FedGTF-EF employs the communication

efficient designs of block randomized update and gradient compression with error-feedback, which encompassed two levels of uplink communication reduction: reduced number of blocks and reduced per-element communication. We further reduce the communication rounds by periodic averaging to develop the FedGTF-EF-PC algorithm. The convergence guarantee is provided under common assumptions applied not only to generalized tensor factorization problems but also to more general machine learning problems possessing a multi-block structure. Our algorithm can maintain low computational and storage complexity while occupying much lower uplink communication cost. We demonstrate its superior efficiency and uncompromised quality on synthetic and two real-world EHR datasets.

From the aspect of the topology of the healthcare network, we proposed **CiderTF**, which is the first decentralized generalized tensor factorization framework. It employs four levels of communication reduction with gradient compression, block randomization, and periodic communication combined with an event-driven communication strategy. Meanwhile, `CiderTF` enjoys low computational and memory complexity due to the two levels of randomization: random fiber sampling and random block selection. Experiments show that `CiderTF` preserves the quality of the extracted phenotypes and converges to similar points as the decentralized SGD baselines with theoretical guarantees.

## 5.2 Future Work

The proposed methods can be further extended from the following directions:

**Federated Tensor Factorization with byzantine robustness.** Under the federated learning setting of FedGTF-EF/FedGTF-EF-PC for healthcare data analytics, the clients within the system would be susceptible to malicious attacks [65, 97, 28], resulting in the attacked clients sending arbitrary information to the server. This

would lead to the failure of the federated learning system (e.g. bad convergence, dissatisfactory performance, etc.), which is modeled as the Byzantine failure. It is important that we develop a Byzantine-robust mechanism to guarantee the convergence of the FTF framework.

**Federated Tensor Factorization with heterogeneous data distributions.** We have proposed the efficient Federated Tensor Factorization frameworks (FedGTF-EF/FedGTF-EF-PC and CiderTF) for the topologies with and without a central server, where data is assumed to be horizontally distributed among multiple medical institutions. However, this assumption is somewhat narrow for the real-world scenarios, where data from different medical institutions may follow different distributions [69]. With the flexibility of the generalized TF, each client with different data distribution will be able to employ different loss functions.

**Asynchronization of FedGTF-EF/FedGTF-EF-PC and CiderTF.** The training of FedGTF-EF/FedGTF-EF-PC and CiderTF involves a periodic synchronization step to update the global (averaged) model with the compressed updates to achieve consensus. However, different clients (medical institutions) may have access to different computation resources, meaning that some of the clients with sufficient computation power can finish the computation faster than those with restricted computation power, who can be recognized as stragglers [12, 52]. Current training of FedGTF-EF/FedGTF-EF-PC and CiderTF requires all clients to be available for each synchronization step, otherwise, the system has to wait for the stragglers. It would be important to investigate the asynchronization of FedGTF-EF/FedGTF-EF-PC and CiderTF to guarantee the convergence and reduce the idle training time.

# Appendix A

# DPFact

This section provides supplementary information.

## A.1 *Lipschitz* Constant

Below is the calculation of the *Lipschitz* constant in objective function (2.15), which will be used for calculating $\beta$ in section 2.4.2. Since $\mathbf{B}^{[t]}$ and $\mathbf{C}^{[t]}$ play the same role in the objective function in (2.15), we take $\mathbf{B}^{[t]}$ as an example. $f(\mathbf{B}^{[t]})$ can be rewritten as

$$f(\mathbf{B}^{[t]}) = \underbrace{\frac{1}{2}\left\|O_{(n)}^{[t]} - \mathbf{B}^{[t]}\mathbf{\Pi}_B\right\|_F^2}_{\mathcal{F}_B} + \underbrace{\frac{\gamma}{2}\left\|\mathbf{B}^{[t]} - \mathbf{B}\right\|_F^2}_{\mathcal{Q}_B}, \tag{A.1}$$

where $\mathcal{O}_{(n)}^{[t]}$ is the matricization of the local observed tensor $\mathcal{O}^{[t]}$, $\mathbf{\Pi}_B = \mathbf{C}^{[t]} \odot \mathbf{A}^{[t]}$. Thus $f(\mathbf{B}^{[t]})$ is the combination of $\mathcal{F}_B$ and $\mathcal{Q}_B$. We provide analysis of *Lipschitz* continuity of (A.1) separately for $\mathcal{F}_B$ and $\mathcal{Q}_B$. The analysis for $\mathcal{F}_B$ could also be adopted into equation (2.9) in section 2.4.1.

The gradient of $\mathcal{F}_B(\mathbf{B}^{[t]})$ is calculated as $\nabla\mathcal{F}_B(\mathbf{B}^{[t]}) = -O_{(n)}^{[t]}\mathbf{\Pi}_B + \mathbf{B}^{[t]}\mathbf{\Gamma}_B$, where

$\mathbf{\Gamma}_B = (\mathbf{A}^{[t]\top}\mathbf{A}^{[t]}) * (\mathbf{C}^{[t]\top}\mathbf{C}^{[t]})$. Furthermore, for any $\mathbf{B}_1^{[t]}, \mathbf{B}_2^{[t]} \in R_+^{j\times n}$, we have

$$\begin{aligned}\left\|\nabla\mathcal{F}_B(\mathbf{B}_1^{[t]}) - \nabla\mathcal{F}_B(\mathbf{B}_2^{[t]})\right\|_F &= \left\|\mathbf{B}_1^{[t]}\mathbf{\Gamma}_B - \mathbf{B}^{[t]}{}_2\mathbf{\Gamma}_B\right\|_F \\ &= \left\|(\mathbf{B}_1^{[t]} - \mathbf{B}_2^{[t]})\mathbf{\Gamma}_B\right\|_F \\ &\leq \|\mathbf{\Gamma}_B\|_F \left\|\mathbf{B}_1^{[t]} - \mathbf{B}_2^{[t]}\right\|_F.\end{aligned} \tag{A.2}$$

The gradient of $\mathcal{Q}_B$ is calculated as $\nabla\mathcal{Q}_B(\mathbf{B}^{[t]}) = \gamma\mathbf{B}^{[t]}$. Similar to $\nabla\mathcal{F}_B(\mathbf{B}^{[t]})$, for any $\mathbf{B}_1^{[t]}, \mathbf{B}_2^{[t]} \in R_+^{j\times n}$, we have

$$\begin{aligned}\left\|\nabla\mathcal{F}_B(\mathbf{B}_1^{[t]}) - \nabla\mathcal{F}_B(\mathbf{B}_2^{[t]})\right\|_F &= \left\|\gamma\mathbf{B}_1^{[t]} - \gamma\mathbf{B}_2^{[t]}\right\|_F \\ &= \left\|\gamma I_j(\mathbf{B}_1^{[t]} - \mathbf{B}_2^{[t]})\right\|_F \\ &\leq \|\gamma I_j\|_F \left\|\mathbf{B}_1^{[t]} - \mathbf{B}_2^{[t]}\right\|_F\end{aligned} \tag{A.3}$$

By combining the results in (A.2) and (A.3), we thus get the *Lipschitz* constant of $\nabla f(\mathbf{B}^{[t]})$ as the Frobenius norm of

$$(\mathbf{A}^{[t]\top}\mathbf{A}^{[t]}) * (\mathbf{C}^{[t]\top}\mathbf{C}^{[t]}) + \gamma I_j.$$

## A.2 $L_{2,1}$ Regularization Parameter

Figure A.1 illustrates the effect of choosing different value of $\mu$ on the column norm of the patient matrix for each ICUs in MIMIC-III dataset. We observe that smaller $\mu$ has minimal effect on the column sparsity, as there are no columns that are set to 0. However, if we set $\mu$ to be too high (i.e., $\mu = \{5, 6, 8, 6, 5, 0.9\}$ for each ICU respectively), then it "turns off" a large portion of the factors and prevents DPFact from generating useful phenotypes. Based on the figure, we choose $\mu = \{1, 1.8, 3.2, 1.8, 1.5, 0.6\}$ for TSICU, SICU, MICU, CSRU, CCU, NICU respectively, as the optimal solution for MIMIC-III as there are still noticeable differences in the column magnitude (i.e, the

phenotypes have a natural ordering within each location) but also provides flexibility to have at least one unshared column (see component 2 and 4).

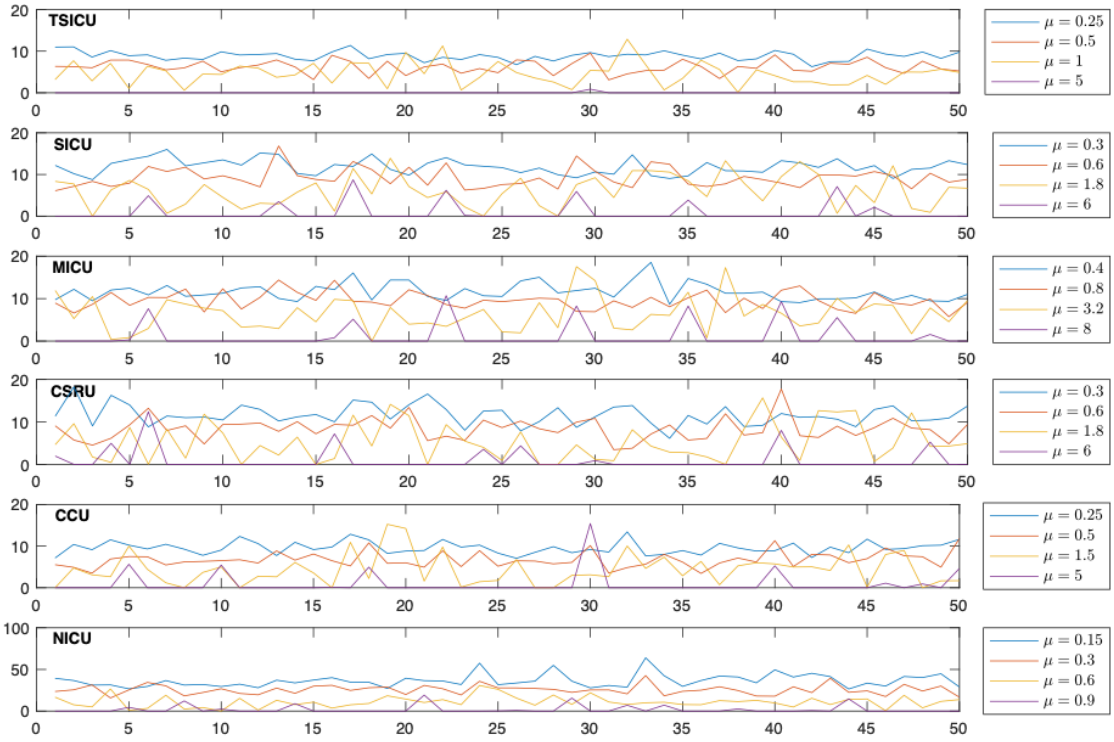



Figure A.1: Norm of each ICU with different regularization term $\mu$ (y-axis is the norm, x-axis is the rank)

## A.3 Phenotype Selection

Table A.1 provides information in supplementary for phenotype selection of the overall pattern. Similar to table 2.4, among the 50 phenotypes generated by DPFact, we selected 20 phenotypes that are statistically significant for mortality prediction. We reported 6 representative phenotypes which has the highest factor weights ($\lambda$).

| **Phenotypes** | **Coef** | **p-value** | $\lambda$ | **Prevalence** |
|---|---|---|---|---|
| 2 | 1.46 | <0.001 | 163 | 21.67 |
| 3: Hypertension | -1.53 | <0.001 | 249 | 20.57 |
| 6 | -1.19 | <0.001 | 145 | 23.53 |
| 7 | -2.34 | <0.001 | 49 | 18.17 |
| 8 | 2.89 | <0.001 | 162 | 24.31 |
| 9 | 2.88 | <0.001 | 116 | 22.81 |
| 14 | -1.72 | <0.001 | 94 | 18.25 |
| 16 | 2.29 | <0.001 | 79 | 17.48 |
| 17 | 4.21 | <0.001 | 166 | 21.96 |
| 18 | 3.66 | <0.001 | 69 | 16.79 |
| 20 | -1.51 | <0.001 | 95 | 20.72 |
| 22 | 2.17 | <0.001 | 137 | 19.19 |
| 25: Heart failure | -6.56 | <0.001 | 278 | 16.40 |
| 30: Acute kidney injury | 0.46 | <0.001 | 203 | 26.35 |
| 32 | 1.65 | <0.001 | 109 | 18.73 |
| 37 | 1.42 | <0.001 | 116 | 22.50 |
| 42: Gastritis and gastroduodenitis | -2.24 | <0.001 | 187 | 22.88 |
| 47: Cardiac surgery | -2.94 | <0.001 | 223 | 16.55 |
| 49 | 2.91 | <0.001 | 113 | 22.73 |
| 50: Chronic ischemic heart disease | 1.38 | <0.001 | 207 | 27.45 |

Table A.1: Logistic regression results for phenotype selection

| **# of Sites** | **MIMIC-III** | **CMS** | **Synthetic** |
|---|---|---|---|
| 1 | 18.73 | 22.89 | 1.55 |
| 5 | 93.62 | 114.42 | 7.75 |
| 10 | 189.83 | 228.83 | 15.50 |

Table A.2: Communication cost of DPFact for different number of sites (Seconds)

# Appendix B

# FedGTF-EF/FedGTF-EF-PC

## B.1 Additional Materials for Experiments

### B.1.1 Parameter Settings

For MIMIC-III, CMS and synthetic datasets, each algorithm is run for 500 iterations per epoch until converge, while for delicious dataset, each algorithm is run for 1000 iterations per epoch. For GCP algorithm, we tune the stepsize within the range of $\{10^{-8}, 10^{-9}, 10^{-10}, 10^{-11}\}$, while for the rest algorithms, we tune the stepsize by grid search through $\{2^2, 2^1, 2^0, 2^{-1}, 2^{-2}, ..., 2^{-11}\}$. The parameter for the proximal operator is set to $10^{-4}$ for all the algorithms with the proximal operators (FedGTF-EF-prox, DPFact-prox). For all the federated algorithms, we by default horizontally partition the tensor (along $I_1$ mode) into 8 tensors without overlapping and distribute each of them to 8 client nodes respectively. We also test different numbers of workers (16 workers and 32 workers), where the stepsizes are set to the same as for 8 workers. The best stepsizes for each algorithms for different datasets are set as in Table B.1 and B.2.

