Reliability and robustness of federated learning in practical applications

Yann Fraboni

UNIVERSITÉ CÔTE D'AZUR

ÉCOLE DOCTORALE
SCIENCES ET TECHNOLOGIES DE L'INFORMATION ET DE LA COMMUNICATION

THÈSE DE DOCTORAT

Fiabilité et Robustesse de l'Apprentissage Fédéré pour Applications Concrètes

Yann FRABONI

INRIA, Équipe EPIONE

Thèse dirigée par Marco LORENZI

Soutenue le : Jeudi 11 Mai 2023

Présentée en vue de l'obtention du grade de DOCTEUR EN AUTOMATIQUE, TRAITEMENT DU SIGNAL ET DES IMAGES de l'UNIVERSITÉ CÔTE D'AZUR.

Devant le jury composé de :

Pietro MICHIARDI	EURECOM	Rapporteur
Thrasyvoulos SPYROPOULOS	Technical University of Crete	Rapporteur
Marco LORENZI	INRIA	Directeur de thèse
Laetitia KAMENI	Accenture Labs	Co-encadrant
Richard VIDAL	Accenture Labs	Co-encadrant
Sébastien OURSELIN	King's College London	Examinateur
Sai Praneeth KARIMIREDDY	UC Berkeley	Examinateur
Sebastian STICH	CISPA	Examinateur
Aurélien BELLET	INRIA	Invité
Giovanni NEGLIA	INRIA	Invité

FIABILITÉ ET ROBUSTESSE DE L'APPRENTISSAGE FÉDÉRÉ POUR APPLICATIONS CONCRÈTES

Reliability and Robustness of Federated Learning in Practical Applications

Yann FRABONI

Jury :

Président du jury

Pietro MICHIARDI, Professeur, EURECOM

Rapporteurs

Pietro MICHIARDI, Professeur, EURECOM
Thrasyvoulos SPYROPOULOS, Professeur, Technical University of Crete

Examinateurs

Sébastien OURSELIN, Professeur, King's College London
Sai Praneeth KARIMIREDDY, Docteur, University of California, Berkeley
Sebastian STICH, Maître de conférence, CISPA Helmholtz Center for Information Security

Directeur de thèse

Marco LORENZI, Chargé de recherche, Centre INRIA de l'Université Côte d'Azur

Co-encadrant de thèse

Laetitia KAMENI, Cheffe de projet, Accenture Labs Sophia Antipolis
Richard VIDAL, Chef de projet, Accenture Labs Sophia Antipolis

Membres invités

Aurélien BELLET, Chargé de recherche, Centre INRIA de l'Université de Lille
Giovanni NEGLIA, Directeur de recherche, Centre INRIA de l'Université Côte d'Azur

Abstract

Federated Learning has gained popularity in the last years as it enables different clients to jointly learn a global model without sharing their respective data. FL specializes the classical problem of distributed learning, to account for the private nature of clients information (i.e. data and surrogate features), and for the potential data and hardware heterogeneity across clients, which is generally unknown to the server. Within this context, the main objective of this thesis is to present new theoretical and practical results to quantify the impact of the clients' data heterogeneity on the convergence guarantees of federated learning, while investigating the feasibility of critical components for deployment of federated learning in real-world applications.

In the first part of the thesis we study the robustness and variability of federated learning to heterogeneous conditions. To this end, we introduce the notion of stochastic aggregation weights to generalize the aggregation scheme proposed in FEDAVG, along with a novel theory to account asymptotically for the impact of a client sampling scheme on the federated learning convergence guarantees. We then introduce "clustered sampling", a novel client selection scheme generalizing and outperforming the state-of-the-art sampling methods in terms of improved representativity and lower variability. We provide a theoretical justification of clustered sampling, and show faster and smoother convergence as compared to the standard approaches. We further extend the stochastic aggregation scheme of clustered sampling to account for asynchronous client updates and provide the close-form solution of the aggregation weights for unbiased federated optimization of federated learning procedures, such as synchronous and asynchronous federated learning, FEDFIX, or FEDBUFF. In the second part of the thesis, we investigate the reliability of federated learning in practical applications. We introduce informed federated unlearning (IFU), a novel federated unlearning scheme, allowing to remove (unlearn) the contribution of a client from a federated model, with statistical guarantees on the unlearning effectiveness. Finally, we propose two strategies for free-riding attacks and introduce a novel theoretical framework to prove their efficiency. Overall, the work presented in this thesis highlights novel theoretical properties of federated learning, which ultimately allow to deepen our understanding on the robustness and reliability of the federated optimization process in practical application scenarios.

Keywords: federated learning, distributed optimization, heterogeneous data, bias, privacy.

Résumé

L'apprentissage fédéré a gagné en popularité ces dernières années car il permet à différents clients d'apprendre conjointement un modèle global sans partager leurs données respectives. FL se spécialise dans le problème classique de l'apprentissage distribué, pour tenir compte de la nature privée des informations des clients et de l'hétérogénéité potentielle des données et du matériel entre les clients, qui est généralement inconnue du serveur. Dans ce contexte, l'objectif principal de cette thèse est de présenter de nouveaux résultats théoriques et pratiques pour quantifier l'impact de l'hétérogénéité des données clients sur les garanties de convergence de l'apprentissage fédéré, tout en étudiant la faisabilité de composants critiques pour le déploiement de l'apprentissage fédéré dans des applications concrètes.

Dans la première partie de la thèse, nous étudions la robustesse et la variabilité de l'apprentissage fédéré aux données hétérogènes. Nous introduisons la notion de coefficients stochastiques d'agrégation pour généraliser le schéma d'agrégation proposé dans FEDAVG, ainsi qu'une nouvelle théorie pour tenir compte asymptotiquement de l'impact d'une méthode de sélection de clients sur les garanties de convergence de l'apprentissage fédéré. Nous introduisons ensuite « clustered sampling », une nouvelle méthode de sélection de clients généralisant et surpassant les méthodes de l'état de l'art en améliorant la représentativité des clients et en réduisant leur variabilité de sélection. Nous fournissons une justification théorique de clustered sampling et montrons une convergence plus rapide et plus stable par rapport aux approches standard. Nous étendons davantage les coefficients stochastique d'agrégation de clustered sampling pour prendre en compte des contributions asynchrones de clients et fournissons l'expression des poids d'agrégation pour une optimisation fédérée juste des méthodes d'apprentissage standarad, telles que l'apprentissage fédéré synchrone et asynchrone, FEDFIX ou FEDBUFF. Dans la deuxième partie de la thèse, nous étudions la fiabilité de l'apprentissage fédéré dans des applications concrètes. Nous introduisons IFU, un nouveau schéma de désapprentissage fédéré, permettant de supprimer la contribution d'un client à un modèle fédéré, avec des garanties statistiques sur l'efficacité du désapprentissage. Enfin, nous proposons deux stratégies pour les attaques de « free-riding » et introduisons un nouveau cadre théorique pour prouver leur efficacité. Dans l'ensemble, les travaux présentés dans cette thèse mettent en évidence de nouvelles propriétés théoriques de l'apprentissage fédéré, qui permettent d'approfondir notre compréhension de la robustesse et de la fiabilité du processus d'optimisation fédérée dans des scénarios d'applications concrètes.

Mots-clefs : apprentissage fédéré, optimisation distribuée, données hétérogènes, biais, protection des données.

Acknowledgement

I would like to express my sincere gratitude to Marco Lorenzi, Richard Vidal, and Laetitia Kameni for providing me with the opportunity to pursue my thesis in the Epione team at Inria and the Accenture Lab of Sophia Antipolis. I am immensely thankful for the trust and confidence you placed in me for this ambitious project. Marco, I am truly appreciative of your guidance, trust, advice, and insightful discussions. Your emphasis on rigor has significantly enhanced my research skills and instilled in me a commitment to consistently produce work of high standards. Richard and Laetitia, I extend my thanks for believing in my ideas and granting me the autonomy to explore them. Your perspectives on approaching research from an industrial standpoint have been detrimental in ensuring the practical applicability of each project undertaken during my PhD.

I would also like to acknowledge the trust placed in me by Sebastien Ourselin and Michela Antonelli, who recognized the potential of my work on federated unlearning and sought to validate its relevance in the context of medical data. I am proud of having worked with you and of the progress we made.

Furthermore, I am grateful to the jury, particularly Pietro Michiardi and Thrasyvoulos Spyropulos, for dedicating their time to review my thesis, attending the defense, and providing valuable comments and advice. The opportunity to defend my work and engage in discussions with experts in the field has been both challenging and rewarding.

Throughout my PhD journey, I have been fortunate to establish meaningful relationships within Inria, Accenture, and beyond. A special thanks goes to Etrit Haxholli, with whom I was lucky to share an office during this remarkable adventure. Our discussions have had a profound impact on my personal and professional growth. I would also like to express my gratitude to Raphael Paulello, who has been an incredible friend for many years and a constant source of support in my life. I would like to express my appreciation to the Epione team and the Accenture Lab team for their warm welcome, with a special mention of Dimitri Hamzaoui, Hugo Tardiou, Laura Degioanni, Johanna Pinard, Hugo Borne, Jérémy Suini, Alexis Hoss, Charlotte Rodriguez, Haris Pasic, Lucia Innocenti, Riccardo Taiello, Othmane Marfoq, Constantin Philippenko, Lauriane Savaron, and Santiago Silva. Additionally, I would like to express my thanks to all the friends I have made in Nice, with whom I have enjoyed countless vibrant moments: Julie Salles, Clément Piccini, Ambre Jagut, Stéphane Grondin, Leslie Dugast, Sofia Ahmed Ahamada, and Sardor Israelov.

Finally, I would like to convey my sincere appreciation to my family for their unwavering support throughout my journey—before, during, and hopefully after my PhD. Their presence has been a constant source of encouragement.

Financial Support

This work has been supported by the French government, through the 3IA Côte d'Azur Investments in the Future project managed by the National Research Agency (ANR) with the reference number ANR-19-P3IA-0002, and by the ANR JCJC project Fed-BioMed 19-CE45-0006-01. The project was also supported by Accenture. The authors are grateful to the OPAL infrastructure from Université Côte d'Azur for providing resources and support.

Contents

1 Introduction **1**

1.1 Federated Learning 1
1.2 Federated Optimization and FEDAVG 3
1.3 Main Challenges of Federated Learning 4
1.3.1 Challenge 1: Data Heterogeneity 5
1.3.2 Challenge 2: Privacy 6
1.3.3 Challenge 3: Communication 6
1.3.4 Challenge 4: Compliance with Regulations 7
1.4 Objectives and Organization of the Thesis 8
1.5 Publications 10

I Robustness and Variability of Federated Learning to Heterogeneous Conditions **11**

2 A General Theory for Client Sampling in Federated Learning **13**

2.1 Introduction 13
2.2 Background 15
2.2.1 Aggregating clients local updates 16
2.2.2 Unbiased data agnostic client samplings 17
2.2.3 Advanced client sampling techniques 19
2.3 Convergence Guarantees 19
2.3.1 Asymptotic FL convergence with respect to client sampling 19
2.3.2 Application to current client sampling schemes 21
2.4 Experiments on real data 24
2.5 Conclusion 25

3 Clustered Sampling: Low-Variance and Improved Representativity for Clients Selection in Federated Learning **27**

3.1 Introduction 28
3.2 Related Work 29
3.2.1 Aggregating clients local updates 30

3.2.2 Clients' sampling . . . 30
3.2.3 FL convergence with MD client sampling . . . 31
3.2.4 Sampling schemes comparison . . . 32
3.3 Clustered Sampling . . . 32
3.3.1 Definition of clustered sampling . . . 33
3.3.2 Improvements provided by clustered sampling . . . 35
3.4 Clustered Sampling based on Sample Size . . . 36
3.5 Clustered Sampling based on Similarity . . . 38
3.6 Experiments . . . 40
3.7 Discussion and Conclusion . . . 44

4 A General Theory for Federated Optimization with Asynchronous and Heterogeneous Clients Updates 45
4.1 Introduction . . . 46
4.2 Background . . . 48
4.2.1 Federated Optimization Problem . . . 48
4.2.2 Asynchronicity in Clients Updates . . . 49
4.2.3 Server Aggregation Scheme . . . 49
4.2.4 Expressing FL as cumulative GD steps . . . 51
4.2.5 Asynchronous FL as a Sequence of Optimization Problems . . . 51
4.2.6 Formalizing Heterogeneity across Clients . . . 52
4.3 Convergence of Federated Problem (4.2) . . . 53
4.3.1 Assumptions and Property . . . 54
4.3.2 Convergence of Algorithm 4 . . . 55
4.3.3 Sufficient Conditions for Minimizing the Federated Problem (4.2) . . . 56
4.3.4 Relaxed Sufficient Conditions for Minimizing the Federated Problem (4.2) . . . 58
4.4 Applications . . . 62
4.4.1 Heterogeneity of clients hardware and data distributions . . . 62
4.4.2 FEDAVG, Synchronous Federated Learning . . . 64
4.4.3 Asynchronous FEDAVG . . . 64
4.4.4 FEDFIX . . . 66
4.4.5 Generalization . . . 68
4.5 Experiments . . . 69
4.5.1 Experimental Setting . . . 69
4.5.2 Experimental Results . . . 70
4.6 Discussion . . . 73

II Reliability of Federated Learning in Practical Applications 75

5 Sequential Informed Federated Unlearning: Efficient and Provable Client Unlearning in Federated Optimization 77

5.1 Introduction . . . 78
5.2 Background and Related Work . . . 80
5.2.1 Machine Unlearning . . . 80
5.2.2 Federated Optimization and FEDAVG . . . 81
5.2.3 Federated Unlearning . . . 82
5.3 Unlearning a FL client with IFU . . . 83
5.3.1 Theorem 5.1, Bounding the Model Sensitivity . . . 83
5.3.2 Improving the Tightness of the Sensitivity Bound . . . 84
5.3.3 Satisfying Definition 5.1 . . . 85
5.3.4 Informed Federated Unlearning (IFU) . . . 86
5.4 Sequential FU with SIFU . . . 87
5.5 Experiments . . . 90
5.5.1 Experimental Setup . . . 90
5.5.2 Experimental Results . . . 91
5.5.3 Verifying Unlearning through Watermarking . . . 92
5.5.4 Impact of the noise perturbation on SIFU . . . 93
5.6 Conclusions . . . 93

6 Free-rider Attacks on Model Aggregation in Federated Learning 95

6.1 Introduction . . . 96
6.2 Methods . . . 98
6.2.1 Federated learning through model aggregation: FedAvg and FedProx 98
6.2.2 Formalizing Free-rider attacks . . . 98
6.2.3 SGD perturbation of the fair clients local model . . . 99
6.2.4 Plain free-riding . . . 102
6.2.5 Disguised free-riding . . . 104
6.2.6 FedProx . . . 107
6.3 Experiments . . . 107
6.3.1 Experimental Details . . . 108
6.3.2 Free-rider attacks: convergence and performances . . . 110
6.4 Conclusion and discussion . . . 111

7 Conclusion and Perspectives 113

7.1 Summary of the Main Contributions . . . 113
7.2 Perspectives and Future Applications . . . 116
7.2.1 Fully Decentralized Federated Learning . . . 116

7.2.2 Ensuring Fairness . . . 117
7.2.3 Model Personalization with Federated Learning . . . 118
7.3 Final Remarks . . . 119

Bibliography **121**

A Appendix of Chapter 2 **135**
A.1 Client Sampling Schemes Calculus . . . 136
A.1.1 Property A.1 . . . 136
A.1.2 No sampling scheme . . . 137
A.1.3 MD sampling . . . 137
A.1.4 Uniform Sampling . . . 138
A.1.5 Poisson Binomial Distribution . . . 139
A.1.6 Binomial Distribution . . . 140
A.1.7 Clustered Sampling . . . 140
A.1.8 Optimal Sampling . . . 142
A.2 FL Convergence . . . 142
A.2.1 Notations . . . 144
A.2.2 Useful Lemmas . . . 145
A.2.3 Intermediary Theorem . . . 148
A.2.4 Synthesis of local learning rate η_l conditions for Theorem A.1 . . . 153
A.2.5 Theorem 2.1 . . . 153
A.2.6 Application to Clustered Sampling . . . 155
A.2.7 Proof of Corollary 2.1 . . . 157
A.3 Additional experiments . . . 158
A.3.1 Shakespeare dataset . . . 158
A.3.2 CIFAR10 dataset . . . 158

B Appendix of Chapter 3 **163**
B.1 Proof of Theorem 3.2 . . . 163
B.1.1 Proof of Lemma B.1 for Theorem 3.1 . . . 163
B.1.2 Proof of Lemma B.1 for Theorem 3.2 . . . 165
B.2 MD and Clustered Sampling Comparison . . . 166
B.2.1 Client aggregation weight variance . . . 166
B.2.2 Probability for a client to be sampled at least once . . . 167
B.3 Explaining Algorithm 2 and 3 . . . 168
B.3.1 Algorithm 2 . . . 168
B.3.2 Algorithm 3 . . . 169
B.4 Additional Experiments . . . 170
B.4.1 CIFAR10 partitioning illustration . . . 170

B.4.2 Influence of the similarity measure . . . 170
B.4.3 More details on Figure 3.2 . . . 173
B.4.4 Influence of m the number of sampled clients, and N the number of SGD run . . . 173
B.4.5 Local regularization . . . 173

C Appendix of Chapter 4 **179**
C.1 Proof of Theorem 4.1 . . . 179
C.1.1 Bounding the convergence residual Σ . . . 179
C.1.2 Basic Inequalities . . . 180
C.1.3 Additional Notation . . . 181
C.1.4 Useful Lemmas . . . 181
C.1.5 Proof of Theorem 4.1 . . . 188
C.1.6 Simplifying the constraint on the learning rate . . . 190
C.2 Proof of Theorem 4.2 . . . 190
C.2.1 Useful Lemma . . . 190
C.2.2 Proof of Theorem 4.2 . . . 191
C.3 Applying Theorem 4.3 . . . 192
C.3.1 Centralized Learning . . . 192
C.3.2 Unbiased client sampling ($q_i(n) = p_i$) . . . 192
C.3.3 Biased client sampling ($q_i(n) \neq p_i$) . . . 193
C.4 Additional Experiments . . . 193

D Appendix of Chapter 5 **197**
D.1 When fine tuning does not guarantee unlearning: example on linear regression 197
D.2 Forgetting a Single Client with IFU, Proof of Theorem 5.1 . . . 198
D.2.1 Proof of Theorem 5.1 for $K = 1$ . . . 198
D.2.2 Proof of Theorem 5.1 for $K \geq 1$ . . . 199
D.2.3 Local Loss Functions' Regularization and Strong Convexity, Proof of Corollary 5.1 . . . 200
D.2.4 Generalization . . . 200
D.2.5 Calculus simplification with uniform importance . . . 201
D.3 Convergence of SIFU, Theorem 5.3 . . . 201
D.3.1 Intermediate results . . . 201
D.3.2 Proof of Theorem 5.3 . . . 202
D.4 Experiments . . . 203

E Appendix of Chapter 6 **207**
E.1 Complete Proofs for FedAvg . . . 207
E.1.1 Proof of Theorem 6.1 . . . 207

E.1.2 Proof of Theorem 6.2 210
E.1.3 Proof of Theorem 6.3 214
E.1.4 Proof of Corollary 6.1 215
E.1.5 Proof of Corollary 6.2 216
E.2 Complete Proofs for FedProx 218
E.3 Additional experimental results 220
E.3.1 Accuracy Performances 220

1 Introduction

Contents

1.1 Federated Learning 1
1.2 Federated Optimization and FedAvg 3
1.3 Main Challenges of Federated Learning 4
1.3.1 Challenge 1: Data Heterogeneity 5
1.3.2 Challenge 2: Privacy 6
1.3.3 Challenge 3: Communication 6
1.3.4 Challenge 4: Compliance with Regulations 7
1.4 Objectives and Organization of the Thesis 8
1.5 Publications 10

1.1 Federated Learning

The way healthcare is delivered to patients is undergoing great change with electronic health records becoming ubiquitous across almost all medical institutions. Artificial Intelligence (AI)-supported medical analysis entails a large potential to provide insight into diagnosi (Kononenko, 2001; Savage, 2012; Roque et al., 2011) and prognosi (Ebadollahi et al., 2010; Jensen et al., 2012), and can assist in the development of treatments (Bennett et al., 2012; Ramakrishnan et al., 2010). Nevertheless, a typical bottleneck for the development of data-driven approaches in biomedical applications is represented by the need for large datasets to achieve robust and reliable models (R. Wang et al., 2019). As a results, AI-based modeling approaches developed on mono-centric data often fail in generalizing to external cohorts, and lack in robustness and reliability (Sheller et al., 2020). To overcome this issue, an AI model should be ideally trained on very large cohorts ensuring proper representativity of the data variability across clinical conditions, data acquisition protocols, and biases. This is currently a major challenge in healthcare since, besides current research efforts such as the UK Biobank (Sudlow et al., 2015), or ADNI (Petersen et al., 2010), we generally lack data lakes providing large data collections of heterogeneous medical measurements. One of the main reason is practical, since institutions are reluctant to share their data due to the private and sensitive nature of biomedical information. For this reason, during

the past years researchers turned their attention to the paradigm of collaborative learning. Among the different collaborative learning paradigms, federated learning (FL) (McMahan, Moore, et al., 2017) has become increasingly popular as it enables to collaboratively train an AI model without requiring data sharing, thus addressing the problem of data ownership and governance in multi-centric studies. Under the orchestration of a central server, the participating institutions to a FL project collaboratively train a model without ever sharing their data with the server, another institution or a third party. Each institution stores its data locally and shares instead the result of the local training.

The application potential for federated learning in the medical field is important as this technology may allow to operate directly with the patients instead of institutions by using the data that their wearable devices and smartphone collects (J. Xu et al., 2021). In addition, the data samples of a participant are characterized by specific features and variations, which prevents a model trained locally from generalizing to the other clients' data. Instead, with federated learning, the trained model can fit the samples of every client (Jianyu Wang, Q. Liu, et al., 2020; Xiang Li et al., 2020), which leads to better generalization to unknown data. More generally, federated learning can also be used for finance risk prediction for reinsurance, pharmaceuticals discovery, electronic health records mining, medical data segmentation, smart manufacturing, or features on mobile phones (Kairouz et al., 2019; T. Li, Sahu, Talwalkar, et al., 2020).

In spite of the large potential and interest of FL in medical applications, there are currently numerous challenges that must be addressed in order to allow the successful adoption of this technology in real-world applications. In particular, the dataset of the different participants often has a data distribution specific to each of them. This heterogeneity in the clients' datasets makes more challenging for federated learning to accommodate the participants' data in the global model than with centralized learning. In this setting, federated learning often leads to slower and less stable learning. In addition, a federated optimization scheme needs to be properly designed to fit every data point without favoring some participants. Also, due to the physical locations of the participants and their hardware constraints, clients may compute and communicate their contributions in significantly different time. Finally, work is needed to investigate the robustness of federated learning to current type of attacks and ensure that no information from the participants' data can be leaked by sharing their local information. Given the set of challenges related to the effective exploitation of federated learning in real-world applications, the goal of this thesis is to investigate novel methodology to address the problem of heterogeneity, stability and security of FL in critical setting.

In the rest of this chapter, we introduce the theoretical background of federated learning (Section 1.2), and illustrate the main methodological bottlenecks addressed in this work (Section 1.3). Finally we provide an outline of the rest of the manuscript (Section 1.4).

1.2 Federated Optimization and FedAvg

The work of Kairouz et al. (2019) proposes the following broad definition of federated learning.

> **Federated learning** is a machine learning setting where multiple entities (clients) collaborate in solving a machine learning problem, under the coordination of a central server or service provider. Each client's raw data is stored locally and not exchanged or transferred; instead, focused updates intended for immediate aggregation are used to achieve the learning objective.

This definition highlights the key point behind federated learning: a client never shares its data. Hence, clients have total governance over their data as they are guaranteed by design that their data is solely used to compute their local work. Indeed, the main drawback behind centralizing the clients' data on a server is that clients lose control on how the model is trained and on whether the server uses their data for other applications or not, e.g. selling it to a third-party.

We formalize the optimization problem jointly minimized across clients with federated learning and its learning setup as follows. We consider a set I of n clients each respectively owning a dataset $\mathcal{D}_i$ composed of n_i data samples. Federated learning aims at optimizing the average of each client's local loss function $\mathcal{L}_i$ weighted by factors p_i such that $\sum_{i=1}^{n} p_i = 1$, i.e.

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{i \in I} p_i \mathcal{L}_i(\boldsymbol{\theta}), \tag{1.1}$$

where $\boldsymbol{\theta}$ represents the model parameters. The weight p_i can be interpreted as the importance given by the server to client i in the federated optimization problem. While any combination of $\{p_i\}$ is possible, we note that in practice, either (a) every client has equal importance, i.e. $p_i = 1/n$, or (b) every data point is equally important, i.e. $p_i = n_i/M$ with $M = \sum_{i \in I} n_i$.

Among the numerous federated optimization schemes proposed in the federated learning literature, federated averaging (FedAvg) (McMahan, Moore, et al., 2017) is currently the most popular approach to solve the optimization problem (1.1). FedAvg is based on the iterative averaging of the clients models' parameters, after locally updating each client's model for a given number of training epochs.

We consider the following optimization procedure that covers FedAvg and other federated optimization schemes. A server orchestrates the training procedure to estimate a global model across clients, by repeating the steps in Algorithm 1 at each iteration step t until training is completed. Algorithm 1 highlights the coordinating role of the central server in

Algorithm 1 Federated optimization procedure

Require: A set of I participating clients

1: **Client selection:** In function of the clients availability, the server selects all or a subset of the clients for participation.
2: **Broadcast:** Participating clients download the current global model weights $\boldsymbol{\theta}^t$ and the training instructions of the server for this optimization round.
3: **Client computation:** Each participating client locally computes its update to the global model $\boldsymbol{\theta}_i^{t+1}$ by executing its training instructions, e.g. a fixed amount of stochastic gradient descents (SGD) with FEDAVG.
4: **Model update:** The server collects the updated model $\boldsymbol{\theta}_i^{t+1}$ of each participating clients and creates the new global model $\boldsymbol{\theta}^{t+1}$ based on these updates. With FEDAVG, the new global model parameters $\boldsymbol{\theta}^{t+1}$ are estimated as a weighted average of the clients' ones, i.e.

$$\boldsymbol{\theta}^{t+1} = \sum_{i \in I} p_i \boldsymbol{\theta}_i^{t+1}. \tag{1.2}$$

federated learning. Indeed, the server orchestrates the optimization and decides how the new global model parameters are computed based on the information provided by the clients. Several production-oriented federated learning platforms based on this optimization procedure are being developed to enable industrial applications of federated learning including FED-BIOMED [1] (Silva, Altmann, et al., 2020), NVIDIA Clara [2] , and Flower (Beutel et al., 2020) .

Surveys on federated learning like the one of Kairouz et al. (2019) distinguish two types of federated learning applications: (i) the *cross-silo* setting where few clients are participating, typical of medical applications involving a limited number of hospitals, and (ii) the *cross-device* one, where the number of clients can be very high, typical of smartphone applications and connected devices. The approaches in this thesis are general, and can be seamlessly applied to both cross-silo and cross-device setting. We note however that this research was conducted with a specific focus on medical applications, while contributing to the development of the federated learning plateform for medical data FED-BIOMED (Silva, Altmann, et al., 2020).

1.3 Main Challenges of Federated Learning

We next describe four of the core challenges of federated optimization, which distinguishes federated learning from distributed learning. Whenever describing a new challenge, we

[1] https://fedbiomed.gitlabpages.inria.fr
[2] https://developer.nvidia.com/blog/federated-learning-clara/

also discuss the currently available technologies to solve or mitigate its impact on federated optimization.

1.3.1 Challenge 1: Data Heterogeneity

The combination of each patient's anatomy, physiology, and medical history, makes every patient's data unique. Moreover, the amount of measurements across patients may vary significantly, as, for example, an healthy individual is less likely to undergo medical examination. Within this setting, federated learning cannot rely on the assumption frequently used in distributed optimization that clients have independent and identically distributed (iid) data. This constraint introduces important challenges in the study of the theoretical properties of federated optimization, and requires the design of federated learning frameworks to specifically account for data heterogeneity. Indeed, it can be demonstrated that otherwise the federated training procedure converges to a suboptimal model of the federated problem described in equation (1.1), where clients are not represented according to their importance p_i, thus introducing a *bias* (Jianyu Wang, Q. Liu, et al., 2020). As a result, some clients may also be underrepresented and, as such, the training procedure is also characterized as *unfair* to these clients. For example, the work of Jianyu Wang, Q. Liu, et al. (2020) proves that FEDAVG is biased when clients perform heterogeneous amount of local work, or when performing local optimization with solvers like Adam (Kingma and Ba, 2014). Nonetheless, the authors of Jianyu Wang, Q. Liu, et al. (2020) also prove that, under minor modifications to the learning procedure, FEDAVG remains unbiased with these learning scenarios and converges to a stationary point of the federated problem (1.1). Another example is the work of Cho et al. (2020) which also shows that, to guarantee fairness, clients should be sampled according to their importance p_i in expectation.

Other definitions of fairness have been proposed in the federated learning literature. For example, the works of T. Li, Sanjabi, et al. (2019) and T. Li, Hu, et al. (2021) focus on minimizing the discrepancy between performances across clients. In that case, a trained model is deemed fair if it has identical training loss or testing accuracy for every client. To this end, the server dynamically updates the way a client's local work is considered in the aggregation to create the new global model, thus solving an optimization problem different from the one in equation (1.1). This example shows how the definition of fairness may impact the federated solution, which may ultimately not correspond with a stationary point of the federated problem (1.1).

In this thesis we focus on the notion of fairness based on preserving the client representativity associated with the optimization problem (1.1). In this setting, proving the fairness of a

federated training routine thus requires to demonstrate the convergence of federated learning to the related.

1.3.2 Challenge 2: Privacy

Federated learning guarantees to every client governance over its data without requiring sharing with the server or any third party. Nevertheless, federated learning still requires the sharing of model updates with the server, which is supposed to carry significantly less sensitive information than the training data itself (Carlini et al., 2019b). Nevertheless, model parameters can nonetheless reveal sensitive information, either to a malicious client, or to the central server itself (McMahan, Ramage, et al., 2017). Moreover, without being able to verify clients' data and local work, the server does not have guarantees about the compliance of the clients' work to the prescribed federated routine. This critical aspect opens the way to ill intentioned participants to disguise their contribution to the FL process (Lyu et al., 2020). Numerous types of attacks on federated learning have been proposed in the literature to allow an ill intentioned participant to recover information regarding other clients' data (Z. Wang et al., 2019; Hitaj et al., 2017; Matt Fredrikson et al., 2015). Different kind of attacks aim instead at manipulating/sabotaging the federated learning routine associated with problem (1.1) to tamper the predictive capabilities of the trained model. For example, the attacker's aim can be to favor some classes or to misclassify a set of chosen inputs with high-confidence during inference (Bhagoji et al., 2019; B. Li et al., 2016; Yin et al., 2018; Xie et al., 2019; Shen et al., 2016). Current research aim at developing methods to enhance the safety and security of federated learning and prevent these attacks, based on the introduction of cryptographic primitives such as secure multiparty computation (Cramer, I. B. Damgård, et al., 2015; I. Damgård et al., 2012; Lindell, 2005) or by using privacy-preserving optimization based on differential privacy (Dwork, 2008; Noble et al., 2022; Wei et al., 2020), or decentralization techniques (Cyffers and Bellet, 2022; Zantedeschi et al., 2020). Nevertheless, these approaches often provide privacy at the cost of reduced model performance or system efficiency. Understanding and balancing these trade-offs, both theoretically and empirically, is a considerable challenge in realizing secure federated learning systems.

1.3.3 Challenge 3: Communication

Communication can also be a primary bottleneck for federated learning since wireless and other end-user connections may operate at variable communication rates while being potentially unreliable. For example, when jointly optimizing a model with participants

from remote geographic locations, the work of Silva, Altmann, et al. (2020) shows that a client's communication time is proportional to the distance to the server. Moreover, the bandwidth capacity of the aggregating server may impose constraints on the number of clients the server can communicate with at the same time. These considerations lead to significant interest in studying novel approaches to optimize the number and bandwidth of communications at every step of the federated learning process. One of the most popular communication reduction strategies proposed with FEDAVG consists in limiting the frequency of communications at the expense of increased computation on the clients side. This is usually achieved by asking the clients to perform multiple iterations of local gradient descent before communicating their updates. To further reduce the number of communications, the server can select a subset of clients participating at every iteration. This strategy, called *client sampling* (T. Li, Sahu, Zaheer, Sanjabi, Talwalkar, and Smith, 2018a; W. Chen et al., 2020; Cho et al., 2020; Xiang Li et al., 2020), enables to speed-up the aggregation process of FEDAVG, while requiring less contributions from participants. Another drawback of FEDAVG concerns the time needed to complete an optimization round, as the server must wait for all the participating clients to perform their local work to synchronize their update and create a new global model. As a consequence, due to the potential heterogeneity of the hardware across clients, the time for an optimization round is conditioned to the one of the slowest update, while the fastest clients stay idle once they have sent their updates to the server. To address these limitations, asynchronous federated learning has been proposed to take full advantage of the clients computing capabilities (Y. Chen et al., 2020; W. Wu et al., 2020; Lu et al., 2020). In the asynchronous setting, whenever the server receives a client's contribution, it creates a new global model and sends it back to the client. In this way, clients are never idle and always perform local work on a different version of the global model.

1.3.4 Challenge 4: Compliance with Regulations

With the emergence of new data regulations, such as the EU General Data Protection Regulation (GDPR) (Voigt and Von dem Bussche, 2017) or the California Consumer Privacy Act (CCPA) (Harding et al., 2019), the storage and processing of sensitive personal data is often subject of strict constraints and restrictions. For example, the "right to be forgotten" requires that personal data must be erased upon request from the concerned individuals, with subsequent potential implications on machine learning models trained by using this data. Machine Unlearning (MU) is an emerging discipline that studies methods to ideally remove the contribution of a given data instance used to train a machine learning model. Current MU approaches are essentially based on routines that modify the weights of models trained on a given dataset in order to guarantee the unlearning of a given data point, i.e. to

	Challenge 1: Data Hetero-geneity	Challenge 2: Privacy	Challenge 3: Communic-ation	Challenge 4: Compliance with Regulations
Chapter 2 on Client Sampling	✓		✓	
Chapter 3 on Clustered Sampling	✓		✓	
Chapter 4 on Asynchronous FEDAVG	✓		✓	
Chapter 5 on Federated Unlearning	✓	✓	✓	✓
Chapter 6 on Free-riders	✓	✓	✓	✓

Tab. 1.1.: Summary of the main contributions of this thesis.

obtain a model equivalent to an hypothetical one trained without this data point (Cao and J. Yang, 2015; Bourtoule et al., 2021). Providing risk-less deployments of federated learning in the real-world, it is of crucial importance to extend MU to guarantee the unlearning of clients wishing to opt-out from a collaborative training routine.

1.4 Objectives and Organization of the Thesis

We have seen that federated learning is essential to provide optimization for applications where the data cannot be centralized, as well as to guarantee to clients the governance of their data while increasing data privacy. Within this context, the main objective of this thesis is to present new theoretical and practical results to quantify the impact of the clients' data heterogeneity on the convergence guarantees of federated learning, while investigating the feasibility of critical components for deployment of federated learning in real-world applications. This manuscript is organized in two parts. In the first part of the thesis, we study the robustness and variability of federated learning to heterogeneous setting (Chapter 2 to 4), while, in the second part, we investigate the reliability of federated learning in practical applications (Chapter 5 and 6).

First, we present in **Chapter 2** a novel decomposition theorem for the convergence of federated learning, allowing to clearly quantify the impact of a client sampling scheme on the global model update. We provide a theoretical ground on the relationship between federated learning convergence and the covariance between the aggregation weights. We show that our theory is general and can be applied to existing client sampling schemes. When applied to Multinomial Distribution (MD) and Uniform sampling, the two default

client sampling schemes of federated learning, our results suggest that MD sampling should be used as default sampling scheme, due to the resilience to the changes in data ratio during the learning process, while Uniform sampling is superior only in the special case when clients have the same amount of data.

Second, we present in **Chapter 3** *clustered sampling*, a novel client sampling scheme. When compared with MD sampling, the state-of-the-art investigated in Chapter 2, we prove that clustered sampling leads to better client representatitivity and to reduced variance of the clients' stochastic aggregation weights in federated learning. Through a series of experiments in non-iid and unbalanced scenarios, we demonstrate that model aggregation through clustered sampling consistently leads to better training convergence and variability when compared to standard sampling approaches.

Third, we present in **Chapter 4**, a novel framework to study asynchronous federated learning optimization with delays in gradient updates. Our theoretical framework extends the standard FEDAVG aggregation scheme by introducing stochastic aggregation weights to represent the variability of the clients update time. Our formalism applies to the general federated setting where clients have heterogeneous datasets and perform at least one step of stochastic gradient descent (SGD). We demonstrate convergence for such a scheme and provide sufficient conditions for the related minimum to be the optimum of the federated problem. We show that our general framework applies to existing optimization schemes including centralized learning, FEDAVG, asynchronous FEDAVG, and FEDBUFF (J. Nguyen et al., 2021). The theory here provided allows drawing meaningful guidelines for designing a federated learning experiment in heterogeneous conditions. In particular, we develop in this work FEDFIX, a novel extension of FEDAVG enabling efficient asynchronous federated training while preserving the convergence stability of synchronous aggregation.

Fourth, we present in **Chapter 5** Informed Federated Unlearning (IFU), a novel efficient and quantifiable federated unlearning (FU) approach. Upon unlearning request from a given client, IFU identifies the optimal federated learning iteration from which federated learning has to be reinitialized, with unlearning guarantees obtained through a randomized perturbation mechanism. The theory of IFU is also extended to account for sequential unlearning requests. Experimental results on different tasks and dataset show that IFU leads to more efficient unlearning procedures as compared to state-of-the-art FU approaches.

Fifth, we introduce in **Chapter 6** the first theoretical and experimental analysis of free-rider attacks on federated learning and provide formal guarantees for these attacks to converge to the aggregated models of the fair participants. We first show that a straightforward implementation of this attack can be simply achieved by not updating the local parameters during the iterative federated optimization. As this attack can be detected by adopting

simple countermeasures at the server level, we subsequently study more complex disguising schemes based on stochastic updates of the free-rider parameters.

Finally, we conclude the manuscript in Chapter 7 by summarizing the main contributions of this work. We also present potential applications of our methods and build upon their limitations to propose further research perspectives.

1.5 Publications

The contributions of this manuscript led to the following publications and submissions in conferences and peer-reviewed journals.

- (Fraboni, Vidal, Kameni, et al., 2022a) *A General Theory for Client Sampling in Federated Learning*. Yann Fraboni, Richard Vidal, Laetitia Kameni, Marco Lorenzi. International Workshop on Trustworthy Federated Learning in Conjunction with IJCAI 2022

- (Fraboni, Vidal, Kameni, et al., 2021) *Clustered Sampling: Low-Variance and Improved Representativity for Clients Selection in Federated Learning*. Yann Fraboni, Richard Vidal, Laetitia Kameni, Marco Lorenzi. Proceedings of the 38th International Conference on Machine Learning, PMLR 139:3407-3416, 2021.

- (Fraboni, Vidal, Kameni, et al., 2022b) *A General Theory for Federated Optimization with Asynchronous and Heterogeneous Clients Updates*. Yann Fraboni, Richard Vidal, Laetitia Kameni, Marco Lorenzi. ArXiv. Under review.

- (Fraboni, Vidal, Kameni, et al., 2022c) *Sequential Informed Federated Unlearning: Efficient and Provable Client Unlearning in Federated Optimization*. Yann Fraboni, Richard Vidal, Laetitia Kameni, Marco Lorenzi. ArXiv. Under review.

- (Fraboni, Vidal, and Lorenzi, 2021) *Free-rider Attacks on Model Aggregation in Federated Learning*. Yann Fraboni, Richard Vidal, Marco Lorenzi. Proceedings of The 24th International Conference on Artificial Intelligence and Statistics, PMLR 130:1846-1854, 2021.

Part I

Robustness and Variability of Federated Learning to Heterogeneous Conditions

2 A General Theory for Client Sampling in Federated Learning

Contents

2.1 Introduction **13**
2.2 Background **15**
2.2.1 Aggregating clients local updates 16
2.2.2 Unbiased data agnostic client samplings 17
2.2.3 Advanced client sampling techniques 19
2.3 Convergence Guarantees **19**
2.3.1 Asymptotic FL convergence with respect to client sampling . . 19
2.3.2 Application to current client sampling schemes 21
2.4 Experiments on real data **24**
2.5 Conclusion **25**

In this chapter, we provide a general theoretical framework to quantify the impact of a client sampling scheme on the federated optimization. Our goal is to measure the impact of the clients data heterogeneity on the convergence speed of federated learning. First, we provide a unified theoretical ground for previously reported sampling schemes experimental results on the relationship between FL convergence and the variance of the aggregation weights. Second, we prove for the first time that the quality of FL convergence is also impacted by the resulting *covariance* between aggregation weights. This chapter is published at the International Workshop on Trustworthy Federated Learning in Conjunction with IJCAI 2022 (FL-IJCAI'22) as Fraboni, Vidal, Kameni, et al. (2022a).