Each experiment is averaged over 5 repetitions. All experiments are run on Matlab 2019a on an `r5.12xlarge` instance of AWS EC2 with Tensor Toolbox Version 3.1 [4].

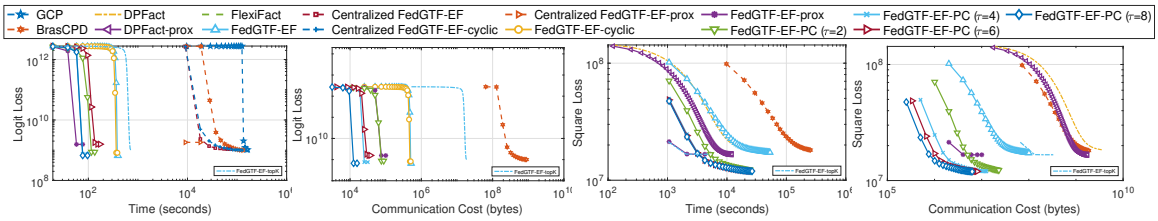



Figure B.1: Bernoulli Logit Loss and Square Loss with respect to computation time and communication for synthetic data.

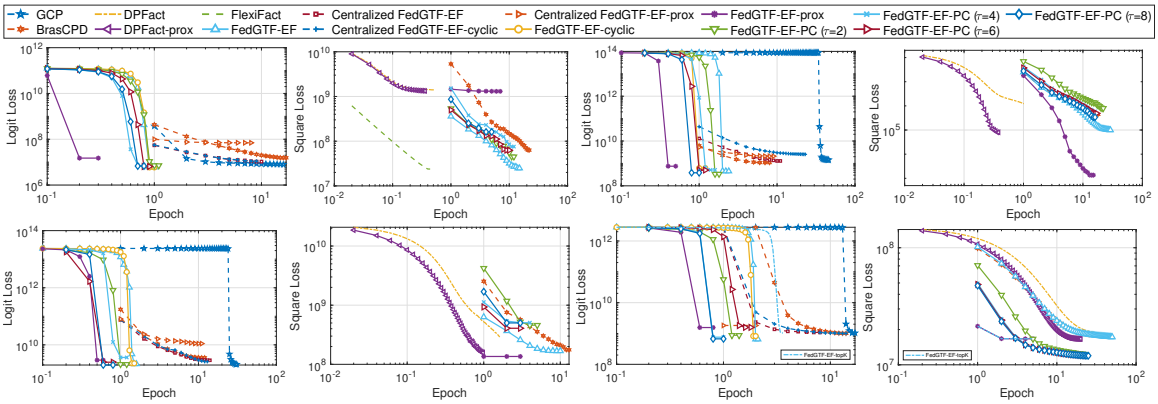



Figure B.2: Bernoulli logit loss (column 1,2) and Least Square loss (column 3,4) decrease with respect to epochs.

### B.1.2 Additional Experiments

Two additional groups of figures are presented here. Fig.B.1 shows the loss decrease for both the Bernoulli loss and the Least Square loss with respect to time and communication for the synthetic data. Fig.B.2 shows the Bernoulli loss and the Least Square loss decrease with respect to epochs in supplementary to the figures showed in the main paper with respects to time and communication. Similar conclusions can be drawn with the real-world EHR datasets in the main paper. That is, the proposed

Table B.1: Best Stepsizes for the Bernoulli Logit Loss

| **Algorithm** | **MIMIC-III** | **4th order CMS** | **3rd order CMS** | **Synthetic** |
|---|---|---|---|---|
| GCP | $10^{-10}$ | $10^{-10}$ | $10^{-10}$ | $10^{-9}$ |
| BrasCPD | $2^{-4}$ | $2^{-1}$ | $2^{-4}$ | $2^{-5}$ |
| Centralized FedGTF-EF | $2^{-3}$ | $2^{-1}$ | $2^{-2}$ | $2^{-4}$ |
| Centralized FedGTF-EF-cyclic | $2^{-2}$ | $2^{-2}$ | $2^{-2}$ | $2^{-4}$ |
| Centralized FedGTF-EF-prox | $2^{-2}$ | $2^{-0}$ | $2^{-2}$ | $2^{-2}$ |
| FedGTF-EF | $2^{-3}$ | $2^{-2}$ | $2^{-2}$ | $2^{-4}$ |
| FedGTF-EF-cyclic | $2^{-4}$ | $2^{-2}$ | $2^{-2}$ | $2^{-4}$ |
| FedGTF-EF-prox | $2^{-2}$ | $2^{-3}$ | $2^{-4}$ | $2^{-1}$ |
| FedGTF-EF-PC($\tau = 2$) | $2^{-5}$ | $2^{-5}$ | $2^{-2}$ | $2^{-4}$ |
| FedGTF-EF-PC($\tau = 4$) | $2^{-5}$ | $2^{-5}$ | $2^{-2}$ | $2^{-4}$ |
| FedGTF-EF-PC($\tau = 6$) | $2^{-5}$ | $2^{-5}$ | $2^{-2}$ | $2^{-4}$ |
| FedGTF-EF-PC($\tau = 8$) | $2^{-5}$ | $2^{-5}$ | $2^{-2}$ | $2^{-4}$ |

Table B.2: Best Stepsizes for the Least Square Loss

| **Algorithm** | **MIMIC-III** | **4-th order CMS** | **3-rd order CMS** | **Synthetic** |
|---|---|---|---|---|
| BrasCPD | $2^{-5}$ | $2^{0}$ | $10^{-4}$ | $2^{-2}$ |
| FlexiFact | - | - | 2 | - |
| DPFact | $2^{-4}$ | $2^{1}$ | $2^{-10}$ | $2^{-2}$ |
| DPFact-prox | $2^{-4}$ | $2^{1}$ | $2^{-10}$ | $2^{-2}$ |
| FedGTF-EF | $2^{-4}$ | $2^{0}$ | $2^{-11}$ | $2^{-2}$ |
| FedGTF-EF-prox | $2^{-5}$ | $2^{0}$ | $2^{-10}$ | $2^{-2}$ |
| FedGTF-EF-PC($\tau = 2$) | $2^{-4}$ | $2^{0}$ | $2^{-10}$ | $2^{-2}$ |
| FedGTF-EF-PC($\tau = 4$) | $2^{-4}$ | $2^{0}$ | $2^{-10}$ | $2^{-2}$ |
| FedGTF-EF-PC($\tau = 6$) | $2^{-4}$ | $2^{0}$ | $2^{-10}$ | $2^{-2}$ |
| FedGTF-EF-PC($\tau = 8$) | $2^{-4}$ | $2^{0}$ | $2^{-10}$ | $2^{-2}$ |

algorithms achieve more efficient convergence than the centralized baselines under the Bernoulli logit loss and the distributed baseline under the least square loss. It is also more communication-efficient than the algorithms without gradient compressor (BrasCPD distributed version) and without the block randomized mechanism (DPFact and its variants).

## B.2 Additional Materials for Convergence Analysis of Algorithm 1

### B.2.1 Proof of Theorem 4.1

For the smooth case, the proof follows [43] to regard the iteration with error feedback as SGD with delayed variable. The new development is to allow randomized block-wise update.

#### Auxiliary variables for the proof and iterative relation

The following auxillary variables and virtual iterations are introduced only for the proof.

$$\widetilde{\mathbf{A}}_{(d)}[t] := \mathbf{A}_{(d)}[t] - \frac{1}{K}\sum_{k=1}^{K} \mathbf{E}_{(d)}^{k}[t]. \tag{B.1}$$

Given the auxiliary variable $\widetilde{\mathbf{A}}_{(d)}[t]$, we have the following iterative relation: if

$d = d_{\xi[t]}$,

$$\begin{aligned}
\widetilde{\mathbf{A}}_{(d)}[t+1] &= \mathbf{A}_{(d)}[t+1] - \frac{1}{K}\sum_{k=1}^{K}\mathbf{E}_{(d)}^{k}[t+1] \\
&= \Big(\mathbf{A}_{(d)}[t] - \frac{1}{K}\sum_{k=1}^{K}\boldsymbol{\Delta}_{(d)}^{k}[t]\Big) - \frac{1}{K}\sum_{k=1}^{K}\mathbf{E}_{(d)}^{k}[t+1] \\
&= \mathbf{A}_{(d)}[t] - \frac{1}{K}\sum_{k=1}^{K}\mathbf{P}_{(d)}^{k}[t] \\
&= \mathbf{A}_{(d)}[t] - \Big(\frac{1}{K}\sum_{k=1}^{K}\mathbf{E}_{(d)}^{k}[t] + \gamma[t]\frac{1}{K}\sum_{k=1}^{K}\mathbf{G}_{(d)}^{k}[t]\Big) \\
&= \widetilde{\mathbf{A}}_{(d)}[t] - \gamma[t]\frac{1}{K}\sum_{k=1}^{K}\mathbf{G}_{(d)}^{k}[t];
\end{aligned} \tag{B.2}$$

else if $d \neq d_{\xi[t]}$,

$$\widetilde{\mathbf{A}}_{(d)}[t+1] = \mathbf{A}_{(d)}[t+1] - \frac{1}{K}\sum_{k=1}^{K}\mathbf{E}_{(d)}^{k}[t+1] = \mathbf{A}_{(d)}[t] - \frac{1}{K}\sum_{k=1}^{K}\mathbf{E}_{(d)}^{k}[t] = \widetilde{\mathbf{A}}_{(d)}[t]. \tag{B.3}$$

**Additional Lemma**

The following lemma extends Lemma 3 in [43] to our block randomized case.

**Lemma B.2.1.** *(Bounding the expectation of the block-wise feedback error averaged among client nodes) For $d = 1, ..., D$ and for $t = 0, ..., T$, assuming constant step size $\gamma[t] = \gamma$, we have*

$$\mathbb{E}\Big[\|\frac{1}{K}\sum_{k=1}^{K}\mathbf{E}_{(d)}^{k}[t+1]\|_F^2\Big] \leq \frac{4(1-\delta)}{\delta^2}\gamma^2(\sigma_d^2 + \omega_d^2). \tag{B.4}$$

*Proof.*

$$
\begin{aligned}
&\mathbb{E}\Big[\|\frac{1}{K}\sum_{k=1}^{K}\mathbf{E}_{(d)}^{k}[t+1]\|_F^2\Big] \leq \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\Big[\|\mathbf{E}_{(d)}^{k}[t+1]\|_F^2\Big] \\
&= \frac{1}{K}\sum_{k=1}^{K}\frac{1}{D}\mathbb{E}\Big[\|\mathbf{P}_{(d)}^{k}[t]-\texttt{Compress}(\mathbf{P}_{(d)}^{k}[t])\|_F^2\Big]+(1-\frac{1}{D})\mathbb{E}\Big[\|\mathbf{E}_{(d)}^{k}[t]\|_F^2\Big] \\
&\leq \frac{1}{K}\sum_{k=1}^{K}\frac{1}{D}(1-\delta)\mathbb{E}\Big[\|\mathbf{P}_{(d)}^{k}[t]\|_F^2\Big]+(1-\frac{1}{D})\mathbb{E}\Big[\|\mathbf{E}_{(d)}^{k}[t]\|_F^2\Big] \\
&= \frac{1}{K}\sum_{k=1}^{K}\frac{1}{D}(1-\delta)\mathbb{E}\Big[\|\mathbf{E}_{(d)}^{k}[t]+\gamma[t]\mathbf{G}_{(d)}^{k}[t]\|_F^2\Big]+(1-\frac{1}{D})\mathbb{E}\Big[\|\mathbf{E}_{(d)}^{k}[t]\|_F^2\Big] \\
&\leq \frac{1}{K}\sum_{k=1}^{K}\frac{1}{D}(1-\delta)(1+\eta)\mathbb{E}\Big[\|\mathbf{E}_{(d)}^{k}[t]\|_F^2\Big]+(\gamma[t])^2\frac{1}{D}(1-\delta)(1+1/\eta)\mathbb{E}\Big[\|\mathbf{G}_{(d)}^{k}[t]\|_F^2\Big] \\
&+(1-\frac{1}{D})\mathbb{E}\Big[\|\mathbf{E}_{(d)}^{k}[t]\|_F^2\Big] \\
&= \frac{1}{K}\sum_{k=1}^{K}\Big(\frac{1}{D}(1-\delta)(1+\eta)+(1-\frac{1}{D})\Big)\mathbb{E}\Big[\|\mathbf{E}_{(d)}^{k}[t]\|_F^2\Big]+(\gamma[t])^2\frac{1}{D}(1-\delta)(1+1/\eta)\mathbb{E}\Big[\|\mathbf{G}_{(d)}^{k}[t]\|_F^2\Big].
\end{aligned}
\tag{B.5}
$$

Unrolling the above recursive relation and let $\gamma[t]=\gamma$, we have

$$
\begin{aligned}
&\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\Big[\|\mathbf{E}_{(d)}^{k}[t+1]\|_F^2\Big] \\
&\leq \frac{1}{K}\sum_{k=1}^{K}\Big(\frac{1}{D}(1-\delta)(1+\eta)+(1-\frac{1}{D})\Big)^{t-i}(\gamma[i])^2\Big(\frac{1}{D}(1-\delta)(1+1/\eta)\mathbb{E}\Big[\|\mathbf{G}_{(d)}^{k}[i]\|_F^2\Big]\Big)
\end{aligned}
\tag{B.6}
$$

Let $\gamma[t]=\gamma$ and $\mathbb{E}\Big[\|\mathbf{G}^k_{(d)}[i]\|_F^2\Big]\leq \sigma_d^2+\omega_d^2$. We have

$$\begin{aligned}
&\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\Big[\|\mathbf{E}^k_{(d)}[t+1]\|_F^2\Big]\leq \frac{1}{K}\sum_{k=1}^{K}\sum_{i=0}^{t}\Big(\frac{1}{D}(1-\delta)(1+\eta)+(1-\frac{1}{D})\Big)^{t-i}\\
&\cdot\Big(\gamma^2\frac{1}{D}(1-\delta)(1+1/\eta)(\sigma_d^2+\omega_d^2)\Big)\\
&\leq \frac{1}{K}\sum_{k=1}^{K}\sum_{i=0}^{+\infty}\Big(\frac{1}{D}(1-\delta)(1+\eta)+(1-\frac{1}{D})\Big)^{t-i}\Big(\gamma^2\frac{1}{D}(1-\delta)(1+1/\eta)(\sigma_d^2+\omega_d^2)\Big) \qquad \text{(B.7)}\\
&\leq \frac{(1-\delta)(1+1/\eta)}{1-[\frac{1}{D}(1-\delta)(1+\eta)+(1-\frac{1}{D})]}\gamma^2\frac{1}{D}(\sigma_d^2+\omega_d^2)\\
&=\frac{(1-\delta)(1+1/\eta)}{1-(1-\delta)(1+\eta)}\gamma^2(\sigma_d^2+\omega_d^2).
\end{aligned}$$

In the above, to satisfy $0<(1-\delta)(1+\eta)<1$, let $\eta=\frac{\delta}{2(1-\delta)}$ and $1+\frac{1}{\eta}=\frac{2-\delta}{\delta}\leq\frac{2}{\delta}$, we have

$$\mathbb{E}\Big[\|\frac{1}{K}\sum_{k=1}^{K}\mathbf{E}^k_{(d)}[t+1]\|_F^2\Big]\leq\frac{4(1-\delta)}{\delta^2}\gamma^2(\sigma_d^2+\omega_d^2). \qquad \text{(B.8)}$$

□

**Main proof of Theorem 4.1**

By block-wise Lipschitz smoothness assumption of the loss function:

$$\begin{aligned}
F(\widetilde{\mathbf{A}}[t+1])&\leq F(\widetilde{\mathbf{A}}[t])+\langle\nabla_{\mathbf{A}_{(d_\xi[t])}}F(\widetilde{\mathbf{A}}[t]),\widetilde{\mathbf{A}}_{(d_\xi[t])}[t+1]-\widetilde{\mathbf{A}}_{(d_\xi[t])}[t]\rangle\\
&\quad+\frac{L_{d_\xi[t]}}{2}\|\widetilde{\mathbf{A}}_{(d_\xi[t])}[t+1]-\widetilde{\mathbf{A}}_{(d_\xi[t])}[t]\|_F^2\\
&=F(\widetilde{\mathbf{A}}[t])-\gamma[t]\langle\nabla_{\mathbf{A}_{(d_\xi[t])}}F(\widetilde{\mathbf{A}}[t]),\frac{1}{K}\sum_{k=1}^{K}\mathbf{G}^k_{(d_\xi[t])}[t]\rangle+\frac{L_{d_\xi[t]}(\gamma[t])^2}{2}\|\frac{1}{K}\sum_{k=1}^{K}\mathbf{G}^k_{(d_\xi[t])}[t]\|_F^2.
\end{aligned} \qquad \text{(B.9)}$$

By Assumption 4.2 that $\mathbb{E}_{\zeta[t]}\Big[\frac{1}{K}\sum_{k=1}^{K}\mathbf{G}^k_{d_{\xi[t]}}[t]\Big|\mathcal{F}[t],\xi[t]\Big]=\nabla_{\mathbf{A}_{(d_{\xi[t]})}}F(\mathbf{A}[t])$, we

have

$$\mathbb{E}_{\zeta[t]}\Big[\|\frac{1}{K}\sum_{k=1}^{K}\mathbf{G}^k_{(d_\xi[t])}[t]-\nabla_{\mathbf{A}_{(d_{\xi[t]})}}F(\mathbf{A}[t])\|_F^2\Big|\mathcal{F}[t],\xi[t]\Big]=\mathbb{E}_{\zeta[t]}\Big[\|\frac{1}{K}\sum_{k=1}^{K}\mathbf{G}^k_{(d_\xi[t])}[t]\|_F^2\Big|\mathcal{F}[t],\xi[t]\Big]$$
$$-\|\nabla_{\mathbf{A}_{(d_{\xi[t]})}}F(\mathbf{A}[t])\|_F^2. \tag{B.10}$$

Taking conditional expectation on both sides of eq.(B.9) with respect to filtration $\mathcal{F}[t]$ and randomness of $\zeta[t]$ during the stochastic gradient computation and plugging eq.(B.10) in, we have

$$\begin{aligned}
&\mathbb{E}_{\zeta[t]}\Big[F(\widetilde{\mathbf{A}}[t+1])\Big|\mathcal{F}[t],\xi[t]\Big]\leq F(\widetilde{\mathbf{A}}[t])-\gamma[t]\mathbb{E}_{\zeta[t]}\Big[\langle\nabla_{\mathbf{A}_{(d_{\xi[t]})}}F(\widetilde{\mathbf{A}}[t]),\frac{1}{K}\sum_{k=1}^{K}\mathbf{G}^k_{(d_\xi[t])}[t]\rangle\Big|\mathcal{F}[t],\xi[t]\Big]\\
&\quad+\frac{L_{d_\xi[t]}(\gamma[t])^2}{2}\mathbb{E}_{\zeta[t]}\Big[\|\frac{1}{K}\sum_{k=1}^{K}\mathbf{G}^k_{(d_\xi[t])}[t]-\nabla_{\mathbf{A}_{(d_{\xi[t]})}}F(\mathbf{A}[t])\|_F^2\Big|\mathcal{F}[t],\xi[t]\Big]\\
&\quad+\frac{L_{d_\xi[t]}(\gamma[t])^2}{2}\|\nabla_{\mathbf{A}_{(d_{\xi[t]})}}F(\mathbf{A}[t])\|_F^2\\
&\leq F(\widetilde{\mathbf{A}}[t])-\gamma[t]\langle\nabla_{\mathbf{A}_{(d_{\xi[t]})}}F(\widetilde{\mathbf{A}}[t]),\nabla_{\mathbf{A}_{(d_{\xi[t]})}}F(\mathbf{A}[t])\rangle\\
&\quad+\frac{L_{d_\xi[t]}(\gamma[t])^2}{2}\|\nabla_{\mathbf{A}_{(d_{\xi[t]})}}F(\mathbf{A}[t])\|_F^2+\frac{L_{d_\xi[t]}(\gamma[t])^2}{2K}\sigma^2_{d_\xi[t]}\\
&=F(\widetilde{\mathbf{A}}[t])-\gamma[t]\langle\nabla_{\mathbf{A}_{(d_{\xi[t]})}}F(\mathbf{A}[t]),\nabla_{\mathbf{A}_{(d_{\xi[t]})}}F(\mathbf{A}[t])\rangle\\
&\quad+\gamma[t]\langle\nabla_{\mathbf{A}_{(d_{\xi[t]})}}F(\mathbf{A}[t])-\nabla_{\mathbf{A}_{(d_{\xi[t]})}}F(\widetilde{\mathbf{A}}[t]),\nabla_{\mathbf{A}_{(d_{\xi[t]})}}F(\mathbf{A}[t])\rangle\\
&\quad+\frac{L_{d_\xi[t]}(\gamma[t])^2}{2}\|\nabla_{\mathbf{A}_{(d_{\xi[t]})}}F(\mathbf{A}[t])\|_F^2+\frac{L_{d_\xi[t]}(\gamma[t])^2}{2K}\sigma^2_{d_\xi[t]}\\
&=F(\widetilde{\mathbf{A}}[t])-\gamma[t]\Big(1-\frac{L_{d_\xi[t]}\gamma[t]}{2}\Big)\|\nabla_{\mathbf{A}_{(d_{\xi[t]})}}F(\mathbf{A}[t])\|_F^2+\frac{L_{d_\xi[t]}(\gamma[t])^2}{2K}\sigma^2_{d_\xi[t]}\\
&\quad+\gamma[t]\langle\nabla_{\mathbf{A}_{(d_{\xi[t]})}}F(\mathbf{A}[t])-\nabla_{\mathbf{A}_{(d_{\xi[t]})}}F(\widetilde{\mathbf{A}}[t]),\nabla_{\mathbf{A}_{(d_{\xi[t]})}}F(\mathbf{A}[t])\rangle.
\end{aligned}\tag{B.11}$$

We bound $\langle\nabla_{\mathbf{A}_{(d_{\xi[t]})}}F(\mathbf{A}[t])-\nabla_{\mathbf{A}_{(d_{\xi[t]})}}F(\widetilde{\mathbf{A}}[t]),\nabla_{\mathbf{A}_{(d_{\xi[t]})}}F(\mathbf{A}[t])\rangle$ by Young's inequal-

ity,

$$\begin{aligned}
&\langle \nabla_{\mathbf{A}_{(d_{\xi[t]})}} F(\mathbf{A}[t]) - \nabla_{\mathbf{A}_{(d_{\xi[t]})}} F(\widetilde{\mathbf{A}}[t]), \nabla_{\mathbf{A}_{(d_{\xi[t]})}} F(\mathbf{A}[t]) \rangle \\
&\leq \frac{1}{2\rho} \|\nabla_{\mathbf{A}_{(d_{\xi[t]})}} F(\mathbf{A}[t]) - \nabla_{\mathbf{A}_{(d_{\xi[t]})}} F(\widetilde{\mathbf{A}}[t])\|_F^2 + \frac{\rho}{2} \|\nabla_{\mathbf{A}_{(d_{\xi[t]})}} F(\mathbf{A}[t])\|_F^2
\end{aligned} \tag{B.12}$$

Plugging eq.(B.12) into eq.(B.11), we have

$$\begin{aligned}
\mathbb{E}_{\zeta[t]}\Big[F(\widetilde{\mathbf{A}}[t+1])\Big|\mathcal{F}[t],\xi[t]\Big] &\leq F(\widetilde{\mathbf{A}}[t]) - \gamma[t]\Big(1 - \frac{L_{d_\xi[t]}\gamma[t]+\rho}{2}\Big)\|\nabla_{\mathbf{A}_{(d_{\xi[t]})}} F(\mathbf{A}[t])\|_F^2 \\
&+ \frac{L_{d_\xi[t]}(\gamma[t])^2}{2K}\sigma_{d_\xi[t]}^2 + \frac{\gamma[t]}{2\rho}\|\nabla_{\mathbf{A}_{(d_{\xi[t]})}} F(\mathbf{A}[t]) - \nabla_{\mathbf{A}_{(d_{\xi[t]})}} F(\widetilde{\mathbf{A}}[t])\|_F^2 \\
&\leq F(\widetilde{\mathbf{A}}[t]) - \gamma[t]\Big(1 - \frac{L_{d_\xi[t]}\gamma[t]+\rho}{2}\Big)\|\nabla_{\mathbf{A}_{(d_{\xi[t]})}} F(\mathbf{A}[t])\|_F^2 \\
&+ \frac{L_{d_\xi[t]}(\gamma[t])^2}{2K}\sigma_{d_\xi[t]}^2 + \frac{L_{d_\xi[t]}^2\gamma[t]}{2\rho}\|\frac{1}{K}\sum_{k=1}^{K}\mathbf{E}_{(d_{\xi[t]})}^k[t]\|_F^2.
\end{aligned} \tag{B.13}$$

Taking expectation with respect to $\xi[t]$ conditioned on $\mathcal{F}[t]$ and substituting $L = \max\{L_1, ..., L_D\}$, $\sigma^2 = \sum_{d=1}^{D}\sigma_d^2$ in, we have

$$\begin{aligned}
\mathbb{E}_{\xi[t]}\Big[F(\widetilde{\mathbf{A}}[t+1])\Big|\mathcal{F}[t]\Big] &\leq F(\widetilde{\mathbf{A}}[t]) - \gamma[t]\Big(1 - \frac{L\gamma[t]+\rho}{2}\Big)\frac{1}{D}\sum_{d=1}^{D}\|\nabla_{\mathbf{A}_{(d)}} F(\mathbf{A}[t])\|_F^2 \\
&+ \frac{L(\gamma[t]\sigma)^2}{2KD} + \frac{L^2\gamma[t]}{2\rho}\frac{1}{D}\sum_{d=1}^{D}\|\frac{1}{K}\sum_{k=1}^{K}\mathbf{E}_{(d)}^k[t]\|_F^2.
\end{aligned} \tag{B.14}$$

By Lemma B.2.1 and let $\gamma[t] = t$, we have

$$\begin{aligned}
\mathbb{E}_{\xi[t]}\Big[F(\widetilde{\mathbf{A}}[t+1])\Big|\mathcal{F}[t]\Big] &\leq F(\widetilde{\mathbf{A}}[t]) - \gamma\Big(1 - \frac{L\gamma+\rho}{2}\Big)\frac{1}{D}\sum_{d=1}^{D}\|\nabla_{\mathbf{A}_{(d)}} F(\mathbf{A}[t])\|_F^2 \\
&+ \frac{L(\gamma\sigma)^2}{2KD} + \frac{2L^2\gamma^3(1-\delta)(\sigma^2+\omega^2)}{\rho D\delta^2}.
\end{aligned} \tag{B.15}$$