2.1 Introduction

Federated Learning (FL) has gained popularity in the last years as it enables different clients to jointly learn a global model without sharing their respective data. Among the different FL

approaches, federated averaging (FEDAVG) has emerged as the most popular optimization scheme (McMahan, Moore, et al., 2017). An optimization round of FEDAVG requires data owners, also called clients, to receive from the server the current global model which they update on a fixed amount of Stochastic Gradient Descent (SGD) steps before sending it back to the server. The new global model is then created as the weighted average of the client updates, according to their data ratio. FL specializes the classical problem of distributed learning (DL), to account for the private nature of clients information (i.e. data and surrogate features), and for the potential data and hardware heterogeneity across clients, which is generally unknown to the server.

In FL optimization, FEDAVG was first proven to converge experimentally (McMahan, Moore, et al., 2017), before theoretical guarantees were provided for any non-iid federated dataset (Jianyu Wang, Q. Liu, et al., 2020; Karimireddy et al., 2020; Haddadpour and Mahdavi, 2019; Khaled et al., 2020a). A drawback of naive implementations of FEDAVG consists in requiring the participation of all the clients to every optimization round. As a consequence, the efficiency of the optimization is limited by the communication speed of the slowest client, as well as by the server communication capabilities. To mitigate this issue, the original FEDAVG algorithm already contemplated the possibility of considering a random subset of m clients at each FL round. It has been subsequently shown that, to ensure the convergence of FL to its optimum, clients must be sampled such that in expectation the resulting global model is identical to the one obtained when considering all the clients (Jianyu Wang, Q. Liu, et al., 2020; Cho et al., 2020). Clients sampling schemes compliant with this requirement are thus called *unbiased*. Due to its simplicity and flexibility, the current default unbiased sampling scheme consists in sampling m clients according to a Multinomial Distribution (MD), where the sampling probability depends on the respective data ratio (T. Li, Sahu, Zaheer, Sanjabi, Talwalkar, and Smith, 2018a; Jianyu Wang, Q. Liu, et al., 2020; Xiang Li et al., 2020; Haddadpour and Mahdavi, 2019; T. Li, Sanjabi, et al., 2020; Jianyu Wang and Joshi, 2018; Fraboni, Vidal, Kameni, et al., 2021). Nevertheless, when clients have identical amount of data, clients can also be sampled uniformly without replacement (Xiang Li et al., 2020; Karimireddy et al., 2020; Reddi et al., 2021; Rizk et al., 2020). In this case, Uniform sampling has been experimentally shown to yield better results than MD sampling (Xiang Li et al., 2020).

Previous works proposed unbiased sampling strategies alternative to MD and Uniform sampling with the aim of improving FL convergence. In Fraboni, Vidal, Kameni, et al. (2021), MD sampling was extended to account for clusters of clients with similar data characteristics, while in W. Chen et al. (2020), clients sampling probabilities are defined depending on the Euclidean norm of the clients local work. While these works are based on the definition and analysis of specific sampling procedures, aimed at satisfying a given

FL criterion, there is currently a need for a general theoretical framework to elucidate the impact of client sampling on FL convergence.

The main contribution of this chapter consists in deriving a general theoretical framework for FL optimization allowing to clearly quantify the impact of client sampling on the global model update at any FL round. This contribution has important theoretical and practical implications. First, we demonstrate the dependence of FL convergence on the variance of the aggregation weights. Second, we prove for the first time that the convergence speed is also impacted through sampling by the resulting *covariance* between aggregation weights. From a practical point of view, we establish both theoretically and experimentally that client sampling schemes based on aggregation weights with sum different than 1 are less efficient. We also prove that MD sampling is outperformed by Uniform sampling only when clients have identical data ratio. Finally, we show that the comparison between different client sampling schemes is appropriate only when considering a small number of clients. Our theory ultimately shows that MD sampling should be used as default sampling scheme, due to the favorable statistical properties and to the resilience to FL applications with varying data ratio and heterogeneity.

This chapter is structured as follows. In Section 2.2, we provide formal definitions for FL, unbiased client sampling, and for the server aggregation scheme. In Section 2.3, we introduce our convergence guarantees (Theorem 2.1) relating the convergence of FL to the aggregation weight variance of the client sampling scheme. Consistently with our theory, in Section 2.4, we experimentally demonstrate the importance of the clients aggregation weights variance and covariance on the convergence speed, and conclude by recommending Uniform sampling for FL applications with identical client ratio, and MD sampling otherwise.

2.2 Background

Before investigating in Section 2.3 the impact of client sampling on FL convergence, we recapitulate in Section 2.2 the current theory behind FL aggregation schemes for clients local updates. We then introduce a formalization for *unbiased* client sampling.

2.2.1 Aggregating clients local updates

In FL, we consider a set I of n clients each respectively owning a dataset $\mathcal{D}_i$ composed of n_i samples. FL aims at optimizing the average of each clients local loss function weighted by p_i such that $\sum_{i=1}^{n} p_i = 1$, i.e.

$$\mathcal{L}(\theta) = \sum_{i=1}^{n} p_i \mathcal{L}_i(\theta), \tag{2.1}$$

where θ represents the model parameters. The weight p_i can be interpreted as the importance given by the server to client i in the federated optimization problem. While any combination of $\{p_i\}$ is possible, we note that in practice, either (a) every device has equal importance, i.e. $p_i = 1/n$, or (b) every data point is equally important, i.e. $p_i = n_i/M$ with $M = \sum_{i=1}^{n} n_i$. Unless stated otherwise, in the rest of this work, we consider to be in case (b), i.e. $\exists i,\ p_i \neq 1/n$.

In this setting, to estimate a global model across clients, FEDAVG (McMahan, Moore, et al., 2017) is an iterative training strategy based on the aggregation of local model parameters. At each iteration step t, the server sends the current global model parameters θ^t to the clients. Each client updates the respective model by minimizing the local cost function $\mathcal{L}_i(\theta)$ through a fixed amount K of SGD steps initialized with θ^t. Subsequently each client returns the updated local parameters θ_i^{t+1} to the server. The global model parameters θ^{t+1} at the iteration step $t+1$ are then estimated as a weighted average:

$$\theta^{t+1} = \sum_{i=1}^{n} p_i \theta_i^{t+1}. \tag{2.2}$$

To alleviate the clients workload and reduce the amount of overall communications, the server often considers $m \leq n$ clients at every iteration. In heterogeneous datasets containing many workers, the percentage of sampled clients m/n can be small, and thus induce important variability in the new global model, as each FL optimization step necessarily leads to an improvement on the m sampled clients to the detriment of the non-sampled ones. To solve this issue, Reddi et al. (2021), Karimireddy et al. (2020), and Jianyu Wang, Tantia, et al. (2020) propose considering an additional learning rate η_g to better account for the clients update at a given iteration. We denote by $\omega_i(S_t)$ the stochastic aggregation weight of client i given the subset of sampled clients S_t at iteration t . The server aggregation scheme can be written as:

$$\theta^{t+1} = \theta^t + \eta_g \sum_{i=1}^{n} \omega_i(S_t)(\theta_i^{t+1} - \theta^t). \tag{2.3}$$

2.2.2 Unbiased data agnostic client samplings

Tab. 2.1.: Synthesis of statistical properties of different sampling schemes.

Sampling	$\mathrm{Var}\left[\omega_i(S_t)\right]$	α	$\mathrm{Var}\left[\sum_{i=1}^{n}\omega_i(S_t)\right]$
Full participation	$=0$	$=0$	$=0$
MD	$=-\frac{1}{m}p_i^2+\frac{1}{m}p_i$	$=1/m$	$=0$
Uniform	$=\left(\frac{n}{m}-1\right)p_i^2$	$=\frac{n-m}{m(n-1)}$	$=\frac{n-m}{m(n-1)}\left[n\sum_{i=1}^{n}p_i^2-1\right]$

While FEDAVG was originally based on the uniform sampling of clients (McMahan, Moore, et al., 2017), this scheme has been proven to be biased and converge to a suboptimal minima of problem (2.1) (Jianyu Wang, Q. Liu, et al., 2020; Cho et al., 2020; Xiang Li et al., 2020). This was the motivation for Xiang Li et al. (2020) to introduce the notion of *unbiasedness*, where clients are considered in expectation subject to their importance p_i, according to Definition 2.1 below. Unbiased sampling guarantees the optimization of the original FL cost function, while minimizing the number of active clients per FL round. We note that unbiased sampling is not necessarily related to the clients distribution, as this would require to know beforehand the specificity of the clients' datasets.

Unbiased sampling methods (T. Li, Sahu, Zaheer, Sanjabi, Talwalkar, and Smith, 2018a; Xiang Li et al., 2020; Fraboni, Vidal, Kameni, et al., 2021) are currently among the standard approaches to FL, as opposed to *biased* approaches, known to over- or under-represent clients and lead to suboptimal convergence properties (McMahan, Moore, et al., 2017; Nishio and Yonetani, 2019; Jeon et al., 2020; Cho et al., 2020), or to methods requiring additional computation work from clients (W. Chen et al., 2020).

Definition 2.1 (Unbiased Sampling)**.** *A client sampling scheme is said unbiased if the expected value of the client aggregation is equal to the global deterministic aggregation obtained when considering all the clients, i.e.*

$$\mathbb{E}_{S_t}\left[\sum_{i=1}^{n} w_i(S_t)\theta_i^t\right] \coloneqq \sum_{i=1}^{n} p_i\theta_i^t, \tag{2.4}$$

where $w_j(S_t)$ is the aggregation weight of client j for subset of clients S_t.

The sampling distribution uniquely defines the statistical properties of stochastic weights. In this setting, unbiased sampling guarantees the equivalence between deterministic and stochastic weights in expectation. Unbiased schemes of primary importance in FL are MD

and Uniform sampling, for which we can derive a close form formula for the aggregation weights :

MD sampling. This scheme considers $l_1, ..., l_m$ to be the m iid sampled clients from a Multinomial Distribution with support on $\{1, ..., m\}$ satisfying $\mathbb{P}(l_k = i) = p_i$ (Jianyu Wang, Q. Liu, et al., 2020; T. Li, Sahu, Zaheer, Sanjabi, Talwalkar, and Smith, 2018a; Xiang Li et al., 2020; Haddadpour and Mahdavi, 2019; T. Li, Sanjabi, et al., 2020; Jianyu Wang and Joshi, 2018; Fraboni, Vidal, Kameni, et al., 2021). By definition, we have $\sum_{i=1}^{n} p_i = 1$, and the clients aggregation weights take the form:

$$\omega_i(S_t) = \frac{1}{m} \sum_{k=1}^{m} \mathbb{I}(l_k = i). \tag{2.5}$$

Uniform sampling. This scheme samples m clients uniformly without replacement. Since in this case a client is sampled with probability $p(\{i \in S_t\}) = m/n$, the requirement of Definition 2.1 implies:

$$\omega_i(S_t) = \mathbb{I}(i \in S_t) \frac{n}{m} p_i. \tag{2.6}$$

We note that this formulation for Uniform sampling is a generalization of the scheme previously used for FL applications with identical client importance, i.e. $p_i = 1/n$ (Karimireddy et al., 2020; Xiang Li et al., 2020; Reddi et al., 2021; Rizk et al., 2020). We note that $\mathrm{Var}\left[\sum_{i=1}^{n} \omega_i(S_t)\right] = 0$ if and only if $p_i = 1/n$ for all the clients as, indeed, $\sum_{i=1}^{n} \omega_i(S_t) = m \frac{n}{m} \frac{1}{n} = 1$

With reference to equation (2.3), we note that by setting $\eta_g = 1$, and by imposing the condition $\forall S_t,\ \sum_{i=1}^{n} \omega_i(S_t) = 1$, we retrieve equation (2.2). This condition is satisfied for example by MD sampling and Uniform sampling for identical clients importance.

We finally note that the covariance of the aggregation weights for both MD and Uniform sampling satisfies Assumption 2.1.

Assumption 2.1 (Client Sampling Covariance)**.** *There exists a constant α such that the client sampling covariance satisfies* $\forall i \neq j,\ \mathrm{Cov}\left[\omega_i(S_t)\right] \omega_j(S_t) = -\alpha p_i p_j$.

We provide in Table 2.1 the derivation of α and the resulting covariance for these two schemes with calculus detailed in Appendix A.1. Furthermore, this property is common to a variety of sampling schemes, for example based on Binomial or Poisson Binomial distributions (detailed derivations can be found in Appendix A.1). Following this consideration, in addition to Definition 2.1, in the rest of this chapter we assume the additional requirement for a client sampling scheme to satisfy Assumption 2.1.

2.2.3 Advanced client sampling techniques

Importance sampling for centralized SGD Zhao and T. Zhang (2015) and Csiba and Richtárik (2018) has been developed to reduce the variance of the gradient estimator in the centralized setting and provide faster convergence. According to this framework, each data point is sampled according to a probability based on a parameter of its loss function (e.g. its Lipschitz constant), in opposition to classical sampling where clients are sampled with same probability. These works cannot be seamlessly applied in FL, since in general no information on the clients loss function should be disclosed to the server. Therefore, the operation of client sampling in FL cannot be seen as an extension of importance sampling. Regarding advanced FL client sampling, Fraboni, Vidal, Kameni, et al. (2021) extended MD sampling to account for collections of sampling distributions with varying client sampling probability. From a theoretical perspective, this approach was proven to have identical convergence guarantees of MD sampling, with albeit experimental improvement justified by lower variance of the clients' aggregation weights. In W. Chen et al. (2020), clients probability are set based on the euclidean norm of the clients local work. We show in Appendix A.1 that these advanced client sampling strategies also satisfy our covariance assumption 2.1, and are thus encompassed by the general theory developed in Section 2.3.

2.3 Convergence Guarantees

Based on the assumptions introduced in Section 2.2, in what follows we elaborate a new theory relating the convergence of FL to the statistical properties of client sampling schemes. In particular, Theorem 2.1 quantifies the asymptotic relationship between client sampling and FL convergence.

2.3.1 Asymptotic FL convergence with respect to client sampling

To prove FL convergence with client sampling, this chapter relies on the following three assumptions (Jianyu Wang, Q. Liu, et al., 2020; T. Li, Sahu, Zaheer, Sanjabi, Talwalkar, and Smith, 2018a; Karimireddy et al., 2020; Haddadpour and Mahdavi, 2019; J. Wang et al., 2019; S. Wang et al., 2019):

Assumption 2.2 (Smoothness)**.** *The clients local objective function is L-Lipschitz smooth, that is,* $\forall i \in \{1, ..., n\},\ \|\nabla \mathcal{L}_i(x) - \nabla \mathcal{L}_i(y)\| \leq L \|x - y\|$.

Assumption 2.3 (Bounded Dissimilarity). *There exist constants $\beta^2 \geq 1$ and $\kappa^2 \geq 0$ such that for every combination of positive weights $\{w_i\}$ such that $\sum_{i=1}^n w_i = 1$, we have $\sum_{i=1}^n w_i \left\|\nabla \mathcal{L}_i(x)\right\|^2 \leq \beta^2 \left\|\nabla \mathcal{L}(x)\right\|^2 + \kappa^2$. If all the local loss functions are identical, then we have $\beta^2 = 1$ and $\kappa^2 = 0$.*

Assumption 2.4 (Unbiased Gradient and Bounded Variance). *Every client stochastic gradient $g_i(\boldsymbol{x}|B)$ of a model $\boldsymbol{x}$ evaluated on batch B is an unbiased estimator of the local gradient. We thus have $\mathbb{E}_B\left[\boldsymbol{\xi}_i(B)\right] = 0$ and $0 \leq \mathbb{E}_B\left[\left\|\boldsymbol{\xi}_i(B)\right\|^2\right] \leq \sigma^2$, with $\boldsymbol{\xi}_i(B) = g_i(\boldsymbol{x}|B) - \nabla \mathcal{L}_i(\boldsymbol{x})$.*

We formalize in the following theorem the relationship between the statistical properties of the client sampling scheme and the asymptotic convergence of FL (proof in Appendix A.2).

Theorem 2.1 (FL convergence). *Let us consider a client sampling scheme satisfying Definition 2.1 and Assumption 2.1. Under Assumptions 2.2, 2.3, and 2.4, and with sufficiently small local step size η_l, the following convergence bound holds:*

$$\begin{aligned}
&\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\left\|\nabla \mathcal{L}(\boldsymbol{\theta}^t)\right\|^2\right] \leq \mathcal{O}\left(\frac{1}{\tilde{\eta}KT}\right) \\
&+ \mathcal{O}\left(\eta_l^2(K-1)\sigma^2\right) + \mathcal{O}\left(\tilde{\eta}\left[\Sigma + \sum_{i=1}^n p_i^2\right]\sigma^2\right) \\
&+ \mathcal{O}\left(\eta_l^2 K(K-1)\kappa^2\right) + \mathcal{O}\left(\tilde{\eta}\gamma\left[(K-1)\sigma^2 + K\kappa^2\right]\right),
\end{aligned} \tag{2.7}$$

where $\tilde{\eta} = \eta_g \eta_l$, K is the number of local SGD,

$$\Sigma = \sum_{i=1}^n \mathrm{Var}\left[\omega_i(S_t)\right] \tag{2.8}$$

and

$$\gamma = \sum_{i=1}^n \mathrm{Var}\left[\omega_i(S_t)\right] + \alpha \sum_{i=1}^n p_i^2. \tag{2.9}$$

We first observe that any client sampling scheme satisfying the assumptions of Theorem 2.1 converges to its optimum. Through Σ and γ, equation (2.7) shows that our bound is proportional to the clients aggregation weights through the quantities $\mathrm{Var}\left[\omega_i(S_t)\right]$ and α, which thus should be minimized. These terms are non-negative and are minimized and equal to zero only with full participation of the clients to every optimization round. Theorem 2.1 does not require the sum of the weights $\omega_i(S_t)$ to be equal to 1. Yet, for client sampling satisfying $\mathrm{Var}\left[\sum_{i=1}^n \omega_i(S_t)\right] = 0$, we get $\alpha \propto \Sigma$. Hence, choosing an optimal

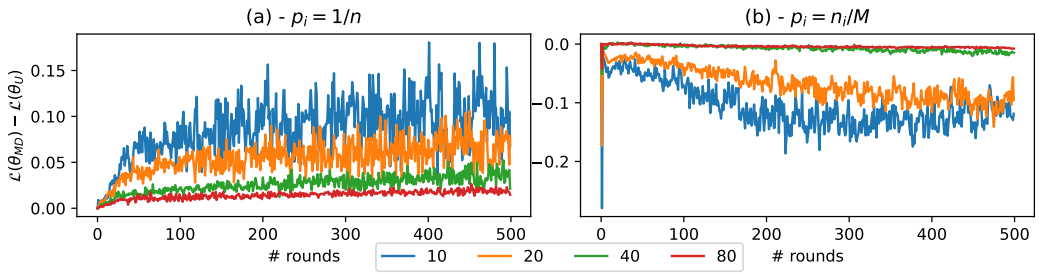

Fig. 2.1.: Difference between the convergence of the global losses resulting from MD and Uniform sampling when considering $n \in \{10, 20, 40, 80\}$ clients and sampling $m = n/2$ of them. In (a), clients have identical importance, i.e. $p_i = 1/n$. In (b), clients importance is proportional to their amount of data, i.e. $p_i = n_i/M$. Differences in global losses are averaged across 30 FL experiments with different model initialization (global losses are provided in Figure 2.2).

client sampling scheme amounts at choosing the client sampling with the smallest Σ. This aspect has been already suggested in Fraboni, Vidal, Kameni, et al. (2021).

The convergence guarantee proposed in Theorem 2.1 extends the work of Jianyu Wang, Q. Liu, et al. (2020) where, in addition of considering FedAvg with clients performing K vanilla SGD, we include a server learning rate η_g and integrate client sampling (equation (2.3)). With full client participation ($\Sigma = \gamma = 0$) and $\eta_g = 1$, we retrieve the convergence guarantees of Jianyu Wang, Q. Liu, et al. (2020). Furthermore, our theoretical framework can be applied to any client sampling satisfying the conditions of Theorem 2.1. In turn, Theorem 2.1 holds for full client participation, MD sampling, Uniform sampling, as well as for the other client sampling schemes detailed in Appendix A.1. Finally, the proof of Theorem 2.1 is general enough to account for FL regularization methods (T. Li, Sahu, Zaheer, Sanjabi, Talwalkar, and Smith, 2018a; T. Li, Sahu, Zaheer, Sanjabi, Talwalkar, and Smithy, 2019; Acar et al., 2021), other SGD solvers (Kingma and Ba, 2014; R. Ward et al., 2019; Xiaoyu Li and Orabona, 2019), and/or gradient compression/quantization (Reisizadeh et al., 2020; Basu et al., 2019; H. Wang et al., 2018). For all these applications, the conclusions drawn for client samplings satisfying the assumptions of Theorem 2.1 still hold.

2.3.2 Application to current client sampling schemes

MD sampling. When using Table 2.1 to compute Σ and γ close-form we obtain:

$$\Sigma_{MD} = \frac{1}{m}\left[1 - \sum_{i=1}^{n} p_i^2\right] \text{ and } \gamma_{MD} = \frac{1}{m}, \tag{2.10}$$

where we notice that $\Sigma_{MD} \leq \frac{1}{m} = \gamma_{MD}$. Therefore, one can obtain looser convergence guarantees than the ones of Theorem 2.1, independently from the amount of participating clients n and set of clients importance $\{p_i\}$, while being inversely proportional to the amount of sampled clients m. The resulting bound shows that FL with MD sampling converges to its optimum for any FL application.

Uniform sampling. Contrarily to MD sampling, the stochastic aggregation weights of Uniform sampling do not sum to 1. As a result, we can provide FL scenarios diverging when coupled with Uniform sampling. Indeed, using Table 2.1 to compute Σ and γ close-form we obtain

$$\Sigma_U = \left[\frac{n}{m} - 1\right] \sum_{i=1}^{n} p_i^2, \tag{2.11}$$

and

$$\gamma_U = \left[1 + \frac{1}{n-1}\right] \left[\frac{n}{m} - 1\right] \sum_{i=1}^{n} p_i^2, \tag{2.12}$$

where we notice that $\gamma_U = \left[1 + \frac{1}{n-1}\right] \Sigma_U$. Considering that $\sum_{i=1}^{n} p_i^2 \leq 1$, we have $\Sigma_U \leq \frac{n}{m} - 1$, which goes to infinity for large cohorts of clients and thus prevents FL with Uniform sampling to converge to its optimum. Indeed, the condition $\sum_{i=1}^{n} p_i^2 \leq 1$ accounts for every possible scenario of client importance $\{p_i\}$, including the very heterogeneous ones. In the special case where $p_i = 1/n$, we have $\sum_{i=1}^{n} p_i^2 = 1/n$, such that Σ_U is inversely proportional to both n and m. Such FL applications converge to the optimum of equation (2.1) for any configuration of n, $\{p_i\}$ and m.

Moreover, the comparison between the quantities Σ and γ for MD and Uniform sampling shows that Uniform sampling outperforms MD sampling when $p_i = 1/n$. More generally, Corollary 2.1 provides sufficient conditions with Theorem 2.1 for Uniform sampling to have better convergence guarantees than MD sampling (proof in Appendix A.2.7).

Corollary 2.1. *Uniform sampling has better convergence guarantees than MD sampling when $\Sigma_U \leq \Sigma_{MD}$, and $\gamma_U \leq \gamma_{MD}$ which is equivalent to*

$$\sum_{i=1}^{n} p_i^2 \leq \frac{1}{n - m + 1}. \tag{2.13}$$

Corollary 2.1 can be related to $\mathrm{Var}\left[\sum_{i=1}^{n} \omega_i(S_t)\right]$, the variance for the sum of the aggregation weights, which is always null for MD sampling, and different of 0 for Uniform sampling except when $p_i = 1/n$ for all the clients.

A last point of interest for the comparison between MD and Uniform sampling concerns the respective time complexity for selecting clients. Sampling with a Multinomial Distribution has time complexity $\mathcal{O}(n + m\log(n))$, where $\mathcal{O}(n)$ comes from building the probability

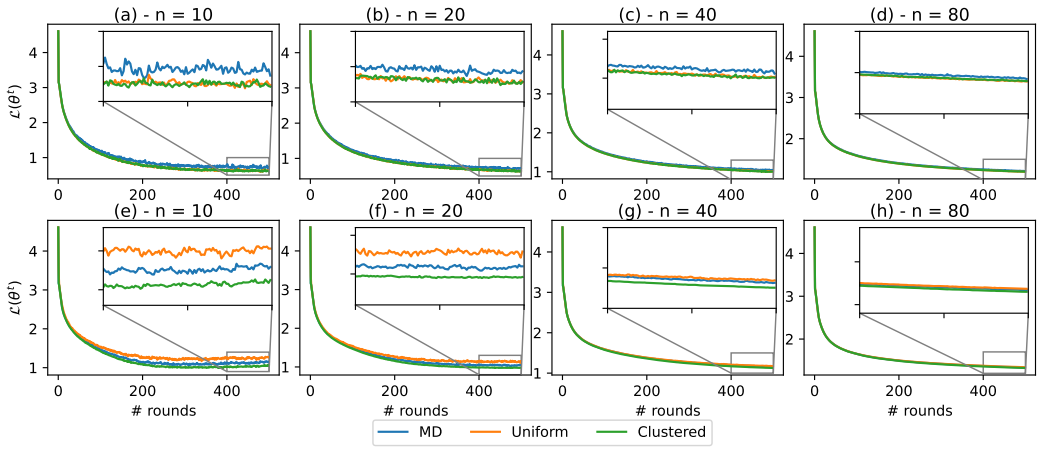

Fig. 2.2.: Convergence of the global losses for MD, Uniform, and Clustered sampling when considering $n \in \{10, 20, 40, 80\}$ clients and sampling $m = n/2$ of them. In (a-d), clients have identical importance, i.e. $p_i = 1/n$. In (e-h), clients importance is proportional to their amount of data, i.e. $p_i = n_i/M$. Zoom of the global losses over the last 100 server aggregations and a variation of 0.5 in the global loss.

density function to sample clients indices (D. Tang, 2019). This makes MD sampling difficult to compute or even intractable for large cohorts of clients. On the contrary sampling m elements without replacement from n states is a reservoir sampling problem and takes time complexity $\mathcal{O}(m(1 + \log(n/m))$(K.-H. Li, 1994). In practice, clients either receive identical importance ($p_i = 1/n$) or an importance proportional to their data ratio, for which we may assume computation $p_i = \mathcal{O}(1/n)$. As a result, for important amount n of participating clients, Uniform sampling should be used as the default client sampling due to its lower time complexity. However, for small amount of clients and heterogeneous client importance, MD sampling should be used by default.

Due to space constraints, we only consider in this manuscript applying Theorem 2.1 to Uniform and MD sampling, which can also be applied to Binomial and Poisson Binomial sampling introduced in Section A.1, and satisfying our covariance assumption. To the best of our knowledge, we could only find *Clustered sampling* introduced in Fraboni, Vidal, Kameni, et al. (2021) not satisfying this assumption. Still, with minor changes, we provide for this sampling scheme a similar bound to the one of Theorem 2.1 (Appendix A.2.6), ultimately proving that clustered sampling improves MD sampling.

2.4 Experiments on real data

In this section, we provide an experimental demonstration of the convergence properties identified in Theorem 2.1. [1] We study a LSTM model for next character prediction on the dataset of *The complete Works of William Shakespeare* (McMahan, Moore, et al., 2017; Caldas et al., 2018). We use a two-layer LSTM classifier containing 100 hidden units with an 8 dimensional embedding layer. The model takes as an input a sequence of 80 characters, embeds each of the characters into a learned 8-dimensional space and outputs one character per training sample after 2 LSTM layers and a fully connected one.

When selected, a client performs $K = 50$ SGD steps on batches of size $B = 64$ with local learning rate $\eta_l = 1.5$. The server considers the clients local work with $\eta_g = 1$. We consider $n \in \{10, 20, 40, 80\}$ clients, and sample half of them at each FL optimization step. While for sake of interpretability we do not apply a decay to local and global learning rates, we note that our theory remains unchanged even in presence of a learning rate decay. In practice, for dataset with important heterogeneity, considering $\eta_g < 1$ can speed-up FL with a more stable convergence.

We compare the impact of MD, Uniform, and Clustered sampling, on the convergence speed of FEDAVG. With Clustered sampling, the server selects m clients from m different clusters of clients created based on the clients importance (Fraboni, Vidal, Kameni, et al., 2021, Algorithm 1). MD sampling is a special case of Clustered sampling, where every cluster is identical.

Clients have identical importance $[p_i = 1/n]$. We note that Uniform sampling consistently outperforms MD sampling due to the lower covariance parameter, while the improvement between the resulting convergence speed is inversely proportional to the number of participating clients n (Figure 2.1a and Figure 2.2a-d). This result confirms the derivations of Section 2.3. Also, with Clustered sampling and identical client importance, every client only belongs to one cluster. Hence, Clustered sampling reduces to Uniform sampling and we retrieve identical convergence for both samplings (Figure 2.2a-d). This point was not raised in Fraboni, Vidal, Kameni, et al. (2021).

Clients importance depends on the respective data ratio $[p_i = n_i/M]$. In this experimental scenario the aggregation weights for Uniform sampling do not always sum to 1, thus leading to the slow-down of FL convergence. Hence, we see in Figure 2.1b that MD always outperforms Uniform sampling. This experiment shows that the impact on FL convergence of the variance of the sum of the stochastic aggregation weights is more relevant than the one due to the covariance parameter α. We also retrieve in Figure 2.2e-h that Clustered

[1] Code and data are available at `https://github.com/Accenture/Labs-Federated-Learning/tree/impact_client_sampling`.

sampling always outperform MD sampling, which confirms that for two client samplings with a null variance of the sum of the stochastic aggregation weights, the one with the lowest covariance parameter α converges faster. We also note that the slow-down induced by the variance is reduced when more clients do participate. This is explained by the fact that the standard deviation of the clients data ratio is reduced with larger clients participation, e.g. $p_i = 1/10 \pm 0.13$ for $n = 10$ and $p_i = 1/80 \pm 0.017$ for $n = 80$. We thus conclude that the difference between the effects of MD, Uniform, and Clustered sampling is mitigated with a large number of participating clients (Figure 2.1b and Figure2.2e-h).

Additional experiments on Shakespeare are provided in Appendix A.3. We show the influence of the amount of sampled clients m and amount of local work K on the convergence speed of MD and Uniform sampling.

Finally, additional experiments on CIFAR10 (Krizhevsky et al., n.d.) are provided in Appendix A.3, where we replicate the experimental scenario previously proposed in Fraboni, Vidal, Kameni, et al. (2021). In these applications, 100 clients are partitioned using a Dirichlet distribution which provides federated scenarios with different level of heterogeneity. For all the experimental scenarios considered, both results and conclusions are in agreement with those here derived for the Shakespeare dataset.

2.5 Conclusion

In this chapter, we highlight the asymptotic impact of client sampling on FL with Theorem 2.1, and shows that the convergence speed is inversely proportional to both the sum of the variance of the stochastic aggregation weights, and to their covariance parameter α. Moreover, to the best of our knowledge, this chapter is the first one accounting for schemes where the sum of the weights is different from 1.

Thanks to our theory, we investigated MD and Uniform sampling from both theoretical and experimental standpoints. We established that when clients have approximately identical importance, i.e $p_i = 1/n$, Uniform outperforms MD sampling, due to the larger impact of the covariance term for the latter scheme. On the contrary, Uniform sampling is outperformed by MD sampling in more general cases, due to the slowdown induced by its stochastic aggregation weights not always summing to 1. Yet, in practical scenario with very large number of clients, MD sampling may be unpractical, and Uniform sampling could be preferred due to the more advantageous time complexity.

In this chapter, we also showed that our theory encompasses advanced FL sampling schemes, such as the one recently proposed in Fraboni, Vidal, Kameni, et al., 2021, and W. Chen et al.,

2020. Finally, while the contribution of this chapter is in the study of the impact of a client sampling on the global optimization objective, further extensions may focus on the analysis of the impact of clients selection method on individual users' performance, especially in presence of heterogeneity.

3

Clustered Sampling: Low-Variance and Improved Representativity for Clients Selection in Federated Learning

Contents

3.1 Introduction **28**
3.2 Related Work **29**
3.2.1 Aggregating clients local updates 30
3.2.2 Clients' sampling 30
3.2.3 FL convergence with MD client sampling 31
3.2.4 Sampling schemes comparison 32
3.3 Clustered Sampling **32**
3.3.1 Definition of clustered sampling 33
3.3.2 Improvements provided by clustered sampling 35
3.4 Clustered Sampling based on Sample Size **36**
3.5 Clustered Sampling based on Similarity **38**
3.6 Experiments **40**
3.7 Discussion and Conclusion **44**

In the previous chapter, we provided a unified framework to measure the impact of a client sampling scheme and of the clients' data heterogeneity on the convergence speed of FL. In this chapter, we introduce *clustered sampling*, a novel client sampling scheme providing optimal FL convergence speed and representativity of clients during FL optimization. In particular, we provide two different clustering approaches enabling clients aggregation based on 1) sample size, and 2) models similarity. Through a series of experiments in non-iid and unbalanced scenarios, we demonstrate that model aggregation through clustered sampling consistently leads to better training convergence and variability when compared to standard sampling approaches. This chapter is published at the International Conference on Machine Learning (ICML) of 2021 as Fraboni, Vidal, Kameni, et al. (2021).

3.1 Introduction

Federated learning (FL) is a training paradigm enabling different clients to jointly learn a global model without sharing their respective data. Communication can be a primary bottleneck for FL since wireless and other end-user internet connections operate at variable communication rates while being potentially unreliable. Moreover, the capacity of the aggregating server may impose constraints on the number of clients the server can communicate with at the same time. These considerations led to significant interest in reducing the number and bandwidth of communications at every step of the FL process.

One of the most popular communication reduction strategies consists in limiting the frequency of communications at the expense of increased computation on the clients side. This is usually achieved by asking the clients to perform multiple iterations of local gradient descent before communicating their updates. In this setting, FedAvg (McMahan, Moore, et al., 2017) is the first and most widely used FL algorithm, for which convergence bounds were given in Jianyu Wang, Q. Liu, et al. (2020), Xiang Li et al. (2020), Karimireddy et al. (2020), Yu, S. Yang, et al. (2019), Khaled et al. (2020b), Woodworth et al. (2020), T. Lin et al. (2020), and Sebastian U. Stich (2019).

To further reduce the number of communications, the server can select a subset of clients participating at every iteration. This strategy, called client sampling, enables reducing communications to the minimum. FedAvg first proposed selecting m clients uniformly without replacement while replacing the contribution of the non-sampled clients with the current global model. However this scheme is known for being biased, since the resulting model is, in expectation, different from the deterministic aggregation of every client. To overcome this issue, T. Li, Sahu, Zaheer, Sanjabi, Talwalkar, and Smith (2018a) proposes an *unbiased* sampling scheme where the new global model is created as the average of the sampled clients work. The sampling is based on a multinomial distribution (MD) whose clients probabilities corresponds to their relative sample size. While other clients sampling schemes have been proposed, most of them require additional server-clients communications and are not proven to be unbiased (Xiang Li et al., 2020; W. Chen et al., 2020; Nishio and Yonetani, 2019).

To the best of our knowledge, FedAvg and MD sampling are the only schemes keeping to a minimum server-clients communications. In particular, MD sampling has been proven to lead to FL optimum and shown experimentally to outperform FedAvg sampling (Xiang Li et al., 2020). In spite of its unbiasedness, MD sampling may still lead to large variance in the clients selection procedure. While unbiasedness guarantees proper clients representativity in expectation, representativity is not necessarily achieved when considering a single FL iteration. Since at each MD sampling instance we select clients with replacement, this

determines a variance in the amount of times a client is selected. This sampling variance is a leading cause of the large variability in the convergence of FL, especially in non-iid applications. Indeed, at each iteration, sampled clients improve the global model based on their data distribution, to the detriment of the data specificity of non-sampled clients.

While the literature mainly focused on the study of the behavior of FL sampling strategies in expectation, to our knowledge this study provides the first theoretical investigation of the variability properties of FL sampling. In what follows, we show that this statistical aspect is crucial to determine convergence stability and quality of FL. The contribution of this chapter is the introduction of *clustered sampling*, a new unbiased client sampling scheme improving MD sampling by guaranteeing smaller client selection variability, while keeping to a minimum server-clients communications. By increasing every client representativity in model aggregations, clustered sampling ensures that clients with unique distributions are more likely of being sampled, leading to smoother and faster FL convergence.

We first derive, in Section 3.2, the theory behind current FL sampling schemes this chapter is built on. We then formally introduce clustered sampling in Section 3.3 and prove its theoretical correctness by extending the work done in Jianyu Wang, Q. Liu, et al. (2020). We finally show the theoretical benefits of clustered sampling over MD sampling. In Section 3.4, we propose an implementation of clustered sampling aggregating clients based on their sample size, showing that this approach leads to reduced variance of the clients' aggregation weights. In Section 3.5, we extend our sampling theory to aggregation schemes based on the similarity between clients updates, showing that this approach further reduces the variance of clients aggregation weights while improving the representation of the clients during each FL aggregation step, as compared to MD sampling. This result leads to an overall improvement of the convergence of FL. Finally, in Section 3.6, we experimentally demonstrate this chapter on a broad range of balanced and unbalanced heterogeneous dataset. The code used for this chapter is available here[1].

3.2 Related Work

Before introducing in Section 3.3 the core idea of clustered sampling, we first recapitulate in Section 3.2 the current theory behind parameter aggregation and sampling schemes for FL.

[1] https://github.com/Accenture//Labs-Federated-Learning/tree/clustered_sampling

3.2.1 Aggregating clients local updates

In FL, we consider a set I of clients respectively owning datasets $\mathcal{D}_i$ composed of n_i samples. FL aims at optimizing the average of each clients local loss function weighted by their importance p_i

$$\mathcal{L}(\theta) = \sum_{i \in I} p_i \mathcal{L}_i(\theta), \tag{3.1}$$

where θ represents the model parameters and $\sum_{i=1}^{n} p_i = 1$. While any combination of $\{p_i\}$ is possible, a common choice consists in defining $p_i = n_i/M$, where $M = \sum_{i \in I} n_i$ is the total number of sample across datasets. In this work, we adopt the same definition of the importance weights, although the theory derived below does not depend on ny specific choice of the parameters $\{p_i\}$.

In this setting, to estimate a global model across clients, FEDAVG (McMahan, Moore, et al., 2017) is an iterative training strategy based on the aggregation of local model parameters θ_i^t. At each iteration step t, the server sends the current global model parameters θ^t to the clients. Each client updates the model by minimizing the local cost function $\mathcal{L}(\theta_i^{t+1}, \mathcal{D}_i)$ through a fixed amount of SGD initialized with θ^t. Subsequently each client returns the updated local parameters θ_i^{t+1} to the server. The global model parameters θ^{t+1} at the iteration step $t+1$ are then estimated as a weighted average, i.e.

$$\theta^{t+1} = \sum_{i \in I} \frac{n_i}{M} \theta_i^{t+1}. \tag{3.2}$$

3.2.2 Clients' sampling

Clients sampling is a central operation of FL. FEDAVG (McMahan, Moore, et al., 2017) proposes to uniformly sample a subset of participating clients S_t at every iteration while the other clients updates are replaced by the current global model, i.e.

$$\theta^{t+1} = \sum_{i \in S_t} \frac{n_i}{M} \theta_i^{t+1} + \sum_{i \notin S_t} \frac{n_i}{M} \theta^t. \tag{3.3}$$

The sampling scheme introduced by FEDAVG is generally slow due to the attrition introduced by non-participating clients. To solve this problem, T. Li, Sahu, Zaheer, Sanjabi, Talwalkar, and Smith (2018a) proposes instead to sample S_t, the subset of clients at iteration t, from a Multinomial Distribution (MD) where each client is sampled according to its relative data ratio $p_i = \frac{n_i}{M}$. The new global model is obtained as the average of each selected client, i.e.

$$\theta^{t+1} = \sum_{i \in S_t} \frac{1}{m} \theta_i^{t+1}. \tag{3.4}$$

By design, MD sampling is such that the aggregation of clients model updates is identical in expectation to the one obtained when considering all the clients, i.e. $\mathbb{E}_{S_t}\left[\theta^{t+1}\right] = \sum_{i\in I} p_i \theta_i^{t+1}$. Sampling schemes following this property are called *unbiased*. Notably, the sampling scheme employed by FEDAVG does not satisfy this property, and it is thus prone to clients-drift (Karimireddy et al., 2020).

3.2.3 FL convergence with MD client sampling

Theoretical guarantees regarding the convergence of FEDAVG were given in Jianyu Wang, Q. Liu, et al. (2020). The proof relies on assumptions classically used in Stochastic Gradient Descent (SGD) analysis (Bottou et al., 2018) (Assumptions 3.1 and 3.2 below), or commonly used in the federated optimization literature (T. Li, Sahu, Zaheer, Sanjabi, Talwalkar, and Smith, 2018a; Xiang Li et al., 2020; Haddadpour, Kamani, Mahdavi, and V. R. Cadambe, 2019; Karimireddy et al., 2020; J. Wang et al., 2019) to capture the dissimilarities of local objectives (Assumption 3.3 below).

Assumption 3.1 (Smoothness)**.** *The clients local objective function is Lipschitz smooth, that is,* $\|\nabla\mathcal{L}_i(x) - \nabla\mathcal{L}_i(y)\| \leq L\|x-y\|, \forall i \in \{1,...,n\}$.

Assumption 3.2 (Unbiased Gradient and Bounded Variance)**.** *For each client* i *local model, the stochastic gradient* $g_i(x|\xi)$ *of model* x *evaluated on batch* ξ *is an unbiased estimator of the local gradient:* $\mathbb{E}_\xi\left[g_i(x|\xi)\right] = \nabla\mathcal{L}_i(x)$*, and has bounded variance* $\mathbb{E}_\xi\left[\|g_i(x|\xi) - \nabla\mathcal{L}_i(x)\|\right] \leq \sigma^2, \forall i \in \{1,...,n\}$ *with* $\sigma^2 \geq 0$.