Taking total expectation with respect to all the random variables in $\mathcal{F}[t]$, we have

$$\mathbb{E}\Big[F(\widetilde{\mathbf{A}}[t+1])\Big] \leq \mathbb{E}\Big[F(\widetilde{\mathbf{A}}[t])\Big] - \gamma\Big(1-\frac{L\gamma+\rho}{2}\Big)\mathbb{E}\Big[\frac{1}{D}\sum_{d=1}^{D}\|\nabla_{\mathbf{A}_{(d)}}F(\mathbf{A}[t])\|_F^2\Big] + \frac{L(\gamma\sigma)^2}{2KD} + \frac{2L^2\gamma^3(1-\delta)(\sigma^2+\omega^2)}{\rho D\delta^2}. \tag{B.16}$$

Averaging the above from $t=0$ to $T$ and letting $\rho < 2-L\gamma$, $F^*$ the optimal value, we have

$$\frac{1}{T+1}\sum_{t=0}^{T}\mathbb{E}\Big[\frac{1}{D}\sum_{d=1}^{D}\|\nabla_{\mathbf{A}_{(d)}}F(\mathbf{A}[t])\|_F^2\Big] \leq \frac{1}{(T+1)\gamma(1-\frac{L\gamma+\rho}{2})}\Big[F(\mathbf{A}[0])-F^*\Big] + \frac{1}{(1-\frac{L\gamma+\rho}{2})}\Big[\frac{L\gamma\sigma^2}{2KD}+\frac{2L^2\gamma^2(1-\delta)(\sigma^2+\omega^2)}{\rho D\delta^2}\Big]. \tag{B.17}$$

Let $\rho=1$ and use $\mathbb{E}\Big[\frac{1}{D}\sum_{d=1}^{D}\|\nabla_{\mathbf{A}_{(d)}}F(\mathbf{A}[\texttt{Output}])\|_F^2\Big] \leq \sum_{t=0}^{T}\frac{1}{T+1}\mathbb{E}\Big[\frac{1}{D}\sum_{d=1}^{D}\|\nabla_{\mathbf{A}_{(d)}}F(\mathbf{A}[t])\|_F^2\Big]$, we have

$$\mathbb{E}\Big[\frac{1}{D}\sum_{d=1}^{D}\|\nabla_{\mathbf{A}_{(d)}}F(\mathbf{A}[\texttt{Output}])\|_F^2\Big] \leq \frac{2}{(T+1)\gamma(1-L\gamma)}\Big[F(\mathbf{A}[0])-F^*\Big]+\frac{2}{(1-L\gamma)}\Big[\frac{L\gamma\sigma^2}{2KD}+\frac{2L^2\gamma^2(1-\delta)(\sigma^2+\omega^2)}{D\delta^2}\Big]. \tag{B.18}$$

Finally, by letting $\gamma = \min\{\frac{1}{2L}, \frac{\varrho}{\sqrt{T+1}/\sqrt{K}+\frac{(1-\delta)^{1/3}}{\delta^{2/3}}T^{1/3}}\}$ for some $\varrho>0$, we complete the

proof of Theorem 4.1:

$$\mathbb{E}[\frac{1}{D}\sum_{d=1}^{D}\|\nabla_{\mathbf{A}_{(d)}}F(\mathbf{A}[\texttt{Output}])\|_F^2] \leq \frac{8L}{T+1}(F(\mathbf{A}[0])-F^*)+\Big[\frac{4}{\varrho}(F(\mathbf{A}[0])-F^*) + \frac{2L\sigma^2\varrho}{D}\Big]\frac{1}{\sqrt{M(T+1)}} + \Big[\frac{4}{\varrho}(F(\mathbf{A}[0])-F^*)+\frac{8L^2\varrho^2(\sigma^2+\omega^2)}{D}\Big]\frac{(1-\delta)^{1/3}}{\delta^{2/3}(T+1)^{2/3}}. \tag{B.19}$$

## B.2.2 Proof of Theorem 4.2

### Auxiliary variables for the proof and iterative relation

We derive the convergence by regarding the iteration as using inexact gradient, which is different from the approach used for the smooth case which is regarded as using delayed variable:

$$\begin{aligned}
&\mathbf{A}_{(d_\xi[t])}[t+1] = \texttt{Prox}\Big(\mathbf{A}_{(d_\xi[t])}[t]-\frac{1}{K}\sum_{k=1}^{K}\boldsymbol{\Delta}^k_{(d_\xi[t])}[t]\Big)\\
&=\texttt{Prox}\Big(\mathbf{A}_{(d_\xi[t])}[t]-\frac{1}{K}\sum_{k=1}^{K}(\gamma[t]\mathbf{G}^k_{(d_\xi[t])}[t]+\mathbf{E}^k_{(d_\xi[t])}[t])+\frac{1}{K}\sum_{k=1}^{K}(\gamma[t]\mathbf{G}^k_{(d_\xi[t])}[t]+\mathbf{E}^k_{(d_\xi[t])}[t])\\
&-\frac{1}{K}\sum_{k=1}^{K}\boldsymbol{\Delta}^k_{(d_\xi[t])}[t]\Big)\\
&=\texttt{Prox}\Big(\mathbf{A}_{(d^k)}[t]-\gamma[t]\frac{1}{K}\sum_{k=1}^{K}\mathbf{G}^k_{(d_\xi[t])}[t]-\frac{1}{K}\sum_{k=1}^{K}\mathbf{E}^k_{(d_\xi[t])}[t]+\frac{1}{K}\sum_{k=1}^{K}(\mathbf{P}^k_{(d_\xi[t])}[t]-\boldsymbol{\Delta}^k_{(d_\xi[t])}[t])\Big)\\
&=\texttt{Prox}\Big(\mathbf{A}_{(d^k)}[t]-\gamma[t]\frac{1}{K}\sum_{k=1}^{K}\big(\mathbf{G}^k_{(d_\xi[t])}[t]+\frac{1}{\gamma[t]}(\mathbf{E}^k_{(d_\xi[t])}[t+1]-\mathbf{E}^k_{(d_\xi[t])}[t])\big)\Big).
\end{aligned} \tag{B.20}$$

We define the generalized gradient $\mathbf{Z}[t] = (\mathbf{Z}_{(1)}[t], ..., (\mathbf{Z}_{(D)}[t])$, where

$$\mathbf{Z}_{(d)}[t] = \frac{1}{\gamma[t]}\Big(\mathbf{A}_{(d)}[t] - \texttt{Prox}_{r_{(d)}}(\mathbf{A}_{(d)}[t] - \gamma[t]\nabla_{\mathbf{A}_{(d)}} F(\mathbf{A}[t]))\Big) \tag{B.21}$$

If $d = d_\xi[t]$,

$$\overline{\mathbf{A}}_{(d)}[t+1] = \texttt{Prox}_{r_{(d)}}(\mathbf{A}_{(d)}[t] - \gamma[t]\nabla_{\mathbf{A}_{(d)}} F(\mathbf{A}[t])), \tag{B.22}$$

else if $d \neq d_\xi[t]$

$$\overline{\mathbf{A}}_{(d)}[t+1] = \mathbf{A}_{(d)}[t]. \tag{B.23}$$

let

$$\Phi(\mathbf{A}[t]) = F(\mathbf{A}[t]) + r(\mathbf{A}[t]); \tag{B.24}$$

**Additional Lemma**

We need the following Lemma 1 from [71].

**Lemma B.2.2.** *Let* $\mathbf{y} = \texttt{Prox}_{\gamma r}(\mathbf{x} - \gamma\mathbf{g})$*, for some* $\mathbf{g}$*. Then for* $\mathbf{y}$*, the following inequality holds,*

$$r(\mathbf{y}) + \langle \mathbf{y} - \mathbf{z}, \mathbf{g}\rangle \leq r(\mathbf{z}) + \frac{1}{2\gamma}[\|\mathbf{z} - \mathbf{x}\|_2^2 - \|\mathbf{y} - \mathbf{x}\|_2^2 - \|\mathbf{y} - \mathbf{z}\|_2^2], \tag{B.25}$$

*for any* $\mathbf{z}$*.*

**Main Proof of Theorem 4.2**

By the block-wise smoothness of $F$,

$$\begin{aligned} F(\overline{\mathbf{A}}[t+1]) \leq F(\mathbf{A}[t]) &+ \langle \nabla_{\mathbf{A}_{(d_\xi[t])}} F(\mathbf{A}[t]), \overline{\mathbf{A}}_{(d_\xi[t])}[t+1] - \mathbf{A}_{(d_\xi[t])}[t]\rangle \\ &+ \frac{L_{(d_\xi[t])}}{2}\|\overline{\mathbf{A}}_{(d_\xi[t])}[t+1] - \mathbf{A}_{(d_\xi[t])}[t]\|_F^2 \end{aligned} \tag{B.26}$$

By the convexity of $r_{(d)}(\cdot)$

$$r_{(d_\xi[t])}(\overline{\mathbf{A}}_{(d_\xi[t])}[t+1])+\langle \partial r_{(d_\xi[t])}(\overline{\mathbf{A}}_{(d_\xi[t])}[t+1]), \mathbf{A}_{(d_\xi[t])}[t]-\overline{\mathbf{A}}_{(d_\xi[t])}[t+1]\rangle \leq r_{(d_k)}(\mathbf{A}_{(d_\xi[t])}[t]). \tag{B.27}$$

and the optimality of $\overline{\mathbf{A}}_{(d_\xi[t])}[t+1]$ for $\texttt{Prox}_{r_{(d)}}(\mathbf{A}_{(d)}[t] - \gamma \nabla_{\mathbf{A}_{(d)}} F(\mathbf{A}[t]))$,

$$\partial r_{(d_\xi[t])}(\overline{\mathbf{A}}_{(d_\xi[t])}[t+1]) + \frac{1}{\gamma[t]}(\overline{\mathbf{A}}_{(d_\xi[t])}[t+1] - (\mathbf{A}_{(d_\xi[t])}[t] - \gamma[t]\nabla_{\mathbf{A}_{(d_\xi[t])}} F(\mathbf{A}[t]))) = 0, \tag{B.28}$$

we have

$$\begin{aligned} & r_{(d_\xi[t])}(\overline{\mathbf{A}}_{(d_\xi[t])}[t+1]) + \langle \frac{1}{\gamma[t]}(\overline{\mathbf{A}}_{(d_\xi[t])}[t+1] - (\mathbf{A}_{(d_\xi[t])}[t] \\ & - \gamma[t]\nabla_{\mathbf{A}_{(d_\xi[t])}} F(\mathbf{A}[t]))), \overline{\mathbf{A}}_{(d_\xi[t])}[t+1] - \mathbf{A}_{(d_\xi[t])}[t]\rangle \\ & \leq r_{(d_k)}(\mathbf{A}_{(d_\xi[t])}[t]), \end{aligned} \tag{B.29}$$

which amounts to

$$\begin{aligned} & r_{(d_\xi[t])}(\overline{\mathbf{A}}_{(d_\xi[t])}[t+1]) + \frac{1}{\gamma[t]}\|\overline{\mathbf{A}}_{(d_\xi[t])}[t+1] - \mathbf{A}_{(d_\xi[t])}[t]\|_F^2 \\ & + \langle \nabla_{\mathbf{A}_{(d_\xi[t])}} F(\mathbf{A}[t]), \overline{\mathbf{A}}_{(d_\xi[t])}[t+1] - \mathbf{A}_{(d_\xi[t])}[t]\rangle \\ & \leq r_{(d_k)}(\mathbf{A}_{(d_\xi[t])}[t]). \end{aligned} \tag{B.30}$$

Thus,

$$F(\overline{\mathbf{A}}[t+1])+r(\overline{\mathbf{A}}[t+1]) \leq F(\mathbf{A}[t])+r(\mathbf{A}[t])+(\frac{L_{(d_\xi[t])}}{2}-\frac{1}{\gamma[t]})\|\overline{\mathbf{A}}_{(d_\xi[t])}[t+1]-\mathbf{A}_{(d_\xi[t])}[t]\|_F^2, \tag{B.31}$$

which amounts to

$$\Phi(\overline{\mathbf{A}}[t+1]) \leq \Phi(\mathbf{A}[t]) + (\frac{L_{(d_\xi[t])}}{2} - \frac{1}{\gamma[t]})\|\overline{\mathbf{A}}_{(d_\xi[t])}[t+1] - \mathbf{A}_{(d_\xi[t])}[t]\|_F^2. \tag{B.32}$$

By Lemma B.2.2, we have

$$
\begin{aligned}
&F(\mathbf{A}_{(d_\xi[t])}[t+1], \mathbf{A}_{(-d_\xi[t])}[t]) + r_{(d_\xi[t])}(\mathbf{A}_{(d_\xi[t])}[t+1]) \leq F(\overline{\mathbf{A}}_{(d_\xi[t])}[t+1], \mathbf{A}_{(-d_\xi[t])}[t]) \\
&+ r_{(d_\xi[t])}(\overline{\mathbf{A}}_{(d_\xi[t])}[t+1]) + \langle \mathbf{A}_{(d_\xi[t])}[t+1] - \overline{\mathbf{A}}_{(d_\xi[t])}[t+1], \nabla_{(d_\xi[t])} F(\mathbf{A}_{(d_\xi[t])}[t]) \\
&- \frac{1}{K}\sum_{k=1}^{K}(\mathbf{G}^k_{(d_\xi[t])}[t] + \frac{1}{\gamma[t]}(\mathbf{E}^k_{(d_\xi[t])}[t+1] - \mathbf{E}_{(d^k_\xi[t])}[t]))\rangle \\
&+ (\frac{L_{(d_\xi[t])}}{2} - \frac{1}{2\gamma[t]})\|\mathbf{A}_{(d_\xi[t])}[t+1] - \mathbf{A}_{(d_\xi[t])}[t]\|_F^2 + (\frac{L_{(d_\xi[t])}}{2} + \frac{1}{2\gamma[t]})\|\overline{\mathbf{A}}_{(d_\xi[t])}[t+1] - \mathbf{A}_{(d_\xi[t])}[t]\|_F^2 \\
&- \frac{1}{2\gamma[t]}\|\overline{\mathbf{A}}_{(d_\xi[t])}[t+1] - \mathbf{A}_{(d_\xi[t])}[t+1]\|_F^2.
\end{aligned}
\tag{B.33}
$$

For the second row in the above inequality, we further have:

$$
\begin{aligned}
&\langle \mathbf{A}_{(d_\xi[t])}[t+1] - \overline{\mathbf{A}}_{(d_\xi[t])}[t+1], \nabla_{(d_\xi[t])} F(\mathbf{A}_{(d_\xi[t])}[t]) \\
&- \frac{1}{K}\sum_{k=1}^{K}(\mathbf{G}^k_{(d_\xi[t])}[t] + \frac{1}{\gamma[t]}(\mathbf{E}^k_{(d_\xi[t])}[t+1] - \mathbf{E}^k_{(d_\xi[t])}[t]))\rangle \\
&= \langle \mathbf{A}_{(d_\xi[t])}[t+1] - \overline{\mathbf{A}}_{(d_\xi[t])}[t+1], \nabla_{(d_\xi[t])} F(\mathbf{A}_{(d_\xi[t])}[t]) - \frac{1}{K}\sum_{k=1}^{K}\mathbf{G}^k_{(d_\xi[t])}[t]\rangle \\
&\quad - \frac{1}{\gamma[t]}\langle \mathbf{A}_{(d_\xi[t])}[t+1] - \overline{\mathbf{A}}_{(d_\xi[t])}[t+1], \frac{1}{K}\sum_{k=1}^{K}(\mathbf{E}^k_{(d_\xi[t])}[t+1] - \mathbf{E}^k_{(d_\xi[t])}[t])\rangle \\
&\leq \frac{1}{2\rho_1}\|\mathbf{A}_{(d_\xi[t])}[t+1] - \overline{\mathbf{A}}_{(d_\xi[t])}[t+1]\|_F^2 + \frac{\rho_1}{2}\|\nabla_{(d_\xi[t])} F(\mathbf{A}_{(d_\xi[t])}[t]) - \frac{1}{K}\sum_{k=1}^{K}\mathbf{G}^k_{(d_\xi[t])}[t]\|_F^2 \\
&\quad + \frac{1}{2\gamma[t]\rho_2}\|\mathbf{A}_{(d_\xi[t])}[t+1] - \overline{\mathbf{A}}_{(d_\xi[t])}[t+1]\|_F^2 + \frac{\rho_2}{2\gamma[t]}\|\frac{1}{K}\sum_{k=1}^{K}(\mathbf{E}^k_{(d_\xi[t])}[t+1] - \mathbf{E}^k_{(d_\xi[t])}[t])\|_F^2 \\
&= (\frac{1}{2\rho_1} + \frac{1}{2\gamma[t]\rho_2})\|\mathbf{A}_{(d_\xi[t])}[t+1] - \overline{\mathbf{A}}_{(d_\xi[t])}[t+1]\|_F^2 + \frac{\rho_1}{2}\|\nabla_{(d_\xi[t])} F(\mathbf{A}_{(d_\xi[t])}[t]) \\
&\quad - \frac{1}{K}\sum_{k=1}^{K}\mathbf{G}^k_{(d_\xi[t])}[t]\|_F^2 + \frac{\rho_2}{2\gamma[t]}\|\frac{1}{K}\sum_{k=1}^{K}(\mathbf{E}^k_{(d_\xi[t])}[t+1] - \mathbf{E}^k_{(d_\xi[t])}[t])\|_F^2.
\end{aligned}
\tag{B.34}
$$

By choosing $\rho_1 = 2\gamma[t]$ and $\rho_2 = 2$, we further have

$$\begin{aligned}\Phi(\mathbf{A}[t+1]) \leq \Phi(\overline{\mathbf{A}}[t+1]) &+ \gamma[t]\|\nabla_{(d_\xi[t])}F(\mathbf{A}_{(d_\xi[t])}[t]) - \frac{1}{K}\sum_{k=1}^{K}\mathbf{G}^k_{(d_\xi[t])}[t]\|_F^2 \\ &+ \frac{1}{\gamma[t]}\|\frac{1}{K}\sum_{k=1}^{K}(\mathbf{E}^k_{(d_\xi[t])}[t+1] - \mathbf{E}^k_{(d_\xi[t])}[t])\|_F^2 \\ &+ (\frac{L_{(d_\xi[t])}}{2} - \frac{1}{2\gamma[t]})\|\mathbf{A}_{(d_\xi[t])}[t+1] - \mathbf{A}_{(d_\xi[t])}[t]\|_F^2 \\ &+ (\frac{L_{(d_\xi[t])}}{2} + \frac{1}{2\gamma[t]})\|\overline{\mathbf{A}}_{(d_\xi[t])}[t+1] - \mathbf{A}_{(d_\xi[t])}[t]\|_F^2\end{aligned} \tag{B.35}$$

Combining it with eq.(B.32) and let $\gamma[t] \leq \frac{1}{2L_{(d_\xi[t])}}$, we have

$$\begin{aligned}\Phi(\mathbf{A}[t+1]) &\leq \Phi(\mathbf{A}[t]) + (L_{(d_\xi[t])} - \frac{1}{2\gamma[t]})\|\overline{\mathbf{A}}_{(d_\xi[t])}[t+1] - \mathbf{A}_{(d_\xi[t])}[t]\|_F^2 \\ &\quad + \gamma[t]\|\nabla_{(d_\xi[t])}F(\mathbf{A}_{(d_\xi[t])}[t]) - \frac{1}{K}\sum_{k=1}^{K}\mathbf{G}^k_{(d_\xi[t])}[t]\|_F^2 \\ &\quad + \frac{1}{\gamma[t]}\|\frac{1}{K}\sum_{k=1}^{K}(\mathbf{E}^k_{(d_\xi[t])}[t+1] - \mathbf{E}^k_{(d_\xi[t])}[t])\|_F^2 \\ &\leq \Phi(\mathbf{A}[t]) + (L_{(d_\xi[t])} - \frac{1}{2\gamma[t]})\|\overline{\mathbf{A}}_{(d_\xi[t])}[t+1] - \mathbf{A}_{(d_\xi[t])}[t]\|_F^2 \\ &\quad + \gamma[t]\frac{1}{K}\sum_{k=1^K}\|\nabla_{(d_\xi[t])}F(\mathbf{A}_{(d_\xi[t])}[t]) - \mathbf{G}^k_{(d_\xi[t])}[t]\|_F^2 \\ &\quad + \frac{1}{\gamma[t]}\frac{1}{K}\sum_{k=1}^{K}\|\mathbf{E}^k_{(d_\xi[t])}[t+1] - \mathbf{E}^k_{(d_\xi[t])}[t]\|_F^2.\end{aligned} \tag{B.36}$$

Taking conditional expectation on both sides with respect to $\xi[t]$ conditioned on

filtration $\mathcal{F}[t]$, we have

$$\begin{aligned}
&\mathbb{E}_{\xi[t]}[\Phi(\mathbf{A}[t+1])|\mathcal{F}[t]] \leq \Phi(\mathbf{A}[t]) + (L_{(d_\xi[t])} - \frac{1}{2\gamma[t]})\mathbb{E}_{\xi[t]}[\|\overline{\mathbf{A}}_{(d_\xi[t])}[t+1] - \mathbf{A}_{(d_\xi[t])}[t]\|_F^2|\mathcal{F}[t]] \\
&+ \gamma[t]\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}_{\xi[t]}[\|\nabla_{(d_\xi[t])}F(\mathbf{A}_{(d_\xi[t])}[t]) - \mathbf{G}^k_{(d_\xi[t])}[t]\|_F^2|\mathcal{F}[t]] \\
&+ \frac{1}{\gamma[t]}\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}_{\xi[t]}[\|\mathbf{E}^k_{(d_\xi[t])}[t+1] - \mathbf{E}^k_{(d_\xi[t])}[t]\|_F^2|\mathcal{F}[t]] \\
&\leq \Phi(\mathbf{A}[t]) + (L - \frac{1}{2\gamma[t]})\frac{1}{D}\sum_{d=1}^{D}\|\|\overline{\mathbf{A}}_{(d)}[t+1] - \mathbf{A}_{(d)}[t]\|_F^2 \\
&\quad + \frac{\gamma[t]\sigma^2}{D} + \frac{1}{\gamma[t]D}\frac{1}{K}\sum_{k=1}^{K}\Big(\sum_{d=1}^{D}(\|\mathbf{E}^k_{(d)}[t+1]\|_F^2 + \|\mathbf{E}^k_{(d)}[t]\|_F^2)\Big)
\end{aligned} \tag{B.37}$$

By Lemma B.2.1 and let $\gamma[t] = t$, we have

$$\begin{aligned}
\Phi(\mathbf{A}[t+1]) \leq \Phi(\mathbf{A}[t]) + (L - \frac{1}{2\gamma})\frac{1}{D}\sum_{d=1}^{D}\|\|\overline{\mathbf{A}}_{(d)}[t+1] - \mathbf{A}_{(d)}[t]\|_F^2 \\
+ \frac{\gamma\sigma^2}{D} + \frac{1}{D}\frac{8(1-\delta)}{\delta^2}\gamma(\sigma^2+\omega^2).
\end{aligned} \tag{B.38}$$

Taking total expectation (i.e. with respect to all random variables in $\mathcal{F}[t]$), we have

$$\begin{aligned}
\gamma^2(\frac{1}{2\gamma} - L)\mathbb{E}[\sum_{d=1}^{D}\frac{1}{D}\mathbb{E}[\|(\overline{\mathbf{A}}_{(d)}[t+1] - \mathbf{A}_{(d)}[t])/\gamma\|_F^2] \leq \mathbb{E}[\Phi(\mathbf{A}[t])] - \mathbb{E}[\Phi(\mathbf{A}[t+1])] \\
+ \frac{\gamma\sigma^2}{D} + \frac{1}{D}\frac{8(1-\delta)}{\delta^2}\gamma(\sigma^2+\omega^2)
\end{aligned} \tag{B.39}$$

Averaging the above equation from $t = 0$ to $T$ and by $\mathbb{E}\Big[\sum_{d=1}^{D}\frac{1}{D}\|\widetilde{\mathbf{G}}_{(d)}[\texttt{Output}]\|_F^2\Big] \leq \frac{1}{T+1}\mathbb{E}[\sum_{d=1}^{D}\frac{1}{D}\|\widetilde{\mathbf{G}}_{(d)}[t]\|_F^2] = \frac{1}{T+1}\mathbb{E}[\sum_{d=1}^{D}\frac{1}{D}\|(\overline{\mathbf{A}}_{(d)}[t+1] - \mathbf{A}_{(d)}[t])/\gamma\|_F^2]$, we have

$$\begin{aligned}
\mathbb{E}\Big[\sum_{d=1}^{D}\frac{1}{D}\|\widetilde{\mathbf{G}}_{(d)}[\texttt{Output}]\|_F^2\Big] \leq \frac{2}{\gamma(1-2\gamma L)(T+1)}(\Phi(\mathbf{A}^0) - \Phi^*) \\
+ \frac{2\sigma^2}{D(1-2\gamma L)} + \frac{16(1-\delta)}{D\delta^2(1-2\gamma L)}(\sigma^2+\omega^2).
\end{aligned} \tag{B.40}$$

Finally, by setting $\gamma = \frac{1}{4L}$, we complete our proof:

$$\mathbb{E}\Big[\sum_{d=1}^{D}\frac{1}{D}\|\widetilde{\mathbf{G}}_{(d)}[\texttt{Output}]\|_F^2\Big] \leq \frac{16L}{T+1}(\Phi(\mathbf{A}[0]) - \Phi^*) + \frac{4\sigma^2}{DK} + \frac{32(1-\delta)}{D\delta^2}(\sigma^2+\omega^2). \tag{B.41}$$

## B.3 Additional Materials for Convergence Analysis of Algorithm 2

### B.3.1 Proof of Theorem 4.3

This proof extends the proof of Theorem 1 in [5] to our block randomized case.

#### Auxillary variables for the proof and iterative relation

Let $\widetilde{\mathbf{A}}^k_{(d)}[t+1] = \widetilde{\mathbf{A}}^k_{(d)}[t] - \gamma[t] G^k_{(d)}[t]$, with $\widetilde{\mathbf{A}}^k_{(d)}[0] = \mathbf{A}^k_{(d)}[0]$; $\widetilde{\mathbf{A}}^{avg}_{(d)}[t+1] = \frac{1}{K}\sum_{k=1}^{K} \widetilde{\mathbf{A}}^k_{(d)}[t+1] = \widetilde{\mathbf{A}}^{avg}_{(d)}[t] - \gamma[t]\frac{1}{K}\sum_{k=1}^{R} G^k_{(d)}[t]$; $\mathbf{A}^{avg}_{(d)}[t] = \frac{1}{K}\sum_{k=1}^{K} \mathbf{A}^k_{(d)}[t]$. It is easy to see that $\mathbf{A}^{avg}_{(d)}[t] - \widetilde{\mathbf{A}}^{avg}_{(d)}[t] = \frac{1}{K}\sum_{k=1}^{K} \mathbf{E}^k_{(d)}[t]$.