Assumption 3.3 (Bounded Dissimilarity)**.** *For any set of weights* $\{w_i \geq 0\}_{i=1}^n$ *such that* $\sum_{i=1}^n w_i = 1$*, there exists constants* $\beta^2 \geq 1$ *and* $\kappa^2 \geq 0$ *such that* $\sum_{i=1}^n w_i \|\nabla\mathcal{L}_i(x)\|^2 \leq \beta^2 \|\sum_{i=1}^n w_i \nabla\mathcal{L}_i(x)\|^2 + \kappa^2$*. If all the local loss functions are identical, then we have* $\beta^2 = 1$ *and* $\kappa^2 = 0$.

The following theorem was proven in Jianyu Wang, Q. Liu, et al. (2020) and provides theoretical guarantees for MD client sampling.

Theorem 3.1. *Under Assumption 3.1 to 3.3, and local learning rate* $\eta = \sqrt{m/NT}$*, FL with* $FedAvg$ *when sampling* m *clients with MD converges to a stationary point of* $\mathcal{L}(\theta)$*:*

$$\frac{1}{T}\sum_{t=0}^{T}\mathbb{E}\left[\left\|\nabla\mathcal{L}(\theta^t)\right\|^2\right] \leq \mathcal{O}(\frac{1}{\sqrt{mNT}}) + \mathcal{O}(\frac{mN}{T}). \tag{3.5}$$

The proof of Theorem 3.1 can be found in Jianyu Wang, Q. Liu, et al. (2020) and shows that considering a subset of workers with MD client sampling is enough to ensure convergence of the global model to a local minimum of the federated loss function, equation (3.1). Following the conclusions of that work, to avoid optimizing a surrogate loss function instead of the federated one in equation (3.1), the server asks from every client to compute the same amount of SGD steps N.

3.2.4 Sampling schemes comparison

Other client sampling schemes have been proposed. For example, with Xiang Li et al. (2020), the server sends the global model to every client before creating the new global model out of the first m updated models the server receives; with W. Chen et al. (2020), the server waits for every client to send the norm of their work before selecting the m clients with the most relevant updates; with Nishio and Yonetani (2019), clients transmit information about their available computation resources before the server selects m of them in function of their availability. Contrarily to FEDAVG and MD sampling, these sampling schemes require additional communications and sometimes even computation from all the clients. On the contrary, FEDAVG and MD sampling have the appealing property of maintaining to a minimum the amount of clients-server communications at each iteration. The global model is sent only to the sampled clients, and the amount of local work is set for those clients to only N SGD updates. To the best of our knowledge, FEDAVG and MD sampling schemes are the only approaches minimizing the effective work and communications asked to the clients. Moreover, while FEDAVG sampling has no theoretical guarantees regarding its convergence, neither regarding the unbiasedness of its global model, MD sampling is shown to converge to a stationary point of the global loss function (3.1) (Theorem 3.1). Based on these considerations and given that Xiang Li et al. (2020) shows experimentally that MD sampling outperforms FEDAVG , in the rest of this work we consider MD sampling as reference sampling technique.

3.3 Clustered Sampling

In Section 3.3.1, we first introduce clustered sampling and prove the convergence of FL under this scheme. In Section 3.3.2, we show the statistical improvements brought by clustered sampling as compared to MD client sampling, in terms of reduced sampling variance, and better clients representativity across the entire FL process.

3.3.1 Definition of clustered sampling

Let us consider n clients participating to FL. With MD sampling, m clients are sampled from a multinomial distribution supported on $\{1, ..., n\}$ where a client is selected in function of its data ratio p_i.

Assumption 3.4 (Unbiased Sampling)**.** *A client sampling scheme is said unbiased if the expected value of the client aggregation is equal to the global deterministic aggregation obtained when considering all the clients, i.e.*

$$\mathbb{E}_{S_t}\left[\theta^t\right] = \mathbb{E}_{S_t}\left[\sum_{j\in S_t} w_j(S_t)\theta_j^t\right] := \sum_{i=1}^{n} p_i\theta_i^t, \tag{3.6}$$

where $w_j(S_t)$ is the aggregation weight of client j for subset of clients S_t.

In Xiang Li et al. (2020), the notion of unbiased sampling is introduced by means of Assumption 3.4. MD sampling follows this assumption, and thus provides at every iteration an unbiased global model. However, MD sampling enables a client to be sampled from 0 to m times with non-null probability at each iteration, giving aggregation weights for every client ranging from 0 to 1. As a result, MD sampling provides appropriate representation for every client in expectation, with however potentially large variance in the amount of times a client is selected. As a consequence, the representativity of a client at any given realization of a FL iteration may not guaranteed. In the following we introduce clustered sampling, and show that this strategy leads to decreasing clients aggregation weight variance and better clients representativity.

We denote by W_0 the multinomial distribution with support on $\{1, ..., n\}$ used to sample one client according to its data ratio p_i. MD sampling can be seen as sampling m times with W_0. With clustered sampling, we propose to generalize MD sampling by sampling m clients according to m independent distributions $\{W_k(t)\}_{k=1}^m$ each of them privileging a different subset of clients based on opportune selection criteria (Section 3.4 and 3.5). With clustered sampling, the m clients can be sampled with different distributions and, at two different iterations, the set of distributions can differ. MD sampling is a special case of clustered sampling when $\forall t,\ \forall k \in \{1, ..., m\},\ W_k(t) = W_0$.

In the rest of this work, we denote by $r_{k,i}^t$ the probability for client i to be sampled in distribution $W_k(t)$. By construction, we have:

$$\forall k \in \{1, ..., m\},\ \sum_{i=1}^{n} r_{k,i}^t = 1 \text{ with } r_{k,i}^t \geq 0. \tag{3.7}$$

We also require clustered sampling to be unbiased. Extending Assumption 3.4 to m independent sampling distributions $\{W_k(t)\}_{k=1}^m$, we obtain the property:

$$\forall i \in \{1, ..., n\}, \sum_{k=1}^{m} r_{k,i}^t = m\, p_i. \tag{3.8}$$

Proposition 3.1. *Equations (3.7) and (3.8) are sufficient conditions for clustered sampling to satisfy Assumption 3.4.*

Proof. Satisfying equation (3.7) ensures the m distributions used with clustered sampling are feasible. When sampling one client from one of the m distributions $W_k(t)$, we get:

$$\mathbb{E}_{W_k(t)}\left[\sum_{j \in W_k(t)} w_j(W_k(t))\theta_j^t\right] = \sum_{i=1}^{n} r_{k,i}^t \theta_i^t. \tag{3.9}$$

By linearity of the expected value, the expected new global model is the average between the weighted models obtained according to each distribution $\{W_k\}_{k=1}^m$ derived in equation (3.9), i.e.

$$\mathbb{E}_{S_t}\left[\theta^t\right] = \sum_{k=1}^{m} \frac{1}{m} \sum_{i=1}^{n} r_{k,i}^t \theta_i^t = \sum_{i=1}^{n} p_i \theta_i^t, \tag{3.10}$$

where the second equality comes from equation (3.8). □

In Theorem 3.2, we prove that FedAvg with clustered sampling satisfying Assumptions 3.1 to 3.3 and Proposition 3.1 has the same convergence bound to a FL local optimum as with FedAvg and MD sampling. The proof of Theorem 3.2 can be found in Appendix B.1.

Theorem 3.2. *Under Assumption 3.1 to 3.3, and local learning rate $\eta = \sqrt{m/NT}$, let's consider FL with $FedAvg$ when sampling m clients with clustered sampling scheme satisfying Proposition 3.1. The same asymptotic behavior of MD sampling holds :*

$$\frac{1}{T}\sum_{t=0}^{T} \mathbb{E}\left[\left\|\nabla \mathcal{L}(\theta^t)\right\|^2\right] \leq \mathcal{O}(\frac{1}{\sqrt{mNT}}) + \mathcal{O}(\frac{mN}{T}) \tag{3.11}$$

In the proof of Theorem 3.2 (Appendix B.1), we show that this convergence bound holds for any clustered sampling scheme satisfying Proposition 3.1. Moreover, the convergence bound of MD sampling is a bound itself for the convergence of a general clustered sampling scheme satisfying Proposition 3.1. Therefore, clustered sampling enjoys better convergence guarantees than MD sampling.

3.3.2 Improvements provided by clustered sampling

We introduced clustered sampling in Section 3.3.1 and showed that under the condition of Proposition 3.1 it provides the same convergence bound of MD sampling. In this section, we investigate the statistical benefits of clustered sampling with respect to MD sampling.

We define by $\omega_i(S)$ the aggregation weight of client i with subset of sampled clients S, and by S_{MD} and $S_C(t)$ the subset of clients sampled at iteration t with respectively MD and clustered sampling.

We consider a clustered sampling scheme following Proposition 3.1. Hence, for both MD and clustered sampling, the expected aggregation equals the deterministic aggregation when considering all the clients leading to:

$$\mathbb{E}_{S_{MD}(t)}\left[\omega_i(S_{MD})\right] = \mathbb{E}_{S_C(t)}\left[\omega_i(S_C(t))\right] = p_i. \tag{3.12}$$

With clustered sampling, we first show that every client has a smaller aggregation weight variance. A client's aggregation weight can be written as $\omega_i(S) = \frac{1}{m}\sum_{k=1}^{m}\{l_k = i\}$, where l_k is the index of the k^{th} sampled client. With MD sampling, m clients are iid sampled according to $\mathcal{B}(p_i)$, a Bernoulli distribution with probability p_i, giving the following variance:

$$\begin{aligned}\mathrm{Var}_{S_{MD}}\left[\omega_i(S_{MD})\right] &= \frac{1}{m^2} m \,\mathrm{Var}\left[\mathcal{B}(p_i)\right] &(3.13)\\ &= \frac{1}{m^2} m p_i(1-p_i). &(3.14)\end{aligned}$$

Clustered sampling instead selects independently m clients according to the distributions $\{W_k(t)\}_{k=1}^{m}$. Therefore, each client is sampled according to $\mathcal{B}(r_{k,i}^t)$ giving:

$$\begin{aligned}\mathrm{Var}_{S_C(t)}\left[\omega_i(S_C(t))\right] &= \frac{1}{m^2}\sum_{k=1}^{m}\mathrm{Var}\left[\mathcal{B}(r_{k,i}^t)\right] &(3.15)\\ &= \frac{1}{m^2}\sum_{k=1}^{m} r_{k,i}^t(1-r_{k,i}^t). &(3.16)\end{aligned}$$

By the Cauchy-Schwartz inequality, one can prove that

$$\mathrm{Var}_{S_{MD}}\left[\omega_i(S_{MD})\right] \geq \mathrm{Var}_{S_C(t)}\left[\omega_i(S_C(t))\right], \tag{3.17}$$

with equality if and only if all the m distributions are equal to the one of MD sampling, i.e. $\forall k,\ W_k(t) = W_0$. Exact derivation is given in Appendix B.2. Therefore, with clustered sampling, every client has a smaller aggregation weight variance. Another interesting statistical measure of representativity is the probability for a client to be sampled, i.e.

$\mathbf{P}(\{i \in S\})$. In particular, increasing the probability for every client to be sampled is mandatory to allow a proper representation of each client's data specificity in the global model, especially in heterogeneous setting, such as with non-iid and unbalanced clients. For this statistical measure, clustered sampling also provides better guarantees than MD sampling. With MD sampling, clients are iid sampled giving

$$p(i \in S_{MD}) = 1 - p(\{i \notin S_{MD}\}) \tag{3.18}$$
$$= 1 - p(\{i \notin W_0\})^m \tag{3.19}$$
$$= 1 - (1 - p_i)^m. \tag{3.20}$$

Similarly, with clustered sampling we get:

$$p(i \in S_C(t)) = 1 - \prod_{k=1}^{m} p(\{i \notin W_k(t)\}) \tag{3.21}$$
$$= 1 - \prod_{k=1}^{m} (1 - r_{k,i}^t). \tag{3.22}$$

Since we assume here that clustered sampling follows Proposition 3.1, from equation (3.8), and from the inequality of arithmetic and geometric means, we get:

$$p(\{i \in S_{MD}(t)\}) \leq p(\{i \in S_C(t)\}), \tag{3.23}$$

with equality if and only if all the m distributions are equal to the one of MD sampling, i.e. $\forall k,\ W_k(t) = W_0$ (derivation in Appendix B.2). Therefore, with clustered sampling, every client has an higher probability of being sampled and thus is better represented throughout the FL process.

In conclusion, clustered sampling reduces clients aggregation weights variance and increases their representativity. These results are important for FL applications with heterogeneous federated dataset. Increasing a client representativity ensures that clients with unique distributions are more likely of being sampled, and can potentially lead to smoother and faster FL convergence.

3.4 Clustered Sampling based on Sample Size

We introduced and showed the convergence of unbiased clustered sampling in Section 3.3. Clustered sampling schemes compatible with Proposition 3.1 are numerous, including MD sampling. In this section, we first provide an unbiased clustered sampling scheme based on the number of samples n_i owned by each client. The proposed scheme, illustrated in

Algorithm 2 Clustered sampling based on sample size

Require: $\{n_i\}_{i=1}^n$ clients number of samples

Order clients by descending importance of n_i.
$k \leftarrow 1$ distribution index.
$q \leftarrow 0$ sum of samples.
$M \leftarrow \sum_{i=1}^n n_i$ total number of samples.
for each client $i = 1$ **to** n **do**
    $q \leftarrow q + mn_i$
    $q = a_i M + b_i$ with a_i and b_i non negative integers
    if $a_i > k$ **then**
        $r'_{k,i} \leftarrow M - b_{i-1}$
        $\forall l \geq k+1$ s.t. $(a_i - 1) - l \geq 0$, $r'_{k,i} \leftarrow M$
    end if
    $r'_{a_i,i} \leftarrow b_i$
    $k \leftarrow a_i$
end for

Ensure: Sampling probabilities $r_{k,i} = r'_{k,i}/M$.

Algorithm 2 , is compatible with Proposition 3.1. In particular, we have the following theorem:

Theorem 3.3. *Algorithm 2 outputs m distributions for a clustered sampling satisfying Proposition 3.1. The complexity of the algorithm is $\mathcal{O}(n \log(n))$.*

Proof. Algorithm 2 identifies the m distributions W_k by defining m sets q_k of cardinality M in which each client i is represented with probability $r_{k,i}$. The sets are constructed as follows. We define by $n'_i = mn_i$ the total number of samples to be allocated for each client. We thus have mM samples to allocate over the m sets q_k. For each client, the integer division $n'_i = Ma_i + b_i$, identifies a_i sets for which the client must be represented with probability 1. The remaining b_i samples are allocated to the remaining $m - \sum_i a_i$ sets. This is possible by observing that $Mm = \sum_i n'_i = M(\sum_i a_i) + \sum_i b_i$, and therefore $M(m - \sum_i a_i) = \sum_i b_i$. By construction, Proposition 3.1 is satisfied: $|q_k| = M$ implies equation (3.7), while equation (3.8) is met since, for each client, the total number of samples distributed across the sets q_k is $n'_i = mn_i$, and thus each client is represented with proportion mp_i across all the distributions. Algorithm 2 provides the practical implementation of this scheme.

The complexity of Algorithm 2 is derived in Appendix B.3, where we also provide a schematic illustration of the allocation procedure. □

We note that with Algorithm 2 a client i can be sampled up to $\lfloor mp_i \rfloor + 2$ times. This is an improvement from MD sampling where clients can be instead sampled up to m times.

Algorithm 3 Clustered sampling based on model similarity

Require: $\{n_i\}_{i=1}^n$ clients number of samples, $\{G_i\}_{i=1}^n$ clients representative gradient, m number of sampled clients, *clustering method* (e.g. Ward method), s similarity function (e.g. Arccos)

Estimated hierarchical clustering P with *clustering method* from similarity matrix ρ with $\rho_{i,j} = s(G_i, G_j)$.
Cut P to determine $K \geq m$ groups $\{B_k\}_{k=1}^K$. We define q_k as the total number of samples of the corresponding clients: $q_k = \sum_{i \in B_k} mn_i \leq M$.
Order the groups $\{B_k\}_{k=1}^K$ by decreasing q_k.
Define clients number of samples in the m distributions $\{W_k\}_{k=1}^m$ based on the ranking of $q_k : \forall k \leq m, \forall i \in B_k,\ r'_{k,i} \leftarrow mn_i$.
Create a set with the clients of the remaining groups $S = \{\{i, u_i = mn_i\},\ \forall i \in B_{m+1} \cup ... \cup B_K\}$
$k \leftarrow 1$ Start considering the first distribution W_k
while $S \neq \emptyset$ **do**
    Select first client i in S with u_i samples to allocate
    Determine a_i and b_i the quotient and remainder of the euclidean division of $q_k + u_i$ by M
    if $a_i = 0$ **then**
        $r'_{k,i} \leftarrow b_i$ and i removed from S
    else if
        then$r'_{k,i} \leftarrow M - q_k$
        $u_i \leftarrow u_i - r'_{k,i}$
        Remove i from S if $u_i = 0$
        $k \leftarrow k + 1$
    end if
end while

Ensure: Sampling probabilities $r_{k,i} = r'_{k,i}/M$.

Since clustering is performed according to the clients sample size n_i, unless n_i changes during the learning process, Algorithm 2 needs to be run only once at the beginning of the learning process, i.e. $\{W_k(t)\}_{k=1}^m = \{W_k\}_{k=1}^m$.

3.5 Clustered Sampling based on Similarity

We have shown, in Section 3.3, that unbiased clustered sampling is a generalization of MD sampling providing smaller aggregation weight variance for every client, and we proposed in Section 3.4 an algorithm to practically fulfill Proposition 3.1 to obtain m distributions grouping clients based on their number of samples n_i.

In this section we extend the approach of Section 3.3 to define a novel clustered sampling scheme where sampling distributions are defined based on the similarity across clients. In

what follows we define clients similarity based on the measure of *representative gradient*. The representative gradient is the difference between a client's updated model and the global model. Comparing clients' representative gradients at a given iteration is shown to be an effective approach for detecting similarity between FL participants (Sattler et al., 2019).

Algorithm 3 adopts this concept to define a clustered sampling scheme compatible with Proposition 3.1. We have:

Theorem 3.4. *If for every client $p_i \leq 1/m$, Algorithm 3 outputs m distributions for a clustered sampling satisfying Proposition 3.1. The complexity of the algorithm is in $\mathcal{O}(n^2 d + X)$, where d is the number of parameters in the model, and X is the complexity of the clustering method.*

Proof. Algorithm 3 is similar to Algorithm 2, with the additional constraint that the number of clusters K can differ from the number m of distributions. If $K = m$, the clients are already allocated in q_k sets, and the same reasoning of Algorithm 2 can be applied. If $K > m$, we consider again the partitioning problem over m sets q_k of cardinality M. We define again by $n_i' = mn_i$ the total number of samples to be allocated for each client, and we have mM samples to allocate over the m sets q_k. Differently from Algorithm 2, we initialize the allocation with the clustering. In particular, we assign to each set q_k the n_i' samples of the clients included in cluster k. By construction, each of these sets q_k has cardinality $|q_k| \leq M$. We consider the m largest sets, and distribute in these sets the remaining samples of the $K - m$ clusters until $|q_k| = M$, for each k. By construction, this allocation is possible since we have mM total number of samples to be distributed across m sets of cardinality M. As for Algorithm 2, Proposition 3.1 is satisfied: $|q_k| = M$ implies equation (3.7), while equation (3.8) is met since, for each client, the total number of samples distributed across sets q_k is $n_i' = mn_i$. Appendix B.3 completes the proof on the complexity of the algorithm. □

As for Algorithm 2, Appendix B.3 provides a schematic for a better illustration of the algorithm. Being a clustered sampling scheme, the variance of the clients aggregation weights of Algorithm 3 is bounded (equation (3.16)). Moreover, since the distributions are obtained from the similarity tree resulting from the representative gradients, this scheme explicitly promotes the sampling of clients based on their similarity. Finally, with Algorithm 3, the sampling from the distributions $\{W_k\}_{k=1}^m$ can be performed even when no representative gradient is available for the clients, for example if clients have not been sampled during FL yet. In this case we simply consider a constant 0 representative gradient for those clients, and thus group them together to promote their representativity in the same distribution.

We recall that Algorithm 3 does not require to share gradients across clients, but only the difference between local and global models (a.k.a. representative gradients). Thus, the communication cost is the same of standard FL while the privacy properties of FL privacy remain identical.

We emphasize that any valid hierarchical clustering algorithm can be used in Algorithm 3. Without loss of generality, in the rest of this chapter we consider the Ward hierarchical clustering method (J. H. Ward, 1963), which allows to obtain a similarity tree by minimizing at every node the variance of its depending clients. This method has complexity $\mathcal{O}(n^2 \log(n))$. We finally observe that the time complexity of Algorithm 3 is not necessarily an issue, even in presence of an important amount of clients. After aggregation of the new global model, the server can sample the clients, and transmit it to them. While waiting for their local work to be completed, the server can therefore estimate the new partitioning. In this way, Algorithm 3 is equivalent to MD sampling for what concerns the process of receiving the updated models, and transmitting the new global model to the clients.

As a final observation, while Algorithm 3 is originally designed for sampling scenarios where $p_i \leq 1/m$, with few modifications it can be also used for federated datasets composed of clients with larger sample size, i.e. when $I = \{i : p_i \geq 1/m\} \neq \emptyset$, or equivalently $I = \{i : mn_i \geq M\} \neq \emptyset$. In this case, we can simply allocate those clients in specific distributions, where they are sampled with probability 1. In total, we obtain $\lfloor m\frac{n_i}{M} \rfloor$ distributions of this kind. The remaining samples $mn_i - \lfloor m\frac{n_i}{M} \rfloor M < M$ will be then redistributed according to Algorithm 3.

3.6 Experiments

We first show on a standard classification problem on MNIST (LeCun et al., 1998), the advantages of clustered sampling obtained with Algorithm 2 and 3 with respect to MD sampling. We consider a fully connected network with one hidden layer of 50 nodes. We create a federated dataset composed of 100 clients where each one has 500 training and 100 testing samples composed by one digit only. Each digit is owned by 10 clients, every client has the same number of samples, and the server samples 10 clients at every iteration.

We note that an ideal clustering method for this FL problem consists in creating 10 clusters each containing the 10 clients with same classes. At each FL round, we should sample a client from each cluster in order to obtain a fair representation of all the digits in the model aggregations. We call 'target' sampling this ideal FL scenario. In practice, the server cannot adopt 'target' sampling as this requires to know the clients data distributions in advance. As

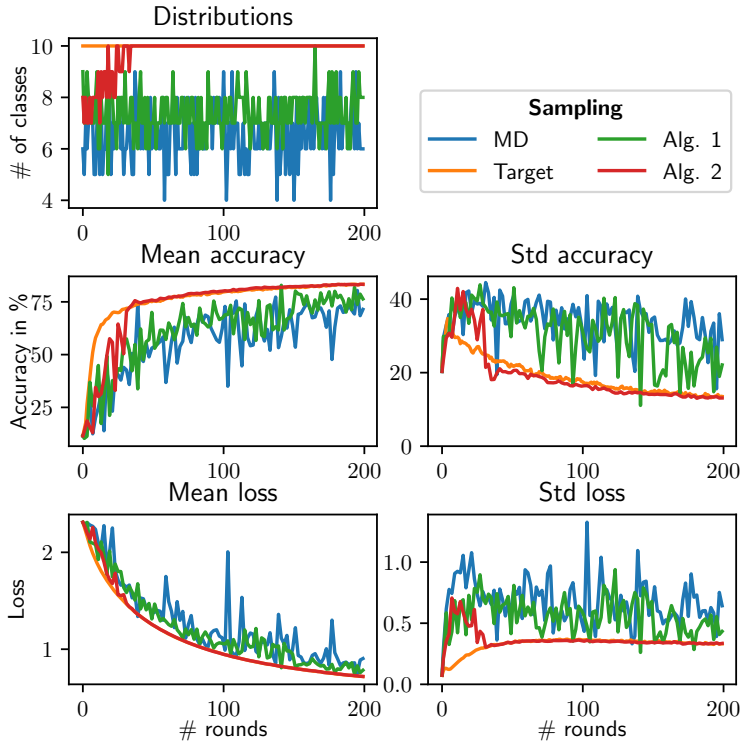

Fig. 3.1.: Comparison of MD sampling with clustered sampling of Algorithm 2 and 3 using cosine angle for the similarity measure. $n = 100$ clients from which $m = 10$ are sampled to perform $N = 50$ SGD with learning rate $lr = 0.01$ and batch size $B = 50$.

we shall see in the rest of this section, the controlled nature of this example allows to clearly appreciate the practical benefits of clustered sampling.

We first show in Figure 3.1 that the FL processes obtained with Algorithm 2 and 3 both outperform MD sampling in terms of training global loss, testing accuracy, and representativity of the sampled classes. Moreover, we note that Algorithm 3 converges to the same ideal performances of 'target' sampling.

We also note that, with MD sampling, between 6 and 8 clients with different digits are generally chosen at each iteration round (Figure 3.1, top left panel). This is a practical demonstration of the sub-optimal representation of the clients heterogeneity. From a statistical perspective, with MD sampling the probability of sampling 10 different clients is $p = \frac{100!}{90!100^{10}} \sim 63\%$. Thus, for 37% of FL iterations, the new global model results from aggregation of less than 10 distinct clients. On the contrary, clustered sampling guarantees by construction that the aggregation will be always performed on 10 different clients. Indeed, since the dataset is balanced and the number of sampled clients $m = 10$ is a divider of the number of clients $n = 100$, every client can be allocated to one distribution only, and can be thus sampled up to once. Moreover, clustered sampling ensures that all the clients have identical aggregation weight variance. This improved data representation translates in less convergence variability at every iteration. Figure 3.1 illustrates this result by showing noticeable improvements in terms of convergence with lower variance and better performance for training loss and testing accuracy. Moreover, with Algorithm 3, although at the early training steps some classes are not represented, the clustering strategy allows to quickly partition the 100 clients in 10 clusters and converge to the ideal distribution of 'target' (Figure 3.1, top left). As a consequence of this improved representativity of clients and classes, Algorithm 3 is associated with smoother and faster convergence processes for training loss and testing throughout iterations.

To demonstrate the benefits of clustered sampling beyond the controlled setting of MNIST, we conduct additional experiments on CIFAR10 (Krizhevsky et al., n.d.) to investigate clustered sampling on more complex data distributions and models. CIFAR10 is composed of 32x32 images with three RGB channels of 10 different classes with 60000 samples. We use the same classifier of McMahan, Moore, et al. (2017) composed of 3 convolutional layers and 2 fully connected ones, including dropout after every convolutional layer.

To measure the influence of non-iid data distributions on the effectiveness of clustered sampling, we partition CIFAR10 using a Dirichlet distribution, $Dir(\alpha)$, giving to each client the respective partitioning across classes. The parameter α monitors the heterogeneity of the created dataset: $\alpha = 0$ assigns one class only to every client, while $\alpha \to +\infty$ gives a uniform partitioning of classes to each client. Harry Hsu et al. (2019) provides graphical illustration of datasets obtained with such a process, and we provide in Appendix B.4 similar

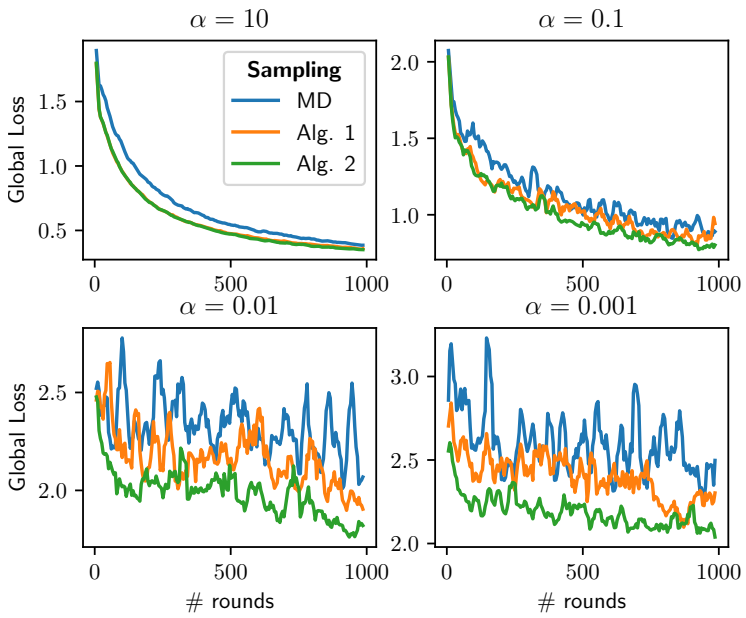

Fig. 3.2.: We investigate the improvement provided by clustered sampling on federated unbalanced datasets partitioned from CIFAR10 using a Dirichlet distribution with parameter $\alpha \in \{0.001, 0.01, 0.1, 10\}$. We use $N = 100$, $m = 10$, and respective learning rate for each dataset $lr = \{0.05, 0.05, 0.05, 0.1\}$.

illustrations for the parameters $\alpha \in \{0.001, 0.01, 0.1, 10\}$ considered in this chapter. To create an unbalanced federated dataset, we consider 100 clients where 10, 30, 30, 20 and 10 clients have respectively 100, 250, 500, 750, and 1000 training samples, and testing samples amounting to a fifth of their training size. All the clients consider a batch size of 50. For every CIFAR10 dataset partition, we report in this chapter experiments with learning rate in $\{0.001, 0.005, 0.01, 0.05, 0.1\}$ minimizing FEDAVG with MD sampling training loss at the end of the learning process.

In Figure 3.2, we show how heterogeneity determines the improvements of clustered sampling over MD sampling for any of the four datasets. We note that the more heterogeneous a dataset is, i.e. the smaller α, the larger is the improvement provided by clustered sampling. Theorem 3.4 shows that clustered sampling has an identical bound as MD sampling. This is retrieved for $\alpha = 10.$ and $\alpha = 0.1$ where the final performances for the two samplings are close with faster convergence for clustered sampling. However, with $\alpha = 0.01$ and $\alpha = 0.001$, clustered sampling provides faster and better convergence. Overall, with clustered sampling, the evolution of the training loss and testing accuracy are smoother processes than with MD sampling.

For sake of clarity, we note that the training losses displayed in Figure 3.2 is computed as the rolling mean over 50 iterations, while we provide in Appendix B.4 the original training

loss evolution. Furthermore, Appendix B.4 reports a larger panel of experiments providing additional verification of the improvements brought by clustered sampling. In Figure 3.1 and 3.2, Algorithm 3 is computed with Arccos similarity. We show in Appendix B.4 that with L2 and L1 we get similar improvements. We also show that increasing the amount of local work N enables clients to update models fitting better their data distribution. As a result, measuring clients similarity is easier, enabling better clustering, and leading to better performances. We also show that for any amount of sampled clients clustered sampling improves MD sampling. Finally, it is worth noticing that in none of the experimental settings considered for this paper clustered sampling underperformed with respect to MD sampling, providing further experimental evidence for our theoretical results.

3.7 Discussion and Conclusion

In this chapter, we introduced clustered sampling, a novel client selection scheme in FL generalizing MD sampling, the current scheme from the state-of-the-art. We proved the correctness of clustered sampling and proposed two clustering methods implementing aggregation based on the clients number of samples, in Algorithm 2, or model similarity, in Algorithm 3. Both algorithms provide smaller weight variance for the clients aggregation process leading to better client representativity. Consistently, clustered sampling is experimentally shown to have faster and smoother convergence in heterogeneous dataset.

The generality of clustered sampling paves the way to further investigation of clients clustering methods based on different criteria than clients sample size or model similarity. To the best of our knowledge, this chapter is also the first one introducing model similarity detection when sampling clients, as opposed to current approaches considering all clients at every iteration.

Finally, clustered sampling is an unbiased sampling scheme simple to implement, while not requiring to modify neither server nor clients behavior during FL training. This aspects makes clustered sampling readily compatible with existing methods and technologies for privacy enhancement and communication reduction.

4 A General Theory for Federated Optimization with Asynchronous and Heterogeneous Clients Updates

Contents

4.1 Introduction **46**
4.2 Background **48**
4.2.1 Federated Optimization Problem 48
4.2.2 Asynchronicity in Clients Updates 49
4.2.3 Server Aggregation Scheme 49
4.2.4 Expressing FL as cumulative GD steps 51
4.2.5 Asynchronous FL as a Sequence of Optimization Problems . . 51
4.2.6 Formalizing Heterogeneity across Clients 52
4.3 Convergence of Federated Problem (4.2) **53**
4.3.1 Assumptions and Property 54
4.3.2 Convergence of Algorithm 4 55
4.3.3 Sufficient Conditions for Minimizing the Federated Problem (4.2) 56
4.3.4 Relaxed Sufficient Conditions for Minimizing the Federated Problem (4.2) 58
4.4 Applications **62**
4.4.1 Heterogeneity of clients hardware and data distributions . . . 62
4.4.2 FEDAVG, Synchronous Federated Learning 64
4.4.3 Asynchronous FEDAVG 64
4.4.4 FEDFIX 66
4.4.5 Generalization 68
4.5 Experiments **69**
4.5.1 Experimental Setting 69

4.5.2 Experimental Results 70
4.6 Discussion **73**

In this chapter, we propose a novel framework to study asynchronous federated learning optimization with delays in gradient updates. In Chapter 2, we extended the standard FEDAVG aggregation scheme by introducing stochastic aggregation weights to represent the variability of a client sampling scheme. In this chapter, we further extend the stochastic aggregation weights definition to represent the variability of the clients update time, due for example to heterogeneous hardware capabilities. In particular, we develop in this chapter FEDFIX, a novel extension of FEDAVG enabling efficient asynchronous federated training while preserving the convergence stability of synchronous aggregation. This chapter is published at the Journal of Machine Learning Research as (Fraboni, Vidal, Kameni, et al., 2022b).

4.1 Introduction

Federated learning (FL) is a training paradigm enabling different clients to jointly learn a global model without sharing their respective data. Federated learning is a generalization of distributed learning (DL), which was first introduced to optimize a given model in star-shaped networks composed of a server communicating with computing machines (Bertsekas and Tsitsiklis, 1989; Nedić et al., 2001; Zinkevich et al., 2009). In DL, the server owns the dataset and distributes it across machines. At every optimization round, the machines return the estimated gradients, and the server aggregates them to perform an SGD step. DL was later extended to account for SGD, and FL extends DL to enable optimization without sharing data between clients. Typical federated training schemes are based on the averaging of clients model parameters optimized locally by each client, such as in FEDAVG (McMahan, Moore, et al., 2017), where at every optimization round clients perform a fixed amount of stochastic gradient descent (SGD) steps initialized with the current global model parameters, and subsequently return the optimized parameters to the server. The server computes the new global model as the average of the clients updates weighted by their respective data ratio.

A key methodological difference between the optimization problem solved in FL and the one of DL lies in the assumption of potentially non independent and identically distributed (iid) data instances (Kairouz et al., 2019; Q. Yang et al., 2019). Proving convergence in the non-iid setup is more challenging, and in some settings, FEDAVG has been shown to converge to a sub-optimum, e.g. when each client performs a different amount of local

work (Jianyu Wang, Q. Liu, et al., 2020), or when clients are not sampled in expectation according to their importance (Cho et al., 2020).

A major drawback of FEDAVG concerns the time needed to complete an optimization round, as the server must wait for all the clients to perform their local work to *synchronize* their update and create a new global model. As a consequence, due to the potential heterogeneity of the hardware across clients, the time for an optimization round is conditioned to the one of the slowest update, while the fastest clients stay idle once they have sent their updates to the server. To address these limitations, asynchronous FL has been proposed to take full advantage of the clients computation capabilities (C. Xu et al., 2021; L. Nguyen et al., 2018; Koloskova et al., 2019; De Sa et al., 2015). In the asynchronous setting, whenever the server receives a client's contribution, it creates a new global model and sends it back to the client. In this way, clients are never idle and always perform local work on a different version of the global model. While asynchronous FL has been investigated in the iid case (Sebastian U Stich and Karimireddy, 2020), a unified theoretical and practical investigation in the non-iid scenario is currently missing.

This chapter introduces a novel theoretical framework for asynchronous FL based on the generalization of the aggregation scheme of FEDAVG, where asynchronicity is modeled as a stochastic process affecting clients' contribution at a given federated aggregation step. More specifically, our framework is based on a stochastic formulation of FL, where clients are given stochastic aggregation weights dependent on their effectiveness in returning an update. Based on this formulation, we provide sufficient conditions for asynchronous FL to converge, and we subsequently give sufficient conditions for convergence to the FL optimum of the associated synchronous FL problem. Our conditions depend on the clients computation time (which can be eventually estimated by the server), and are independent from the clients data heterogeneity, which is usually unknown to the server.

With asynchronous FL, the server only waits for one client contribution to create the new global. As a result, optimization rounds are potentially faster even though the new global improves only for the participating client at the detriment of the other ones. This aspect may affect the stability of asynchronous FEDAVG as compared to synchronous FEDAVG and, as we demonstrate in this work, even diverge in some cases. To tackle this issue, we propose FEDFIX, a robust asynchronous FL scheme, where new global models are created with all the clients contributions received after a fixed amount of time. We prove the convergence of FEDFIX and verify experimentally that it outperforms standard asynchronous FEDAVG in the considered experimental scenarios.

The chapter is structured as follows. In Section 4.2, we introduce our aggregation scheme and the close-form of its aggregation weights in function of the clients computation capabilities and the considered FL optimization routine. Based on our aggregation scheme, in Section

4.3, we provide convergence guarantees, and we give sufficient conditions for the learning procedure to converge to the optimum of the FL optimization problem. In Section 4.4, we apply our theoretical framework to synchronous and asynchronous FEDAVG, and show that this chapter extends current state-of-the-art approaches to asynchronous optimization in FL. Finally, in Section 4.5, we demonstrate experimentally our theoretical results.

4.2 Background

We define here the formalism required by the theory that will be introduced in the following sections. We first introduce in Section 4.2.1 the FL optimization problem, and we adapt it in section 4.2.2 to account for delays in client contributions. We then generalize in Section 4.2.3 the FEDAVG aggregation scheme to account for contributions delays. In Section 4.2.4, we introduce the notion of virtual global models as a direct generalization of gradient descent, and introduce in Section 4.2.5 the final asynchronous FL optimization problem. Finally, we introduce in Section 4.2.6 a formalization of the concept of data heterogeneity across clients.

4.2.1 Federated Optimization Problem

We have M participants owning n_i data points $\{\boldsymbol{z}_{k,i}\}_{k=1}^{n_i}$ independently sampled from a fixed unknown distribution over a sample space $\{\mathcal{Z}_i\}_{i=1}^{M}$. We have $\boldsymbol{z}_{k,i} = (\boldsymbol{x}_{k,i}, \boldsymbol{y}_{k,i})$ for supervised learning, where $\boldsymbol{x}_{k,i}$ is the input of the statistical model, and $\boldsymbol{y}_{k,i}$ its desired target, while we denote $\boldsymbol{z}_{k,i} = \boldsymbol{x}_{k,i}$ for unsupervised learning. Each client optimizes the model's parameters $\boldsymbol{\theta}$ based on the estimated local loss $l(\boldsymbol{\theta}, \boldsymbol{z}_{k,i})$. The aim of FL is solving a distributed optimization problem associated with the averaged loss across clients

$$\mathcal{L}(\boldsymbol{\theta}) \coloneqq \mathbb{E}_{z\sim\hat{\mathcal{Z}}}\left[l(\boldsymbol{\theta}, \boldsymbol{z})\right] = \frac{1}{\sum_{i=1}^{M} n_i} \sum_{i=1}^{M} \sum_{k=1}^{n_i} l(\boldsymbol{\theta}, \boldsymbol{z}_{k,i}), \tag{4.1}$$

where the expectation is taken with respect to the sample distribution $\hat{\mathcal{Z}}$ across the M participating clients. We consider a general form of the federated loss of equation (4.1) where clients local losses are weighted by an associated parameter p_i such that $\sum_{i=1}^{n} p_i = 1$, i.e.

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^{M} p_i \mathcal{L}_i(\boldsymbol{\theta}) \text{ s.t. } \mathcal{L}_i(\boldsymbol{\theta}) = \frac{1}{n_i} \sum_{k=1}^{n_i} l(\boldsymbol{\theta}, \boldsymbol{z}_{k,i}). \tag{4.2}$$

The weight p_i can be interpreted as the importance given by the server to client i in the federated optimization problem. While any combination of $\{p_i\}$ is possible, we note that in

typical FL formulations, either (a) every client has equal importance, i.e. $p_i = 1/M$, or (b) every data point is equally important, i.e. $p_i = n_i / \sum_{i=1}^{M} n_i$.

4.2.2 Asynchronicity in Clients Updates

An optimization round starts at time t^n with global model $\boldsymbol{\theta}^n$, finishes at time t^{n+1} with the new global model $\boldsymbol{\theta}^{n+1}$, and takes $\Delta t^n = t^{n+1} - t^n$ time to complete. No assumptions are made on Δt^n, which can be a random variable, and we set for convenience $t^0 = 0$. In this section, we introduce the random variables needed to develop in Section 4.2.3 the server aggregation scheme connecting two consecutive global models $\boldsymbol{\theta}^n$ and $\boldsymbol{\theta}^{n+1}$.

We define the random variable T_i representing the update time needed for client i to perform its local work and send it to the server for aggregation. T_i depends on the client computation and communication hardware, and is assumed to be independent from the current optimization round n. If the server sets the FL round time to $\Delta t^n = \max_i T_i$, the aggregation is performed by waiting for the contribution of every client, and we retrieve the standard client-server communication scheme of synchronous FedAvg.