#### Additional Lemma

**Lemma B.3.1.** *(Bounding the expectation of the block-wise feedback error averaged among client nodes) For $d = 1, ..., D$ and for $t = 0, ..., T$, assuming constant step size $\gamma[t] = \gamma$, we have*

$$\mathbb{E}[\| \frac{1}{K} \sum_{k=1}^{K} \mathbf{E}^k_{(d)}[t] \|_F^2] \leq \frac{4(1-\delta^2)(\sigma_d^2 + \omega_d^2)\tau^2}{\delta^2} \tag{B.42}$$

*Proof.* For $t^\dagger \texttt{ mod } \tau = 0$, we have

$$
\begin{aligned}
&\mathbb{E}[\|\frac{1}{K}\sum_{k=1}^{K}\mathbf{E}_{(d)}^k[t^\dagger+\tau]\|_F^2] \le \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}[\|\mathbf{E}_{(d)}^k[t^\dagger+\tau]\|_F^2] \\
&= \frac{1}{K}\sum_{k=1}^{K}(1-\frac{1}{D})\mathbb{E}[\|\mathbf{E}_{(d)}^k[t^\dagger+\tau-1]\|_F^2 \\
&+ \frac{1}{D}\mathbb{E}[\|\mathbf{E}_{(d)}^k[t^\dagger+\tau-1]+\mathbf{A}_{(d)}^g[t^\dagger+\tau-1]-\mathbf{A}_{(d)}^k[t^\dagger+\tau-\frac{1}{2}]-\Delta_{(d)}^k[t^\dagger+\tau-1]\|_F^2] \\
&\le \frac{1}{K}\sum_{k=1}^{K}(1-\frac{1}{D})\mathbb{E}[\|\mathbf{E}_{(d)}^k[t^\dagger+\tau-1]\|_F^2 \\
&+ \frac{1}{D}(1-\delta)\mathbb{E}[\|\mathbf{E}_{(d)}^k[t^\dagger+\tau-1]+\mathbf{A}_{(d)}^g[t^\dagger+\tau-1]-\mathbf{A}_{(d)}^k[t^\dagger+\tau-\frac{1}{2}]\|_F^2] \\
&= \frac{1}{K}\sum_{k=1}^{K}(1-\frac{1}{D})\mathbb{E}[\|\mathbf{E}_{(d)}^k[t^\dagger+\tau-1]\|_F^2 \\
&+ \frac{1}{D}(1-\delta)\mathbb{E}[\|\mathbf{E}_{(d)}^k[t^\dagger+\tau-1]+\mathbf{A}_{(d)}^k[t^\dagger]-\mathbf{A}_{(d)}^k[t^\dagger+\tau-\frac{1}{2}]\|_F^2] \\
&\le \frac{1}{K}\sum_{k=1}^{K}(1-\frac{1}{D})\mathbb{E}[\|\mathbf{E}_{(d)}^k[t^\dagger+\tau-1]\|_F^2 \\
&+ \frac{1}{D}(1-\delta)\Big(\mathbb{E}[(1+\frac{\delta}{2})\|\mathbf{E}_{(d)}^k[t^\dagger+\tau-1]\|_f^2]+(1+\frac{2}{\delta})\mathbb{E}[\|\mathbf{A}_{(d)}^k[t^\dagger]-\mathbf{A}_{(d)}^k[t^\dagger+\tau-\frac{1}{2}]\|_F^2]\Big) \\
&\le \frac{1}{K}\sum_{k=1}^{K}(1-\frac{1}{D})\mathbb{E}[\|\mathbf{E}_{(d)}^k[t^\dagger+\tau-1]\|_F^2 \\
&+ \frac{1}{D}(1-\frac{\delta}{2})\mathbb{E}[(1+\frac{\delta}{2})\|\mathbf{E}_{(d)}^k[t^\dagger+\tau-1]\|_f^2]+\frac{1}{D}\frac{2(1-\delta^2)}{\delta}\mathbb{E}[\|\mathbf{A}_{(d)}^k[t^\dagger]-\mathbf{A}_{(d)}^k[t^\dagger+\tau-\frac{1}{2}]\|_F^2] \\
&\le \frac{1}{K}\sum_{k=1}^{K}(1-\frac{1}{D})\mathbb{E}[\|\mathbf{E}_{(d)}^k[t^\dagger+\tau-1]\|_F^2 \\
&+ \frac{1}{D}(1-\frac{\delta}{2})\mathbb{E}[(1+\frac{\delta}{2})\|\mathbf{E}_{(d)}^k[t^\dagger+\tau-1]\|_f^2]+\frac{1}{D}\frac{2(1-\delta^2)}{\delta}(\sigma_d^2+\omega_d^2)\gamma^2\tau^2.
\end{aligned}
\tag{B.43}
$$

By unrolling the above recursion in a similar way as the proof of Lemma B.2.1, we have

$$
\mathbb{E}[\|\frac{1}{K}\sum_{k=1}^{K}\mathbf{E}_{(d)}^k[t^\dagger+\tau]\|_F^2] \le \frac{4(1-\delta^2)(\sigma_d^2+\omega_d^2)\tau^2}{\delta^2}, \tag{B.44}
$$

where the right hand side is indepednet on $t$, thus

$$\mathbb{E}[\|\frac{1}{K}\sum_{k=1}^{K}\mathbf{E}_{(d)}^{k}[t]\|_F^2] \leq \frac{4(1-\delta^2)(\sigma_d^2+\omega_d^2)\tau^2}{\delta^2}, \text{ for } t \in \{1,...,T\}. \tag{B.45}$$

□

**Lemma B.3.2.** *(Lemma 7 in [5]) The deviation of the local sequences has the following upper bound,*

$$\mathbb{E}[\sum_{k=1}^{K}\|\mathbf{A}^{avg}[t]-\mathbf{A}^{k}[t]\|_F^2] \leq \gamma\tau^2(\sigma^2+\omega^2). \tag{B.46}$$

**Main proof of Theorem 4.3**

By the block-wise Lipschitz smoothness assumption of the loss function:

$$\begin{aligned}
&F(\widetilde{\mathbf{A}}^{avg}[t+1]) \leq F(\widetilde{\mathbf{A}}^{avg}[t]) + \langle\nabla_{d_\xi[t]}F(\widetilde{\mathbf{A}}^{avg}[t]), \widetilde{\mathbf{A}}^{avg}[t+1]-\widetilde{\mathbf{A}}^{avg}[t]\rangle \\
&+\frac{L_{d_\xi[t]}}{2}\|\widetilde{\mathbf{A}}^{avg}[t+1]-\widetilde{\mathbf{A}}^{avg}[t]\|_2^2 \\
&= F(\widetilde{\mathbf{A}}^{avg}[t]) - \gamma[t]\langle\nabla_{d_\xi[t]}F(\widetilde{\mathbf{A}}^{avg}[t]), \sum_{k=1}^{K}\mathbf{G}_{(d_\xi[t])}^{k}[t]\rangle + \frac{L_{d_\xi[t]}(\gamma[t])^2}{2}\|\sum_{k=1}^{K}\mathbf{G}_{(d_\xi[t])}^{k}[t]\|_2^2.
\end{aligned} \tag{B.47}$$

Taking conditional expectation with randomness in $\zeta[t]$ conditioned on $\mathcal{F}[t], \xi[t]$,

we have

$$
\begin{aligned}
&\mathbb{E}_{\zeta[t]}[F(\widetilde{\mathbf{A}}^{avg}[t+1])] \leq F(\widetilde{\mathbf{A}}^{avg}[t]) - \gamma[t]\langle \nabla_{d_\xi[t]} F(\widetilde{\mathbf{A}}^{avg}[t]), \frac{1}{K}\sum_{k=1}^{K} \nabla_{(d_\xi[t])} F(\mathbf{A}^k[t])\rangle \\
&+ \frac{L_{d_\xi[t]}(\gamma[t])^2}{2}\mathbb{E}_{\zeta[t]}\Big[\|\frac{1}{K}\sum_{k=1}^{K}(\mathbf{G}^k_{(d_\xi[t])}[t] - \nabla_{(d_\xi[t])}F(\mathbf{A}^k[t])\|_F^2)\Big|\mathcal{F}[t],\xi[t]\Big] \\
&+ \frac{L_{d_\xi[t]}(\gamma[t])^2}{2}\|\frac{1}{K}\sum_{k=1}^{K}(\nabla_{(d_\xi[t])}F(\mathbf{A}^k[t]))\|_F^2 \\
&= F(\widetilde{\mathbf{A}}^{avg}[t]) - \frac{\gamma[t]}{2}\Big(\|\nabla_{d_\xi[t]}F(\widetilde{\mathbf{A}}^{avg}[t])\|_F^2 + \|\frac{1}{K}\sum_{k=1}^{K}\nabla_{(d_\xi[t])}F(\mathbf{A}^k[t])\|_F^2 \\
&- \|\nabla_{d_\xi[t]}F(\widetilde{\mathbf{A}}^{avg}[t]) - \frac{1}{K}\sum_{k=1}^{K}\nabla_{(d_\xi[t])}F(\mathbf{A}^k[t])\|_F^2\Big) \\
&+ \frac{L_{d_\xi[t]}(\gamma[t])^2}{2}\|\frac{1}{K}\sum_{k=1}^{K}\nabla_{(d_\xi[t])}F(\mathbf{A}^k[t])\|_F^2 + \frac{L_{d_\xi[t]}(\gamma[t])^2}{2K}\sigma^2_{d_\xi[t]} \\
&\leq F(\widetilde{\mathbf{A}}^{avg}[t]) - \frac{\gamma[t]}{2}\Big(\|\nabla_{d_\xi[t]}F(\widetilde{\mathbf{A}}^{avg}[t])\|_F^2 - L^2_{d_\xi[t]}\|\widetilde{\mathbf{A}}^{avg}_{(d_\xi[t])}[t] - \frac{1}{K}\sum_{k=1}^{K}\mathbf{A}^k_{(d_\xi[t])}[t]\|_F^2\Big) \\
&+ \frac{L_{d_\xi[t]}(\gamma[t])^2 - \gamma[t]}{2}\|\frac{1}{K}\sum_{k=1}^{K}\nabla_{(d_\xi[t])}F(\mathbf{A}^k[t])\|_F^2 + \frac{L_{d_\xi[t]}(\gamma[t])^2}{2K}\sigma^2_{d_\xi[t]} \\
&\leq F(\widetilde{\mathbf{A}}^{avg}[t]) - \frac{\gamma[t]}{2K}\sum_{k=1}^{K}\Big(L^2_{d_\xi[t]}\|\widetilde{\mathbf{A}}^{avg}_{d_\xi[t]}[t] - \mathbf{A}^k_{d_\xi[t]}[t]\|_F^2 + \frac{1}{2}\|\nabla_{d_\xi[t]}F(\mathbf{A}^k[t])\|_F^2 \\
&+ L^2_{d_\xi[t]}\|\widetilde{\mathbf{A}}^{avg}_{(d_\xi[t])}[t] - \mathbf{A}^k_{(d_\xi[t])}[t]\|_F^2\Big) \\
&+ \frac{L_{d_\xi[t]}(\gamma[t])^2 - \gamma[t]}{2K}\sum_{k=1}^{K}\|\nabla_{(d_\xi[t])}F(\mathbf{A}^k[t])\|_F^2 + \frac{L_{d_\xi[t]}(\gamma[t])^2}{2K}\sigma^2_{d_\xi[t]}.
\end{aligned}
\tag{B.48}
$$

Reorganizing terms and by $\gamma[t] \leq \frac{1}{L_{\xi[t]}}$, we have

$$
\begin{aligned}
&\frac{\gamma[t]}{4K}\sum_{k=1}^{K}\|\nabla_{d_\xi[t]}F(\mathbf{A}^k[t])\|_F^2 \leq F(\widetilde{\mathbf{A}}^{avg}[t]) - \mathbb{E}_{\zeta[t]}[F(\widetilde{\mathbf{A}}^{avg}[t+1])] \\
&+ \frac{\gamma[t]L^2_{d_\xi[t]}}{K}\sum_{k=1}^{K}\|\widetilde{\mathbf{A}}^{avg}_{d_\xi[t]}[t] - \mathbf{A}^k_{d_\xi[t]}[t]\|_F^2 + \frac{L_{d_\xi[t]}(\gamma[t])^2}{2K}\sigma^2_{d_\xi[t]}.
\end{aligned}
\tag{B.49}
$$

Taking expectation with respect to $\xi[t]$ conditioned on $\mathcal{F}[t]$ and substituting $L = \max\{L_1, ..., L_D\}$, $\sigma^2 = \sum_{d=1}^{D} \sigma_d^2$ in, we have

$$\begin{aligned}
&\frac{\gamma[t]}{4DK} \sum_{k=1}^{K} \sum_{d=1}^{D} \|\nabla_d F(\mathbf{A}^k[t])\|_F^2 \leq \mathbb{E}_{\xi[t]}[F(\widetilde{\mathbf{A}}^{avg}[t]) - \mathbb{E}_{\zeta[t]}[F(\widetilde{\mathbf{A}}^{avg}[t+1])]] \\
&+ \frac{\gamma[t]L^2}{DK} \sum_{k=1}^{K} \sum_{d=1}^{D} \|\widetilde{\mathbf{A}}_{(d)}^{avg}[t] - \mathbf{A}_{(d)}^{k}[t]\|_F^2 + \frac{L(\gamma[t])^2}{2DK}\sigma^2 \\
&= \mathbb{E}_{\xi[t]}[F(\widetilde{\mathbf{A}}^{avg}[t]) - \mathbb{E}_{\zeta[t]}[F(\widetilde{\mathbf{A}}^{avg}[t+1])]] + \frac{L(\gamma[t])^2}{2DK}\sigma^2 \\
&+ \frac{\gamma[t]L_{d_{\xi[t]}}^2}{DK} \sum_{k=1}^{K} \|\widetilde{\mathbf{A}}^{avg}[t] - \mathbf{A}^{avg}[t] + \mathbf{A}^{avg}[t] - \mathbf{A}^{k}[t]\|_F^2 \\
&\leq \mathbb{E}_{\xi[t]}[F(\widetilde{\mathbf{A}}^{avg}[t]) - \mathbb{E}_{\zeta[t]}[F(\widetilde{\mathbf{A}}^{avg}[t+1])]] + \frac{L(\gamma[t])^2}{2DK}\sigma^2 \\
&+ \frac{2\gamma[t]L^2}{D} \|\widetilde{\mathbf{A}}^{avg}[t] - \mathbf{A}^{avg}[t]\|_F^2 + \frac{2\gamma[t]L^2}{DK} \sum_{k=1}^{K} \|\mathbf{A}^{avg}[t] - \mathbf{A}^{k}[t]\|_F^2 \\
&\leq \mathbb{E}_{\xi[t]}[F(\widetilde{\mathbf{A}}^{avg}[t]) - \mathbb{E}_{\zeta[t]}[F(\widetilde{\mathbf{A}}^{avg}[t+1])]] + \frac{L(\gamma[t])^2}{2DK}\sigma^2 \\
&+ \frac{2\gamma[t]L^2}{D} \sum_{d=1}^{D} \| \sum_{k=1}^{K} \mathbf{E}_{(d)}^{k}[t]\|_F^2 + \frac{2\gamma[t]L^2}{DK} \sum_{k=1}^{K} \|\mathbf{A}^{avg}[t] - \mathbf{A}^{k}[t]\|_F^2
\end{aligned} \tag{B.50}$$

Taking total expectation with respect to all the random variables in $\mathcal{F}[t]$, by Lemma B.3.1 and B.3.2 and setting $\gamma[t] = \gamma$, we have

$$\begin{aligned}
&\frac{\gamma}{4DK}\mathbb{E}[\sum_{k=1}^{K} \sum_{d=1}^{D} \|\nabla_d F(\mathbf{A}^k[t])\|_F^2] \leq \mathbb{E}[F(\widetilde{\mathbf{A}}^{avg}[t])] - \mathbb{E}[F(\widetilde{\mathbf{A}}^{avg}[t+1])] \\
&+ \frac{L\sigma^2}{2DK}\gamma^2 + \frac{8L^2(1-\delta^2)(\sigma^2+\omega^2)\tau^2}{D\delta^2}\gamma^3 + \frac{2L^2(\sigma^2+\omega^2)\tau^2}{DK}\gamma^3.
\end{aligned} \tag{B.51}$$

Averaging the above from $t = 0$ to $T$ and denoting $F^*$ the optimal value, we have

$$\frac{1}{DK(T+1)}\sum_{t=0}^{T}\mathbb{E}[\sum_{k=1}^{K}\sum_{d=1}^{D}\|\nabla_d F(\mathbf{A}^k[t])\|_F^2] \leq \frac{4}{(T+1)\gamma}[F(\widetilde{\mathbf{A}}^{avg}[t]) - F^*] + 2L\sigma^2\gamma + \frac{32L^2(1-\delta^2)(\sigma^2+\omega^2)\tau^2}{D\delta^2}\gamma^2 + \frac{8L^2(\sigma^2+\omega^2)\tau^2}{D^2K}\gamma^2. \tag{B.52}$$

Let $\mathbf{A}[\texttt{Output}]$ be sampled from $A^k[t]$ for $t = 0, ..., T$ and $k = 1, ..., K$ with probability $\frac{1}{K(T+1)}$, and let $\gamma = \frac{C}{\sqrt{T+1}}$, we have

$$\mathbb{E}[\sum_{d=1}^{D}\frac{1}{D}\|\nabla_d F(\mathbf{A}[\texttt{Output}])\|_F^2] \leq \frac{4C}{\sqrt{T+1}}[F(\widetilde{\mathbf{A}}^{avg}[t]) - F^*] + \frac{2CL\sigma^2}{\sqrt{T+1}} + \frac{32C^2L^2(1-\delta^2)(\sigma^2+\omega^2)\tau^2}{D\delta^2(T+1)} + \frac{8C^2L^2(\sigma^2+\omega^2)\tau^2}{DK(T+1)}. \tag{B.53}$$

## B.4 Additional Materials for Complexity Analysis of Algorithm 1 & 2

### B.4.1 Proof of Theorem 4.4 (Computational Complexity)

*Proof.* The per-iteration complexity for each client can be broken down as follows:

1) $\mathbf{G}_{(d)}^k[t]$ takes $O(I_d|\mathcal{S}|)+R|\mathcal{S}|(D-1)+|\mathcal{S}|RI_d$, where the first term is for computing the sampled $\mathbf{Y}_{<d>}(:,|\mathcal{S}|)$, the second is for computing the sampled $\mathbf{H}_d(|\mathcal{S}|,:)$ and the third is for computing the matrix multiplication;

2) all other steps are element-wise operations of variable size $I_d \times R$, which takes $O(I_dR)$. (Here, we assume the proximal operator is also element-wise (e.g., soft-threholding) but it can take higher complexity if more computational demanding proximal operator is used, e.g. singular value thresholding operator for the nuclear norm regularization).

Finally, considering the randomness of sampling $d$, the averaged per-iteration per-client computational complexity is $O(\frac{1}{D}(\sum_{d=1}^D I_d)R|\mathcal{S}|)$ □

### B.4.2 Proof of Theorem 4.5 (Uplink Communication Complexity)

*Proof.* For Algorithm 1, each client requires $32 + \frac{1}{D}R\sum_{d=2}^D I_d$ bits per-iteration on average for the uplink communication, where 32-bits is for sending the $\ell_1$ norm and the $\frac{1}{D}R\sum_{d=2}^D I_d$ is the average cost for communicating the sign of each element of $\mathbf{\Delta}_{(d)}^k$. Compared to the full precision distributed SGD, the per-iteration uplink communication cost reduction is $1 - \frac{32+\frac{1}{D}R\sum_{d=2}^D I_d}{32\cdot(\sum_{d=1}^D I_d)} \approx 1 - \frac{1}{32D}$. Algorithm 2 further reduces $1/\tau$ of the communication upon Algorithm 1, which is approximately $1 - \frac{1}{32D\tau}$. □

# Bibliography

[1] https://www.cms.gov/Research-Statistics-Data-and-Systems/Downloadable-Public-Use-Files/SynPUFs/DE_Syn_PUF.

[2] Evrim Acar, Daniel M Dunlavy, Tamara G Kolda, and Morten Mørup. Scalable tensor factorizations for incomplete data. *Chemometrics and Intelligent Laboratory Systems*, 106(1):41–56, 2011.

[3] Naman Agarwal, Ananda Theertha Suresh, Felix Xinnan X Yu, Sanjiv Kumar, and Brendan McMahan. cpsgd: Communication-efficient and differentially-private distributed sgd. In *Advances in Neural Information Processing Systems*, 2018.

[4] Brett W. Bader, Tamara G. Kolda, et al. Matlab tensor toolbox version 3.0-dev. Available online, August 2017. URL https://gitlab.com/tensors/tensor_toolbox.

[5] Debraj Basu, Deepesh Data, Can Karakus, and Suhas Diggavi. Qsparse-local-sgd: Distributed sgd with quantization, sparsification, and local computations. In *Advances in Neural Information Processing Systems*, 2019.

[6] Casey Battaglino, Grey Ballard, and Tamara G Kolda. A practical randomized cp tensor decomposition. *SIAM Journal on Matrix Analysis and Applications*, 39(2):876–901, 2018.

[7] Amir Beck and Luba Tetruashvili. On the convergence of block coordinate descent type methods. *SIAM journal on Optimization*, 23(4):2037–2060, 2013.

[8] Arnaud Berlioz, Arik Friedman, Mohamed Ali Kaafar, Roksana Boreli, and Shlomo Berkovsky. Applying differential privacy to matrix factorization. In *Proceedings of the 9th ACM Conference on Recommender Systems*, pages 107–114. ACM, 2015.

[9] Alex Beutel, Partha Pratim Talukdar, Abhimanu Kumar, Christos Faloutsos, Evangelos E Papalexakis, and Eric P Xing. Flexifact: Scalable flexible factorization of coupled tensors on hadoop. In *Proceedings of the 2014 SDM*, pages 109–117, 2014.

[10] Mark Bun and Thomas Steinke. Concentrated differential privacy: Simplifications, extensions, and lower bounds. In *Theory of Cryptography Conference*, pages 635–658. Springer, 2016.

[11] J Douglas Carroll and Jih-Jie Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. *Psychometrika*, 35(3):283–319, 1970.

[12] Zheng Chai, Yujing Chen, Liang Zhao, Yue Cheng, and Huzefa Rangwala. Fedat: A communication-efficient federated learning method with asynchronous tiers under non-iid data. *arXiv preprint arXiv:2010.05958*, 2020.

[13] Tianyi Chen, Xiao Jin, Yuejiao Sun, and Wotao Yin. Vafl: a method of vertical asynchronous federated learning. *arXiv preprint arXiv:2007.06081*, 2020.

[14] Weisheng Chen and Wei Ren. Event-triggered zero-gradient-sum distributed consensus optimization over directed networks. *Automatica*, 65:90–97, 2016.

[15] Eric C Chi and Tamara G Kolda. On tensors, sparsity, and nonnegative factorizations. *SIAM Journal on Matrix Analysis and Applications*, 33(4):1272–1299, 2012.

[16] Joon Hee Choi and S Vishwanathan. Dfacto: Distributed factorization of tensors. In *NIPS*, pages 1296–1304, 2014.

[17] Patrick L Combettes and Jean-Christophe Pesquet. Proximal splitting methods in signal processing. In *Fixed-point algorithms for inverse problems in science and engineering*, pages 185–212. Springer, 2011.

[18] Dimos V Dimarogonas, Emilio Frazzoli, and Karl H Johansson. Distributed event-triggered control for multi-agent systems. *IEEE Transactions on Automatic Control*, 57(5):1291–1297, 2011.

[19] Wen Du, Xinlei Yi, Jemin George, Karl H Johansson, and Tao Yang. Distributed optimization with dynamic event-triggered mechanisms. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 969–974. IEEE, 2018.

[20] Cynthia Dwork and Guy N Rothblum. Concentrated differential privacy. *arXiv preprint arXiv:1603.01887*, 2016.

[21] Cynthia Dwork, Guy N Rothblum, and Salil Vadhan. Boosting and differential privacy. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 51–60. IEEE, 2010.

[22] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4): 211–407, 2014.

[23] Vera Ehrenstein, Hadi Kharrazi, Harold Lehmann, and Casey Overby Taylor. Obtaining data from electronic health records. In *Tools and Technologies for*

*Registry Interoperability, Registries for Evaluating Patient Outcomes: A User's Guide, 3rd Edition, Addendum 2 [Internet]*. Agency for Healthcare Research and Quality (US), 2019.

[24] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1322–1333. ACM, 2015.

[25] Xiao Fu, Shahana Ibrahim, Hoi-To Wai, Cheng Gao, and Kejun Huang. Block-randomized stochastic proximal gradient for low-rank tensor factorization. *IEEE Transactions on Signal Processing*, 68:2170–2185, 2020.

[26] Xiao Fu, Shahana Ibrahim, Hoi-To Wai, Cheng Gao, and Kejun Huang. Block-randomized stochastic proximal gradient for low-rank tensor factorization. *IEEE Transactions on Signal Processing*, 68:2170–2185, 2020.

[27] Aimilia Gastounioti, Meng-Kang Hsieh, Eric Cohen, Lauren Pantalone, Emily F Conant, and Despina Kontos. Incorporating breast anatomy in computational phenotyping of mammographic parenchymal patterns for breast cancer risk estimation. *Scientific reports*, 8(1):1–10, 2018.

[28] Avishek Ghosh, Raj Kumar Maity, Swanand Kadhe, Arya Mazumdar, and Kannan Ramchandran. Communication-efficient and byzantine-robust distributed learning. In *2020 Information Theory and Applications Workshop (ITA)*, pages 1–28. IEEE, 2020.

[29] Trisha Greenhalgh, Susan Hinder, Katja Stramer, Tanja Bratan, and Jill Russell. Adoption, non-adoption, and abandonment of a personal electronic health record: case study of healthspace. *Bmj*, 341:c5814, 2010.

[30] Yuhong Guo and Wei Xue. Probabilistic multi-label classification with sparse feature learning. In *IJCAI*, pages 1373–1379, 2013.

[31] Richard A Harshman et al. Foundations of the parafac procedure: Models and conditions for an" explanatory" multimodal factor analysis. 1970.

[32] Huan He, Jette Henderson, and Joyce C Ho. Distributed tensor decomposition for large scale health analytics. In *The World Wide Web Conference*, 2019.

[33] WPMH Heemels, Karl Henrik Johansson, and Paulo Tabuada. An introduction to event-triggered and self-triggered control. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 3270–3285. IEEE, 2012.

[34] Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz. Deep models under the gan: information leakage from collaborative deep learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 603–618. ACM, 2017.

[35] Joyce C Ho, Joydeep Ghosh, Steve R Steinhubl, Walter F Stewart, Joshua C Denny, Bradley A Malin, and Jimeng Sun. Limestone: High-throughput candidate phenotype generation via tensor factorization. *Journal of biomedical informatics*, 52:199–211, 2014.

[36] Joyce C Ho, Joydeep Ghosh, and Jimeng Sun. Marble: high-throughput phenotyping from electronic health records via sparse nonnegative tensor factorization. In *Proceedings of the 20th ACM SIGKDD*, pages 115–124. ACM, 2014.