With asynchronous FedAvg, we need to relate T_i to the server aggregation time Δt^n. We introduce $\rho_i(n)$ the index of the most recent global model received by client i at optimization round n and, by construction, we have $0 \leq \rho_i(n) \leq n$. We define by

$$T_i^n := T_i - (t^n - t^{\rho_i(n)}) \tag{4.3}$$

the remaining time at optimization round n needed by client i to complete its local work.

Comparing T_i^n with Δt^n indicates whether a client is participating to the optimization round or not, through the stochastic event $\mathbb{I}(T_i^n \leq \Delta t^n)$. When $\mathbb{I}(T_i^n \leq \Delta t^n) = 1$, the local work of client i is used to create the new global model $\boldsymbol{\theta}^{n+1}$, while client i does not contribute when $\mathbb{I}(T_i^n \leq \Delta t^n) = 0$. With synchronous FedAvg, we retrieve $\mathbb{I}(T_i^n \leq \Delta t^n) = \mathbb{I}(T_i \leq \max_i T_i) = 1$ for every client.

Figure 4.1 illustrates the notations described in this section in a FL process with $M = 2$ clients.

4.2.3 Server Aggregation Scheme

We consider $\boldsymbol{\Delta}_i(n)$ the contribution of client i received by the server at optimization round n. In the rest of this chapter, we consider that clients perform K steps of SGD on the model they receive from the server. By calling their trained model $\boldsymbol{\theta}_i^{n,k}$ after k SGD, we can rewrite

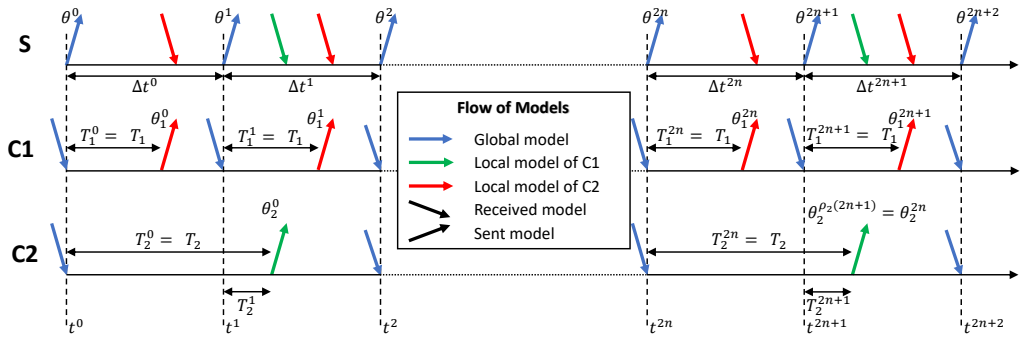

Fig. 4.1.: Illustration of the time notations introduced in Section 4.2.2 with $M = 2$ clients. The frequency of the updates of Client 1 (C1) is twice the one of Client 2 (C2). If the server (S) creates the new global model after every fixed waiting time ($\Delta t^n = \Delta t$), C1 contributes at every optimization round, while C2 contributes once every two rounds. This aggregation policy define the federated learning strategy FEDFIX (Section 4.4.4).

clients contribution for FEDAVG as $\boldsymbol{\Delta}_i(n) \coloneqq \boldsymbol{\theta}_i^{n,K} - \boldsymbol{\theta}^n$, and the FEDAVG aggregation scheme as

$$\boldsymbol{\theta}^{n+1} \coloneqq \boldsymbol{\theta}^n + \sum_{i=1}^{M} p_i \boldsymbol{\Delta}_i(n). \tag{4.4}$$

With FEDAVG, the server waits for every client to send its contribution $\boldsymbol{\Delta}_i(n)$ to create the new global model. To allow for partial computation within the server aggregation scheme, we introduce the aggregation weight $d_i(n)$ corresponding to the weight given by the server to client i at optimization round n. We can then define the stochastic aggregation weight $\omega_i(n)$ given to client i at optimization step n as

$$\omega_i(n) \coloneqq \mathbb{I}(T_i^n \leq \Delta t^n) d_i(n), \tag{4.5}$$

with $\omega_i(n) = d_i(n)$ if client i updated its work at optimization round n and $\omega_i(n) = 0$ otherwise. In the general setting, client i receives $\boldsymbol{\theta}^{\rho_i(n)}$ and its contribution is $\boldsymbol{\Delta}_i(\rho_i(n)) = \boldsymbol{\theta}_i^{\rho_i(n),K} - \boldsymbol{\theta}^{\rho_i(n)}$. By weighting each delayed contribution $\boldsymbol{\Delta}_i(\rho_i(n))$ with its stochastic aggregation weight $\omega_i(n)$, we propose the following aggregation scheme

$$\boldsymbol{\theta}^{n+1} \coloneqq \boldsymbol{\theta}^n + \eta_g \sum_{i=1}^{M} \omega_i(n) \boldsymbol{\Delta}_i(\rho_i(n)), \tag{4.6}$$

where η_g is a global learning rate that the server can use to mitigate the disparity in clients contributions (Reddi et al., 2021; Karimireddy et al., 2020; Jianyu Wang, Tantia, et al., 2020). Equation (4.6) generalizes FedAvg aggregation scheme (4.4) ($\eta_g = 1$ and $\Delta t^n = \max_i T_i$), and the one of Fraboni, Vidal, Kameni, et al. (2022a) based on client sampling.

We introduce with Algorithm 4 the implementation of the optimization schemes satisfying aggregation scheme (4.6) with stochastic aggregation weights satisfying equation (4.5).

Algorithm 4 Asynchronous Federated Learning based on equation (4.6)

Require: server learning rate η_g, aggregation weights $\{d_i(n)\}$, number of SGD K, learning rate η_l, batch size B, aggregation time policy Δt^n.

1: The server sends to the M clients the learning parameters (K, η_l, B) and the initial global model $\boldsymbol{\theta}^0$.
2: **for** $n \in \{0, ..., N-1\}$ **do**
3: Clients in $S_n = \{i : T_i^n \leq \Delta t^n\}$ send their contribution $\boldsymbol{\Delta}_i(\rho_i(n)) = \boldsymbol{\theta}_i^{\rho_i(n),K} - \boldsymbol{\theta}^{\rho_i(n)}$ to the server.
4: The server creates the new global model $\boldsymbol{\theta}^{n+1} = \boldsymbol{\theta}^n + \eta_g \sum_{i \in S_n} d_i(n) \boldsymbol{\Delta}_i(\rho_i(n))$, equation (4.6).
5: The global model $\boldsymbol{\theta}^{n+1}$ is sent back to the clients in S_n.
6: **end for**

4.2.4 Expressing FL as cumulative GD steps

To obtain the tightest possible convergence bound, we consider a convergence framework similar to the one of Xiang Li et al. (2020) and Khaled et al. (2020a). We introduced the aggregation rule for the server global models $\{\boldsymbol{\theta}^n\}$ in Section 4.2.3, and we generalize it in this section by introducing the virtual sequence of global models $\boldsymbol{\theta}^{n,k}$. This sequence corresponds to the *virtual* global model that would be obtained with the clients contribution at optimization round n computed on $k \leq K$ SGD, i.e.

$$\boldsymbol{\theta}^{n,k} := \boldsymbol{\theta}^n + \eta_g \sum_{i=1}^{M} \omega_i(n) \left[\boldsymbol{\theta}_i^{\rho_i(n),k} - \boldsymbol{\theta}^{\rho_i(n)} \right]. \tag{4.7}$$

We retrieve $\boldsymbol{\theta}^{n,0} = \boldsymbol{\theta}^n$ and $\boldsymbol{\theta}^{n,K} = \boldsymbol{\theta}^{n+1,0} = \boldsymbol{\theta}^{n+1}$. The server has not access to $\boldsymbol{\theta}^{n,k}$ when $k \neq 0$ or $k \neq K$. Hence the name virtual for the model $\boldsymbol{\theta}^{n,k}$.

The difference between two consecutive global models in our virtual sequence depends on the sum of the differences between local models $\boldsymbol{\theta}_i^{\rho_i(n),k+1} - \boldsymbol{\theta}_i^{\rho_i(n),k} = -\eta_l \nabla \mathcal{L}_i(\boldsymbol{\theta}_i^{\rho_i(n),k}, \boldsymbol{\xi}_i)$, where $\boldsymbol{\xi}_i$ is a random batch of data samples of client i. Hence, we can rewrite the aggregation process as a GD step with

$$\boldsymbol{\theta}^{n,k+1} = \boldsymbol{\theta}^{n,k} - \eta_g \eta_l \sum_{i=1}^{M} \omega_i(n) \nabla \mathcal{L}_i(\boldsymbol{\theta}_i^{\rho_i(n),k}, \boldsymbol{\xi}_i). \tag{4.8}$$

4.2.5 Asynchronous FL as a Sequence of Optimization Problems

For the rest of this work, we define $q_i(n) := \mathbb{E}[\omega_i(n)]$, the expected aggregation weight of client i at optimization round n. No assumption is made on $q_i(n)$ which can vary across

optimization rounds. The expected clients contribution $\sum_{i=1}^{M} q_i(n)\boldsymbol{\Delta}_i(n)$ help minimizing the optimization problem $\mathcal{L}^n$ defined as

$$\mathcal{L}^n(\boldsymbol{\theta}) \coloneqq \sum_{i=1}^{M} q_i(n)\mathcal{L}_i(\boldsymbol{\theta}). \tag{4.9}$$

We denote by $\bar{\boldsymbol{\theta}}^n$ the optimum of $\mathcal{L}^n$ and by $\boldsymbol{\theta}^*$ the optimum of the optimization problem $\mathcal{L}$ defined in equation (4.2). Finally, we define by $q_i = \frac{1}{N}\sum_{n=0}^{N-1} q_i(n)$ the expected importance given to client i over the N server aggregations during the FL process, and by $\tilde{q}_i(n)$ the normalized expected importance $\tilde{q}_i(n) = q_i(n)/(\sum_{i=1}^{M} q_i(n))$. We define by $\bar{\mathcal{L}}$ the associated optimization problem

$$\bar{\mathcal{L}}(\boldsymbol{\theta}) \coloneqq \sum_{i=1}^{M} q_i\mathcal{L}_i(\boldsymbol{\theta}) = \frac{1}{N}\sum_{n=0}^{N-1} \mathcal{L}^n(\boldsymbol{\theta}), \tag{4.10}$$

and we denote by $\bar{\boldsymbol{\theta}}$ the associated optimum.

Finally, we introduce the following expected convergence residual, which quantifies the variance at the optimum in function of the relative clients importance $q_i(n)$

$$\Sigma \coloneqq \sum_{i=1}^{M} q_i \, \mathbb{E}_{\boldsymbol{\xi}_i}\left[\left\|\nabla\mathcal{L}_i(\bar{\boldsymbol{\theta}}, \boldsymbol{\xi}_i)\right\|^2\right]. \tag{4.11}$$

The convergence guarantees provided in this chapter (Section 4.3) are proportional to the expected convergence residual and extend the ones provided for the synchronous setting in the work of Khaled et al. (2020a). The quantity Σ is finite and serves as a natural measure of variance in local optimization methods. Σ is positive and null only when clients have the same loss function and perform GD steps for local optimization. The work of Khaled et al. (2020a) shows how considering Σ provides tighter convergence guarantees than when assuming, for each client's gradient estimator, a bounded variance σ^2. This is a common assumption in synchronous FL (Xiang Li et al., 2020; Jianyu Wang, Q. Liu, et al., 2020). A thorough analysis of the relationship between Σ and σ^2 is provided in Appendix C.1.1.

4.2.6 Formalizing Heterogeneity across Clients

We assume the existence of $J \leq M$ different clients feature spaces $\mathcal{Z}_i$ and, without loss of generality, assume that the first J clients feature spaces are different. This formalism allows us to represent the heterogeneity of data distribution across clients. In DL problems, we have $J < M$ when the same dataset split is accessible to many clients. When clients share the same distribution, we assume that their optimization problem is equivalent. In this case,

	Client i	Sample distribution j
Importance	p_i	r_j
Stochastic aggregation weight	$\omega_i(n)$	-
Aggregation weight	$d_i(n)$	-
Expected agg. weight	$q_i(n)$	$s_j(n)$
Normalized expected agg. weight	$\tilde{q}_i(n)$	$\tilde{s}_j(n)$
Expected agg. weight over N rounds	q_i	s_j

Tab. 4.1.: The different weights used to account for the importance of clients or data distributions at every optimization round and during the full FL process.

we call $F_j(\boldsymbol{\theta})$ their loss function with optimum $\boldsymbol{\theta}_j^*$. The federated problem of equation (4.2) can thus be formalized with respect to the discrepancy between the clients feature spaces $\mathcal{Z}_i$. To this end, we define Q_j the set of clients with the same feature space of client j, i.e. $Q_j \coloneqq \{i : \mathcal{Z}_i = \mathcal{Z}_j\}$. Each feature space as thus importance $r_j = \sum_{i \in Q_j} p_i$, and expected importance $s_j(n) = \sum_{i \in Q_j} q_i(n)$ such that

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{j=1}^{J} r_j F_j(\boldsymbol{\theta}) \text{ and } \mathcal{L}^n(\boldsymbol{\theta}) = \sum_{j=1}^{J} s_j(n) F_j(\boldsymbol{\theta}). \tag{4.12}$$

As for $\tilde{q}_i(n)$, we define $\tilde{s}_j(n) = s_j(n) / \sum_{i=1}^{M} s_j(n)$.

In Table 4.1, we summarize the different weights used to adapt the federated optimization problem to account respectively for heterogeneity in clients importance and data distributions across rounds.

4.3 Convergence of Federated Problem (4.2)

In this section, we prove the convergence of the optimization based on the stochastic aggregation scheme defined in equation (4.6), with implementation given in Algorithm 4. We first introduce in Section 4.3.1 the necessary assumptions and then prove with Theorem 4.1 the convergence of the sequence of optimized models (Section 4.3.2). We show in Section 4.3.3 the implications of Theorem 4.1 on the convergence of the federated problem (4.2), and propose sufficient conditions for the learnt model to be the associated optimum. Finally, with two additional assumptions, we propose in Section 4.3.4 simpler and practical sufficient conditions for FL convergence to the optimum of the federated problem (4.2).

4.3.1 Assumptions and Property

We make the following assumptions regarding the Lipschitz smoothness and convexity of the clients local loss functions (Assumption 4.1 and 4.2), unbiased gradients estimators (Assumption 4.3), and finite answering time for the clients (Assumption 4.4). Assumption 4.3 (Khaled et al., 2020a) considers unbiased gradient estimators without assuming bounded variance, giving in turn more interpretable convergence bounds.

Assumption 4.1 (Smoothness)**.** *Clients local objective functions are L-Lipschitz smooth, that is,* $\forall i \in \{1, ..., n\},\ \|\nabla \mathcal{L}_i(\boldsymbol{x}) - \nabla \mathcal{L}_i(\boldsymbol{y})\| \leq L \|\boldsymbol{x} - \boldsymbol{y}\|$.

Assumption 4.2 (Convexity)**.** *Clients local objective functions are convex.*

Assumption 4.3 (Unbiased Gradient)**.** *Every client stochastic gradient $g_i(\boldsymbol{x}) = \nabla \mathcal{L}_i(\boldsymbol{x}, \boldsymbol{\xi}_i)$ of a model with parameters $\boldsymbol{x}$ evaluated on batch $\boldsymbol{\xi}_i$ is an unbiased estimator of the local gradient, i.e.* $\mathbb{E}_{\boldsymbol{\xi}_i}\left[g_i(\boldsymbol{x})\right] = \nabla \mathcal{L}_i(\boldsymbol{x})$.

Assumption 4.4 (Finite Answering Time)**.** *The server receives a client local work in at most $\tau \coloneqq \max_{i,n}(n - \rho_i(n))$ optimization steps, which satisfy $\mathbb{P}(\tau < \infty) = 1$.*

Finally, before focusing our attention on the convergence of Algorithm 4, we introduce Property 4.1 which states that the covariance between two aggregation weights can be expressed as the product of their expected aggregation weight up to a multiplicative factor α.

Property 4.1. *There exists $\alpha \in [-1, 1]$ such that $\mathbb{E}\left[\omega_i(n)\omega_j(n)\right] \leq \alpha q_i(n) q_j(n)$.*

The proof of Property 4.1 follows from the definition of the clients aggregation weights, equation (4.5), which gives

$$\mathbb{E}\left[\omega_i(n)\omega_j(n)\right] = \mathbb{P}(T_i^n \leq \Delta t^n, T_j^n \leq \Delta t^n) d_i(n) d_j(n) \leq q_i(n) q_j(n). \tag{4.13}$$

This last equality shows that Property 4.1 is always verified by $\alpha = 1$. In Section 4.4, we show that there exists α such that Property 4.1 is an equality for synchronous FL, asynchronous FL, and FedFix. We also derive such an α in close-form as function of the different training parameters. The work of Fraboni, Vidal, Kameni, et al. (2022a) shows that Property 4.1 also holds as an equality for numerous client samplings and provides for each of them related α in close-form.

4.3.2 Convergence of Algorithm 4

Before providing convergence guarantees for the federated optimization problem (4.2), we first prove with Theorem 4.1 the convergence of Algorithm 4.

Theorem 4.1. *Under Assumptions 4.1 to 4.4, with* $\eta_l \leq 1/48KL\min\left(1, 1/3\rho^2\eta_g(\tau+1)\right)$*, we obtain the following convergence bound:*

$$\frac{1}{N}\sum_{n=0}^{N-1}\frac{1}{K}\sum_{k=0}^{K-1}\left[\mathbb{E}\left[\mathcal{L}^n(\boldsymbol{\theta}^{n,k})\right] - \mathcal{L}^n(\bar{\boldsymbol{\theta}}^n)\right] \leq R(\{\mathcal{L}^n\}) + \epsilon_F + \epsilon_K + \epsilon_\alpha + \epsilon_\beta, \quad (4.14)$$

where

$$R(\{\mathcal{L}^n\}) = \frac{1}{N}\sum_{n=0}^{N-1}\left[\mathcal{L}^n(\bar{\boldsymbol{\theta}}) - \mathcal{L}^n(\bar{\boldsymbol{\theta}}^n)\right], \qquad \epsilon_F = \frac{1}{\tilde{\eta}KN}\left\|\boldsymbol{\theta}^0 - \bar{\boldsymbol{\theta}}\right\|^2, \quad (4.15)$$

$$\epsilon_K = \mathcal{O}\left(\eta_l^2(K-1)^2\left[R(\{\mathcal{L}^n\}) + \Sigma\right]\right), \quad \epsilon_\alpha = \mathcal{O}\left(\alpha\left[\tilde{\eta} + \tilde{\eta}^2K^2\tau^2\right]\left[R(\{\mathcal{L}^n\}) + \max q_i(n)\Sigma\right]\right), \quad (4.16)$$

$$\epsilon_\beta = \mathcal{O}\left(\beta\left[\tilde{\eta} + \tilde{\eta}^2K^2\tau^2\right]\left[R(\{\mathcal{L}^n\}) + \Sigma\right]\right), \quad \tilde{\eta} = \eta_g\eta_l, \quad \beta \coloneqq \max\{d_i(n) - \alpha q_i(n)\}, \quad (4.17)$$

and $\mathcal{O}$ *accounts for numerical constants and the loss function Lipschitz smoothness* L*.*

Theorem 4.1 is proven in Appendix C.1. The convergence guarantee provided in Theorem 4.1 is composed of 5 terms: $R(\{\mathcal{L}^n\})$, ϵ_F, ϵ_K, ϵ_α, ϵ_β. In the following, we describe these terms and explain their origin in a given optimization scheme.

Optimized expected residual $R(\{\mathcal{L}^n\})$. The residual $R(\{\mathcal{L}^n\})$ quantifies the sensitivity of $\mathcal{L}^n$ between its optimum $\bar{\boldsymbol{\theta}}^n$ and the optimum $\bar{\boldsymbol{\theta}}$ of the overall expected minimized problem across optimization rounds $\tilde{\mathcal{L}}$. As such, the residual accounts for the heterogeneity in the history of optimized problems, and is minimized to 0 when the same optimization problem is minimized at every round n, i.e. $\mathcal{L}^n = \tilde{\mathcal{L}}$. This condition is always satisfied when clients have identical data distributions, but requires for the server to set properly every client aggregation weight $d_i(n)$ in function of the server waiting time policy Δt^n and the clients hardware capabilities T_i^n in the general case (Section 4.3.3 and 4.3.4).

Initialization quality ϵ_F. ϵ_F only depends of the quality of the initial model $\boldsymbol{\theta}^0$ through its distance with respect to the optimum $\bar{\boldsymbol{\theta}}$ of the overall expected minimized problem across

optimization rounds $\tilde{\mathcal{L}}$. This convergence term can only be minimized by performing as many serial SGD steps KN.

Clients data heterogeneity ϵ_K. This term accounts for the disparity in the clients updated models, and is proportional to the clients amount of local work K (quadratically) and to the heterogeneity of their data distributions $\mathcal{Z}_i$ through Σ_1. When $K = 1$, every client perform its SGD on the same model, which reduces the server aggregation to a traditional centralized SGD. We retrieve $\epsilon_K = 0$.

Gradient delay τ **through** ϵ_α **and** ϵ_β. Decreasing the server time policy Δt^n allows faster optimization rounds but decreases a client's participation probability $\mathbb{P}(T_i^n \leq \Delta t^n)$ resulting in an increased maximum answering time τ. In turn, we note that ϵ_α and ϵ_β are quadratically proportional to the maximum amount of serial SGD $K\tau$. This latter terms quantifies the maximum amount of SGD integrated in the global model $\boldsymbol{\theta}^n$.

4.3.3 Sufficient Conditions for Minimizing the Federated Problem (4.2)

Theorem 4.1 provides convergence guarantees for the history of optimized models $\{\mathcal{L}^n\}$. Under the same assumptions of Theorem 4.1, we can provide convergence guarantees for the original FL problem $\mathcal{L}(\boldsymbol{\theta})$ (proof in Appendix C.2).

Theorem 4.2. *Under the same conditions of Theorem 4.1, we have*

$$\frac{1}{N}\sum_{n=0}^{N-1}\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\left[\left\|\nabla\mathcal{L}(\boldsymbol{\theta}^{n,k})\right\|^2\right] \leq \mathcal{O}\left(R(\{\mathcal{L}^n\})\right) + P(\{\mathcal{L}^n\}) + U(\{\mathcal{L}^n\}) + \mathcal{O}\left(\epsilon_F\right) + \epsilon_K + \epsilon_\alpha + \epsilon_\beta, \quad (4.18)$$

where

$$P(\{\mathcal{L}^n\}) = \mathcal{O}\left(\frac{1}{N}\sum_{n=0}^{N-1}\chi_n^2\sum_{j\in W_n}\tilde{s}_j(n)\left[F_j(\bar{\boldsymbol{\theta}}^n) - F_j(\boldsymbol{\theta}_j^*)\right]\right), \quad (4.19)$$

$$U(\{\mathcal{L}^n\}) = \mathcal{O}\left(\frac{1}{N}\sum_{n=0}^{N-1}\frac{1}{K}\sum_{k=0}^{K-1}\sum_{j\notin W_n} r_j\left[\mathbb{E}\left[F_j(\boldsymbol{\theta}^{n,k})\right] - F_j(\boldsymbol{\theta}_j^*)\right]\right), \quad (4.20)$$

$\chi_n^2 = \sum_{j\in W_n}(r_j - \tilde{s}_j(n))^2/\tilde{s}_j(n)$, *and* $W_n = \{j : s_j(n) > 0\}$.

Theorem 4.2 provides convergence guarantees for the optimization problem (4.2) and generalizes the Theorem 2 in the work of Jianyu Wang, Q. Liu, et al. (2020) developed for the synchronous setting.

We retrieve the components of the convergence bound of Theorem 4.1. The terms ϵ_F to ϵ_τ can be mitigated by choosing an appropriate local learning rate η_l, but the same cannot be said for $R(\{\mathcal{L}^n\})$, $P(\{\mathcal{L}^n\})$, $U(\{\mathcal{L}^n\})$. Behind these three quantities, Theorem 4.2 shows that proper expected representation of every dataset type is needed, i.e. $s_j(n) = r_j$. Indeed, if a client is poorly represented, i.e. $s_j(n) \neq r_j$, then $R(\{\mathcal{L}^n\}) > 0$ and $P(\{\mathcal{L}^n\}) > 0$, while if a client is not represented at all, i.e. $s_j(n) = 0$, then $U(\{\mathcal{L}^n\}) > 0$. Therefore, we propose, with Corollary 4.1, sufficient conditions for any FL optimization scheme satisfying Algorithm 4 to converge to the optimum of the federated problem (4.2).

We also note that the discussions made in Section 4.3.2 on the implications of Theorem 4.1 to provide tighter convergence guarantees (regarding the expected residuals, initialization quality, data heterogeneity, and gradient delay) can be translated to Theorem 4.2 and Corollary 4.1, therefore providing relevant insights on the rate of convergence to reach the optimum in asynchronous FL.

Corollary 4.1. *Under the conditions of Theorem 4.1, if every client data distribution satisfies $\tilde{s}_j(n) = r_j$, the following convergence bound for optimization problem (4.2) can be obtained*

$$\frac{1}{N}\sum_{n=0}^{N-1}\frac{1}{K}\sum_{k=0}^{K-1}\left[\mathbb{E}\left[\mathcal{L}(\boldsymbol{\theta}^{n,k})\right] - \mathcal{L}(\boldsymbol{\theta}^*)\right] \leq \epsilon_F + \epsilon_K + \epsilon_\alpha + \epsilon_\beta. \tag{4.21}$$

Proof. Follows directly. $\tilde{s}_j(n) = r_j$ implies $\chi_n^2 = 0$, $W_n = \emptyset$, $\mathcal{L}^n = q(n)\mathcal{L}$, and $\bar{\boldsymbol{\theta}}^n = \boldsymbol{\theta}^*$. □

These theoretical results provide relevant insights for different FL scenarios.

iid data distributions, $\mathcal{Z}_i = \mathcal{Z}$. Consistently with the extensive literature on synchronous and asynchronous distributed learning, when clients have data points sampled from the same data distribution, FL always converges to its optimum (Corollary 4.1). Indeed, $\tilde{s}_j(n) = r_j = 1$ regardless of which clients are participating, and what importance p_i or aggregation weight $d_i(n)$ a client is given.

non-iid data distributions. The convergence of FL to the optimum requires to optimize by considering every data distribution type fairly at every optimization round, i.e. $\tilde{s}_j(n) = r_j$ (Corollary 4.1). This condition is weaker than requiring to treat fairly every client at every optimization round, i.e. $q_i(n) = p_i$. Ideally, only one client per data type needs to have a non-zero participating probability, i.e. $\mathbb{P}(T_i^n \leq \Delta t^n) > 0$, and an appropriate $d_i(n)$ such that $\tilde{s}_j(n) = r_j$ is satisfied. In practice, knowing the clients data distribution is not possible. Therefore, ensuring FL convergence to its optimum requires at every optimization round $\tilde{q}_i(n) = p_i$ (Jianyu Wang, Q. Liu, et al., 2020).

We provide in Example 4.1 an illustration on these results based on quadratic loss functions to show that considering fairly data distributions is sufficient for an optimization scheme satisfying Algorithm 4 to converge to the optimum of the optimization problem (4.2), since $\tilde{s}_j(n) = r_j$ is satisfied at every optimization round, while $\tilde{q}_i(n) \neq p_i$ may not be satisfied.

Example 4.1. *Let us consider four clients with data distributions such that their loss can be expressed as* $\mathcal{L}_i(\boldsymbol{\theta}) = \frac{1}{2}\|\boldsymbol{\theta} - \boldsymbol{\theta}_i^*\|^2$ *with* $\boldsymbol{\theta}_1^* = \boldsymbol{\theta}_2^*$ *(*$\mathcal{Z}_1$*),* $\boldsymbol{\theta}_3^* = \boldsymbol{\theta}_4^*$ *(*$\mathcal{Z}_2$*), and identical client importance, i.e.* $p_i = 1/4$*. Therefore, each data type has identical importance, i.e.* $r_j = 1/2$*, and the optimum satisfies* $\boldsymbol{\theta}^* = \frac{1}{2}[\boldsymbol{\theta}_1^* + \boldsymbol{\theta}_3^*]$*. We consider that clients with odd index participate at odd optimization rounds while the ones with even index at even optimization rounds, i.e.* $q_1^{2n+1} = q_3^{2n+1} = q_2^{2n} = q_4^{2n} = 1/2$ *and* $q_1^{2n} = q_2^{2n} = q_3^{2n+1} = q_4^{2n+1} = 0$ *which gives* $\tilde{s}_1(n) = \tilde{s}_2(n) = 1/2$ *and* $\tilde{q}_i(n) = 0$ *or* $\tilde{q}_i(n) = 1/2$ *but not* $\tilde{q}_i(n) = 1/4$*. With* $\eta_g = 1$*, equation (4.6) can be rewritten as*

$$\boldsymbol{\theta}^{n+2} = \boldsymbol{\theta}^{n+1} + \frac{1}{2}\left[(\boldsymbol{\theta}_1^{n+1} - \boldsymbol{\theta}^n) + (\boldsymbol{\theta}_3^{n+1} - \boldsymbol{\theta}^n)\right]. \tag{4.22}$$

Clients update can be rewritten as $\boldsymbol{\theta}_i^{n+1} - \boldsymbol{\theta}^n = \phi(\boldsymbol{\theta}_i^* - \boldsymbol{\theta}^n)$*, where* $\phi = 1 - (1 - \eta_l)^K$*. Equation (*4.22*) can thus be rewritten as*

$$\boldsymbol{\theta}^{n+2} - \boldsymbol{\theta}^{n+1} + \phi\boldsymbol{\theta}^n = \phi\boldsymbol{\theta}^*. \tag{4.23}$$

*Solving equation (*4.23*) proves FL asymptotic convergence to the optimum* $\boldsymbol{\theta}^*$*.*

4.3.4 Relaxed Sufficient Conditions for Minimizing the Federated Problem (4.2)

Theorem 4.2 holds for any client's update time T_i and optimization scheme satisfying Algorithm 4, and provides finite convergence guarantees for the optimization problem (4.2). Corollary 4.1 shows that for the asymptotic convergence of FL, data distribution types should be treated fairly in expectation, i.e. $\tilde{s}_j(n) = r_j$. This sufficient condition is not necessarily realistic, since the server cannot know the clients data distributions and participation time, and thus needs to give to every client an aggregation weight $d_i(n)$ such that $\tilde{q}_i(n) = p_i$ without knowing T_i.

In Example 4.1, we note that we have $\frac{1}{2}\left[q_i^{2n} + q_i^{2n+1}\right] = p_i$. Therefore, every client is given proper consideration every two optimization rounds. Based on Example 4.1, in Theorem 4.3 we provide weaker sufficient conditions than the ones of Corollary 4.1. To this end, we introduce Property 4.2 stating that, in a window of size W optimization rounds, clients are always contributing according to their expected importance q_i.

Property 4.2 (Window). $\exists W \geq 1$ *such that* $\forall s,\ \frac{1}{W}\sum_{n=sW}^{(s+1)W-1} q_i(n) = q_i$.

Property 4.2 states that over a cycle of W aggregations, the sum of a client's expected aggregation weights $q_i(n)$ is constant. By definition of q_i, Property 4.2 is always satisfied for $W = N$. In addition, we show in Section 4.4 that Property 4.2 holds for all the asynchronous optimization schemes used in our work, and provide W in close-form depending on M, the amount of participating clients, and their associated update time.

Finally, we consider with Assumption 4.5 that clients gradients are bounded. This assumption has been considered in previous work on federated optimization including Xiang Li et al. (2020) and Sebastian U. Stich (2019), and can be justified by the use of gradient clipping during the practical optimization of deep learning models to prevent exploding gradients. With gradient clipping, a given threshold B is introduced, and gradients with norm exceeding this threshold are clipped to norm B.

Assumption 4.5 (Bounded Gradients). *The expected squared norm of gradients is uniformly bounded, i.e.* $\exists B > 0$ *such that* $\mathbb{E}_{\xi_i}\left[\|\nabla \mathcal{L}_i(\boldsymbol{x}, \xi_i)\|^2\right] \leq B^2$.

Therefore, using Assumption 4.2, Assumption 4.5 and the Cauchy Schwartz inequality gives

$$\mathbb{E}\left[\mathcal{L}_i(\boldsymbol{\theta}^{n,k+1})\right] - \mathbb{E}\left[\mathcal{L}_i(\boldsymbol{\theta}^{n,k})\right] \leq \mathbb{E}\left[\langle \nabla \mathcal{L}_i(\boldsymbol{\theta}^{n,k+1}), \boldsymbol{\theta}^{n,k+1} - \boldsymbol{\theta}^{n,k}\rangle\right] \leq \eta_g \eta_l q(n) B^2. \tag{4.24}$$

Finally, using equation (4.24) and Property 4.2, the performance history on the optimized problem can be bounded as follows

$$\sum_{n=sW}^{(s+1)W-1}\sum_{k=0}^{K-1} q_i\, \mathbb{E}\left[\mathcal{L}_i(\boldsymbol{\theta}^{n,k})\right] \leq \sum_{n=sW}^{(s+1)W-1}\sum_{k=0}^{K-1} q_i(n)\left[\mathbb{E}\left[\mathcal{L}_i(\boldsymbol{\theta}^{n,k})\right] + \eta_g \eta_l K(W-1)B^2\right]. \tag{4.25}$$

Theorem 4.3. *Under Assumption 4.1 to 4.5, and considering that* W *is a divider of* N*, we get the following convergence bound for the optimization problem* (4.10)*:*

$$\frac{1}{N}\sum_{n=0}^{N-1}\frac{1}{K}\sum_{k=0}^{K-1}\left[\mathbb{E}\left[\bar{\mathcal{L}}(\boldsymbol{\theta}^{n,k})\right] - \bar{\mathcal{L}}(\bar{\boldsymbol{\theta}})\right] \leq \epsilon \coloneqq \epsilon_F + \epsilon_K + \epsilon_\alpha + \epsilon_\beta + \epsilon_W, \tag{4.26}$$

where $\epsilon_W = \mathcal{O}(\eta_g \eta_l (W-1)K)$. *Furthermore, we obtain the following convergence guarantees for the federated problem* (4.2)*:*

$$\frac{1}{N}\sum_{n=0}^{N-1}\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\left[\left\|\nabla\mathcal{L}(\boldsymbol{\theta}^{n,k})\right\|^2\right] \leq \epsilon + \mathcal{O}(\chi^2[\bar{\mathcal{L}}(\bar{\boldsymbol{\theta}}) - \sum_{j=1}^{J} s_j F_j(\boldsymbol{\theta}_j^*)]), \tag{4.27}$$

where $\chi^2 = \sum_{j=1}^{J} \frac{(r_j - \tilde{s}_j)^2}{\tilde{s}_j}$.

Proof.

$$\begin{aligned}
&\frac{1}{N}\sum_{n=0}^{N-1}\frac{1}{K}\sum_{k=0}^{K-1}\left[\mathbb{E}\left[\bar{\mathcal{L}}(\boldsymbol{\theta}^{n,k})\right] - \bar{\mathcal{L}}(\bar{\boldsymbol{\theta}})\right] \\
&\leq \frac{1}{N}\sum_{n=0}^{N-1}\frac{1}{K}\sum_{k=0}^{K-1} q_i(n)\left[\mathbb{E}\left[\mathcal{L}_i(\boldsymbol{\theta}^{n,k})\right] + \tilde{\eta}K(W-1)B^2\right] - \bar{\mathcal{L}}(\bar{\boldsymbol{\theta}}) \qquad (4.28)\\
&\leq R(\{\mathcal{L}^n\}) + \epsilon + \frac{1}{N}\sum_{n=0}^{N-1}\mathcal{L}^n(\bar{\boldsymbol{\theta}}^n) - \bar{\mathcal{L}}(\bar{\boldsymbol{\theta}}) = \epsilon, \qquad (4.29)
\end{aligned}$$

where we use equation (4.25) for the first inequality and Theorem 4.1 for the second inequality.

Finally, we can obtain convergence guarantees on the optimization problem (4.2) with Theorem 4.2 by considering the minimization of the optimization problem $\bar{\mathcal{L}}$. Therefore, the bound of Theorem 4.2 can be simplified noting that $\mathcal{L}^n = \bar{\mathcal{L}}$, $\bar{\boldsymbol{\theta}}^n = \bar{\boldsymbol{\theta}}$, $W_n = \emptyset$, $\chi_n^2 = \chi^2$, and by adding ϵ_W, which completes the proof. □

Theorem 4.3 shows that the condition $\tilde{s}_j = r_j$ is sufficient to minimize the optimization problem (4.2). In practice, for privacy concerns, clients may not want to share their data distribution with the server, and thus the relaxed sufficient condition becomes $\tilde{q}_i = p_i$. This condition is weaker than the one obtained with Corollary 4.1, at the detriment of a looser convergence bound including an additional asymptotic term ϵ_W linearly proportional to the window size W. Therefore, for a given learning application, the maximum local work delay τ and the window size W need to be considered when selecting an FL optimization scheme satisfying Algorithm 4. Also, the server needs to properly allocate clients aggregation weight $d_i(n)$ such that Property 4.2 is satisfied while keeping at a minimum the window size W. We note that W depends of the considered FL optimization scheme and clients hardware capabilities. Based on the results of Theorem 4.3, in the following section, we introduce FEDFIX, a novel asynchronous FL setting based on a waiting policy over fixed time windows Δt^n.

Finally, the following example illustrates a practical application of the condition $\tilde{q}_i = p_i$.

Example 4.2. *We consider two clients, $i = 1, 2$, with $\mathcal{L}_i(\boldsymbol{\theta}) = \frac{1}{2}\|\boldsymbol{\theta} - \boldsymbol{\theta}_i^*\|^2$ where clients have identical importance, i.e. $p_1 = p_2 = 1/2$. Client 1 contributes at even optimization rounds and Client 2 at odd ones, i.e. $q_1^{2n} = q_1$, $q_2^{2n+1} = q_2$, and $q_1^{2n+1} = q_2^{2n} = 0$. Hence, we have*

$$\boldsymbol{\theta}^n \xrightarrow{n\to\infty} \frac{q_1\boldsymbol{\theta}_1^* + q_2\boldsymbol{\theta}_2^*}{q_1 + q_2}, \tag{4.30}$$

which converges to the optimum of problem (4.2) if and only if $\frac{1}{2}\left[\tilde{q}_i^{2n} + \tilde{q}_i^{2n+1}\right] = p_i$ (Theorem 4.3).

The conditions of Corollary 4.1 and Theorem 4.3 are equivalent when $W = 1$, where we retrieve $\epsilon_W = 0$. They are also equivalent when clients have the same data distributions, and we retrieve $\tilde{s}_j = r_j = 1$ at every optimization round, which also implies that $W = 1$.

The convergence guarantee proposed in Theorem 4.3 depends on the window size W, and to the maximum amount of optimizations needed for a client to update its work τ. We provide sufficient conditions in Corollary 4.2 for the parameters W, and τ, such that an optimization scheme satisfying Algorithm 4 converges to the optimum of the optimization problem (4.2).

Corollary 4.2. *Let us assume there exists $a \geq 0$ and $b \geq 0$ such that $W = \mathcal{O}(N^a)$, $\tau = \mathcal{O}(N^b)$, and $\eta_l \propto N^{-c}$. The convergence bound of Theorem 4.3 asymptotically converges to 0 if*

$$W = o(N), \tau = o(N), \text{ and } \max(a, b) < c < 1. \tag{4.31}$$

Proof. The bound of Theorem 4.3 converges to 0 if the following quantities also do: $\eta_l W$, $\frac{1}{\eta_l N}$, $\tau\eta_l$, η_l. We get the following conditions on a, b, and c: $-c + a < 0$, $c - 1 < 0$, $b - c < 0$, $-c < 0$, which completes the proof. □

By construction and definition of q_i, Property 4.2 is always satisfied with $W = N$. However, Corollary 4.2 shows that when $W = N$, no learning rate η_l can be chosen such that the learning process converges to $\boldsymbol{\theta}^*$. Also, Corollary 4.2 shows that Assumption 4.4 can be relaxed. Indeed, Assumption 4.4 implies that $\tau = \mathcal{O}(1)$ and Corollary 4.2 shows that $\tau = o(N)$ is sufficient. We show in Section 4.4 that all the considered optimization schemes satisfy $\tau = \mathcal{O}(1)$ and $W = \mathcal{O}(1)$, and also depend of the clients hardware capabilities and amount of participating clients M.

4.4 Applications

In this section, we show that the formalism of Section 4.2 can be applied to a wide-range of optimization schemes, demonstrating the validity of the conclusions of Corollary 4.1 and Theorem 4.3 (Section 4.3). When clients have identical data distributions, the sufficient conditions of Corollary 4.1 are always satisfied (Section 4.3). In the heterogeneous case, these conditions can also (theoretically) be satisfied. It suffices that every client has a non-null participation probability, i.e. $\mathbb{P}(T_i^n \leq \Delta t^n) > 0$ such that there exists an appropriate $d_i(n)$ satisfying $\tilde{q}_i(n) = p_i$. Yet, in practice clients generally may not even know their update time distribution $\mathbb{P}(T_i^n)$ making the computation of $d_i(n)$ intractable. In what follows, we thus focus on Theorem 4.3 to obtain the close-form of ϵ, which only requires from the server to know the clients time τ_i.

Theorem 4.3 provides a close-form for the convergence bound ϵ of an optimization scheme in function of the amount of server aggregation rounds N. We first introduce in Section 4.4.1 our considerations for the clients hardware and data to instead express ϵ in function of the training time T. The quantity ϵ also depends on the optimization scheme time policy Δt^n through α, β and τ, and on the clients data heterogeneity through $R(\{\mathcal{L}^n\})$ and W. We provide their close-form for synchronous FedAvg (Section 4.4.2), asynchronous FedAvg (Section 4.4.3), and FedFix (Section 4.4.4), a novel asynchronous optimization scheme motivated by Section 4.3.4. Finally, in Section 4.4.5, we show that the conclusions drawn for synchronous/asynchronous FedAvg and FedFix can also be extended to other distributed optimization schemes with delayed gradients. Of course, similar bounds can seamlessly be derived for centralized learning and client sampling, which we differ to Appendix C.3 to focus on asynchronous FL in this section.

4.4.1 Heterogeneity of clients hardware and data distributions

Clients importance. We restrict our investigation to the case where clients have identical aggregation weights during the learning process, i.e. $d_i(n) = d_i$. We also consider identical client importance $p_i = 1/M$. We can therefore define the averaged optimum residual Σ defined as the average of the clients SGD evaluated on the global optimum, i.e.

$$\Sigma \coloneqq \frac{1}{M} \sum_{i=1}^{M} \mathbb{E}_{\boldsymbol{\xi}_i} \left[\left\| \nabla \mathcal{L}_i(\boldsymbol{\theta}^*, \boldsymbol{\xi}_i) \right\|^2 \right]. \tag{4.32}$$

When clients have identical data distributions, Σ can be simplified as $\Sigma = \mathbb{E}_{\boldsymbol{\xi}} \left[\left\| \nabla \mathcal{L}(\boldsymbol{\theta}^*, \boldsymbol{\xi}) \right\|^2 \right]$, and $\Sigma = 0$ when clients perform GD. We note that in the DL and FL literature Σ is often

	Sync. FEDAVG	Async. FEDAVG	FEDFIX
d_i	$= p_i$	$= \left[\sum_{i=1}^{M} \frac{1}{\tau_i}\right] \tau_i p_i$	$= \lceil \tau_i / \Delta t \rceil p_i$
N	T/τ_M	$\sum_{i=1}^{M} T/\tau_i$	$T/\Delta t$
Δt	$= \max T_i^n$	$= \min T_i^n$	$= \Delta t$
α	1	0	1
β	0	$\max d_i \leq \tau_m/\tau_0$	0
τ	0	$\Omega(M)$, $\mathcal{O}(M\tau_M/\tau_0)$	0, $\lfloor \tau_m/\tau_0 \rfloor$
W	1	$\Omega(M)$, $\mathcal{O}(M(\tau_M)^M)$	1, $M\lceil \tau_m/\tau_0 \rceil^M$

Tab. 4.2.: The different variables used to account for the importance of clients or data distributions at every optimization round and during the full FL process. For τ and W, we give two values which correspond to their respective lower and upper bound.

simplified by assuming bounded variance of the stochastic gradients, i.e. $\Sigma \leq \sigma^2$, where σ^2 is the bounded variance of any client SG.

Clients computation time. In the rest of this chapter, we consider that clients guarantee reliable computation and communication, although with heterogeneous hardware capabilities, i.e. $\exists \tau_i \in \mathbb{R},$ s.t. $T_i = \tau_i$. Without loss of generality, we assume that clients are ordered by increasing τ_i, i.e. $\tau_i \leq \tau_{i+1}$, where the unit of τ_i is such that τ_i is an integer. In what follows, we provide the close form of d_i for all the considered optimization schemes. This derivation still holds for applications where clients have unreliable hardware capabilities that can be modeled as an exponential distribution, i.e. $T_i \sim \exp(\tau_i^{-1})$ which gives $\mathbb{E}[T_i] = \tau_i$.

Clients data distributions. Unless stated otherwise, we will consider the FL setting where each client has its unique data distribution. Therefore, clients have heterogeneous hardware and non-iid data distributions. The obtained results can be simplified for the DL setting where a dataset is made available to M processors. In this special case, clients have iid data distributions ($\mathcal{Z}_i = \mathcal{Z}_1$) , and identical computation times ($\tau_i = \tau_1$, $W = M$, and $\tau = M - 1$).

Learning rates. For sake of clarity, we ignore the server learning rate when expressing the convergence bounds ϵ, i.e. $\eta_g = 1$. Also, we consider a local learning rate η_l inversely proportional to the serial amount of SGD included in the global model, i.e. $\eta_l \propto 1/\sqrt{KN}$, consistently with the rest of the distributed optimization literature.

We propose Table 4.2 to summarize the close form or bounds of the different parameters used in Section 4.3.

4.4.2 FedAvg, Synchronous Federated Learning

As described for FedAvg in Section 4.2.3, at every optimization round, the server sends to the clients the current global model to perform K SGD steps on their own data before returning the resulting model to the server. Once every client performs its local work, the new global model is created as the weighted average of the clients contribution. The time required for an optimization step is therefore the one of the slowest client ($\Delta t^n = \max_i(T_i^n)$), and every client is considered ($\mathbb{P}(T_i^n \leq \Delta t^n) = 1$). Hence, $\alpha = 1$, $\beta = 0$, and setting $d_i = p_i$ is sufficient to satisfy the conditions of Corollary 4.1 (and thus the ones of Theorem 4.3) ensuring that FL converges to its optimum (Jianyu Wang, Q. Liu, et al., 2020). The term ϵ then reduces to

$$\epsilon_{\text{FedAvg}} = \frac{1}{\sqrt{KN}} \left\| \boldsymbol{\theta}^0 - \boldsymbol{\theta}^* \right\|^2 + \mathcal{O}\left(\frac{K-1}{N} \Sigma \right) + \mathcal{O}\left(\frac{1}{\sqrt{KN}} \frac{1}{M} \Sigma \right). \tag{4.33}$$

The second element of equation (4.33) accounts for the clients update disparity through their amount of local work K between two server aggregations, and is proportional to the SG variance Σ. The third element benefits of the distributed computation by being proportional to $1/M$. Equation (4.33) is consistent with literature on convex distributed optimization with FedAvg including Jianyu Wang, Q. Liu, et al. (2020) and Khaled et al. (2020a).

4.4.3 Asynchronous FedAvg

With FedAvg, every client waits for the slowest one to perform its local work, and cannot contribute to the learning process during this waiting time. To remove this bottleneck, with asynchronous FedAvg, the server creates a new global model whenever it receives a client contribution before sending it back to this client. For in depth discussion of Asynchronous FedAvg, please refer to C. Xu et al. (2021).

With asynchronous FedAvg, clients always compute their local work but each on a different global model, giving $\Delta t^n = \min_i T_i^n$, $\alpha = 0$, and $\beta = \max_i d_i$. In addition, while the slowest client updates its local work, other clients performs a fix amount of updates (up to $\lceil \tau_M / \tau_i \rceil$). By scaling this amount of updates by the amount of clients sending updates to the server, we have

$$\tau = \mathcal{O}\left(\frac{\tau_M}{\tau_0} (M-1) \right). \tag{4.34}$$

We define $lcm(\{x_i\})$ the function returning the least common multiplier of the set of integers $\{x_i\}$. Hence, after every $\nu \coloneqq lcm(\{\tau_i\})$ time, each client has performed ν/τ_i

optimization rounds and the cycle of clients update repeats itself. Thus, the smallest window W satisfies

$$W = \sum_{i=1}^{M} \nu/\tau_i. \tag{4.35}$$

By construction, $\nu \geq \tau_M$ and thus $W = \Omega(M)$, with $W = M$ when clients have homogeneous hardware ($\tau_M = \tau_0$). In the worse case, every τ_i is a prime number, and we have $\nu/\tau_i \leq (\tau_M)^{M-1}$, which gives $W = \mathcal{O}(M\,(\tau_M)^{M-1})$. In a cycle of W optimization rounds, every client participates ν/τ_i times to the creation of a new global model. Therefore, we have $q_i(n) = d_i$ for the ν/τ_i participation of client i, and $q_i(n) = 0$ otherwise. Hence, the sufficient conditions of Theorem 4.3 are satisfied when

$$q_i = \frac{1}{W} \sum_{n=kW}^{(k+1)W-1} q_i(n) = \frac{1}{\sum_{i=1}^{M} \nu/\tau_i} \frac{\nu}{\tau_i} d_i = p_i \Rightarrow d_i = \left[\sum_{i=1}^{M} \frac{1}{\tau_i} \right] \tau_i p_i. \tag{4.36}$$

The client weight calculated in equation (4.36) is constant and only depends on the client importance p_i (set and thus known by the server), and on the clients computation time τ_i (eventually estimated by the server after some clients updates). The condition on d_i can be further simplified by accounting for the server learning rate η_g. Coupling equation (4.6) with equation (4.36) gives $\eta_g d_i \propto \tau_i p_i$, which is sufficient to guarantee the convergence of asynchronous FL to its optimum. Finally, by bounding τ_i, we also have $\beta = \max_i d_i \leq \tau_M/\tau_0$, bounded the hardware computation time heterogeneity.

The disparity between the optimized objectives $R(\{\mathcal{L}^n\})$ at different optimization rounds also slows down the learning process. Indeed, at every optimization round, only a single client can participate with probability 1. As such, we have $\mathcal{L}^n = d_i \mathcal{L}_i$ which, thanks to the assumption $p_i = 1/M$, yields

$$R(\{\mathcal{L}^n\}) = \frac{1}{M} \sum_{i=1}^{M} \left[\mathcal{L}_i(\boldsymbol{\theta}^*) - \mathcal{L}_i(\boldsymbol{\theta}_i^*) \right]. \tag{4.37}$$

Finally, we simplify the close-form of ϵ (Theorem 4.3) for asynchronous FEDAVG to get

$$\begin{aligned} \epsilon_{Async} = & \frac{1}{\sqrt{KN}} \left\| \boldsymbol{\theta}^0 - \boldsymbol{\theta}^* \right\|^2 + \mathcal{O}\left(\frac{K-1}{N} \Sigma \right) + \mathcal{O}\left(\frac{\tau_M}{\tau_0} \frac{1}{\sqrt{KN}} \left[R(\{\mathcal{L}^n\}) + \Sigma \right] \right) \\ & + \mathcal{O}\left(\left(\frac{\tau_M}{\tau_0} \right)^3 \frac{K}{N} M^2 \left[R(\{\mathcal{L}^n\}) + \Sigma \right] \right) + \mathcal{O}\left(\frac{1}{\sqrt{KN}} (W-1) \right). \end{aligned} \tag{4.38}$$

With equation (4.38), we can compare synchronous and asynchronous FEDAVG. The first and second asymptotic terms are identical for the two learning algorithms, while the third asymptotic term is scaled by the hardware characteristics τ_M/τ_0 instead of $1/M$ in FEDAVG, with the addition of a non null residual $R(\{\mathcal{L}^n\})$ for asynchronous FEDAVG. However, the fourth and fifth term are unique to asynchronous FEDAVG, and explains why

its convergence gets more challenging as the amount of clients M increases. The impact of the hardware heterogeneity is also identified through the importance of τ_M/τ_0 in the third term. With no surprise, for a given optimization round, synchronous FEDAVG outperforms its asynchronous counterpart. However, in T time, the server performs

$$N = \sum_{i=1}^{M} T/\tau_i \tag{4.39}$$

aggregations with asynchronous FEDAVG against T/τ_M for synchronous FEDAVG. With asynchronous FEDAVG, the server thus performs at least M times more aggregations than with synchronous FEDAVG. As a result, the first two terms of equation (4.38), which are proportional to how good the initial model is $\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}^*\|$, decrease faster with asynchronous FEDAVG at the detriment of an higher convergence residual coming for the two last terms.

Comparison with asynchronous DL and FEDAVG literature. The convergence rates obtained in the convex distributed optimization literature relies on additional assumptions to ours, with which we retrieve their proposed convergence rate. To the best of our knowledge, only Zinkevich et al. (2009) considers non-iid data distributions for the clients. When assuming $W = \mathcal{O}(\tau)$ and $\eta_l \propto 1/\sqrt{\tau N}$, we retrieve a convergence rate $\sqrt{\tau/N}$.

We also match convergence rates for literature with iid client data distributions and $K = 1$. With $M = \mathcal{O}(\sqrt{N})$, then we have $\mathcal{O}(1/\sqrt{N})$ (Agarwal and Duchi, 2011; Lian, Huang, et al., 2015). When $\eta_l = \mathcal{O}(1/\tau\sqrt{KN})$, we retrieve $\tau/N + 1/\sqrt{N}$ (Sebastian U Stich and Karimireddy, 2020; S. Stich et al., 2021).

4.4.4 FEDFIX

The analysis of asynchronous FEDAVG (Section 4.4.3) and its comparison with synchronous FEDAVG (Section 4.4.2), shows that asynchronous FEDAVG is not scalable to large cohort of clients. We thus propose FEDFIX combining the strong points of synchronous and asynchronous FEDAVG, where the server creates the new global model at a fixed time t^n with the contributions received since t^{n-1}. Therefore, the server does not wait for every client, contrarily to synchronous FEDAVG, and considers more than one client per aggregation to have more stable aggregations, contrarily to asynchronous FEDAVG. We provide in Figure 4.1 an illustration of FEDFIX with two clients.

With FEDFIX, an iteration time $\Delta t^n = t^{n+1} - t^n$ is decided by the server and is independent from the clients remaining update time T_i^n. For sake of convenience, we further assume that the time between optimization rounds is identical, i.e. $\Delta t^n = \Delta t$, but the following results

can be derived for other fixed time policies $\{\Delta t^n\}$. Therefore, T_i^n and T_j^n are independent, and so are ω_i and ω_j, which gives $\alpha = 1$ and $\beta = 0$.

Every client sends an update to the server in $N_i' = \lceil T_i/\Delta t \rceil$ optimization steps. Contrarily to asynchronous FEDAVG, we thus have $\tau = \lceil \tau_m/\Delta t \rceil = \mathcal{O}(1)$, which is independent from the amount of participating clients M. In this case, the smallest window W satisfies $W = lcm(\{N_i'\})$, and clients update W/N_i' times their work to the server during the window W. Therefore, satisfying the conditions of Theorem 4.3 requires

$$d_i = \lceil \frac{\tau_i}{\Delta t} \rceil p_i. \tag{4.40}$$

With equation (4.40), we can notice the relationship between FEDFIX and synchronous or asynchronous FEDAVG. When $\Delta t \geq \tau_i$, client i participates to every optimization round and is thus considered synchronously, which gives $d_i = p_i$. When $\Delta_t \geq \tau_M$, we retrieve synchronous FL and $d_i = p_i$ for every client. On the contrary, for asynchronous FL, when $\Delta t \ll \tau_i$, we obtain $\lceil \tau_i/\Delta t \rceil \approx \tau_i/\Delta t$ and we retrieve $\eta_g d_i = \eta_g \lceil \tau_i/\Delta t \rceil p_i \propto \tau_i p_i$.

Regarding the disparity between the local objectives $R\{\mathcal{L}^n\}$, we know that a client participates to an optimization round with $q_i(n) = d_i$. We thus have $\mathcal{L}^n = \sum_{i \in S_n} d_i \mathcal{L}_i$, where S_n is the set of the participating clients at optimization step n. Considering that $\mathcal{L}^n(\bar{\boldsymbol{\theta}}^n) \geq \sum_{i \in S_n} d_i \mathcal{L}_i(\boldsymbol{\theta}_i^*)$, the close form of FEDFIX is bounded by the one of of asynchronous FEDAVG, i.e.

$$R(\{\mathcal{L}^n\}) \leq \frac{1}{M} \sum_{i=1}^{M} \left[\mathcal{L}_i(\boldsymbol{\theta}^*) - \mathcal{L}_i(\boldsymbol{\theta}_i^*) \right]. \tag{4.41}$$

Finally, we simplify the close-form of ϵ (Theorem 4.3) for FEDFIX to get

$$\begin{aligned} \epsilon_{\text{FEDFIX}} = & \frac{1}{\sqrt{KN}} \mathbb{E}\left[\left\| \boldsymbol{\theta}^0 - \boldsymbol{x} \right\|^2 \right] + \mathcal{O}\left(\frac{K-1}{N} \left[R(\{\mathcal{L}^n\}) + \Sigma \right] \right) \\ & + \mathcal{O}\left(\left[\frac{1}{\sqrt{KN}} + \frac{K}{N} \lceil \frac{\tau_m}{\Delta t} \rceil^2 \right] \left[R(\{\mathcal{L}^n\}) + \lceil \frac{\tau_m}{\Delta t} \rceil \frac{1}{M} \Sigma \right] \right) + \mathcal{O}\left(\frac{1}{\sqrt{KN}} (W-1) \right). \end{aligned} \tag{4.42}$$

The first two elements of equation (4.42) are identical for FEDFIX, synchronous and asynchronous FEDAVG. However, thanks to lower values for the different variables (cf Table 4.2), the last two asymptotic terms of the convergence bound are smaller for FEDFIX than for asynchronous FEDAVG, equation (4.42). Similarly, these two terms are larger with FEDFIX than with synchronous FEDAVG. The hardware heterogeneity and the amount of participating clients still impacts the convergence bound through $\lceil \tau_M/\Delta t \rceil$ and W, but can be mitigated with proper selection of Δt. Therefore, after N optimization rounds, synchronous FEDAVG outperforms FEDFIX which outperforms in turn asynchronous FEDAVG.

However, in T time, the server performs $N = T/\Delta t$ aggregations with FEDFIX against T/τ_M for synchronous FEDAVG. With asynchronous FEDAVG, the server thus performs at least $\tau_M/\Delta t$ times more aggregations than with synchronous FEDAVG. Overall, Δt can be considered as the level of asynchronicity given to Algorithm 4, with FEDAVG when $\Delta t = \tau_M$ and asynchronous FEDAVG when $\Delta t \geq \tau_M$.

In the DL case, clients have identical computation time ($\tau_1 = \tau_m$), and we retrieve the convergence bound of synchronous FEDAVG.

In addition, we can increase the waiting time for the clients update, since the learning process converges and gets closer to the optimum of optimization problem (4.2), to reach a behavior similar to the one of synchronous FL. Indeed, for Theorem 4.3 to hold, we only need the same optimization time rounds Δt over a window W

4.4.5 Generalization

Coupled with the theoretical method developed in Jianyu Wang, Q. Liu, et al., 2020, the proof of Theorem 4.1 can account for FL regularization methods (T. Li, Sahu, Zaheer, Sanjabi, Talwalkar, and Smith, 2018a; T. Li, Sahu, Zaheer, Sanjabi, Talwalkar, and Smithy, 2019; Acar et al., 2021), other SGD solvers (Kingma and Ba, 2014; R. Ward et al., 2019; Xiaoyu Li and Orabona, 2019; Yu, Jin, et al., 2019; Yu, S. Yang, et al., 2019; Haddadpour, Kamani, Mahdavi, and V. Cadambe, 2019), and/or gradient compression/quantization (Reisizadeh et al., 2020; Basu et al., 2019; H. Wang et al., 2018; Koloskova* et al., 2020).

We also note that Theorem 4.3 can be applied to other distributed optimization schemes using different waiting time policy Δt^n. With FEDBUFF (J. Nguyen et al., 2021), the server waits for m client updates to create the new global model. The server then communicates to these clients the new global model, while the other clients keep performing local work on the global model they received.

In this section, the sufficient conditions of Theorem 4.3 regarding the expected aggregation weights $q_i(n)$ were applied to obtain proper aggregation weight d_i. We keep identical clients local learning rate η_l and amount of local work K. We could instead get the close-form of a client specific learning rate $\eta_l(i)$ or amount of local work $K(i)$ using the gradient formalization of Jianyu Wang, Q. Liu, et al. (2020). Specifically, our conclusions can also be applied to federated optimization schemes where clients perform the same amount of SGD steps on their data over the whole optimization process while asynchronously aggregating the clients' delayed updates (Lian, W. Zhang, et al., 2018; Avdiukhin and Kasiviswanathan, 2021). Finally, with minor modifications to the aggregation scheme (4.6), our convergence guarantees can also be extended to federated optimization schemes where the server balances

the clients' hardware heterogeneity by using every client latest contribution during each aggregation step (Gu et al., 2021; H. Yang et al., 2022).

4.5 Experiments

In this section, we experimentally demonstrate the theoretical claims of Section 4.3 and 4.4. We first introduce the information needed to understand how the experiments are run in Section 4.5.1. Finally, in Section 4.5.2, we provide our experiments and their interpretation.

4.5.1 Experimental Setting

We introduce in this subsection the dataset and the predictive models used for federated optimization, the hardware scenarios proposed to simulate hardware heterogeneity, the clients aggregation weights strategies, and how the different hyperparameters are set[1].

Optimization Problems. We consider learning a predictive model for optimization problem (4.2) where clients have identical importance ($p_i = 1/M$) based on the following datasets with their associated learning scenarios.

- **MNIST iid** (Lecun et al., 1998) and **MNIST non-iid**. MNIST is a dataset of 28x28 pixel grayscale images of handwritten single digits between 0 and 9 composed of 60 000 training and 10 000 testing samples split equally among the clients. We use a logistic regression to predict the images class. Clients are randomly allocated digits to match their number of samples. With MNIST non-iid, we split instead data samples among clients using a Dirichlet distribution of parameter 0.1, i.e. $Dir(0.1)$.Therefore, with MNIST iid and non-iid, we evaluate our theory on a convex optimization problem.

- **CIFAR10/100** (Krizhevsky et al., n.d.). The dataset consists of 10/100 classes of 32x32 images with three RGB channels. There are 50000 training and 10000 testing examples. The model architecture was taken from (McMahan, Moore, et al., 2017) which consists of two convolutional layers and a linear transformation layer to produce logits. Clients get the same amount of samples but their percentage for each class vary and is determined with a Dirichlet distribution of parameter 0.1, i.e. $Dir(0.1)$ (Harry Hsu et al., 2019).

[1] Code and data are available at `https://github.com/Accenture/Labs-Federated-Learning/tree/asynchronous_FL`

- **CIFAR*10/100**. Clients get the same samples as with CIFAR10/100. However, with CIFAR*10/100, we use a logistic regression to predict the image class to evaluate our theory on a convex optimization problem as for MNIST iid and non-iid.

- **Shakespeare** (Caldas et al., 2018). We study a LSTM model for next character prediction on the dataset of *The Complete Works of William Shakespeare*. The model architecture is composed of a two-layer LSTM classifier containing 100 hidden units with an 8 dimensional embedding layer taken from (T. Li, Sahu, Zaheer, Sanjabi, Talwalkar, and Smith, 2018a). The model takes as an input a sequence of 80 characters, embeds each of the characters into a learned 8-dimensional space and outputs one character per training sample after 2 LSTM layers and a fully connected one.

Hardware Scenarios. In the following experimental scenarios, clients computation time are obtained according to the time policy FX. We consider that clients have fixed update times that can be up to X% faster than the slowest client. Clients computation time are uniformly distributed from the lower to the upper bound set at 1 unit of time. Clients have thus identical hardware with $F0$. To simulate heterogeneous clients hardware, we consider the time scenario $F80$.

Clients Aggregation Weights. To compare asynchronous FL with and without the close-form of d_i provided in Section 4.4, we introduce IDENTICAL where $d_i = 1$ for every client regardless of the time scenario FX, and TIME-BASED where d_i satisfies equation (4.36) derived in Section 4.4.

Hyperparameters. Unless specified otherwise, we consider a global learning rate $\eta_g = 1$. We finetune the local learning rate η_l with values ranging from 10^{-5} to 1. We consider a batch size $B = 64$ for every dataset. We report mean and standard deviation on 5 random seeds. Every comparison of IDENTICAL with TIME-BASED is done using the same local learning rate. We give an advantage to IDENTICAL by finetuning the learning rate on this clients aggregation weight scenario.

4.5.2 Experimental Results

We experimentally show that asynchronous FL has better performances with TIME-BASED than with IDENTICAL, and thus we demonstrate the correctness of Theorem 4.3 with Figure 4.2 in Section 4.5.2. Finally, we compare synchronous FEDAVG and asynchronous FEDAVG in Figure 4.3.

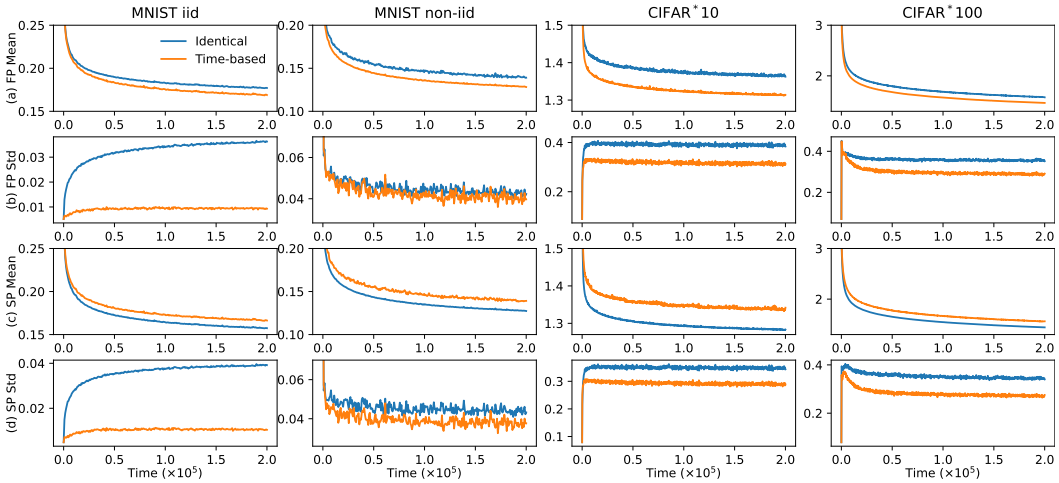

Fig. 4.2.: We consider the loss evolution over time of federated problem (4.2) (FP) and surrogate problem (4.10) (SP) for MNIST iid, MNIST non-iid, CIFAR*10, and CIFAR*100; and the respective standard deviation of the loss over clients in (b) and (d). We consider $M = 10$ for a time scenario $F80$ with $K = 1$.

Impact of the Clients Aggregation Weights on Asynchronous FedAvg

Figure 4.2(a) experimentally shows the interest of coupling asynchronous FL with TIME-BASED instead of IDENTICAL for different convex applications (MNIST iid, MNIST non-iid, CIFAR*10, and CIFAR*100). The learnt model with TIME-BASED has better minima on the federated problem (4.2). In addition, Figure 4.2(b) shows that losses across clients are more homogeneous with TIME-BASED, resulting in generally lower standard deviations.

Focusing on MNIST iid and non-iid, we see the impact of data heterogeneity on the learnt model performances. With IDENTICAL, asynchronous FL converges to a suboptimum point and the differences between the learnt model losses is twice as large for MNIST non-iid than for MNIST iid, Figure 4.2(a). Figure 4.2(b) shows a similar result concerning the clients loss heterogeneity. Therefore, data heterogeneity degrades the suboptimum loss and cannot be ignored in asynchronous FL applications. Indeed, IDENTICAL and TIME-BASED curves are significantly different even for the simplest application on MNIST iid, where the dataset is uniformly distributed across $M = 10$ clients. Hence, the assumption of identical data distributions should generally not be made and the aggregation scheme TIME-BASED should be used instead for any asynchronous FL (or DL).

With Figure 4.2(c), we can also appreciate the performances of the learning procedure on the surrogate problem (4.10) based on the clients computation times T_i. Due to clients hardware heterogeneity, in the scenario $F80$, clients communicate with the server up to 5 more times than the slowest one. TIME-BASED balances this amount of updates disparity

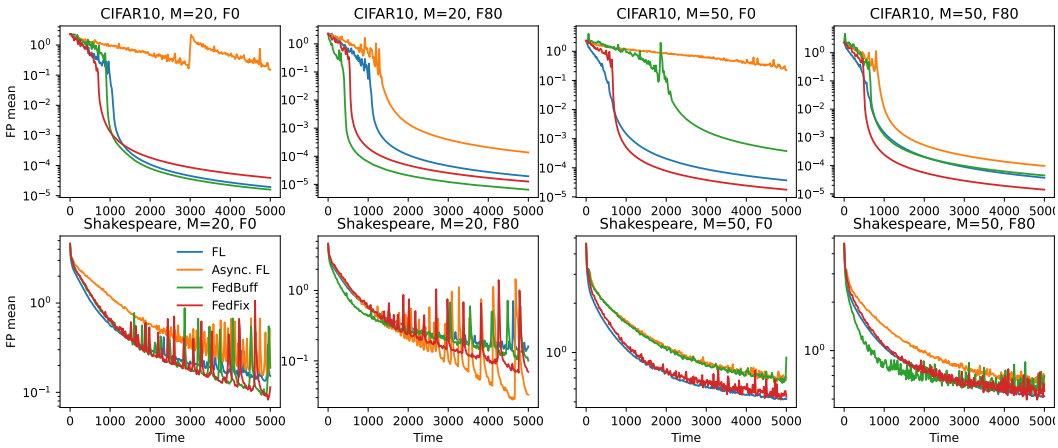

Fig. 4.3.: Evolution of federated problem (4.2) (FP) loss for CIFAR10 and Shakespeare and time scenario $F0$ and $F80$, with $M = 20$ and $M = 50$. We consider $\eta_g = 1$ and $K = 10$. The server creates the new global model after $\Delta t = 0.5$ for FEDFIX and after receiving $c = 10$ delayed contributions with FEDBUFF.

across clients. As a result, IDENTICAL has better performances than TIME-BASED on the surrogate problem (4.10) for MNIST iid/non-iid and CIFAR$^*10/100$.

Partial Asynchronicity with FEDFIX

The theory derived in Section 4.3 can be applied to asynchronous FL but also synchronous FL, FEDAVG, and other asynchronous FL schemes like FEDFIX (Section 4.4) and FEDBUFF (J. Nguyen et al., 2021). We show with Figure 4.3 that allowing asynchronicity does not necessarily provide faster learning processes, e.g. comparison between synchronous and asynchronous FEDAVG above, but FEDFIX outperforms FEDAVG by balancing convergence speed and stability.

With a small enough learning rate η_l, asynchronous FEDAVG outperforms FEDFIX and FEDBUFF, which outperforms synchronous FL (see Figure C.1 to C.4 in Appendix C.4). Indeed, in this case, global models change slowly and we can consider that the server receives contributions with no gradient delay. As such, the learning procedure including the most serial contributions in the global model is the fastest. However, in the other cases, the learning rate η_l does not mitigate the discrepancy between clients update, which slows down convergence for asynchronous FL, and can even prevent it.

Identifying the fastest optimization scheme must be done by comparing optimization schemes based on their best local learning rate η_l (Figure 4.3). Synchronous FL outperforms asynchronous FL when clients have homogeneous ($F0$) or heterogeneous ($F80$) hardware. Indeed, the server needs to increase the amount of contributions at each aggregation to balance convergence speed and convergence stability. We see that FEDFIX-0.5 provides

this trade-off and outperforms synchronous FL by performing twice as many server aggregations in the heterogeneous hardware scenario ($F80$). We also see that FEDBUFF always outperforms asynchronous FL by considering more clients at every aggregation without necessarily outperforming synchronous FL. Hence, FEDFIX better balances convergence speed and stability than FEDBUFF.

We note that, even for synchronous FL, FL convergence is not monotonous. Indeed, for synchronous FL to have a better convergence speed than asynchronous FL, the server needs to consider a high local learning rate leading to convergence instability. Figure 4.3 shows this instability for Shakespeare and $t > 2500$, and Figure C.1 to C.4 in Appendix C.4 provides the evolution of this instability as the learning rate η_l increases.

We note that even when clients have homogeneous hardware ($F0$), FEDFIX can outperform synchronous FL. This can be explained by the close-form of FEDFIX weights d_i, equation (4.40), which accounts for server aggregations where no client participates. As a result, FEDFIX behaves as asynchronous FL but with an higher server learning rate $\eta_g = 2$ which provides faster convergence.

4.6 Discussion

This chapter introduces equation (4.6) which generalizes the expression of FEDAVG aggregation scheme by introducing stochastic aggregation weights $\omega_i(n)$ to account for asynchronous client updates. We prove the convergence of FL schemes satisfying equation (4.6) with Theorem 4.1. A similar aggregation scheme has been derived in Fraboni, Vidal, Kameni, et al. (2022a) for unbiased client sampling, which this chapter generalizes. In addition, we show that aggregation scheme (4.6) is satisfied by asynchronous FL, FEDFIX, and FEDBUFF, Section 4.4. Finally, we assume fixed clients update time T_i such that we can consider $d_i(n) = d_i$, and give in Section 4.4 its close-form to ensure any FL optimization scheme converges to the optimum of problem (4.2). Thsi chapter remains relevant for applications with $d_i(n) = d_i$ but we let the specific derivations to the reader.

This chapter shows theoretically and experimentally that asynchronous FEDAVG does not always outperform its synchronous counterpart. By creating the new global model with the contribution of only one client, asynchronous FEDAVG convergence speed is very sensitive to the choice of learning rate and amount of local work K. These two hyperparameters need to be fine-tuned to properly balance convergence speed and stability. Due to the hardware constraints inherent to the FL setting, fine-tuning is a challenging step for FL and is not necessarily feasible. Therefore, we proposed FEDFIX, an FL algorithm where the server, after a fixed amount of time, creates the new global model with the contribution

of all the participating clients. We prove the convergence of FEDFIX with our theoretical framework, and experimentally demonstrate its improvement over FEDAVG in all the considered scenarios.

Part II

Reliability of Federated Learning in Practical Applications

5 Sequential Informed Federated Unlearning: Efficient and Provable Client Unlearning in Federated Optimization

Contents

5.1 Introduction **78**
5.2 Background and Related Work **80**
5.2.1 Machine Unlearning 80
5.2.2 Federated Optimization and FEDAVG 81
5.2.3 Federated Unlearning 82
5.3 Unlearning a FL client with IFU **83**
5.3.1 Theorem 5.1, Bounding the Model Sensitivity 83
5.3.2 Improving the Tightness of the Sensitivity Bound 84
5.3.3 Satisfying Definition 5.1 85
5.3.4 Informed Federated Unlearning (IFU) 86
5.4 Sequential FU with SIFU **87**
5.5 Experiments **90**
5.5.1 Experimental Setup 90
5.5.2 Experimental Results 91
5.5.3 Verifying Unlearning through Watermarking 92
5.5.4 Impact of the noise perturbation on SIFU 93
5.6 Conclusions **93**

In this chapter, we present Informed Federated Unlearning (IFU), a novel efficient and quantifiable approach to unlearn the data specificities of a client from a mode trained with federated learning. Upon receiving an unlearning request from a given client, IFU identifies the optimal FL iteration from which FL has to be reinitialized, with unlearning guarantees obtained through a randomized perturbation mechanism. The theory of IFU is also extended

to account for sequential unlearning requests. Experimental results on different tasks and dataset show that IFU leads to more efficient unlearning procedures as compared to basic re-training and state-of-the-art federated unlearning approaches. This chapter is under review and currently available as a preprint (Fraboni, Vidal, Kameni, et al., 2022c)

5.1 Introduction

With the emergence of new data regulations, such as the EU General Data Protection Regulation (GDPR) (Voigt and Von dem Bussche, 2017) and the California Consumer Privacy Act (CCPA) (Harding et al., 2019), the storage and processing of sensitive personal data is often subject of strict constraints and restrictions. In particular, the "right to be forgotten" states that personal data must be erased upon request from the concerned individuals, with subsequent potential implications on machine learning models trained by using this data. Machine Unlearning (MU) is an emerging discipline that studies methods to ideally remove the contribution of a given data instance used to train a machine learning model. Current MU approaches are essentially based on routines that modify the model weights in order to guarantee the "forgetting" of a given data point, i.e. to obtain a model equivalent to an hypothetical one trained without this data point (Cao and J. Yang, 2015; Bourtoule et al., 2021).

Motivated by data governance and confidentiality concerns, Federated learning (FL) has gained popularity in the last years to allow data owners to collaboratively learn a model without sharing their respective data. Among the different FL approaches, federated averaging (FedAvg) has emerged as the most popular optimization scheme (McMahan, Moore, et al., 2017). An optimization round of FedAvg requires data owners, also called clients, to receive from the server the current global model which they update before sending it back to the server. The new global model is then created as the weighted average of the client updates, according to their data ratio. FL communication design guarantees to clients that their data is solely used to compute their model update, while theoretical work guarantees FL convergence to a stationary point of the clients' joint optimization problem (Jianyu Wang, Q. Liu, et al., 2020; Xiang Li et al., 2020).

With the current deployments of FL in the real-world, it is of crucial importance to extend MU to guarantee the unlearning of clients wishing to opt-out from a collaborative training routine. This is not straightforward, since current MU schemes have been proposed essentially in the centralized learning setting, and cannot be seamlessly applied to the federated one. For example, several MU methods require the estimation of the Hessian of the loss function (Guo et al., 2020; Izzo et al., 2021; Golatkar, Achille, and Soatto, 2020a; Golatkar,

Achille, and Soatto, 2020b; Golatkar, Achille, Ravichandran, et al., 2021), an operation which is notoriously computationally heavy and intractable for high dimensional models. Moreover, sharing the Hessian would require clients to share with the server additional information about their data, thus exposing the federated setting to information leakage and attacks, for example under the form of model inversion (Matt Fredrikson et al., 2015). Alternative MU methods draw from the concept of differential privacy Dwork and Roth (2014) and are based on a Gaussian noise perturbation of the trained model (Neel et al., 2021; Guo et al., 2020; Gupta et al., 2021). The magnitude of the noise perturbation should be estimated directly from the clients data, which is by construction inaccessible to the server in the FL regime. We also note that while recent federated unlearning (FU) methods have been proposed to unlearn a client from the global FL model (G. Liu et al., 2021; Junxiao Wang et al., 2021; Halimi et al., 2022; C. Wu et al., 2022), these approaches do not come with theoretical guarantees on the effectiveness of the unlearning.

The main contribution of this chapter consists in Informed Federated Unlearning (IFU), a novel efficient FU approach to unlearn a client's contribution with quantifiable unlearning guarantees. IFU requires minimal additional computations to the server in a standard FedAvg procedure. Specifically, the server quantifies at every optimization round each client's contribution to the global model. Upon receiving an unlearning request from a client, the server identifies in the FL training history the optimal FL iteration and associated intermediate global model from which re-initializing the unlearning procedure. Unlearning guarantees are provided by introducing a novel randomized mechanism to perturb the selected intermediate model with client-specific noise. We also extend IFU to Sequential Informed Federated Unlearning (SIFU), to account for realistic unlearning scenarios where the server receives sequential unlearning requests from one or more clients at different times (Neel et al., 2021; Gupta et al., 2021).

This manuscript is structured as follows. In Section 5.2, we provide formal definitions for MU, FL, and FU, and introduce the randomized mechanism with associated unlearning guarantees. In Section 5.3, we introduce sufficient conditions for IFU to unlearn a client from the FL routine (Theorem 5.2). In Section 5.4, we extend IFU to the sequential unlearning setting with Sequential IFU (SIFU). Finally, in Section 5.5, we experimentally demonstrate on different tasks and datasets that SIFU leads to more efficient unlearning procedures as compared to basic re-training and state-of-the-art FU approaches.

5.2 Background and Related Work

In Section 5.2.1, we introduce the state-of-the art behind Machine Unlearning, while in Section 5.2.2, we introduce FL and FEDAVG. Finally, we introduce Federated Unlearning (FU) in Section 5.2.3.

5.2.1 Machine Unlearning

Let us consider a dataset $\mathcal{D}$ composed of two disjoint datasets: $\mathcal{D}_f$, the cohort of data samples on which unlearning must be applied after FL training, and $\mathcal{D}_k$, the remaining data samples. Hence, we have $\mathcal{D} = \mathcal{D}_f \cup \mathcal{D}_k$. We also consider $\mathcal{M}(\mathcal{D})$, the ML model parameters resulting from training with optimization scheme $\mathcal{M}$ on dataset $\mathcal{D}$. We introduce in this section the different unlearning baselines and methods currently used to unlearn $\mathcal{D}_f$ from the trained model $\mathcal{M}(\mathcal{D})$.

MU through retraining. Within this setting, a new training is performed from scratch with only $\mathcal{D}_k$ as training data. As the initial model contains no information from $\mathcal{D}_f$, the new trained model $\mathcal{M}(\mathcal{D}_k)$ also contains no information from $\mathcal{D}_f$. We note however that this procedure wastes the contribution of $\mathcal{D}_k$ already available by training originally on $\mathcal{D}$. Hence, this method is considered sub-optimal, and represents a basic baseline for unlearning approaches.

MU through fine-tuning. Fine-tuning on the remaining data $\mathcal{D}_k$ has been proposed as a practical approach to unlearn the specificities of $\mathcal{D}_f$. However, fine-tuning does not provide guarantees about the effectiveness of the unlearning. We provide an example of this issue in Appendix D.1.

MU through model scrubbing. Another unlearning approach consists in applying a "scrubbing" transformation h to the model $\mathcal{M}(\mathcal{D})$ such that the resulting model is as close as possible to the one that would be trained with only $\mathcal{D}_k$, i.e. $h(\mathcal{M}(\mathcal{D})) \approx \mathcal{M}(\mathcal{D}_k)$ (Ginart et al., 2019). To define a scrubbing method h, existing work mostly relies on the following Assumption 5.1, which considers a quadratic approximation of the loss function.

Assumption 5.1. *For model parameters $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$, the gradient of the loss function of a given data point $\mathcal{D}_x$ satisfies*

$$\nabla f_{\mathcal{D}_x}(\boldsymbol{\phi}) = \nabla f_{\mathcal{D}_x}(\boldsymbol{\theta}) + H_{\mathcal{D}_x}(\boldsymbol{\theta})(\boldsymbol{\phi} - \boldsymbol{\theta}), \tag{5.1}$$

where $H_{\mathcal{D}_x}(\boldsymbol{\theta})$ is positive semi-definite.