[37] David Hong, Tamara G Kolda, and Jed A Duersch. Generalized canonical polyadic tensor decomposition. *arXiv preprint arXiv:1808.07452*, 2018.

[38] Jingyu Hua, Chang Xia, and Sheng Zhong. Differentially private matrix factorization. In *IJCAI*, pages 1763–1770, 2015.

[39] Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3:160035, 2016.

[40] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

[41] U Kang, Evangelos Papalexakis, Abhay Harpale, and Christos Faloutsos. Gigatensor: scaling tensor analysis up by 100 times-algorithms and discoveries. In *Proceedings of the 18th ACM SIGKDD*, pages 316–324. ACM, 2012.

[42] Alexandros Karatzoglou, Xavier Amatriain, Linas Baltrunas, and Nuria Oliver. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 79–86, 2010.

[43] Sai Praneeth Karimireddy, Quentin Rebjock, Sebastian Stich, and Martin Jaggi. Error feedback fixes signsgd and other gradient compression schemes. In *International Conference on Machine Learning*, pages 3252–3261, 2019.

[44] Yejin Kim, Robert El-Kareh, Jimeng Sun, Hwanjo Yu, and Xiaoqian Jiang. Discriminative and distinct phenotyping by constrained tensor factorization. *Scientific reports*, 7(1):1114, 2017.

[45] Yejin Kim, Jimeng Sun, Hwanjo Yu, and Xiaoqian Jiang. Federated tensor factorization for computational phenotyping. In *Proceedings of the 23rd ACM SIGKDD*, pages 887–895. ACM, 2017.

[46] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[47] Tamara G Kolda and David Hong. Stochastic gradients for large-scale tensor decomposition. *arXiv preprint arXiv:1906.01687*, 2019.

[48] Anastasia Koloskova, Sebastian Stich, and Martin Jaggi. Decentralized stochastic optimization and gossip algorithms with compressed communication. In *ICML*, pages 3478–3487. PMLR, 2019.

[49] Anastasia Koloskova, Nicolas Loizou, Sadra Boreiri, Martin Jaggi, and Sebastian Stich. A unified theory of decentralized sgd with changing topology and local updates. In *ICML*, pages 5381–5393. PMLR, 2020.

[50] Jun Li, Yumeng Shao, Ming Ding, Chuan Ma, Kang Wei, Zhu Han, and H Vincent Poor. Blockchain assisted decentralized federated learning (blade-fl) with lazy clients. *arXiv preprint arXiv:2012.02044*, 2020.

[51] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. *arXiv preprint arXiv:1705.09056*, 2017.

[52] Xiangru Lian, Wei Zhang, Ce Zhang, and Ji Liu. Asynchronous decentralized parallel stochastic gradient descent. In *International Conference on Machine Learning*, pages 3043–3052. PMLR, 2018.

[53] Katherine P Liao, Fina Kurreeman, Gang Li, Grant Duclos, Shawn Murphy, Raul Guzman, Tianxi Cai, Namrata Gupta, Vivian Gainer, Peter Schur, et al. Associations of autoantibodies, autoimmune risk alleles, and clinical diagnoses from the electronic medical records in rheumatoid arthritis cases and non–rheumatoid arthritis controls. *Arthritis & Rheumatism*, 65(3):571–581, 2013.

[54] Tao Lin, Sebastian U Stich, Kumar Kshitij Patel, and Martin Jaggi. Don't use large mini-batches, use local sgd. *arXiv preprint arXiv:1808.07217*, 2018.

[55] Jun Liu, Shuiwang Ji, and Jieping Ye. Multi-task feature learning via efficient $\ell_{2,1}$-norm minimization. In *UAI*, pages 339–348, 2009.

[56] Yaohua Liu, Cameron Nowzari, Zhi Tian, and Qing Ling. Asynchronous periodic event-triggered coordination of multi-agent systems. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 6696–6701. IEEE, 2017.

[57] Ziqi Liu, Yu-Xiang Wang, and Alexander Smola. Fast differentially private matrix factorization. In *Proceedings of the 9th ACM RecSys*, pages 171–178, 2015.

[58] Jian Lou and Yiu-ming Cheung. Uplink communication efficient differentially private sparse optimization with feature-wise distributed data. In *AAAI*, volume 32, 2018.

[59] Jian Lou and Yiu-ming Cheung. An uplink communication-efficient approach to featurewise distributed sparse optimization with differential privacy. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[60] Yucheng Lu and Christopher De Sa. Moniqua: Modulo quantized communication in decentralized sgd. In *ICML*, pages 6415–6425. PMLR, 2020.

[61] Yuan Luo, Faraz S Ahmad, and Sanjiv J Shah. Tensor factorization for precision medicine in heart failure with preserved ejection fraction. *Journal of cardiovascular translational research*, 10(3):305–312, 2017.

[62] Jing Ma, Qiuchen Zhang, Jian Lou, Joyce C Ho, Li Xiong, and Xiaoqian Jiang. Privacy-preserving tensor factorization for collaborative health data analysis. *arXiv preprint arXiv:1908.09888*, 2019.

[63] Jing Ma, Qiuchen Zhang, Joyce C. Ho, and Li Xiong. Spatio-temporal tensor sketching via adaptive sampling. *CoRR*, abs/2006.11943, 2020. URL `https://arxiv.org/abs/2006.11943`.

[64] Jing Ma, Qiuchen Zhang, Jian Lou, Li Xiong, and Joyce C Ho. Communication efficient federated generalized tensor factorization for collaborative health data analytics. In *Proceedings of the Web Conference 2021*, pages 171–182, 2021.

[65] Luis Muñoz-González, Kenneth T Co, and Emil C Lupu. Byzantine-robust federated machine learning through adaptive model averaging. *arXiv preprint arXiv:1909.05125*, 2019.

[66] Yu Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

[67] Feiping Nie, Heng Huang, Xiao Cai, and Chris H Ding. Efficient and robust feature selection via joint $\ell_{2,1}$-norms minimization. In *NeurIPS*, pages 1813–1821, 2010.

[68] Ioakeim Perros, Evangelos E Papalexakis, Fei Wang, Richard Vuduc, Elizabeth Searles, Michael Thompson, and Jimeng Sun. Spartan: Scalable parafac2 for large & sparse data. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 375–384, 2017.

[69] Rimma Pivovarov, Adler J Perotte, Edouard Grave, John Angiolillo, Chris H Wiggins, and Noémie Elhadad. Learning probabilistic phenotypes from heterogeneous ehr data. *Journal of biomedical informatics*, 58:156–165, 2015.

[70] Meisam Razaviyayn, Mingyi Hong, and Zhi-Quan Luo. A unified convergence analysis of block successive minimization methods for nonsmooth optimization. *SIAM Journal on Optimization*, 23(2):1126–1153, 2013.

[71] Sashank J Reddi, Suvrit Sra, Barnabas Poczos, and Alexander J Smola. Proximal stochastic methods for nonsmooth nonconvex finite-sum optimization. In *Advances in Neural Information Processing Systems*, pages 1145–1153, 2016.

[72] Rachel L Richesson, Jimeng Sun, Jyotishman Pathak, Abel N Kho, and Joshua C Denny. Clinical phenotyping in selected national networks: demonstrating the need for high-throughput, portable, and computational methods. *Artificial intelligence in medicine*, 71:57–61, 2016.

[73] Daniel Rothchild, Ashwinee Panda, Enayat Ullah, Nikita Ivkin, Ion Stoica, Vladimir Braverman, Joseph Gonzalez, and Raman Arora. Fetchsgd: Communication-efficient federated learning with sketching. In *ICML*, pages 8253–8265. PMLR, 2020.

[74] Georg S Seyboth, Dimos V Dimarogonas, and Karl H Johansson. Event-based broadcasting for multi-agent average consensus. *Automatica*, 49(1):245–252, 2013.

[75] Kijung Shin, Lee Sael, and U Kang. Fully scalable methods for distributed tensor factorization. *IEEE Transactions on Knowledge and Data Engineering*, 29(1):100–113, 2016.

[76] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pages 3–18. IEEE, 2017.

[77] Nicholas D Sidiropoulos, Lieven De Lathauwer, Xiao Fu, Kejun Huang, Evangelos E Papalexakis, and Christos Faloutsos. Tensor decomposition for signal processing and machine learning. *IEEE Transactions on Signal Processing*, 65 (13):3551–3582, 2017.

[78] Navjot Singh, Deepesh Data, Jemin George, and Suhas Diggavi. Sparq-sgd: Event-triggered and compressed communication in decentralized optimization. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 3449–3456. IEEE, 2020.

[79] Navjot Singh, Deepesh Data, Jemin George, and Suhas Diggavi. Squarm-sgd: Communication-efficient momentum sgd for decentralized optimization. *arXiv preprint arXiv:2005.07041*, 2020.

[80] Sebastian U Stich. Local sgd converges fast and communicates little. In *International Conference on Learning Representations*, 2018.

[81] Sebastian U Stich and Sai Praneeth Karimireddy. The error-feedback framework: Better rates for sgd with delayed gradients and compressed communication. *arXiv preprint arXiv:1909.05350*, 2019.

[82] Sebastian U Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. Sparsified sgd with memory. In *Advances in Neural Information Processing Systems*, pages 4447–4458, 2018.

[83] Hanlin Tang, Shaoduo Gan, Ce Zhang, Tong Zhang, and Ji Liu. Communication compression for decentralized training. *arXiv preprint arXiv:1803.06443*, 2018.

[84] Hanlin Tang, Xiangru Lian, Ming Yan, Ce Zhang, and Ji Liu. $d^2$: Decentralized training over decentralized data. In *ICML*, pages 4848–4856. PMLR, 2018.

[85] Minal Thakkar and Diane C Davis. Risks, barriers, and benefits of ehr systems: a comparative study based on size of hospital. *Perspectives in Health Information Management/AHIMA, American Health Information Management Association*, 3, 2006.

[86] M Alex O Vasilescu and Demetri Terzopoulos. Multilinear analysis of image ensembles: Tensorfaces. In *European conference on computer vision*. Springer, 2002.

[87] Ashish Vulimiri, Carlo Curino, P Brighten Godfrey, Thomas Jungblut, Jitu Padhye, and George Varghese. Global analytics in the face of bandwidth and regulatory constraints. In *12th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 15)*, pages 323–336, 2015.

[88] Yichen Wang, Robert Chen, Joydeep Ghosh, Joshua C Denny, Abel Kho, You Chen, Bradley A Malin, and Jimeng Sun. Rubik: Knowledge guided tensor factorization and completion for health data analytics. In *Proceedings of the 21th ACM SIGKDD*, pages 1265–1274. ACM, 2015.

[89] Yining Wang and Anima Anandkumar. Online and differentially-private tensor decomposition. In *NeurIPS*, pages 3531–3539, 2016.

[90] Wei-Qi Wei and Joshua C Denny. Extracting research-quality phenotypes from electronic health records to support precision medicine. *Genome medicine*, 7(1): 41, 2015.

[91] Xi Wu, Fengan Li, Arun Kumar, Kamalika Chaudhuri, Somesh Jha, and Jeffrey Naughton. Bolt-on differential privacy for scalable stochastic gradient descent-based analytics. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 1307–1322. ACM, 2017.

[92] Chen Xu, Jianqiang Yao, Zhouchen Lin, Wenwu Ou, Yuanbin Cao, Zhirong Wang, and Hongbin Zha. Alternating multi-bit quantization for recurrent neural networks. *ICLR-2018, arXiv preprint arXiv:1802.00150*, 2018.

[93] Xiao Xu, Shu-Xia Li, Haiqun Lin, SL Normand, Tara Lagu, Nihar Desai, Michael Duan, Eugene A Kroch, and Harlan M Krumholz. Hospital phenotypes in the

management of patients admitted for acute myocardial infarction. *Medical care*, 54(10):929–936, 2016.

[94] Yangyang Xu and Wotao Yin. Block stochastic gradient iteration for convex and nonconvex optimization. *SIAM Journal on Optimization*, 2015.

[95] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019.

[96] Yi Yang, Heng Tao Shen, Zhigang Ma, Zi Huang, and Xiaofang Zhou. $\ell_{2,1}$-norm regularized discriminative feature selection for unsupervised learning. In *IJCAI*, volume 22, page 1589, 2011.

[97] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. In *International Conference on Machine Learning*, pages 5650–5659. PMLR, 2018.

[98] Doaa Youssef, Hadeel Abd-Elrahman, Mohamed M Shehab, Mohamed Abd-Elrheem, et al. Incidence of acute kidney injury in the neonatal intensive care unit. *Saudi journal of kidney diseases and transplantation*, 26(1):67, 2015.

[99] Lei Yu, Ling Liu, Calton Pu, Mehmet Emre Gursoy, and Stacey Truex. Differentially private model publishing for deep learning. *arXiv preprint arXiv:1904.02200*, 2019.

[100] Jinshan Zeng, Tim Tsz-Kit Lau, Shaobo Lin, and Yuan Yao. Global convergence of block coordinate descent in deep learning. In *International Conference on Machine Learning*, pages 7313–7323, 2019.

[101] Sixin Zhang, Anna E Choromanska, and Yann LeCun. Deep learning with elastic averaging sgd. In *NeurIPS*, pages 685–693, 2015.

[102] Shandian Zhe, Kai Zhang, Pengyuan Wang, Kuang-chih Lee, Zenglin Xu, Yuan Qi, and Zoubin Ghahramani. Distributed flexible nonlinear tensor factorization. In *Advances in neural information processing systems*, pages 928–936, 2016.

[103] Shuai Zheng, Ziyue Huang, and James T Kwok. Communication-efficient distributed blockwise momentum sgd with error-feedback. In *Advances in Neural Information Processing Systems*, 2019.