The scrubbed model is the new optimum obtained when unlearning data samples in $\mathcal{D}_f$. Hence, under Assumption 5.1, the new optimum can be obtained by setting $\nabla f_{\mathcal{D}_k}(h_{\mathcal{D}_k}(\boldsymbol{\theta})) = 0$, which gives

$$h_{\mathcal{D}_k}(\boldsymbol{\theta}) = \boldsymbol{\theta} - H^{-1}_{\mathcal{D}_k}(\boldsymbol{\theta}) \nabla f_{\mathcal{D}_k}(\boldsymbol{\theta}). \tag{5.2}$$

With equation (5.2), h reduces to performing a Newton step, and has been derived in previous MU works (Guo et al., 2020; Izzo et al., 2021; Golatkar, Achille, and Soatto, 2020a; Golatkar, Achille, and Soatto, 2020b; Golatkar, Achille, Ravichandran, et al., 2021; Mahadevan and Mathioudakis, 2021a) under different theoretical assumptions that can be generalized with Assumption 5.1. The main drawback behind the use of the scrubbing function (5.2) is the computation of the Hessian, which can be unfeasible for high dimensional model. Finally, the scrubbing function (5.2) is often coupled with Gaussian noise perturbation on the resulting weights (Golatkar, Achille, and Soatto, 2020a; Golatkar, Achille, and Soatto, 2020b; Golatkar, Achille, Ravichandran, et al., 2021), to compensate the quadratic approximation of the loss function or the approximation of the Hessian.

MU through noise perturbation. This unlearning method consists in randomly perturbing the trained model $\mathcal{M}(\mathcal{D})$ to unlearn specificities from data samples in $\mathcal{D}_f$ (Neel et al., 2021; Gupta et al., 2021; Mahadevan and Mathioudakis, 2021b). The noise is set such that the guarantees of Definition 5.1 are satisfied, where (ϵ, δ) are parameters quantifying the unlearning guarantees.

Definition 5.1. *Let f_m be a randomized mechanism taking model parameters as an input. (ϵ, δ)-Unlearning trough f_m of a data point $\{x_m, y_m\}$ from a model $\mathcal{M}(D)$ is achieved if, for any subset $\mathcal{S}$ of the model parameters space and $D_{-m} \coloneqq D \setminus \{x_m, y_m\}$, we have*

$$\mathbb{P}(f_m(\mathcal{M}(D)) \in \mathcal{S}) \leq e^{\epsilon} \mathbb{P}(f_m(\mathcal{M}(D_{-m})) \in \mathcal{S}) + \delta \tag{5.3}$$

$$\textit{and } \mathbb{P}(f_m(\mathcal{M}(D_{-m})) \in \mathcal{S}) \leq e^{\epsilon} \mathbb{P}(f_m(\mathcal{M}(D)) \in \mathcal{S}) + \delta. \tag{5.4}$$

Guo et al., 2020 shows the relationship between Definition 5.1 and the definition a randomized mechanism in differential privacy (Dwork and Roth, 2014; X. Chen et al., 2020).

5.2.2 Federated Optimization and FedAvg

In FL, we consider a learning setup with M clients, and define $I = \{1, ..., M\}$ as the set of indices of the participating clients. Each client owns a dataset D_i composed of $|D_i| = n_i$ data samples. We consider a loss $f(\boldsymbol{x}_{i,l}, \boldsymbol{y}_{i,l}, \boldsymbol{\theta})$ assessed on each data sample

Algorithm 5 FEDAVG(I, N)

1: **for** n from 0 to $N-1$ **do**
2: The server sends $\boldsymbol{\theta}^n$ to every client in I.
3: Clients perform K SGDs to compute $\boldsymbol{\theta}_i^{n+1}$.
4: The server creates $\boldsymbol{\theta}^{n+1}$, equation (5.6).
5: **end for**
6: **return** the trained global model $\boldsymbol{\theta}^N$

$(\boldsymbol{x}_{i,l}, \boldsymbol{y}_{i,l}) \in D_i$, and define a client's loss function as $f_i(\boldsymbol{\theta}) \coloneqq 1/n_i \sum_{l=1}^{n_i} f(\boldsymbol{x}_{i,l}, \boldsymbol{y}_{i,l}, \boldsymbol{\theta})$. We define for the joint dataset $D_I \coloneqq \cup_{i \in I} D_i$ the federated loss function

$$f_I(\boldsymbol{\theta}) \coloneqq \frac{1}{|D_I|} \sum_{i \in I} |D_i| f_i(\boldsymbol{\theta}). \tag{5.5}$$

FEDAVG (McMahan, Moore, et al., 2017) optimizes the loss (5.5) with theoretical guarantees for FL convergence to a stationary point (Jianyu Wang, Q. Liu, et al., 2020; Xiang Li et al., 2020). At each iteration step n, the server sends the current global model parameters $\boldsymbol{\theta}^n$ to the clients. Each client updates the model by minimizing its local cost function $f_i(\boldsymbol{\theta})$ with K SGD steps initialized on $\boldsymbol{\theta}^n$. Subsequently each client returns the updated local parameters $\boldsymbol{\theta}_i^{n+1}$ to the server. The global model parameters $\boldsymbol{\theta}^{n+1}$ at the iteration step $n+1$ are then estimated as a weighted average, i.e.

$$\boldsymbol{\theta}^{n+1} = \boldsymbol{\theta}^n + \frac{1}{|D|} \sum_{i \in I} |D_i| \left[\boldsymbol{\theta}_i^{n+1} - \boldsymbol{\theta}^n \right]. \tag{5.6}$$

Algorithm 5 provides the implementation of FEDAVG. For the rest of this work, we define the joint dataset for a subset of client $I_x \subset I$ as $D_{I_x} \coloneqq \cup_{i \in I_x} D_i$.

5.2.3 Federated Unlearning

Existing works (G. Liu et al., 2021; Junxiao Wang et al., 2021; Halimi et al., 2022; C. Wu et al., 2022) already consider the problem of unlearning a client from a model optimized through FEDAVG. However, these works do not provide theoretical nor quantitative guarantees on the unlearning procedure. Also, we note that standard MU methods cannot seamlessly be used in the federated setting. On one hand, federated unlearning (FU) with model scrubbing would require clients to perform only $K = 1$ SGD and share their Hessian with the server. Hence, model scrubbing decreases significantly FL convergence speed, while exposing the clients' data by sharing high order quantities with the risk of model inversion (Matt Fredrikson et al., 2015). Moreover, the computation of the Hessian is unfeasible for highly dimensional models. On the other hand, existing MU approaches

based on model perturbation require to retrain the model after the noise is added to the model's parameters. As such, retraining generally requires a significant amount of SGD steps to guarantee convergence to a new optimum, negatively affecting the effectiveness of the unlearning procedure.

In this chapter, we introduce a novel unlearning paradigm which avoids retraining the final model by identifying the optimal FL iteration where unlearning should be applied. Therefore, retraining is applied to an "early" version of the global model with reduced perturbation, thus minimizing the amount of required SGD steps to achieve convergence.

5.3 Unlearning a FL client with IFU

In this section, we develop our theory for the scenario where a model is trained with FedAvg on the set of clients I, after which a client c requests unlearning of its own data. In Section 5.3.1, we define the sensitivity of the global model with respect to a client's contribution, and provide a bound relating this sensitivity to the FL procedure. In Section 5.3.2, we provide a tighter model sensitivity for some specific FL applications. Using Theorem 5.1, we introduce in Section 5.3.3 the perturbation procedure to unlearn a client c from the model trained with FedAvg (Theorem 5.2). Finally, using Theorem 5.2, we introduce Informed Federated Unlearning (IFU) (Algorithm 6).

5.3.1 Theorem 5.1, Bounding the Model Sensitivity

As defined in Section 5.2.2, $\boldsymbol{\theta}_i^{n+1}$ is the local update of client i sent to the server after performing K SGD steps on its dataset D_i after initialization with global model $\boldsymbol{\theta}^n$. Given the contribution $\boldsymbol{\theta}_i^{n+1} - \boldsymbol{\theta}^n$ of a client i, we define the overall FL increment after aggregations across the set of clients I as

$$\Delta(I, \boldsymbol{\theta}^n) \coloneqq \frac{1}{|D_I|} \sum_{i \in I} |D_i| \left[\boldsymbol{\theta}_i^{n+1} - \boldsymbol{\theta}^n \right]. \tag{5.7}$$

By comparing increments obtained by training on the set of clients I, and on the set $I_{-c} \coloneqq I \setminus \{c\}$ obtained after dropping a given client c, we define the bounded sensitivity after n server aggregations as

$$\Psi(n, c) \coloneqq \sum_{s=0}^{n-1} \left\| \Delta(I, \boldsymbol{\theta}^s) - \Delta(I_{-c}, \boldsymbol{\theta}^s) \right\|_2, \tag{5.8}$$

We show in Theorem 5.1 that the model sensitivity of FEDAVG can be bounded by the bounded sensitivity (5.8).

Theorem 5.1. *Under Assumption 5.1, the model sensitivity of* FEDAVG *when removing a client c after n server aggregations is defined as*

$$\alpha(n, c) \coloneqq \|\text{FEDAVG}(I, n) - \text{FEDAVG}(I_{-c}, n)\|_2 \,, \tag{5.9}$$

where $\text{FEDAVG}(I, n)$ *is the output of Algorithm 5, and*

$$\alpha(n, c) \leq \Psi(n, c). \tag{5.10}$$

Proof. See Appendix D.2. □

5.3.2 Improving the Tightness of the Sensitivity Bound

Theorem 5.1 shows that the bounded sensitivity provides a bound for the model sensitivity, while the computation of (5.8) only requires the clients' updated models, which are already shared with the server by design in FEDAVG. Nevertheless, we note that the bounded sensitivity (5.8) does not necessarily faithfully represent the evolution of the sensitivity across FL rounds. For instance, this quantity does not properly account for the unlearning of previous clients contributions for $s < n - 1$. Indeed, these contributions should decrease across iterations due to the subsequent server aggregations and new clients' local work. To account for this aspect, we provide a tighter lower bound by assuming strongly convex and regularized local loss function, leading to a tighter bound for the model sensitivity of FEDAVG (Corollary 5.1).

Corollary 5.1. *Under Assumption 5.1, when considering that clients loss functions are μ-strongly convex and regularized with an L2 norm of weight λ, we have $\alpha(n, c) \leq \Psi(n, c)$ and*

$$\Psi(n, c) = \sum_{s=0}^{n-1} (1 - \eta(\lambda + \mu))^{[(n-1)-s]K} \times \|\Delta(I, \boldsymbol{\theta}^s) - \Delta(I_{-c}, \boldsymbol{\theta}^s)\|_2 \,, \tag{5.11}$$

where η and K are respectively the clients' local learning rate and amount of local work.

Proof. See Appendix D.2.3. □

The bounded sensitivity of Corollary 5.1 shows the following aspects. (1) The importance of a client's contribution decreases through aggregation rounds. (2) Since FL is guaranteed to converge to a stationary point (Jianyu Wang, Q. Liu, et al., 2020; Xiang Li et al., 2020), so does the bounded sensitivity since $\lambda + \mu > 0$. (3) The bounded sensitivity is not necessarily inversely proportional to K. Indeed, due to data heterogeneity, with an increase in K every local model gets closer to its local optimum and the quantity $\|\Delta(I, \boldsymbol{\theta}^n) - \Delta(I_{-c}, \boldsymbol{\theta}^n)\|_2$ increases with the amount of local work K.

When clients have same data distribution, we retrieve $\Delta(I, \boldsymbol{\theta}^n) = \Delta(I_{-c}, \boldsymbol{\theta}^n)$, which yields null bounded sensitivity for every client, i.e. $\Psi(n, c) = 0$. We also note that the bound provided in Corollary 5.1 is tight, e.g. when considering identical eigenvalues for the Hessian of every local loss. More generally, the bound is tight in the limit case where the local learning rate of the clients is small.

We can draw an analogy between the bounded sensitivity (5.8) and client clustering in FL (Sattler et al., 2021; Fraboni, Vidal, Kameni, et al., 2021), where clients are clustered based on their contribution. In this chapter, the bounded sensitivity (5.8) is used instead to bound the sensitivity of the global model across rounds in FEDAVG.

5.3.3 Satisfying Definition 5.1

In this section, we introduce a randomized mechanism to provide guarantees for the unlearning of a given client c, where the magnitude of the perturbation process (Dwork and Roth, 2014) is defined based on the sensitivity of Theorem 5.1. In practice, we define a Gaussian noise mechanism to perturb each parameter of global model $\boldsymbol{\theta}^n$ such that we achieve (ϵ, δ)-unlearning of client c for the resulting model, according to Definition 5.1. We give in Theorem 5.2 sufficient conditions for the noise perturbation to satisfy Definition 5.1.

Theorem 5.2. *Defining*

$$\sigma(n, c) := \left[2\left(\ln(1.25) - \ln(\delta)\right)\right]^{1/2} \epsilon^{-1} \Psi(n, c), \tag{5.12}$$

the noise perturbation $\sigma(n, c)\boldsymbol{I}_{\boldsymbol{\theta}}$ applied to the global model $\boldsymbol{\theta}^n$, where $\boldsymbol{I}_{\boldsymbol{\theta}}$ is the identity matrix, achieves (ϵ, δ)-unlearnig of client c according to Definition 5.1.

Proof. Follows directly from Theorem 5.1 coupled with Theorem A.1 of Dwork and Roth (2014). □

Algorithm 6 Informed Federated Unlearning (IFU)

DURING LEARNING WITH FEDAVG

FEDAVG(I, N) initialized on initial model $\boldsymbol{\theta}^0$.
for n from 0 to $N-1$, and i from 1 to c **do**
  Compute $\Psi(n, i)$, equation (5.8).
end for

WHEN UNLEARNING CLIENT c

Require: c, ϵ, δ, σ, and amount of retraining steps $\tilde{N}$.
1: Get Ψ^* with equation (5.13).
2: Get $T = \arg\max_n \left(\Psi(n, c) \leq \Psi^*\right)$ with eq. (5.14).
3: The new global model is $\tilde{\boldsymbol{\theta}} = \boldsymbol{\theta}^T + N(0, \sigma^2 \boldsymbol{I}_{\boldsymbol{\theta}})$.
4: Run FEDAVG(I_{-c}, $\tilde{N}$) initialized on $\tilde{\boldsymbol{\theta}}$.

We note that, according to Theorem 5.2, (ϵ, δ)-unlearning a client from a given global model requires a standard deviation for the noise that is client-specific and proportional to its bounded sensitivity.

In what follows, the unlearning procedure will be defined with respect to the sensitivity threshold Ψ^* related to the unlearning budget (ϵ, δ) and standard deviation σ:

$$\Psi^* := \left[2\left(\ln(1.25) - \ln(\delta)\right)\right]^{-1/2} \epsilon\sigma. \tag{5.13}$$

5.3.4 Informed Federated Unlearning (IFU)

Using the bounded sensitivity (5.8) and Theorem 5.2, we introduce Informed Federated Unlearning (IFU) to unlearn the contribution of client $c \in I$ from a FL training procedure based on FEDAVG. Algorithm 6 provides the implementation of IFU on top of FEDAVG. We note that during the FL training, IFU requires the server to compute the bounded sensitivity metric $\Psi(n, i)$ from each client's contribution $\boldsymbol{\theta}_i^{n+1}$ and current global model $\boldsymbol{\theta}^n$. These quantities are tracked throughout FL iterations and are used to identify the optimal unlearning strategy after request from a client c.

To unlearn client c, the server identifies the unlearning index T associated to the history of bounded sensitivity metrics, i.e. the most recent global model index such that a perturbation of size σ satisfies Theorem 5.2:

$$T := \arg\max_n \left(\Psi(n, c) \leq \Psi^*\right). \tag{5.14}$$

The new global model is obtained after perturbation $\tilde{\boldsymbol{\theta}} := \boldsymbol{\theta}^T + \nu$, where $\nu \sim N(0, \sigma^2 \boldsymbol{I}_{\boldsymbol{\theta}})$. Our unlearning criterion 5.1 is therefore satisfied for $\tilde{\boldsymbol{\theta}}$ (Theorem 5.2), and the server can perform $\tilde{N}$ new optimization rounds with FEDAVG initialized on $\tilde{\boldsymbol{\theta}}$. Thanks to the

Algorithm 7 Sequential IFU (SIFU)

DURING LEARNING WITH FEDAVG

FEDAVG(I, N) initialized on initial model $\boldsymbol{\theta}_0^0$.
Compute $\Psi_0(n, i)$, equation (5.15).

UNLEARNING A SERIES OF REQUESTS $\{W_r\}$

Require: $\{W_r\}_{r=1}^R$, ϵ, δ, σ, and $\{N_r\}_{r=1}^R$
$O(0) = \{\emptyset\}$.
Get Ψ^* with equation (5.13).
for r from 1 to R **do**
    $I_r = I_{r-1} \setminus W_r$.
    Compute (ζ_r, T_r) with $O(r-1)$, eq. (5.17) and (5.18).
    Update $O(r)$ with ζ_r, T_r, and $O(r-1)$, eq. (5.19).
    The new global model is $\boldsymbol{\theta}_r^0 = \boldsymbol{\theta}_{\zeta_r}^{T_r} + N(0, \sigma^2 \boldsymbol{I}_{\boldsymbol{\theta}})$.
    Perform FEDAVG(D_r, N_r) initialized on $\boldsymbol{\theta}_r^0$.
    Compute $\Psi_r(n, i)$, eq. (5.15).
end for

contribution of the remaining clients in $\tilde{\boldsymbol{\theta}}$, we expect the retraining with IFU to be generally faster than retraining with a random initial model.

Since $\Psi(n, i)$ is strictly increasing with n, the server can stop from computing the bounded sensitivity (5.8) for client i whenever $\Psi(n_i, i) > \Psi^*$ is verified after n_i optimization rounds. At this point, the model $\boldsymbol{\theta}^{n_i - 1}$ will be selected for the unlearning request of client i, as the models at subsequent iterations do not comply with the desired unlearning budget Ψ^*.

5.4 Sequential FU with SIFU

In this section, we extend IFU to the sequential unlearning setting with Sequential IFU (SIFU). With Algorithm 7, SIFU is designed to satisfy a series of R unlearning requests $\{W_r\}_{r=1}^R$, where W_r is the set of clients to unlearn at request index r. SIFU generalizes IFU for which $R = 1$ and $W_1 = \{c\}$. We provide an illustration of SIFU with an example in Figure 5.1.

The notations introduced thus far need to be generalized to account for our series of unlearning requests $W_1, W_2, \ldots, W_R$. Global models are now referenced by their coordinates (r, n), i.e. $\boldsymbol{\theta}_r^n$, which represent the unlearning request index r and the amount of server aggregations n performed during the retraining. Hence, $\boldsymbol{\theta}_r^0$ is the initialization of the model when unlearning the clients in W_r. Also, we consider that the retraining at request index r requires N_r server aggregations on the remaining clients. Therefore, by construction, $\boldsymbol{\theta}_r^{N_r}$ is the model obtained after using SIFU to (ϵ, δ)-unlearn the sequence of unlearning requests

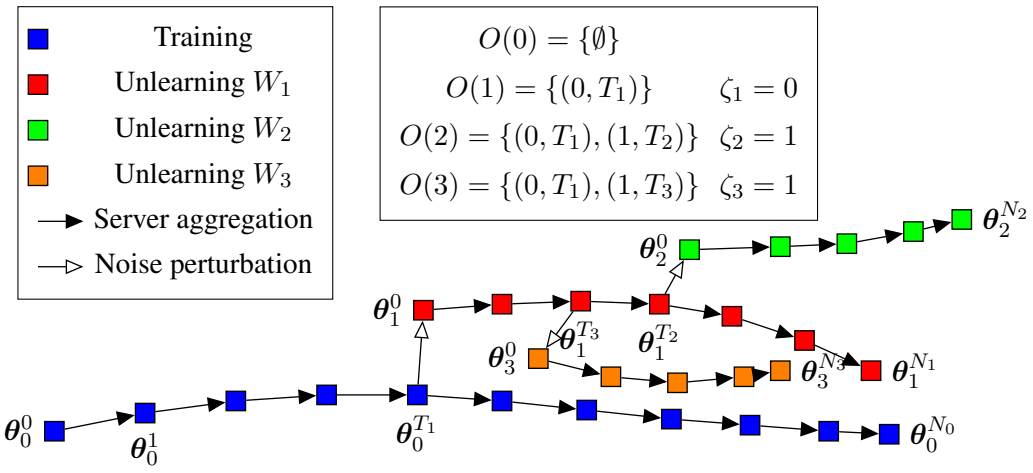

Fig. 5.1.: Illustration of SIFU (Algorithm 7) when the server receives $R = 3$ unlearning requests, through the evolution of the global model parameters $\boldsymbol{\theta}_r^n$ after server aggregation and noise perturbation. After standard federated training via FEDAVG(I, N_0), the oracle is $O(0) = \{\emptyset\}$, and the current training history is $(\boldsymbol{\theta}_0^0, \dots, \boldsymbol{\theta}_0^{N_0})$. At request $r = 1$ the unlearning index is T_1, and the training history becomes $(\boldsymbol{\theta}_0^0, \dots, \boldsymbol{\theta}_0^{T_1}, \boldsymbol{\theta}_1^0, \dots, \boldsymbol{\theta}_1^{N_1})$. The oracle is updated to $O(1) = \{(0, T_1)\}$, and $\zeta_1 = 0$. At request $r = 2$ the unlearning index is T_2 and the training history becomes $(\boldsymbol{\theta}_0^0, \dots, \boldsymbol{\theta}_0^{T_1}, \boldsymbol{\theta}_1^0, \dots, \boldsymbol{\theta}_1^{T_2}, \boldsymbol{\theta}_2^0, \dots \boldsymbol{\theta}_2^{N_2})$. The new node is added to the oracle $O(2) = \{(0, T_1), (1, T_2)\}$, and $\zeta_2 = 1$. Finally, at request $r = 3$, the unlearning index is found at $T_3 < T_2$ in the branch of request $r = 1$. The updated training history is now $(\boldsymbol{\theta}_0^0, \dots, \boldsymbol{\theta}_0^{T_1}, \boldsymbol{\theta}_1^0, \dots, \boldsymbol{\theta}_1^{T_3}, \boldsymbol{\theta}_3^0, \dots \boldsymbol{\theta}_3^{N_3})$, the oracle is updated as $O(3) = \{(0, T_1), (1, T_3)\}$, and $\zeta_3 = 1$.

$\{W_s\}_{s=1}^r$. Finally, we define I_r as the set of remaining clients after unlearning request r, i.e. $I_r := I \setminus \cup_{s=1}^r W_s = I_{r-1} \setminus W_r$ with $I_0 = I$.

We extend the bounded sensitivity (5.8) with $\Psi_r(n, i)$ to compute the metric of client i at unlearning index r with

$$\Psi_r(n, i) := \sum_{s=0}^{n-1} \|\Delta(I_r, \boldsymbol{\theta}_r^s) - \Delta(I_r \setminus \{i\}, \boldsymbol{\theta}_r^s)\|_2 \,. \tag{5.15}$$

When unlearning client c at $r = 1$, the metric at $r = 0$ is equivalent to the previous definition of Ψ. Also, when computing the metric on a client already unlearned, i.e. $i \notin I_r$, we retrieve $\Psi_r(n, i) = 0$. Finally, for a set of clients S, we generalize the bounded sensitivity (5.15) to

$$\Psi_r(n, S) = \max_{i \in S} \Psi_r(n, i). \tag{5.16}$$

With SIFU, the selection of the unlearning index T for a request r depends of the past history of unlearning requests. To keep track of the unlearning history, we introduce the oracle $O(r)$ which returns at each request r the coordinates of the history of global models where

unlearning has been applied. These coordinates represent the nodes of the training history across unlearning requests (Figure 5.1). With reference to Figure 5.1, we start with the original sequence of global models obtained at each FL round, i.e. $(\boldsymbol{\theta}_0^0, \dots, \boldsymbol{\theta}_0^{N_0})$. Similarly to IFU, the first unlearning request requires to identify the unlearning index T_1 for which the corresponding global model $\boldsymbol{\theta}_0^{T_1}$ must be perturbed to obtain $\boldsymbol{\theta}_1^0$ and retrained until convergence, i.e. up to $\boldsymbol{\theta}_1^{N_1}$. The oracle is updated with the coordinates of the branching $O(1) = \{(0, T_1)\}$, and the current training history is now $(\boldsymbol{\theta}_0^0, \dots, \boldsymbol{\theta}_0^{T_1}, \boldsymbol{\theta}_1^0, \dots, \boldsymbol{\theta}_1^{N_1})$. At the next unlearning request, the server needs to identify the coordinates (ζ_r, T_r) in the new training history for which unlearning must be applied on the model $\boldsymbol{\theta}_{\zeta_r}^{T_r}$ to obtain $\boldsymbol{\theta}_r^0 = \boldsymbol{\theta}_{\zeta_r}^{T_r} + \mathcal{N}(0, \sigma^2 \boldsymbol{I}_{\boldsymbol{\theta}})$. The oracle is subsequently updated with the new set of nodes describing the new branching in the training history. By construction, we have $\zeta_r \leq r - 1$ and $T_r \leq N_{\zeta_r}$.

More precisely, we define the index ζ_r associated to the first coordinate in $O(r-1)$ for which the bounded sensitivity (5.15) of clients in W_r exceeds Ψ^*. Formally, we have

$$\zeta_r \coloneqq \min_s \{s : \Psi_s(n, W_r) > \Psi^* \text{ and } (s, n) \in O(r-1), r-1\}. \tag{5.17}$$

The definition of T_r follows directly from the one of ζ_r. Similarly as for IFU, the unlearning index T_r quantifies the maximum amount of server aggregations starting from the unlearning request index ζ_r such that the bounded sensitivity $\Psi_{\zeta_r}(n, W_r)$ on this global model is inferior to Ψ^*, i.e.

$$T_r \coloneqq \arg\max_n \{\Psi_{\zeta_r}(n, W_r) \leq \Psi^*\}. \tag{5.18}$$

Finally, we update the oracle $O(r-1)$ to $O(r)$ with the following recurrent equation

$$O(r) = \{(s, n) \in O(r-1) \text{ s.t. } s < \zeta_r, (\zeta_r, T_r)\}. \tag{5.19}$$

Theorem 5.3 shows that for a model trained with SIFU after a given training request r, (ϵ, δ)-unlearning is guaranteed for every client belonging to the sets W_s, $s \leq r$.

Theorem 5.3. *The model $\boldsymbol{\theta}_r^{N_r}$ obtained with SIFU satisfies (ϵ, δ)-unlearning for every client in current and previous unlearning requests, i.e. clients in $\cup_{s=1}^r W_s$.*

Proof. See Appendix D.3. □

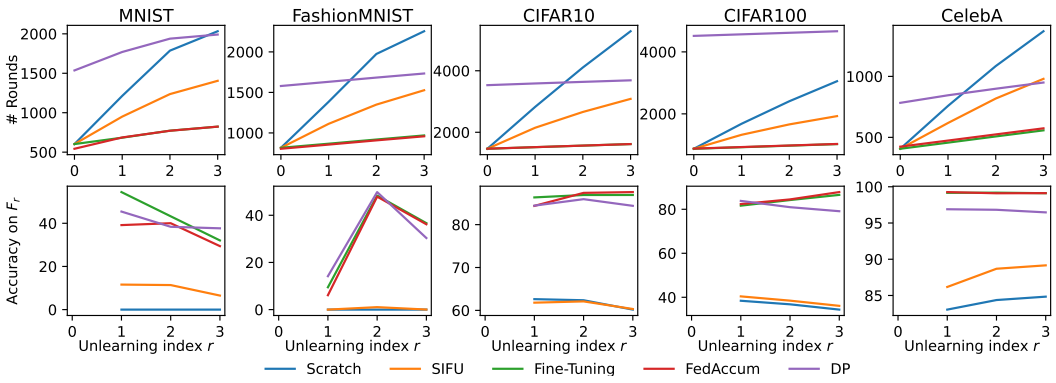

Fig. 5.2.: Total amount of aggregation rounds (1st row) and model accuracy of unlearned clients (2nd row) for MNIST, FashionMNIST, CIFAR10, CIFAR100, and CelebA (the lower the better). The server runs a federated routine with $M = 100$ clients, and unlearns 10 of them at each unlearning request ($R = 3$).

5.5 Experiments

In this section, we experimentally demonstrate the effectiveness of SIFU on a series of benchmarks introduced in Section 5.5.1. In Section 5.5.2, we illustrate and discuss our experimental results. Results and related code are publicly available at URL.

5.5.1 Experimental Setup

Datasets. We report experiments on reduced versions of MNIST (Lecun et al., 1998), FashionMNIST (Xiao et al., 2017), CIFAR-10 (Krizhevsky et al., n.d.), CIFAR-100 (Krizhevsky et al., n.d.), and CelebA (Z. Liu et al., 2015). For each dataset, we consider $M = 100$ clients, with 100 data points each. For MNIST and FashionMNIST, each client has data samples from only one class, so that each class is represented in 10 clients only. For CIFAR10 and CIFAR100, each client has data samples with ratio sampled from a Dirichlet distribution with parameter 0.1 (Harry Hsu et al., 2019). Finally, in CelebA, clients own data samples representing the same celebrity. With these five datasets, we consider different level of heterogeneity based on label and feature distribution.

Models. For MNIST, we train a logistic regression model to consider a convex classification problem, while, for the other datasets, we train a neural network with convolutional layers followed by fully connected ones. More details on the networks are available in Appendix D.4.

Unlearning schemes. In addition to SIFU, we consider the following unlearning schemes from the state-of-the-art: SCRATCH, where retraining of a new initial model is performed on

the remaining clients; FINE-TUNING, where retraining is performed on the current global model with the remaining clients; LAST (Neel et al., 2021), where retraining is performed on the remaining clients via perturbation of the final FL global model; DP (Dwork and Roth, 2014), where training with every client is performed with differential privacy, and FEDACCUM (G. Liu et al., 2021), where retraining is performed on the current global model from which the server removes the updates of the clients to unlearn, by re-aggregating the parameter updates of clients that were stored by the server across FL iterations. We provide in Appendix D.4 the pseudo-code of FEDACCUM with the notation of our paper. We remind that FEDACCUM does not provide quantitative guarantees of the unlearning procedure, and requires the server to store the full sequence of models during the FL procedure.

Experimental scenario. We consider a sequential unlearning scenario in which the server performs the FL training procedure and then receives $R = 3$ sequential unlearning requests to unlearn 10 random clients per request. In the special case of MNIST and FashionMNIST, the server must unlearn 10 clients owning the same class. The server orchestrates each unlearning scheme through retraining until the global model accuracy on the remaining clients exceeds a fixed value specific to each dataset. We set the minimum number of 50 aggregation rounds, and a maximum budget of 10000 rounds when the stopping accuracy criterion is not met. Each unlearning method is applied with the same hyperparameters, i.e. stopping accuracy, local learning rate η, and amount of local work K (Appendix D.4). We define the set of clients requesting unlearning as:

$$F_r = \cup_{s=1}^{r} W_s. \tag{5.20}$$

In our experimental scenario, we have $|F_0| = 0$ during training and $|F_1| = 10$, $|F_2| = 20$, and $|F_3| = 30$ after each unlearning request.

Unlearning quantification. We verify the success of an unlearning scheme with two metrics: (a) the amount of server aggregation rounds needed for retraining, and (b) the resulting model accuracy on the unlearned clients. we note that, by construction, SCRATCH perfectly unlearns the clients from a request W_r. Therefore, we consider an unlearning scheme successful if it reaches similar accuracy of SCRATCH with less aggregation rounds, when tested on the data samples of F_r.

5.5.2 Experimental Results

Figure 5.2 shows that for every dataset and unlearning index, FINE-TUNING, FEDACCUM, and DP provide similar model accuracy for the unlearned clients in F_r (Figure 5.2-2$^{\text{nd}}$ row), albeit significantly higher than for SCRATCH, the unlearning standard. Noteworthy,

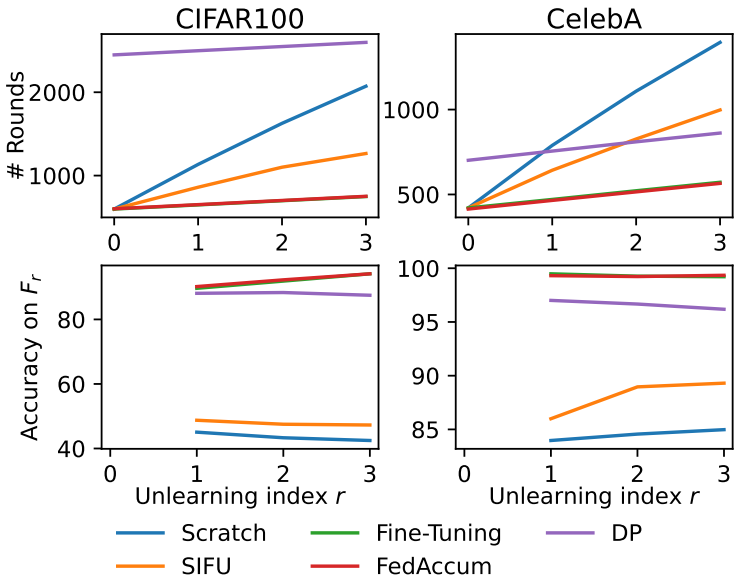

Fig. 5.3.: Total amount of aggregation rounds (1st row) and model accuracy of unlearned clients (2nd row) for the unlearning of watermarked data from CIFAR100 and CelebA.

unlearning with FINE-TUNING, FEDACCUM, and DP results in significantly less aggregation rounds than SCRATCH (Figure 5.2-1st row). We note that SIFU and SCRATCH lead to similar unlearning results, quantified by low accuracy on the unlearned clients F_r (Figure 5.2-2nd row), while SIFU unlearns these clients in roughly half the amount of aggregation rounds needed for SCRATCH (Figure 5.2-1st row). However, the model accuracy of SIFU is slightly higher than the one of SCRATCH, with perfect overlap only for FashionMNIST. This behavior is natural and can be explained by our privacy budget (ϵ, δ), which trades unlearning capabilities for effectiveness of the retraining procedure. With highest unlearning budget, i.e. $\epsilon = 0$ and $\delta = 0$, SIFU would require to retrain from the initial model $\boldsymbol{\theta}_0^0$, thus reducing to SCRATCH.

Finally, we observed that when unlearning with LAST, the retrained model always converged to a local optimum with accuracy inferior to our target after 10000 aggregation rounds. This behavior is likely due to the difficulty of calibrating the noise perturbation due to the numerous heterogeneous contributions of the clients. For this reason, we decided to exclude LAST from the plots of Figure 5.2.

5.5.3 Verifying Unlearning through Watermarking

The work of Sommer et al., 2020 proposes an adversarial approach to verify the efficiency of an unlearning scheme based on watermarking. We apply here this method to our federated

setting, in which watermarking is operated by each client by randomly assigning on all its data samples the maximum possible value to 10 given pixels. To ensure that clients' data heterogeneity is only due to the modification of the pixels, we define heterogeneous data partitioning across clients by randomly assigning the data according to a Dirichlet distribution with parameter 1. Figure 5.3 shows our results for this experimental scenario on CIFAR100 and CelebA, while Appendix D.4 provides similar results for MNIST, FashionMNIST and CIFAR10. We retrieve the same conclusions drawn from Figure 5.2. SIFU and SCRATCH have similar accuracies on the unlearned clients in F_r, to demonstrate the effectiveness of the unlearning. Moreover, SIFU unlearns these clients in significantly less aggregation rounds than SCRATCH.

5.5.4 Impact of the noise perturbation on SIFU

Appendix D.4 illustrates the impact of the perturbation amplitude σ on convergence speed when unlearning with SIFU. We note that when unlearning with a small σ, SIFU has identical behavior to SCRATCH as the unlearning is applied to the initial random model $\boldsymbol{\theta}_0^0$. With large values of σ, SIFU performs instead identically to LAST and applies the unlearning to the finale global model $\boldsymbol{\theta}_r^{N_r}$.

5.6 Conclusions

In this chapter, we introduce informed federated unlearning (IFU), a novel federated unlearning scheme to unlearn a client's contribution from a model trained with federated learning. Upon receiving an unlearning request from a given client, IFU identifies the optimal FL iteration from which FL has to be reinitialised, with statistical unlearning guarantees defined by Definition 5.1. We extend the theory of IFU to account for the practical scenario of sequential unlearning (SIFU), where the server receives a series of forgetting request of one or more clients. We prove that SIFU can unlearn a series of forgetting requests while satisfying our unlearning guarantees, and demonstrate the effectiveness of our methods on a variety of tasks and dataset.

An additional contribution of this chapter consists in a new theory for bounding the clients contribution in FL. The server can compute this bound for every client without asking for any additional computation and communication. The theoretical justification of this approach relies on the linear approximation of the clients' loss function, and its relevance is here demonstrated across several benchmarks. Future extensions of this chapter will focus on generalizing our unlearning framework to more general settings.

6 Free-rider Attacks on Model Aggregation in Federated Learning

Contents

6.1 Introduction **96**

6.2 Methods **98**

6.2.1 Federated learning through model aggregation: FedAvg and FedProx 98

6.2.2 Formalizing Free-rider attacks 98

6.2.3 SGD perturbation of the fair clients local model 99

6.2.4 Plain free-riding 102

6.2.5 Disguised free-riding 104

6.2.6 FedProx 107

6.3 Experiments **107**

6.3.1 Experimental Details 108

6.3.2 Free-rider attacks: convergence and performances 110

6.4 Conclusion and discussion **111**

In this chapter, we introduce the first theoretical and experimental analysis of free-rider attacks on federated learning schemes based on iterative parameters aggregation, such as FEDAVG or FEDPROX, and provide formal guarantees for these attacks to converge to the aggregated models of the fair participants. We demonstrate the effectiveness of free-rider attacks on a number of experimental scenarios, in both iid and non-iid settings. We conclude by providing recommendations to avoid free-rider attacks in real world applications of federated learning, especially in sensitive domains where security of data and models is critical. This work is published at the International Conference in Artificial Intelligence and Statistics (AISTATS) of 2021 as Fraboni, Vidal, and Lorenzi (2021).

6.1 Introduction

Federated learning is a training paradigm that has gained popularity in the last years as it enables different clients to jointly learn a global model without sharing their respective data. It is particularly suited for Machine Learning applications in domains where data security is critical, such as healthcare (Brisimi et al., 2018; Silva, Gutman, et al., 2019). The relevance of this approach is witnessed by current large scale federated learning initiatives under development in the medical domain, for instance for learning predictive models of breast cancer[1], or for drug discovery and development[2].

The participation to this kind of research initiatives is usually exclusive and typical of applications where data is scarce and unique in its kind. In these settings, aggregation results entail critical information beyond data itself, since a model trained on exclusive datasets may have very high commercial or intellectual value. For this reason, providers may not be interested in sharing the model: the commercialization of machine learning products would rather imply the availability of the model as a service through web- or cloud-based API. This is due to the need of preserving the intellectual property on the model components, as well as to avoid potential information leakage, for example by limiting the maximum number of queries allowed to the users (Carlini et al., 2019a; Matt Fredrikson et al., 2015; Ateniese et al., 2015).

This critical aspect can lead to the emergence of opportunistic behaviors in federated learning, where ill-intentioned clients may participate with the aim of obtaining the federated model, without actually contributing with any data during the training process. In particular, the attacker, or free-rider, aims at disguising its participation to federated learning while ensuring that the iterative training process ultimately converges to the wished target: the aggregated model of the fair participants. Free-riding attacks performed by ill-intentioned participants ultimately open federated learning initiatives to intellectual property loss and data privacy breaches, taking place for example in the form of model inversion (Matthew Fredrikson et al., 2014; Matt Fredrikson et al., 2015).

The study of security and safety of federated learning is an active research domain, and several kind of attacks are matter of ongoing studies. For example, an attacker may interfere during the iterative federated learning procedure to degrade/modify models performances (Bhagoji et al., 2019; B. Li et al., 2016; Yin et al., 2018; Xie et al., 2019; Shen et al., 2016), or retrieve information about other clients' data (Z. Wang et al., 2019; Hitaj et al., 2017). Since currently available defense methods such as (Fung et al., 2020; Bhagoji et al., 2019) are generally based on outliers detection mechanisms, they are generally not suitable to

[1] blogs.nvidia.com/blog/2020/04/15/federated-learning-mammogram-assessment/
[2] www.imi.europa.eu/projects-results/project-factsheets/melloddy

prevent free-riding, as this kind of attack is explicitly conceived to stay undetected while not perturbing the FL process. Free-riding may become a critical aspect of future machine learning applications, as federated learning is rapidly emerging as the standard training scheme in current cooperative learning initiatives. To the best of our knowledge, the only investigation is in a preliminary work (J. Lin et al., 2019) focusing on attack strategies operated on federated learning based on gradient aggregation. However, no theoretical guarantees are provided for the effectiveness of this kind of attacks. Furthermore this setup is unpractical in many real world applications, where federated training schemes based on model averaging are instead more common, due to the reduced data exchange across the network. FedAvg (McMahan, Moore, et al., 2017) is the most representative framework of this kind, as it is based on the iterative averaging of the clients models' parameters, after updating each client model for a given number of training epochs at the local level. To improve the robustness of FedAvg in non-iid and heterogeneous learning scenarios, FedProx (T. Li, Sahu, Zaheer, Sanjabi, Talwalkar, and Smith, 2018b) extends FedAvg by including a regularization term penalizing local departures of clients' parameters from the global model.

The contribution of this cahpter consists in the development of a theoretical framework for the study of free-rider attacks in federated learning schemes based on model averaging, such as in FedAvg and FedProx. The problem is here formalized via the reformulation of federated learning as a stochastic process describing the evolution of the aggregated parameters across iterations. To this end, we build upon previous works characterizing the evolution of model parameters in Stochastic Gradient Descent (SGD) as a continuous time process (Mandt et al., 2017; Orvieto and Lucchi, 2018; Qianxiao Li et al., 2017; He et al., 2018). A critical requirement for opportunistic free-rider attacks is to ensure the convergence of the training process to the wished target represented by the aggregated model of the fair clients. We show that the proposed framework allows to derive explicit conditions to guarantee the success of the attack. This is an important theoretical feature as it is of primary interest for the attacker to not interfere with the learning process.

The manuscript is structured as follows. We first derive in Section 6.2.4 a basic free-riding strategy to guarantee the convergence of federated learning to the model of the fair participants. This strategy simply consists in returning at each iteration the received global parameters. As this behavior could easily be detected by the server, we build more complex strategies to disguise the free-rider contribution to the optimization process, based on opportune stochastic perturbations of the parameters. We demonstrate in Section 6.2.5 that this strategy does not alter the global model convergence, and in Section 6.3 we experimentally demonstrate our theory on a number of learning scenarios in both iid and non-iid settings. All proofs and additional material are provided in the Appendix.

6.2 Methods

Before introducing in Section 6.2.2 the core idea of free-rider attacks, we first recapitulate in Section 6.2.1 the general context of parameter aggregation in federated learning.

6.2.1 Federated learning through model aggregation: FedAvg and FedProx

In federated learning, we consider a set I of participating clients respectively owning datasets $\mathcal{D}_i$ composed of M_i samples. During optimization, it is generally assumed that the D elements of the clients' parameters vector $\boldsymbol{\theta}_i^t = (\theta_{i,0}^t, \theta_{i,1}^t, ..., \theta_{i,D}^t)$, and the global parameters $\boldsymbol{\theta}^t = (\theta_0^t, \theta_1^t, ..., \theta_D^t)$ are aggregated independently at each iteration round t. Following this assumption, and for simplicity of notation, in what follows we restrict our analysis to a single parameter entry, that will be generally denoted by θ_i^t and θ^t for clients and server respectively.

In this setting, to estimate a global model across clients, FedAvg (McMahan, Moore, et al., 2017) is an iterative training strategy based on the aggregation of local model parameters θ_i^t. At each iteration step t, the server sends the current global model parameters θ^t to the clients. Each client updates the model by minimizing over E epochs the local cost function $\mathcal{L}(\theta_i^{t+1}, \mathcal{D}_i)$ initialized with θ^t, and subsequently returns the updated local parameters θ_i^{t+1} to the server. The global model parameters θ^{t+1} at the iteration step $t+1$ are then estimated as a weighted average:

$$\theta^{t+1} = \sum_{i \in I} \frac{M_i}{N} \theta_i^{t+1}, \tag{6.1}$$

where $N = \sum_{i \in I} M_i$ represents the total number of samples across distributed datasets. FedProx (T. Li, Sahu, Zaheer, Sanjabi, Talwalkar, and Smith, 2018b) builds upon FedAvg by adding to the cost function a L2 regularization term penalizing the deviation of the local parameters θ_i^{t+1} from the global parameters θ^t. The new cost function is $\mathcal{L}_{Prox}(\theta_i^{t+1}, \mathcal{D}_i, \theta^t) = \mathcal{L}(\theta_i^{t+1}, \mathcal{D}_i) + \frac{\mu}{2} \left\| \theta_i{}^{t+1} - \theta^t \right\|^2$ where μ is the hyperparameter monitoring the regularization by enforcing proximity between local update $\theta_i{}^{t+1}$ and reference model θ^t.

6.2.2 Formalizing Free-rider attacks

Aiming at obtaining the aggregated model of the fair clients, the strategy of a free-rider consists in participating to federated learning by dissimulating local updating through the

Algorithm 8 Free-riding in federated learning

Require: learning rate λ, epochs E, initial model θ^0, batch size S,

$\tilde{\theta}^0 = \theta^0$;
for each round t=0,...,T-1 **do**
    Send the global model $\tilde{\theta}^t$ to all the clients;
    for each fair client $j \in J$ **do**
        $\tilde{\theta}_j^{t+1} = ClientUpdate(\tilde{\theta}^t, E, \lambda)$;
        Send $\tilde{\theta}_j^{t+1}$ to the server;
    end for
    for each free-rider $k \in K$ **do**
        if disguised free-rider **then**
            $\tilde{\theta}_k^{t+1} = \tilde{\theta}^t + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma_k^2 I)$;
        else
            $\tilde{\theta}_k^{t+1} = \tilde{\theta}^t$;
        end if
        Send $\tilde{\theta}_k^{t+1}$ to the server;
    end for
    $\tilde{\theta}^{t+1} = \sum_{j \in J} \frac{M_j}{N} \tilde{\theta}_j^{t+1} + \sum_{k \in K} \frac{M_k}{N} \tilde{\theta}_k^{t+1}$;
end for

sharing of opportune counterfeited parameters. The free-riding attacks investigated in this chapter are illustrated in Algorithm 8, and analysed in the following sections from both theoretical and experimental standpoints.

We denote by J the set of fair clients, i.e. clients following the federated learning strategy of Section 6.2.1 and by K the set of free-riders, i.e. malicious clients pretending to participate to the learning process, such that $I = J \cup K$ and $J \neq \emptyset$. We denote by M_K the number of samples declared by the free-riders.

6.2.3 SGD perturbation of the fair clients local model

To describe the clients' parameters observed during federated learning, we rely on the modeling of Stochastic Gradient Descent (SGD) as a continuous time stochastic process (Mandt et al., 2017; Orvieto and Lucchi, 2018; Qianxiao Li et al., 2017; He et al., 2018).

For a client j, let us consider the following form for the loss function:

$$\mathcal{L}_j(\theta_j) = \frac{1}{M_j} \sum_{n=1}^{M_j} l_{n,j}(\theta_j), \tag{6.2}$$

where M_j is the number of samples owned by the client, and $l_{n,j}$ is the contribution to the overall loss from a single observation $\{x_{n,j}; y_{n,j}\}$. The gradient of the loss function is defined as $g_j(\theta_j) \equiv \nabla \mathcal{L}_j(\theta_j)$.

We represent SGD by considering a minibatch $\mathcal{S}_{j,k}$, composed of a set of S different indices drawn uniformly at random from the set $\{1, \ldots, M_j\}$, each of them indexing a function $l_{n,j}(\theta_j)$ and where k is the index of the minibatch. Based on $\mathcal{S}_{j,k}$, we form a stochastic estimate of the loss,

$$\mathcal{L}_{\mathcal{S}_{j,k}}(\theta_j) = \frac{1}{S} \sum_{n \in \mathcal{S}_{j,k}} l_{n,j}(\theta_j), \tag{6.3}$$

where the corresponding stochastic gradient is defined as $g_{\mathcal{S}_{j,k}}(\theta_j) \equiv \nabla \mathcal{L}_{\mathcal{S}_{j,k}}(\theta_j)$.

By observing that gradient descent is a sum of S independent and uniformly distributed samples, thanks to the central limit theorem, gradients at the client level can thus be modeled by a Gaussian distribution

$$g_{\mathcal{S}_{j,k}}(\theta_j) \sim \mathcal{N}(g_j(\theta_j), \frac{1}{S}\sigma_j^2(\theta_j)), \tag{6.4}$$

where $g_j(\theta_j) = \mathbb{E}_s \left[g_{\mathcal{S}_{j,k}}(\theta_j) \right]$ is the full gradient of the loss function in equation (6.2) and $\sigma_j^2(\theta_j)$ is the variance associated with the loss function in equation (6.3).

SGD updates are expressed as:

$$\theta_j(u_j + 1) = \theta_j(u_j) - \lambda g_{\mathcal{S}_{j,k}}(\theta_j(u_j)), \tag{6.5}$$

where u_j is the SGD iteration index and λ is the learning rate set by the server.

By defining $\Delta\theta_j(u_j) = \theta_j(u_j + 1) - \theta_j(u_j)$, we can rewrite the update process as

$$\Delta\theta_j(u_j) = -\lambda g_j(\theta_j(u_j)) + \frac{\lambda}{\sqrt{S}}\sigma_j(\theta_j)\Delta W_j, \tag{6.6}$$

where $\Delta W_j \sim \mathcal{N}(0, 1)$. The resulting continuous-time model (Mandt et al., 2017; Orvieto and Lucchi, 2018; Qianxiao Li et al., 2017; He et al., 2018) is

$$\mathrm{d}\theta_j = -\lambda g_j(\theta_j)\mathrm{d}u_j + \frac{\lambda}{\sqrt{S}}\sigma_j(\theta_j)\mathrm{d}W_j. \tag{6.7}$$

where W_j is a continuous time Wiener Process.

Similarly as in (Mandt et al., 2017), we assume that $\sigma_j(\theta_j)$ is approximately constant with respect to θ_j for the client's stochastic gradient updates between t and $t + 1$, and will therefore denote $\sigma_j(\theta_j) = \sigma_j^t$. Following (Mandt et al., 2017), we consider a local quadratic approximation for the client's loss, leading to a linear form for the gradient

$g_j(\theta_j) \simeq r_j[\theta_j - \theta_j^*]$, where $r_j \in \mathbb{R}^+$ depends on the approximation of the cost function around the local minimum θ_j^*. This assumption enables rewriting equation (6.7) as an Ornstein-Uhlenbeck process (Uhlenbeck and Ornstein, 1930). Starting from the initial condition represented by θ^t, the global model received at the iteration t, we characterize the local updating of the parameters through equation (6.7), and we follow the evolution up to the time $\frac{EM_j}{S}$, where E is the number of epochs, and M_j is the number of samples owned by the client. Assuming that M_j is a multiple of S, the number of samples per minibatch, the quantity $\frac{EM_j}{S}$ represents the total number of SGD steps run by the client. The updated model θ_j^{t+1} uploaded to the server therefore takes the form:

$$\theta_j^{t+1} = \underbrace{e^{-\lambda r_j \frac{EM_j}{S}}[\theta^t - \theta_j^*] + \theta_j^*}_{\hat{\theta}_j^{t+1}} + \frac{\lambda}{\sqrt{S}} \int_{u=0}^{\frac{EM_j}{S}} e^{-\lambda r_j \left(\frac{EM_j}{S} - u\right)} \sigma_j^t dW_u. \tag{6.8}$$

We note that the relative number of SGD updates for the fair clients, $\frac{EM_j}{S}$, influences the parameter $\eta_j = e^{-\lambda r_j \frac{EM_j}{S}}$, which becomes negligible for large values of E.

The variance introduced by SGD can be rewritten as

$$\mathrm{Var}\left[\theta_j^{t+1} | \theta^t\right] = \underbrace{\frac{\lambda}{S} {\sigma_j^t}^2 \frac{1}{2r_j}\left[1 - e^{-2\lambda r_j \frac{EM_j}{S}}\right]}_{{\rho_j^t}^2}, \tag{6.9}$$

where we can see that the higher $\frac{EM_j}{S}$, the lower the overall SGD noise. The noise depends on the local loss function r_j, on the server parameters (number of epochs E, learning rate λ, and number of samples per minibatch S), and on the clients' data specific parameters (SGD variance ${\sigma_j^t}^2$).

Equation (6.8) shows that clients' parameters observed during federated learning can be expressed as $\theta_j^t = \hat{\theta}_j^t + \rho_j^t \zeta_{j,t}$, where, given θ^t, $\hat{\theta}_j^t$ is a deterministic component corresponding to the model obtained with $\frac{EM_j}{S}$ steps of gradient descents, and $\zeta_{j,t}$ is a delta-correlated Gaussian white noise. We consider in what follows a constant local noise variance σ_j^2 (this assumption will be relaxed in Section 6.2.5 to consider instead time-varying noise functions ρ_j^t).

Based on this formalism, in the next Section we study a basic free-rider strategy simply consisting in returning at each iteration the received global parameters. We call this type of attack *plain free-riding*.

6.2.4 Plain free-riding

We denote by $\tilde{\theta}$ and $\tilde{\theta}_j$ respectively the global and local model parameters obtained in presence of free-riders. The plain free-rider returns the same model parameters as the received ones, i.e. $\forall k \in K,\ \tilde{\theta}_k^{t+1} = \tilde{\theta}^t$. In this setting, the server aggregation process (6.1) can be rewritten as:

$$\tilde{\theta}^{t+1} = \sum_{j \in J} \frac{M_j}{N} \tilde{\theta}_j^{t+1} + \frac{M_K}{N} \tilde{\theta}^t \ , \tag{6.10}$$

where $\tilde{\theta}^t$ is the global model and $\tilde{\theta}_j^t$ are the fair clients' local models uploaded to the server for free-riding.

Free-riders perturbation of the fair clients local model

In this section, we investigate the effect of the free-riders on the local optimization performed by the fair clients at every server iteration. The participation of the free-riders to federated learning implies that the processes of the fair clients are being perturbed by the attacks throughout training. In particular, the initial conditions of the local optimization problems are modified according to the perturbed aggregation of equation (6.10).

Back to the assumptions of Section 6.2.3 , the initial condition $\tilde{\theta}^t$ of the local optimization includes now the aggregated model of the fair clients and a perturbation coming from the free-riders. Thus, equation (6.8) in presence of free-riding can be written as

$$\tilde{\theta}_j^{t+1} = \eta_j[\tilde{\theta}^t - \theta_j^*] + \theta_j^* + \frac{\lambda}{\sqrt{S}} \int_{u=0}^{\frac{EM_j}{S}} e^{-\lambda r_j \left(\frac{EM_j}{S} - u\right)} \tilde{\sigma}_j^t dW_u,$$

where $\tilde{\sigma}_j^t = \sigma_j^t(\tilde{\theta}_j)$ is the SGD variance for free-riding. We consider that $\tilde{\sigma}_j^t = \sigma_j^t = \sigma_j$. This assumption will be relaxed in Section 6.2.5 to consider instead time-varying noise functions. With analogous considerations to those made in Section 6.2.3, the updated parameters take the form:

$$\tilde{\theta}_j^{t+1} = \eta_j[\tilde{\theta}^t - \theta_j^*] + \theta_j^* + \rho_j \tilde{\zeta}_{j,t}, \tag{6.11}$$

where $\tilde{\zeta}_{j,t}$ is a delta-correlated Gaussian white noise. Similarly as for federated learning, $\mathbb{E}\left[\tilde{\theta}_j^{t+1} | \tilde{\theta}^t\right] = \eta_j[\tilde{\theta}^t - \theta_j^*] + \theta_j^*$, and $\text{Var}\left[\tilde{\theta}_j^{t+1} | \tilde{\theta}^t\right] = \rho_j^2$.

We want to express the global optimization process $\tilde{\theta}^t$ due to free-riders in terms of a a perturbation of the equivalent stochastic process θ^t obtained with fair clients only. Theorem 6.1 provides a recurrent form for the difference between these two processes.

Theorem 6.1. *Under the assumptions of Section 6.2.3 and 6.2.4 for the local optimization processes resulting from federated learning with respectively only fair clients and with free-riders, the difference between the aggregation processes of formulas (6.1) and (6.10) takes the following recurrent form:*

$$\tilde{\theta}^t - \theta^t = \sum_{i=0}^{t-1} \left(\epsilon + \frac{M_K}{N}\right)^{t-i-1} f(\theta^i) + \sum_{i=0}^{t-1} \left(\epsilon + \frac{M_K}{N}\right)^{t-i-1} (\tilde{\nu}_i - \nu_i),$$

with $f(\theta^t) = \frac{M_K}{N}\left[\theta^t - \sum_{j\in J} \frac{M_j}{N-M_K}[\eta_j(\theta^t - \theta_j^*) + \theta_j^*]\right]$, $\epsilon = \sum_{j\in J} \frac{M_j}{N}\eta_j$, $\nu_t = \sum_{j\in J} \frac{M_j}{N-M_K}\rho_j\zeta_{j,t}$ *and* $\tilde{\nu}_t = \sum_{j\in J} \frac{M_j}{N}\rho_j\tilde{\zeta}_{j,t}$.

We note that in the special case with no free-riders (i.e. $M_K = 0$), the quantity $\tilde{\theta}^t - \theta^t$ depends on the second term of equation (6.12) only, and represents the comparison between two different realizations of the stochastic process associated to the federated global model. Theorem 6.1 shows that in this case the variance across optimization results is non-zero, and depends on the intrinsic variability of the local optimization processes quantified by the variable ν_t. We also note that in presence of free-riders the convergence to the model obtained with fair clients depends on the relative sample size declared by the free-riders $\frac{M_K}{N}$.

Convergence analysis of plain free-riding

Based on the relationship between the learning processes established in Theorem 6.1, we are now able to prove that federated learning with plain free-riders defined in equation (6.10) converges in expectation to the aggregated model of the fair clients of equation (6.1).

Theorem 6.2 (Plain free-riding)**.** *Assuming FedAvg converges in expectation, and based on the assumption of Theorem 6.1, the following asymptotic properties hold:*

$$\mathbb{E}\left[\tilde{\theta}^t - \theta^t\right] \xrightarrow{t\to+\infty} 0, \tag{6.12}$$

$$\mathrm{Var}\left[\tilde{\theta}^t - \theta^t\right] \xrightarrow{t\to+\infty} \frac{\left[\frac{1}{N^2} + \frac{1}{(N-M_K)^2}\right]\sum_{j\in J}\left(M_j\rho_j\right)^2}{1 - \left(\epsilon + \frac{M_K}{N}\right)^2}. \tag{6.13}$$

As a corollary of Theorem 6.2, in Proof E.1.2 it is shown that the asymptotic variance is strictly increasing with the sample size M_K declared by the free-riders. In practice, the smaller the total number of data points declared by the free-riders, the closer the final aggregation result approaches the model obtained with fair clients only. On the contrary, when the the sample size of the fair clients is negligible with respect to the the

one declared by the free-riders, i.e. $N \simeq M_K$, the variance tends to infinity. This is due to the ratio approaching to 1 in the geometric sum of the second term of equation (6.12). In the limit case when only free-riders participate to federated learning ($J = \emptyset$), we obtain instead the trivial result $\tilde{\theta}^t = \theta^0$ and $\mathrm{Var}\left[\tilde{\theta}^t\right] = 0$. In this case there is no learning throughout the training process. Finally, with no free-riders ($M_K = 0$), we obtain $\mathrm{Var}\left[\tilde{\theta}_1^t - \theta_2^t\right] \xrightarrow{t\to+\infty} \frac{2}{N^2}\frac{1}{1-\epsilon^2}\sum_{j\in J}(M_j\rho_j)^2$, reflecting the variability of the fair aggregation process due to the stochasticity of the local optimization processes.

6.2.5 Disguised free-riding

Plain free-riders can be easily detected by the server, since for each iteration the condition $[\tilde{\theta}_k^{t+1} - \tilde{\theta}^t = 0]$ is true. In what follows, we study improved attack strategies based on the sharing of opportunely disguised parameters, and investigate sufficient conditions on the disguising models to obtain the desired convergence behavior of free-rider attacks.

Additive noise to mimic SGD updates

A disguised free-rider with additive noise generalizes the plain one, and uploads parameters $\tilde{\theta}_k^{t+1} = \tilde{\theta}^t + \varphi_k(t)\epsilon_t$. Here, the perturbation ϵ_t is assumed to be Gaussian white noise, and $\varphi_k(t) > 0$ is a suitable time-varying perturbation compatible with the free-rider attack. As shown in equation (6.8), the parameters uploaded by the fair clients take the general form composed of an expected model corrupted by a stochastic perturbation due to SGD. Free-riders can mimic this update form by adopting a noise structure similar to the one of the fair clients:

$$\varphi_k^2(t) = \frac{\lambda}{S}{\sigma_k^t}^2\frac{1}{2r_k}\left[1 - e^{-2\lambda r_k\frac{EM_k}{S}}\right], \tag{6.14}$$

where r_k and σ_k^t would ideally depend on the (non-existing) free-rider data distribution and thus need to be determined, while M_k is the declared number of samples. Compatibly with the assumptions of constant SGD variance σ_j^2 for the fair clients, we here assume that the free-riders noise is constant and compatible with the SGD form:

$$\varphi_k^2 = \frac{\lambda}{S}\sigma_k^2\frac{1}{2r_k}\left[1 - e^{-2\lambda r_k\frac{EM_k}{S}}\right]. \tag{6.15}$$

The parameters r_k and σ_k affect the noise level and decay of the update, and thus the ability of the free-rider of mimicking a realistic client. These parameters can be ideally estimated by computing a plausible quadratic approximation of the local loss function (Section 6.2.3). While the estimation may require the availability of some form of data for the free-rider,

in Section 6.2.5 we prove that, for any combination of r_k and σ_k, federated learning still converges to the desired aggregated target.

Analogously as for the fair clients, this assumption will be relaxed in Section 6.2.5.

Attacks based on fixed additive stochastic perturbations

In this new setting, we can rewrite the FedAvg aggregation process (6.1) for an attack with a single free-rider with perturbation φ:

$$\tilde{\theta}^{t+1} = \sum_{j \in J} \frac{M_j}{N} \tilde{\theta}_j^{t+1} + \frac{M_K}{N} \tilde{\theta}^t + \frac{M_K}{N} \varphi \epsilon_t. \tag{6.16}$$

Theorem 6.3 extends the results previously obtained for federated learning with plain free-riders to our new case with additive perturbations.

Theorem 6.3 (Single disguised free-rider)**.** *Analogously to Theorem 6.2, the aggregation process under free-riding described in equation (6.16) converges in expectation to the aggregated model of the fair clients of equation (6.1) :*

$$\mathbb{E}\left[\tilde{\theta}^t - \theta^t\right] \xrightarrow{t \to +\infty} 0, \tag{6.17}$$

$$\begin{aligned} \mathrm{Var}\left[\tilde{\theta}^t - \theta^t\right] \xrightarrow{t \to +\infty} & \frac{\left[\frac{1}{N^2} + \frac{1}{(N - M_K)^2}\right] \sum_{j \in J} \left(M_j \rho_j\right)^2}{1 - \left(\epsilon + \frac{M_K}{N}\right)^2} \\ & + \frac{1}{1 - \left(\epsilon + \frac{M_K}{N}\right)^2} \frac{M_K^2}{N^2} \varphi^2. \end{aligned} \tag{6.18}$$

Theorem 6.3 shows that disguised free-riding converges to the final model of federated learning with fair clients, although with a higher variance resulting from the free-rider's perturbations injected at every iteration. The perturbation is proportional to $\frac{M_K}{N}$, the relative number of samples declared by the free-rider.

The extension of this result to the case of multiple free-riders requires to account in equation (6.16) for an attack of the form $\sum_{k \in K} \frac{M_k}{N} \varphi_k \epsilon_{k,t}$, where M_k is the total sample size declared by free-rider k. Corollary 6.1 follows from the linearity of this form.

Corollary 6.1 (Multiple disguised free-riders)**.** *Assuming a constant perturbation factor φ_k for each free-rider k, the asymptotic expectation of Theorem 6.3 still holds, while the variance reduces to*

$$\mathrm{Var}\left[\tilde{\theta}^t - \theta^t\right] \xrightarrow{t\to+\infty} \frac{\left[\frac{1}{N^2} + \frac{1}{(N-M_K)^2}\right]\sum_{j\in J}\left(M_j\rho_j\right)^2}{1-\left(\epsilon+\frac{M_K}{N}\right)^2} + \frac{1}{1-\left(\epsilon+\frac{M_K}{N}\right)^2}\sum_{k\in K}\frac{M_k^2}{N^2}\varphi_k^2. \tag{6.19}$$

Time-varying noise model of fair-clients evolution

To investigate more plausible parameters evolution in federated learning, in this section we relax the assumption made in Section 6.2.3 about the constant noise perturbation of the SGD process across iteration rounds.

We assume here that the standard deviation σ_j^t of SGD decreases at each server iteration t, approaching to zero over iteration rounds: $\sigma_j^t \xrightarrow{t\to+\infty} 0$. This assumption reflects the improvement of the fit of the global model $\tilde{\theta}^t$ to the local datasets over server iterations, and implies that the stochastic process of the local optimization of Section 6.2.3 has noise parameter $\rho_j^t \xrightarrow{t\to+\infty} 0$. We thus hypothesize that, to mimic the behavior of the fair clients, a suitable time-varying perturbation of the free-riders should follow a similar asymptotic behavior: $\varphi_k(t) \xrightarrow{t\to+\infty} 0$. Under these assumptions, Corollary 6.2 shows that the asymptotic variance of model aggregation under free-rider attacks is zero, and that it is thus still possible to retrieve the fair client's model.

Corollary 6.2. *Assuming that fair clients and free-riders evolve according to Section 6.2.3 to 6.2.5, if the conditions $\rho_j^t \xrightarrow{t\to+\infty} 0$ and $\varphi_k(t) \xrightarrow{t\to+\infty} 0$ are met, the aggregation process of federated learning is such that the asymptotic variance of Theorems 6.2 and 6.3 reduce to*

$$\mathrm{Var}\left[\tilde{\theta}^t - \theta^t\right] \xrightarrow{t\to+\infty} 0. \tag{6.20}$$

We assumed in Corollary 6.2 that the SGD noise σ_j^t decreases at each server iteration and eventually converges to 0. In practice, the global model may not fit perfectly the dataset of the different clients $\mathcal{D}_j$ and, after a sufficient number of optimization rounds, may keep oscillating around a local minima. We could therefore assume that $\sigma_j^t \xrightarrow{t\to+\infty} \sigma_j$ leading to $\rho_j^t \xrightarrow{t\to+\infty} \rho_j$. In this case, to mimic the behavior of the fair clients, a suitable time-varying perturbation compatible with the free-rider attacks should converge to a fixed noise level such that $\varphi_k(t) \xrightarrow{t\to+\infty} \varphi_k$. Similarly as for Corollary 6.2, it can be shown that under these

hypothesis federated learning follows the asymptotic behaviors of Theorem 6.2 and 6.3 for respectively plain and disguised free-riders.

6.2.6 FedProx

FedProx includes a regularization term for the local loss functions of the different clients ensuring the proximity between the updated models θ_j^{t+1} and θ^t. This regularization is usually defined as an additional L2 penalty term, and leads to the following form for the local gradient $g_j(\theta_j) \simeq r_j[\theta_j - \theta_j^*] + \mu[\theta_j - \theta^t]$ where μ is a trade-off parameter. Since the considerations in Section 6.2.3 still hold in this setting, we can express the local model contribution for FedProx with a formulation analogous to the one of equation (6.8). Hence, for FedProx, we obtain similar conclusions for Theorem 6.2 and 6.3, as well as for Corollary 6.1 and 6.2, proving that the convergence behavior with free-riders is equivalent to the one obtained with fair clients only, although with a different asymptotic variance (Appendix E.2).

Theorem 6.4. *Assuming convergence in expectation for federated learning with fair clients only, under the assumptions of Theorem 6.1 the asymptotic properties of plain and disguised free-riding of Theorem 6.2, 6.3, and Corollary 6.1, 6.2, still hold with FedProx. In this case we have parameters:*

$$\rho_j{}^2 = \frac{\lambda}{S}\sigma_j{}^2 \frac{1}{2(r_j+\mu)}\left[1 - e^{-2\lambda(r_j+\mu)\frac{EM_j}{S}}\right], \tag{6.21}$$

$$\epsilon = \sum_{j\in J} \frac{M_j}{N}[\gamma_j + \mu\frac{1-\gamma_j}{r_j+\mu}], \tag{6.22}$$

$$\textit{and } \gamma_j = e^{-\lambda(r_j+\mu)\frac{EM_j}{S}}. \tag{6.23}$$

We note that the asymptotic variance is still strictly increasing with the total number of free-riders samples. Moreover, the regularization term monitors the asymptotic variance: a higher regularization leads to a smaller noise parameter ρ_j^2 and to a smaller ϵ, thus decreasing the asymptotic variances of Theorem 6.2, 6.3, and Corollary 6.1, 6.2.

6.3 Experiments

This experimental section focuses on a series of benchmarks for the proposed free-rider attacks. The methods being of general application, the focus here is to empirically

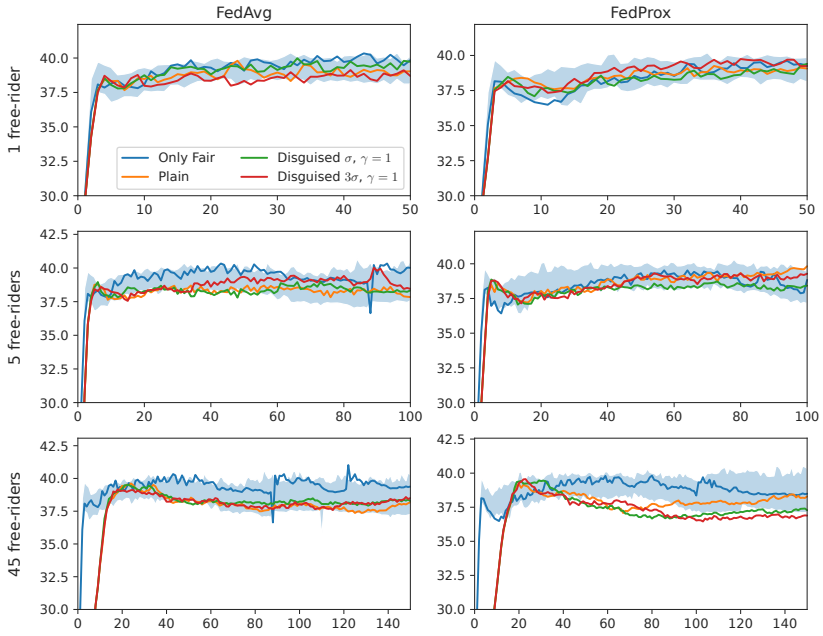

Fig. 6.1.: Plots for Shakespeare and $E = 20$. Accuracy performances for FedAvg and FedProx according to the number of free-riders participating in the learning process: 15% (top), 50% (middle), and 90% (bottom) of the total amount of clients. The shaded blue region indicates the variability of federated learning model with fair clients only, estimated from 30 different training initialization.

demonstrate our theory on diverse experimental setups and model specifications. All code, data and experiments are available at `https://github.com/Accenture/Labs-Federated-Learning/tree/free-rider_attacks`.

6.3.1 Experimental Details

We consider 5 fair clients for each of the following scenarios, investigated in previous works on federated learning (McMahan, Moore, et al., 2017; T. Li, Sahu, Zaheer, Sanjabi, Talwalkar, and Smith, 2018b):

MNIST (classification in iid and non-iid settings). We study a standard classification problem on MNIST (LeCun et al., 1998) and create two benchmarks: an iid dataset (MNIST iid) where we assign 600 training digits and 300 testing digits to each client, and a non-iid dataset (MNIST non-iid), where for each digit we create two shards with 150 training samples and 75 testing samples, and allocate 4 shards for each client. For each scenario, we use a logistic regression predictor.

CIFAR-10(Krizhevsky et al., n.d.) (image classification). The dataset consists of 10 classes of 32x32 images with three RGB channels. There are 50000 training examples and 10000

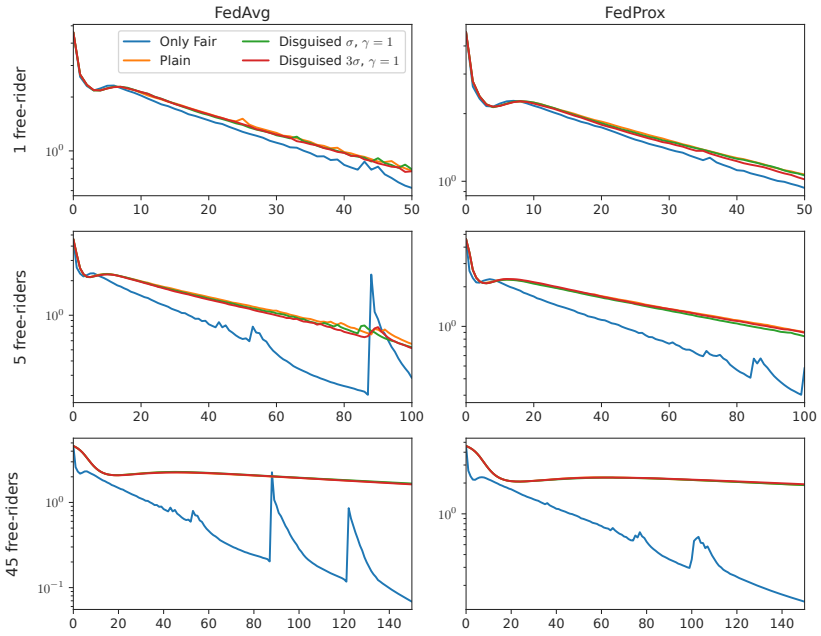

Fig. 6.2.: Plots for Shakespeare and $E = 20$. Loss performances for FedAvg and FedProx according to the number of free-riders participating in the learning process: 15% (top), 50% (middle), and 90% (bottom) of the total amount of clients.

testing examples which we partitioned into 5 clients each containing 10000 training and 2000 testing samples. The model architecture was taken from (McMahan, Moore, et al., 2017) which consists of two convolutional layers and a linear transformation layer to produce logits.

Shakespeare (LSTM prediction). We study a LSTM model for next character prediction on the dataset of *The Complete Works of William Shakespeare* (McMahan, Moore, et al., 2017). We randomly chose 5 clients with more than 3000 samples, and assign 70% of the dataset to training and 30% to testing. Each client has on average 6415.4 samples (± 1835.6) . We use a two-layer LSTM classifier containing 100 hidden units with an 8 dimensional embedding layer. The model takes as an input a sequence of 80 characters, embeds each of the characters into a learned 8-dimensional space and outputs one character per training sample after 2 LSTM layers and a fully connected one.

We train federated models following FedAvg and FedProx aggregation processes. In FedProx, the hyperparameter μ monitoring the regularization is chosen according to the best performing scenario reported in (T. Li, Sahu, Zaheer, Sanjabi, Talwalkar, and Smith, 2018b): $\mu = 1$ for MNIST (iid and non-iid), and $\mu = 0.001$ for Shakespeare. For the free-rider we declare a number of samples equal to the average sample size across fair clients. We test federated learning with 5 and 20 local epochs using SGD optimization with learning rate $\lambda = 0.001$ for MNIST (iid and non-iid), $\lambda = 0.001$ for CIFAR-10, and $\lambda = 0.5$ for

Shakespeare, and batch size of 100. We evaluate the success of the free-rider attacks by quantifying testing accuracy and training loss of the resulting model, as indicators of the effect of the perturbation induced by free-riders on the final model performances. Resulting figures for associated accuracy and loss can be found in Figure 6.1, Figure 6.2 and Appendix E.3.

6.3.2 Free-rider attacks: convergence and performances

In the following experiments, we assume that free-riders do not have any data, which means that they cannot estimate the noise level by computing a plausible quadratic approximation of the local loss function (Section 6.2.5). Therefore, we investigate free-rider attacks taking the simple form $\varphi(t) = \sigma t^{-\gamma}$. The parameter γ is chosen among a panel of testing parameters $\gamma \in \{0.5, 1, 2\}$, while additional experimental material on the influence of γ on the convergence is presented in Appendix E.3. While the optimal tuning of disguised free-rider attacks is out of the scope of this study, in what follows the perturbations parameter σ is defined according to practical hypotheses on the parameters evolution during federated learning. After random initialization at the initial federated learning step, the parameter σ is opportunely estimated to mimic the extent of the distribution of the update $\Delta\tilde{\theta}^0 = \tilde{\theta}^1 - \tilde{\theta}^0$ observed between consecutive rounds of federated learning. We can simply model these increments as a zero-centered univariate Gaussian distribution, and assign the parameter σ to the value of the fitted standard deviation. According to this strategy, the free-rider would return parameters $\tilde{\theta}_k^t$ with perturbations distributed as the ones observed between two consecutive optimization rounds. Figure 6.1, top row, exemplifies the evolution of the models obtained with FedAvg (20 local training epochs) on the Shakespeare dataset with respect to different scenarios: 1) fair clients only, 2) plain free-rider, 3) disguised free-rider with decay parameter $\gamma = 1$, and estimated noise level σ, and 4) disguised free-rider with noise level increased to 3σ. For each scenario, we compare the federated model obtained under free-rider attacks with respect to the equivalent model obtained with the participation of the fair clients only. For this latter setting, to assess the model training variability, we repeated the training 30 times with different parameter initializations. The results show that, independently from the chosen free-riding strategy, the resulting models attains comparable performances with respect to the one of the model obtained with fair clients only (Figure 6.1, top row). Similar results are obtained for the setup with 5 local training epochs and different values of γ, as well as for FedProx with 5 and 20 local epochs (Appendix E.3).

We also investigate the same training setup under the influence of multiple free-riders (Figure 6.1, mid and bottom rows). In particular, we test the scenarios where the free-riders declare respectively 50% and 90% of the total training sample size. In practice, we maintain the same experimental setting composed of 5 fair clients, and we increase the number of

free-riders to respectively 5 and 45, while declaring for each free-rider a sample size equal to the average number of samples of the fair clients. Independently from the magnitude of the perturbation function, the number of free-riders does not seem to affect the performance of the final aggregated model. However, the convergence speed is greatly decreased. Figure 6.2 shows that the convergence in these different settings is not identically affected by the free-riders. When the size of free-riders is moderate, e.g. up to 50% of the total sample size, the convergence speed of the loss is slightly slower than for federated learning with fair clients. The attacks can be still considered successful, as convergence is achieved within the pre-defined iteration budget. However, when the size of free-riders reaches 90%, convergence to the optimum is extremely slow and cannot be achieved anymore in a reasonable amount of iterations. This result is in agreement with our theory, for which the convergence speed inversely proportional to the relative size of the free-riders. Interestingly, we note that the final accuracy obtained in all the scenarios is similar (though a bit slower with 90% of free-riders), and falls within the variability observed in federated learning with fair-clients only (Figure 1). This result is achieved in spite of the incomplete convergence during training. This effect can be explained by observing that this accuracy level is already reached at the early training stages of federated learning with fair clients, while further training does not seem to improve the predictions. This result suggests that, in spite of the very low convergence speed, the averaging process with 90% of free-riders still achieves a reasonable minima compatible with the training path of the fair clients aggregation.

We note that the "peaks" observed in the loss of Figure 2 are common in FL, especially in the considered application when the number of clients is low. It is important to notice that our experiments are performed by using vanilla SGD. As such, the peaks for only fair clients are to be expected in both loss and performances. We also notice that the peaks are smaller for free-riding because of the “regularization” effect of free-riders, which regresses the update towards the global model of the previous iteration.

Analogous results and considerations can be derived from the set of experiments on the remaining datasets, training parameters and FedProx as an aggregation scheme (Appendix E.3).

6.4 Conclusion and discussion

We introduced a theoretical framework for the study of free-riding attacks on model aggregation in federated learning. Based on the proposed methodology, we proved that simple strategies based on returning the global model at each iteration already lead to successful free-rider attacks (plain free-riding), and we investigated more sophisticated disguising

techniques relying on stochastic perturbations of the parameters (disguised free-riding). The convergence of each attack was demonstrated through theoretical developments and experimental results. The threat of free-rider attacks is still under-investigated in machine learning. For example, current defence schemes in federated learning are mainly based on outliers detection mechanisms, to detect malicious attackers providing abnormal updates. These schemes would be therefore unsuccessful in detecting a free-rider update which is, by design, equivalent to the global federated model.

This chapter opens the way to the investigation of optimal disguising and defense strategies for free-rider attacks, beyond the proposed heuristics. Our experiments show that inspection of the client's distribution should be established as a routine practice for the detection of free-rider attacks in federated learning. Further research directions are represented by the improvement of detection at the server level, through better modeling of the heterogeneity of the incoming clients' parameters. This study provides also the theoretical basis for the study of effective free-riding strategies, based on different noise model distributions and perturbation schemes. Finally, in this chapter we relied on a number of hypothesis concerning the evolution of the clients' parameters during federated learning. This choice provides us with a convenient theoretical setup for the formalization of the proposed theory which may be modified in the future, for example, for investigating more complex forms of variability and schemes for parameters aggregation.

Conclusion and Perspectives

7

Contents

7.1 Summary of the Main Contributions **113**
7.2 Perspectives and Future Applications **116**
7.2.1 Fully Decentralized Federated Learning 116
7.2.2 Ensuring Fairness 117
7.2.3 Model Personalization with Federated Learning 118
7.3 Final Remarks **119**

In the first part of this thesis, we studied the robustness and variability of federated learning to heterogeneous dataset and hardware, through our investigation of the impact of clients sampling (Chapter 2 and 3) and delayed updates (Chapter 4) on the convergence speed and guarantees of federated learning. In the second part of this thesis, we investigated the reliability of federated learning in practical applications. We provided a federated unlearning scheme to remove the contribution of a set of clients from a federatively trained model (Chapter 5) and investigated free-rider attacks to federated learning (Chapter 6).

7.1 Summary of the Main Contributions

A General Theory for Client Sampling in Federated Learning

In Chapter 2, we highlighted the asymptotic impact of client sampling on FL. In particular, we showed how the variance and covariance of the clients' stochastic aggregation weights impact FL convergence speed. While our theory holds for any advanced FL sampling scheme, we investigated MD and Uniform sampling from both theoretical and experimental standpoints. We established that when clients have approximately identical importance, Uniform outperforms MD sampling, while MD outperform Uniform sampling otherwise. Yet, in practical scenario with very large number of clients, MD sampling may be unpractical, and Uniform sampling could be preferred due to the more advantageous time complexity.

Main Contributions

- We highlight the asymptotic impact of client sampling on FL with Theorem 2.1, and shows that the convergence speed is inversely proportional to both the sum of the variance of the stochastic aggregation weights, and to their covariance parameter α.
- We established that when clients have approximately identical importance, i.e $p_i = 1/n$, Uniform outperforms MD sampling, due to the larger impact of the covariance term for the latter scheme.
- We showed that our theory encompasses advanced FL sampling schemes, such as the one proposed in Fraboni, Vidal, Kameni, et al. (2021), and W. Chen et al. (2020).

Clustered Sampling: Low-Variance and Improved Representativity for Clients Selection in Federated Learning

In Chapter 3, we introduced clustered sampling, a novel client selection scheme in FL generalizing MD sampling, the current scheme from the state-of-the-art (Chapter 2). Consistently with Chapter 2, we proved the correctness of clustered sampling and proposed two clustering methods implementing aggregation based on the clients number of samples or model similarity. Both algorithms provide smaller weight variance for the clients aggregation process leading to better client representativity. Consistently, clustered sampling is experimentally shown to have faster and smoother convergence in heterogeneous dataset.

Main Contributions

- We introduced clustered sampling, a novel client selection scheme in FL generalizing MD sampling, the current scheme from the state-of-the-art.
- We proved the correctness of clustered sampling and proposed two clustering methods implementing aggregation based on the clients number of samples, in Algorithm 2, or model similarity, in Algorithm 3.
- We showed that clustered sampling has faster and smoother convergence in heterogeneous dataset than MD sampling.

A General Theory for Federated Optimization with Asynchronous and Heterogeneous Clients Updates

In Chapter 4, we generalized the expression of FEDAVG aggregation scheme by introducing stochastic aggregation weights to account for asynchronous client updates. We proved the convergence of FL schemes satisfying this formalization, e.g. synchronous and asynchronous FEDAVG, FEDFIX, and FEDBUFF. Finally, we gave the aggregation weights close-form to ensure the convergence of any FL optimization scheme to the optimum of the federated problem.

Main Contributions

- With equation (4.6), we generalized the expression of FEDAVG aggregation scheme by introducing stochastic aggregation weights $\omega_i(n)$ to account for asynchronous client updates, and proved the convergence of FL schemes satisfying this equation with Theorem 4.1.

- We showed that existing federated optimization procedures satisfy aggregation scheme (4.6) including synchronous FL, asynchronous FL, FEDFIX, FEDBUFF, and client sampling.

- We proposed FEDFIX, an FL algorithm where the server, after a fixed amount of time, creates the new global model with the contribution of all the participating clients, proved its convergence with our theoretical framework, and experimentally demonstrated its improvement over synchronous and asynchronous FEDAVG in all the considered scenarios.

Sequential Informed Federated Unlearning: Efficient and Provable Client Unlearning in Federated Optimization

In Chapter 5, we introduced informed federated unlearning (IFU), a novel federated unlearning scheme to unlearn a client's contribution from a model trained with federated learning with statistical unlearning guarantees. We extended the theory of IFU to account for the practical scenario of sequential unlearning (SIFU), where the server receives a series of forgetting request of one or more clients. We proved that SIFU can unlearn a series of forgetting requests while satisfying our unlearning guarantees, and demonstrated the effectiveness of our methods on a variety of tasks and dataset.

Main Contributions

- We introduced informed federated unlearning (IFU), a novel federated unlearning scheme to unlearn, with statistical guarantees, a client's contribution from a model trained with FL.

- We extended the theory of IFU to account for the practical scenario of sequential unlearning (SIFU), where the server receives a series of forgetting request of one or more clients.

- We provided a new theory for the server to bound the clients contribution in FL without asking clients for any additional computation and communication.

Free-rider Attacks on Model Aggregation in Federated Learning

In Chapter 6, we introduced a theoretical framework for the study of free-riding attacks on model aggregation in federated learning. Based on the proposed methodology, we proved that simple strategies based on returning the global model at each iteration already lead to successful free-rider attacks, and we investigated more sophisticated disguising techniques relying on stochastic perturbations of the parameters. The convergence of each attack was demonstrated through theoretical developments and experimental results.

Main Contributions

- We provided the theoretical basis for the study of effective free-riding strategies, based on different noise model distributions, perturbation schemes and the Ornstein-Uhlenbeck process.
- We proved that simple strategies based on returning the global model at each iteration already lead to successful free-rider attacks (plain free-riding), and we investigated more sophisticated disguising techniques relying on stochastic perturbations of the parameters (disguised free-riding).
- We demonstrated the convergence and success of plain and disguised free-riding attacks through theoretical developments and experimental results.

7.2 Perspectives and Future Applications

7.2.1 Fully Decentralized Federated Learning

In this thesis, we considered that the server orchestrates the federated optimization by receiving the clients' contributions and creating the new global model with them. The server communication capabilities are finite and can be constraining when training with large cohort of clients. In this thesis, we addressed this aspect with the analysis of FL convergence speed when only a subset of clients participates for aggregation. We showed the impact of only considering a subset of clients for participation in Chapter 2, and even provided a faster selection method in Chapter 3. In addition, we introduced FedFix in Chapter 4. By allowing delayed client contribution, the server only considers the received contributions for aggregation every fixed period of time, which alleviates the server communication workload. However, mitigating the server computation workload is not the only server constraint in FL. Indeed, by orchestrating the FL training procedure, the server is also a central point of failure, and the cost of guaranteeing a reliable and powerful central server may not always be possible (Vanhaesebrouck et al., 2017). Hence, the interest of fully decentralized learning for FL training without a server.

The key idea behind fully decentralized learning is to replace communication with the server by peer-to-peer communication between individual clients. The communication topology is represented as a connected graph in which nodes are the clients and an edge indicates a communication channel between two clients. In fully decentralized algorithms, a round corresponds to each client performing a local update and exchanging information with their neighbors. Note that there is no longer a global state of the model as in standard federated learning, but the process can be designed such that all local models converge to the desired global solution, i.e., the individual models gradually reach consensus. While multi-agent optimization has a long history in the control community, fully decentralized variants of SGD and other optimization algorithms have recently been considered in machine learning both for improved scalability in datacenters (Assran et al., 2019) as well as for decentralized networks of devices (Colin et al., 2016; H. Tang et al., 2018). It is worth noting that even in the decentralized setting outlined above, a certain degree of trust is needed to set up the learning task either through a central authority or a consensus scheme. Indeed, to perform training, clients need to know their training instruction and hyperparameters used for training.

While fully decentralized FL removes the server constraints, the ones related to the clients' computation and communication capabilities remain. Clients might still need to use their computation and communication capabilities for other tasks than FL training. The clients heterogeneous hardware forces fast clients to wait for the slow ones to communicate with them. As such, adapting the frameworks developed to account for client sampling, Chapter 2, and asynchronicity, Chapter 4, could be coupled with fully decentralized FL for faster federated optimization without the need of a server orchestrating the training. However, a fairness investigation is needed to identify the scenarios in which fully decentralized FL converges to a stationary point of its federated problem, and otherwise how to modify accordingly the training procedure for fully decentralized FL.

7.2.2 Ensuring Fairness

In this thesis, we investigated the impact of a federated optimization scheme on the fairness of the resulting trained model. Especially, we verified that the trained model is a stationary point of the federated problem (1.1) and not of a surrogate one favoring some clients. Another popular fairness approach guarantees instead that the trained model has identical performances on every client (T. Li, Sanjabi, et al., 2019; T. Li, Hu, et al., 2021). However, another source of unfairness has yet to be investigated in the federated learning literature. To this date, no federated optimization scheme guarantees that every class is treated equally in the trained model. Indeed, often the clients' data samples belong to a client-specific distribution, which makes their dataset prone to lack information regarding some sensitive

attributes or to over-represent some features or classes. With federated learning, practitioners obtain a better estimate of an unbiased sample of the data that match the distribution of the population. Hence, performing federated learning is a first step towards removing this source of unfairness in the trained model but is not sufficient to guarantee that every class is identically represented in the trained model.

In the classical centralized machine learning setting, numerous improvements have been made to train this kind of fair classifiers, by introducing for example constrained optimization, post-shifting approaches, and distributionally-robust optimization (Hardt et al., 2016; Zafar et al., 2017; Hashimoto et al., 2018). However, it remains to be proven if these methods, which have demonstrated their effectiveness for improving fairness in centralized training, could be used for federated learning. Indeed, with federated learning, the clients share no information regarding their data, which makes designing a federated optimization scheme correcting the data imbalance challenging. Hence, developing a framework accounting for this kind of bias while respecting the private protocol of FL is needed to provide fair, private and useful optimization.

7.2.3 Model Personalization with Federated Learning

Thorough this thesis, we consider that the clients' data samples are generated with a client-specific data distribution and consider the convergence guarantees of FL on the federated optimization problem (1.1). When local datasets are small and the data is iid, the model trained with FL outperforms the one obtained solely with local training, which justifies the use of FL for optimization in real-world applications (T. Yang et al., 2018; M. Chen et al., 2019). On the other hand, with non-IID distributions, local models can perform better locally than ones trained with a federated optimization scheme. Hence, further investigation is needed to identify a priori the learning scenarios where federated learning outperforms local optimization, and quantify this improvement.

To provide valid incentives for clients to participate to a federated learning experience, researchers are currently focusing in the problem of "personalization" in federated learning. This approach extends the classical federated optimization routine to account for local data specificities, in order to provide client-specific predictions at inference time (M. Zhang et al., 2020; Qinbin Li et al., 2021). These techniques are shown to be particularly efficient when faced with non-iid data and may outperform even the best possible shared global model. For example, the server can cluster clients according to their specificities, such as geographic location or characteristics of the client's device, before running federated learning on each cluster (Mansour et al., 2020; Sattler et al., 2019). The server can also consider that similar clients are associates with the same machine learning task and apply methods adapted

from multi-task learning to FL (Smith et al., 2017; Marfoq et al., 2021). A client can also personalize the global model by fine-tuning on its local dataset. The development of such algorithms is an important open problem for federated optimization. Especially when dealing with heterogeneous and decentralized datasets. However, fairness must be kept in mind while developing such algorithms, thus requiring to define the optimization routine compatible with the solution of the collaborative optimization problem originally defined by formula (1.1).

7.3 Final Remarks

The main bottleneck for the development of data-driven approaches in biomedical applications is represented by the need for large datasets to achieve robust and reliable models. Federated Learning could bridge that gap by enabling its participants to train a model without sharing or exposing their data. With this thesis, we introduced new methods to improve the robustness and reliability of federated learning to heterogeneous conditions and thus improve its adoption in real-world applications.

Bibliography

Acar, Durmus Alp Emre, Yue Zhao, Ramon Matas, et al. (2021). "Federated Learning Based on Dynamic Regularization". In: *International Conference on Learning Representations* (cit. on pp. 21, 68, 201).

Agarwal, Alekh and John C Duchi (2011). "Distributed Delayed Stochastic Optimization". In: *Advances in Neural Information Processing Systems*. Ed. by J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger. Vol. 24. Curran Associates, Inc. (cit. on p. 66).

Assran, Mahmoud, Nicolas Loizou, Nicolas Ballas, and Mike Rabbat (2019). "Stochastic gradient push for distributed deep learning". In: *International Conference on Machine Learning*. PMLR, pp. 344–353 (cit. on p. 117).

Ateniese, Giuseppe, Luigi V. Mancini, Angelo Spognardi, et al. (2015). "Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers". In: *International Journal of Security and Networks* 10.3, pp. 137–150. arXiv: 1306.4447 (cit. on p. 96).

Avdiukhin, Dmitrii and Shiva Kasiviswanathan (2021). "Federated Learning under Arbitrary Communication Patterns". In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, pp. 425–435 (cit. on p. 68).

Basu, Debraj, Deepesh Data, Can Karakus, and Suhas Diggavi (2019). "Qsparse-local-SGD: Distributed SGD with Quantization, Sparsification and Local Computations". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, et al. Vol. 32. Curran Associates, Inc. (cit. on pp. 21, 68, 201).

Bennett, Casey C, Thomas W Doub, and Rebecca Selove (2012). "EHRs connect research and practice: Where predictive modeling, artificial intelligence, and clinical decision support intersect". In: *Health Policy and Technology* 1.2, pp. 105–114 (cit. on p. 1).

Bertsekas, Dimitri P. and John N. Tsitsiklis (1989). *Parallel and Distributed Computation: Numerical Methods*. USA: Prentice-Hall, Inc. (cit. on p. 46).

Beutel, Daniel J., Taner Topal, Akhil Mathur, et al. (2020). "Flower: A Friendly Federated Learning Research Framework". In: *CoRR* abs/2007.14390. arXiv: 2007.14390 (cit. on p. 4).

Bhagoji, Arjun Nitin, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo (2019). "Analyzing federated learning through an adversarial lens". In: *36th International Conference on Machine Learning, ICML 2019* 2019-June, pp. 1012–1021. arXiv: 1811.12470 (cit. on pp. 6, 96).

Bottou, Léon, Frank E. Curtis, and Jorge Nocedal (2016). "Optimization Methods for Large-Scale Machine Learning". In: *SIAM Review* 60 (cit. on p. 192).

Bottou, Léon, Frank E. Curtis, and Jorge Nocedal (2018). “Optimization methods for large-scale machine learning”. In: *SIAM Review* 60.2, pp. 223–311. arXiv: 1606.04838 (cit. on p. 31).

Bourtoule, Lucas, Varun Chandrasekaran, Christopher A Choquette-Choo, et al. (2021). “Machine unlearning”. In: *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, pp. 141–159 (cit. on pp. 8, 78).

Brisimi, Theodora, Ruidi Chen, Theofanie Mela, et al. (2018). “Federated learning of predictive models from federated Electronic Health Records”. In: *International Journal of Medical Informatics* 112 (cit. on p. 96).

Caldas, Sebastian, Sai Meher Karthik Duddu, Peter Wu, et al. (2018). “LEAF: A Benchmark for Federated Settings”. In: NeurIPS, pp. 1–9. arXiv: 1812.01097 (cit. on pp. 24, 70).

Cao, Yinzhi and Junfeng Yang (2015). “Towards Making Systems Forget with Machine Unlearning”. In: *2015 IEEE Symposium on Security and Privacy*, pp. 463–480 (cit. on pp. 8, 78).

Carlini, Nicholas, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song (2019a). “The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks”. In: *28th USENIX Security Symposium (USENIX Security 19)*. Santa Clara, CA: USENIX Association, pp. 267–284 (cit. on p. 96).

– (2019b). “The secret sharer: Evaluating and testing unintended memorization in neural networks”. In: *28th USENIX Security Symposium (USENIX Security 19)*, pp. 267–284 (cit. on p. 6).

Chen, Mingqing, Rajiv Mathews, Tom Ouyang, and Françoise Beaufays (2019). “Federated learning of out-of-vocabulary words”. In: *arXiv preprint arXiv:1903.10635* (cit. on p. 118).

Chen, Wenlin, Samuel Horvath, and Peter Richtarik (2020). “Optimal Client Sampling for Federated Learning”. In: *Workshop in NeurIPS: Privacy Preserving Machine Learning* (cit. on pp. 7, 14, 17, 19, 25, 28, 32, 114, 142).

Chen, Xiangyi, Steven Z. Wu, and Mingyi Hong (2020). “Understanding Gradient Clipping in Private SGD: A Geometric Perspective”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., pp. 13773–13782 (cit. on p. 81).

Chen, Yujing, Yue Ning, Martin Slawski, and Huzefa Rangwala (2020). “Asynchronous online federated learning for edge devices with non-iid data”. In: *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, pp. 15–24 (cit. on p. 7).

Cho, Yae Jee, Jianyu Wang, and Gauri Joshi (2020). “Client selection in federated learning: Convergence analysis and power-of-choice selection strategies”. In: *arXiv preprint arXiv:2010.01243* (cit. on pp. 5, 7, 14, 17, 47, 193).

Colin, Igor, Aurélien Bellet, Joseph Salmon, and Stéphan Clémençon (2016). “Gossip dual averaging for decentralized optimization of pairwise functions”. In: *International Conference on Machine Learning*. PMLR, pp. 1388–1396 (cit. on p. 117).

Cramer, Ronald, Ivan Bjerre Damgård, et al. (2015). *Secure multiparty computation*. Cambridge University Press (cit. on p. 6).

Csiba, Dominik and Peter Richtárik (2018). “Importance Sampling for Minibatches”. In: *Journal of Machine Learning Research* 19.27, pp. 1–21 (cit. on p. 19).

Cyffers, Edwige and Aurélien Bellet (2022). "Privacy Amplification by Decentralization". In: *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*. Ed. by Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera. Vol. 151. Proceedings of Machine Learning Research. PMLR, pp. 5334–5353 (cit. on p. 6).

Damgård, Ivan, Valerio Pastro, Nigel Smart, and Sarah Zakarias (2012). "Multiparty computation from somewhat homomorphic encryption". In: *Annual Cryptology Conference*. Springer, pp. 643–662 (cit. on p. 6).

De Sa, Christopher M, Ce Zhang, Kunle Olukotun, Christopher Ré, and Christopher Ré (2015). "Taming the Wild: A Unified Analysis of Hogwild-Style Algorithms". In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett. Vol. 28. Curran Associates, Inc. (cit. on p. 47).

Dwork, Cynthia (2008). "Differential privacy: A survey of results". In: *International conference on theory and applications of models of computation*. Springer, pp. 1–19 (cit. on p. 6).

Dwork, Cynthia and Aaron Roth (2014). "The Algorithmic Foundations of Differential Privacy". In: *Found. Trends Theor. Comput. Sci.* 9.3–4, pp. 211–407 (cit. on pp. 79, 81, 85, 91).

Ebadollahi, Shahram, Jimeng Sun, David Gotz, et al. (2010). "Predicting patient's trajectory of physiological data using temporal trends in similar patients: a system for near-term prognostics". In: *AMIA annual symposium proceedings*. Vol. 2010. American Medical Informatics Association, p. 192 (cit. on p. 1).

Fraboni, Yann, Richard Vidal, Laetitia Kameni, and Marco Lorenzi (2022a). "A General Theory for Client Sampling in Federated Learning". In: *International Workshop on Trustworthy Federated Learning in conjunction with IJCAI 2022 (FL-IJCAI'22)* (cit. on pp. 10, 13, 50, 54, 73, 192, 201).

– (2022b). "A General Theory for Federated Optimization with Asynchronous and Heterogeneous Clients Updates". In: *arXiv preprint arXiv:2206.10189* (cit. on pp. 10, 46).

– (2021). "Clustered Sampling: Low-Variance and Improved Representativity for Clients Selection in Federated Learning". In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, pp. 3407–3416 (cit. on pp. 10, 14, 17–19, 21, 23–25, 27, 85, 114, 140, 156, 158).

– (2022c). "Sequential Informed Federated Unlearning: Efficient and Provable Client Unlearning in Federated Optimization". In: *arXiv preprint arXiv:2211.11656* (cit. on pp. 10, 78).

Fraboni, Yann, Richard Vidal, and Marco Lorenzi (2021). "Free-rider Attacks on Model Aggregation in Federated Learning". In: *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*. Ed. by Arindam Banerjee and Kenji Fukumizu. Vol. 130. Proceedings of Machine Learning Research. PMLR, pp. 1846–1854 (cit. on pp. 10, 95).

Fredrikson, Matt, Somesh Jha, and Thomas Ristenpart (2015). "Model Inversion Attacks That Exploit Confidence Information and Basic Countermeasures". In: *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. CCS '15. Denver, Colorado, USA: Association for Computing Machinery, pp. 1322–1333 (cit. on pp. 6, 79, 82, 96).

Fredrikson, Matthew, Eric Lantz, Somesh Jha, et al. (2014). "Privacy in Pharmacogenetics: An End-to-End Case Study of Personalized Warfarin Dosing". In: *23rd USENIX Security Symposium (USENIX Security 14)*. San Diego, CA: USENIX Association, pp. 17–32 (cit. on p. 96).

Fung, Clement, Chris J. M. Yoon, and Ivan Beschastnikh (2020). "The Limitations of Federated Learning in Sybil Settings". In: *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*. San Sebastian: USENIX Association, pp. 301–316 (cit. on p. 96).

Ginart, Antonio, Melody Guan, Gregory Valiant, and James Y Zou (2019). "Making AI Forget You: Data Deletion in Machine Learning". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, et al. Vol. 32. Curran Associates, Inc. (cit. on p. 80).

Golatkar, Aditya, Alessandro Achille, Avinash Ravichandran, Marzia Polito, and Stefano Soatto (2021). "Mixed-Privacy Forgetting in Deep Networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 792–801 (cit. on pp. 79, 81).

Golatkar, Aditya, Alessandro Achille, and Stefano Soatto (2020a). "Eternal Sunshine of the Spotless Net: Selective Forgetting in Deep Networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 78, 81).

– (2020b). *Forgetting Outside the Box: Scrubbing Deep Networks of Information Accessible from Input-Output Observations*. arXiv: 2003.02960 [cs.LG] (cit. on pp. 78, 81).

Gu, Xinran, Kaixuan Huang, Jingzhao Zhang, and Longbo Huang (2021). "Fast federated learning in the presence of arbitrary device unavailability". In: *Advances in Neural Information Processing Systems* 34, pp. 12052–12064 (cit. on p. 69).

Guo, Chuan, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten (2020). "Certified Data Removal from Machine Learning Models". In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, pp. 3832–3842 (cit. on pp. 78, 79, 81).

Gupta, Varun, Christopher Jung, Seth Neel, et al. (2021). "Adaptive Machine Unlearning". In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato, A. Beygelzimer, K. Nguyen, et al. Vol. 34. Curran Associates, Inc., pp. 16319–16330 (cit. on pp. 79, 81).

Haddadpour, Farzin, Mohammad Mahdi Kamani, Mehrdad Mahdavi, and Viveck Cadambe (2019). "Trading Redundancy for Communication: Speeding up Distributed SGD for Non-convex Optimization". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, pp. 2545–2554 (cit. on pp. 68, 201).

Haddadpour, Farzin, Mohammad Mahdi Kamani, Mehrdad Mahdavi, and Viveck R. Cadambe (2019). "Local SGD with periodic averaging: Tighter analysis and adaptive synchronization". In: *Advances in Neural Information Processing Systems* 32.2 (cit. on p. 31).

Haddadpour, Farzin and Mehrdad Mahdavi (2019). *On the Convergence of Local Descent Methods in Federated Learning*. arXiv: 1910.14425 [cs.LG] (cit. on pp. 14, 18, 19).

Halimi, Anisa, Swanand Kadhe, Ambrish Rawat, and Nathalie Baracaldo (2022). "Federated Unlearning: How to Efficiently Erase a Client in FL?" In: *arXiv preprint arXiv:2207.05521* (cit. on pp. 79, 82).

Harding, Elizabeth Liz, Jarno J Vanto, Reece Clark, L Hannah Ji, and Sara C Ainsworth (2019). "Understanding the scope and impact of the California Consumer Privacy Act of 2018". In: *Journal of Data Protection & Privacy* 2.3, pp. 234–253 (cit. on pp. 7, 78).

Hardt, Moritz, Eric Price, and Nati Srebro (2016). "Equality of opportunity in supervised learning". In: *Advances in neural information processing systems* 29 (cit. on p. 118).

Harry Hsu, Tzu Ming, Hang Qi, and Matthew Brown (2019). "Measuring the effects of non-identical data distribution for federated visual classification". In: *arXiv*. arXiv: 1909.06335 (cit. on pp. 42, 69, 90, 158).

Hashimoto, Tatsunori, Megha Srivastava, Hongseok Namkoong, and Percy Liang (2018). "Fairness without demographics in repeated loss minimization". In: *International Conference on Machine Learning*. PMLR, pp. 1929–1938 (cit. on p. 118).

He, Li, Qi Meng, Wei Chen, Zhi Ming Ma, and Tie Yan Liu (2018). "Differential equations for modeling asynchronous algorithms". In: *IJCAI International Joint Conference on Artificial Intelligence* 2018-July.1, pp. 2220–2226. arXiv: 1805.02991 (cit. on pp. 97, 99, 100).

Hitaj, Briland, Giuseppe Ateniese, and Fernando Perez-Cruz (2017). "Deep Models under the GAN: Information leakage from collaborative deep learning". In: *Proceedings of the ACM Conference on Computer and Communications Security*, pp. 603–618. arXiv: 1702.07464 (cit. on pp. 6, 96).

Izzo, Zachary, Mary Anne Smart, Kamalika Chaudhuri, and James Zou (2021). "Approximate Data Deletion from Machine Learning Models". In: *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*. Ed. by Arindam Banerjee and Kenji Fukumizu. Vol. 130. Proceedings of Machine Learning Research. PMLR, pp. 2008–2016 (cit. on pp. 78, 81).

Jensen, Peter B, Lars J Jensen, and Søren Brunak (2012). "Mining electronic health records: towards better research applications and clinical care". In: *Nature Reviews Genetics* 13.6, pp. 395–405 (cit. on p. 1).

Jeon, Joohyung, Soohyun Park, Minseok Choi, et al. (2020). "Optimal User Selection for High-Performance and Stabilized Energy-Efficient Federated Learning Platforms". In: *Electronics* 9.9 (cit. on p. 17).

Kairouz, Peter, H. Brendan McMahan, Brendan Avent, et al. (2019). "Advances and Open Problems in Federated Learning". In: pp. 1–105. arXiv: 1912.04977 (cit. on pp. 2–4, 46).

Karimireddy, Sai Praneeth, Satyen Kale, Mehryar Mohri, et al. (2020). "SCAFFOLD: Stochastic Controlled Averaging for Federated Learning". In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, pp. 5132–5143 (cit. on pp. 14, 16, 18, 19, 28, 31, 50).

Khaled, Ahmed, Konstantin Mishchenko, and Peter Richtarik (2020a). "Tighter Theory for Local SGD on Identical and Heterogeneous Data". In: *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. Ed. by Silvia Chiappa and Roberto Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, pp. 4519–4529 (cit. on pp. 14, 51, 52, 54, 64, 185).

– (2020b). "Tighter Theory for Local SGD on Identical and Heterogeneous Data". In: *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. Ed. by Silvia Chiappa and Roberto Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, pp. 4519–4529 (cit. on p. 28).

Kingma, Diederik P and Jimmy Ba (2014). "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (cit. on pp. 5, 21, 68, 201).

Koloskova, Anastasia, Sebastian Stich, and Martin Jaggi (2019). "Decentralized Stochastic Optimization and Gossip Algorithms with Compressed Communication". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, pp. 3478–3487 (cit. on p. 47).

Koloskova*, Anastasia, Tao Lin*, Sebastian U Stich, and Martin Jaggi (2020). "Decentralized Deep Learning with Arbitrary Communication Compression". In: *International Conference on Learning Representations* (cit. on p. 68).

Kononenko, Igor (2001). "Machine learning for medical diagnosis: history, state of the art and perspective". In: *Artificial Intelligence in medicine* 23.1, pp. 89–109 (cit. on p. 1).

Krizhevsky, Alex, Vinod Nair, and Geoffrey Hinton (n.d.). "CIFAR-10 (Canadian Institute for Advanced Research)". In: () (cit. on pp. 25, 42, 69, 90, 108, 158).

Lecun, Y., L. Bottou, Y. Bengio, and P. Haffner (1998). "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11, pp. 2278–2324 (cit. on pp. 69, 90).

LeCun, Yann, Leon Bottou, Yoshua Bengio, and Patrick Ha (1998). "LeNet". In: *Proceedings of the IEEE* November, pp. 1–46. arXiv: 1102.0183 (cit. on pp. 40, 108).

Li, Bo, Yining Wang, Aarti Singh, and Yevgeniy Vorobeychik (2016). "Data poisoning attacks on factorization-based collaborative filtering". In: *Advances in Neural Information Processing Systems* Nips, pp. 1893–1901. arXiv: 1608.08182 (cit. on pp. 6, 96).

Li, Kim-Hung (1994). "Reservoir-Sampling Algorithms of Time Complexity O(n(1 + Log(N/n)))". In: *ACM Trans. Math. Softw.* 20.4, pp. 481–493 (cit. on p. 23).

Li, Qianxiao, Cheng Tai, and E. Weinan (2017). "Stochastic modified equations and adaptive stochastic gradient algorithms". In: *34th International Conference on Machine Learning, ICML 2017* 5, pp. 3306–3340. arXiv: 1511.06251 (cit. on pp. 97, 99, 100).

Li, Qinbin, Bingsheng He, and Dawn Song (2021). "Model-contrastive federated learning". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10713–10722 (cit. on p. 118).

Li, Tian, Shengyuan Hu, Ahmad Beirami, and Virginia Smith (18–24 Jul 2021). "Ditto: Fair and Robust Federated Learning Through Personalization". In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, pp. 6357–6368 (cit. on pp. 5, 117).

Li, Tian, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith (2020). "Federated Learning: Challenges, Methods, and Future Directions". In: *IEEE Signal Processing Magazine* 37.3, pp. 50–60 (cit. on p. 2).

Li, Tian, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith (2018a). "Federated Optimization in Heterogeneous Networks". In: *Proceedings of the 1 st Adaptive & Multitask Learning Workshop, Long Beach, California, 2019*, pp. 1–28. arXiv: 1812.06127 (cit. on pp. 7, 14, 17–19, 21, 28, 30, 31, 68, 70, 173, 192, 200, 201).

– (2018b). “Federated Optimization in Heterogeneous Networks”. In: *Proceedings of the 1 st Adaptive & Multitask Learning Workshop, Long Beach, California, 2019*, pp. 1–28. arXiv: 1812.06127 (cit. on pp. 97, 98, 108, 109).

Li, Tian, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smithy (2019). “FedDANE: A Federated Newton-Type Method”. In: *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, pp. 1227–1231 (cit. on pp. 21, 68, 200).

Li, Tian, Maziar Sanjabi, Ahmad Beirami, and Virginia Smith (2020). “Fair Resource Allocation in Federated Learning”. In: *International Conference on Learning Representations* (cit. on pp. 14, 18).

– (2019). “Fair resource allocation in federated learning”. In: *arXiv preprint arXiv:1905.10497* (cit. on pp. 5, 117).

Li, Xiang, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang (2020). “On the Convergence of FedAvg on Non-IID Data”. In: *International Conference on Learning Representations* (cit. on pp. 2, 7, 14, 17, 18, 28, 31–33, 51, 52, 59, 78, 82, 85, 192, 201).

Li, Xiaoyu and Francesco Orabona (2019). “On the Convergence of Stochastic Gradient Descent with Adaptive Stepsizes”. In: *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*. Ed. by Kamalika Chaudhuri and Masashi Sugiyama. Vol. 89. Proceedings of Machine Learning Research. PMLR, pp. 983–992 (cit. on pp. 21, 68, 201).

Lian, Xiangru, Yijun Huang, Yuncheng Li, and Ji Liu (2015). “Asynchronous Parallel Stochastic Gradient for Nonconvex Optimization”. In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett. Vol. 28. Curran Associates, Inc. (cit. on p. 66).

Lian, Xiangru, Wei Zhang, Ce Zhang, and Ji Liu (2018). “Asynchronous decentralized parallel stochastic gradient descent”. In: *International Conference on Machine Learning*. PMLR, pp. 3043–3052 (cit. on p. 68).

Lin, Jierui, Min Du, and Jian Liu (2019). “Free-riders in Federated Learning: Attacks and Defenses”. In: arXiv: 1911.12560 (cit. on p. 97).

Lin, Tao, Sebastian U. Stich, Kumar Kshitij Patel, and Martin Jaggi (2020). “Don’t Use Large Mini-batches, Use Local SGD”. In: *International Conference on Learning Representations* (cit. on p. 28).

Lindell, Yehida (2005). “Secure multiparty computation for privacy preserving data mining”. In: *Encyclopedia of Data Warehousing and Mining*. IGI global, pp. 1005–1009 (cit. on p. 6).

Liu, Gaoyang, Xiaoqiang Ma, Yang Yang, Chen Wang, and Jiangchuan Liu (2021). “FedEraser: Enabling Efficient Client-Level Data Removal from Federated Learning Models”. In: *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*, pp. 1–10 (cit. on pp. 79, 82, 91).

Liu, Ziwei, Ping Luo, Xiaogang Wang, and Xiaoou Tang (2015). “Deep Learning Face Attributes in the Wild”. In: *Proceedings of International Conference on Computer Vision (ICCV)* (cit. on p. 90).

Lu, Xiaofeng, Yuying Liao, Pietro Lio, and Pan Hui (2020). "Privacy-preserving asynchronous federated learning mechanism for edge network computing". In: *IEEE Access* 8, pp. 48970–48981 (cit. on p. 7).

Lyu, Lingjuan, Han Yu, and Qiang Yang (2020). "Threats to federated learning: A survey". In: *arXiv preprint arXiv:2003.02133* (cit. on p. 6).

Mahadevan, Ananth and Michael Mathioudakis (2021a). "Certifiable Machine Unlearning for Linear Models". In: *CoRR* abs/2106.15093. arXiv: 2106.15093 (cit. on p. 81).

– (2021b). "Certifiable machine unlearning for linear models". In: *arXiv preprint arXiv:2106.15093* (cit. on p. 81).

Mandt, Stephan, Matthew D. Hof Fman, and David M. Blei (2017). "Stochastic gradient descent as approximate Bayesian inference". In: *Journal of Machine Learning Research* 18, pp. 1–35. arXiv: 1704.04289 (cit. on pp. 97, 99, 100).

Mansour, Yishay, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh (2020). "Three approaches for personalization with applications to federated learning". In: *arXiv preprint arXiv:2002.10619* (cit. on p. 118).

Marfoq, Othmane, Giovanni Neglia, Aurélien Bellet, Laetitia Kameni, and Richard Vidal (2021). "Federated Multi-Task Learning under a Mixture of Distributions". In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan. Vol. 34. Curran Associates, Inc., pp. 15434–15447 (cit. on p. 119).

McMahan, H. Brendan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas (2017). "Communication-Efficient Learning of Deep Networks from Decentralized Data". In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. Ed. by Aarti Singh and Jerry Zhu. Vol. 54. Proceedings of Machine Learning Research. Fort Lauderdale, FL, USA: PMLR, pp. 1273–1282 (cit. on pp. 2, 3, 14, 16, 17, 24, 28, 30, 42, 46, 69, 78, 82, 97, 98, 108, 109).

McMahan, H. Brendan, Daniel Ramage, Kunal Talwar, and Li Zhang (2017). "Learning differentially private recurrent language models". In: *arXiv preprint arXiv:1710.06963* (cit. on p. 6).

Nedić, A., D. P. Bertsekas, and V. S. Borkar (2001). "Distributed asynchronous incremental subgradient methods". English (US). In: *Studies in Computational Mathematics* 8.C, pp. 381–407 (cit. on p. 46).

Neel, Seth, Aaron Roth, and Saeed Sharifi-Malvajerdi (2021). "Descent-to-Delete: Gradient-Based Methods for Machine Unlearning". In: *Proceedings of the 32nd International Conference on Algorithmic Learning Theory*. Ed. by Vitaly Feldman, Katrina Ligett, and Sivan Sabato. Vol. 132. Proceedings of Machine Learning Research. PMLR, pp. 931–962 (cit. on pp. 79, 81, 91).

Nguyen, John, Kshitiz Malik, Hongyuan Zhan, et al. (2021). "Federated Learning with Buffered Asynchronous Aggregation". In: *ArXiv* abs/2106.06639 (cit. on pp. 9, 68, 72).

Nguyen, Lam, PHUONG HA NGUYEN, Marten van Dijk, et al. (2018). "SGD and Hogwild! Convergence Without the Bounded Gradients Assumption". In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, pp. 3750–3758 (cit. on p. 47).

Nishio, T. and R. Yonetani (2019). “Client Selection for Federated Learning with Heterogeneous Resources in Mobile Edge”. In: *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, pp. 1–7 (cit. on pp. 17, 28, 32, 193).

Noble, Maxence, Aurélien Bellet, and Aymeric Dieuleveut (2022). “Differentially Private Federated Learning on Heterogeneous Data”. In: *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*. Ed. by Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera. Vol. 151. Proceedings of Machine Learning Research. PMLR, pp. 10110–10145 (cit. on p. 6).

Orvieto, Antonio and Aurelien Lucchi (2018). “Continuous-time Models for Stochastic Optimization Algorithms”. In: NeurIPS. arXiv: 1810.02565 (cit. on pp. 97, 99, 100).

Petersen, Ronald Carl, Paul S Aisen, Laurel A Beckett, et al. (2010). “Alzheimer’s disease neuroimaging initiative (ADNI): clinical characterization”. In: *Neurology* 74.3, pp. 201–209 (cit. on p. 1).

Ramakrishnan, Naren, David Hanauer, and Benjamin Keller (2010). “Mining electronic health records”. In: *Computer* 43.10, pp. 77–81 (cit. on p. 1).

Reddi, Sashank J., Zachary Charles, Manzil Zaheer, et al. (2021). “Adaptive Federated Optimization”. In: *International Conference on Learning Representations* (cit. on pp. 14, 16, 18, 50).

Reisizadeh, Amirhossein, Aryan Mokhtari, Hamed Hassani, Ali Jadbabaie, and Ramtin Pedarsani (2020). “FedPAQ: A Communication-Efficient Federated Learning Method with Periodic Averaging and Quantization”. In: *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. Ed. by Silvia Chiappa and Roberto Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, pp. 2021–2031 (cit. on pp. 21, 68, 201).

Rizk, Elsa, Stefan Vlaski, and Ali Sayed (2020). “Dynamic Federated Learning”. In: (cit. on pp. 14, 18).

Roque, Francisco S, Peter B Jensen, Henriette Schmock, et al. (2011). “Using electronic patient records to discover disease correlations and stratify patient cohorts”. In: *PLoS computational biology* 7.8, e1002141 (cit. on p. 1).

Sattler, Felix, Klaus-Robert Müller, and Wojciech Samek (2019). “Clustered Federated Learning: Model-Agnostic Distributed Multi-Task Optimization under Privacy Constraints”. In: pp. 1–16. arXiv: 1910.01991 (cit. on pp. 39, 118).

– (2021). “Clustered Federated Learning: Model-Agnostic Distributed Multitask Optimization Under Privacy Constraints”. In: *IEEE Transactions on Neural Networks and Learning Systems* 32.8, pp. 3710–3722 (cit. on p. 85).

Savage, Neil (2012). “Better medicine through machine learning”. In: *Communications of the ACM* 55.1, pp. 17–19 (cit. on p. 1).

Sheller, Micah J, Brandon Edwards, G Anthony Reina, et al. (2020). “Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data”. In: *Scientific reports* 10.1, pp. 1–12 (cit. on p. 1).

Shen, Shiqi, Shruti Tople, and Prateek Saxena (2016). “AUROR: Defending against poisoning attacks in collaborative deep learning systems”. In: *ACM International Conference Proceeding Series*. Vol. 5-9-Decemb, pp. 508–519 (cit. on pp. 6, 96).

Silva, Santiago, Andre Altmann, Boris Gutman, and Marco Lorenzi (2020). "Fed-BioMed: A General Open-Source Frontend Framework for Federated Learning in Healthcare". In: *Domain Adaptation and Representation Transfer, and Distributed and Collaborative Learning: Second MICCAI Workshop, DART 2020, and First MICCAI Workshop, DCL 2020, Held in Conjunction with MICCAI 2020, Lima, Peru, October 4–8, 2020, Proceedings*. Lima, Peru: Springer-Verlag, pp. 201–210 (cit. on pp. 4, 7).

Silva, Santiago, Boris A Gutman, Eduardo Romero, et al. (2019). "Federated learning in distributed medical databases: Meta-analysis of large-scale subcortical brain data". In: *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*. IEEE, pp. 270–274 (cit. on p. 96).

Smith, Virginia, Chao Kai Chiang, Maziar Sanjabi, and Ameet Talwalkar (2017). "Federated multi-task learning". In: *Advances in Neural Information Processing Systems*. Vol. 2017-Decem. Nips, pp. 4425–4435. arXiv: 1705.10467 (cit. on p. 119).

Sommer, David Marco, Liwei Song, Sameer Wagh, and Prateek Mittal (2020). "Towards Probabilistic Verification of Machine Unlearning". In: *CoRR* abs/2003.04247. arXiv: 2003.04247 (cit. on p. 92).

Stich, Sebastian, Amirkeivan Mohtashami, and Martin Jaggi (2021). "Critical Parameters for Scalable Distributed Learning with Large Batches and Asynchronous Updates". In: *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*. Ed. by Arindam Banerjee and Kenji Fukumizu. Vol. 130. Proceedings of Machine Learning Research. PMLR, pp. 4042–4050 (cit. on p. 66).

Stich, Sebastian U and Sai Praneeth Karimireddy (2020). "The Error-Feedback framework: SGD with Delayed Gradients". In: *Journal of Machine Learning Research* 21.237, pp. 1–36 (cit. on pp. 47, 66).

Stich, Sebastian U. (2019). "Local SGD Converges Fast and Communicates Little". In: *International Conference on Learning Representations* (cit. on pp. 28, 59).

Sudlow, Cathie, John Gallacher, Naomi Allen, et al. (2015). "UK biobank: an open access resource for identifying the causes of a wide range of complex diseases of middle and old age". In: *PLoS medicine* 12.3, e1001779 (cit. on p. 1).

Tang, Daniel (2019). "Efficient algorithms for modifying and sampling from a categorical distribution". In: *CoRR* abs/1906.11700. arXiv: 1906.11700 (cit. on p. 23).

Tang, Hanlin, Xiangru Lian, Ming Yan, Ce Zhang, and Ji Liu (2018). "D2: Decentralized training over decentralized data". In: *International Conference on Machine Learning*. PMLR, pp. 4848–4856 (cit. on p. 117).

Uhlenbeck, G. E. and L. S. Ornstein (1930). "On the Theory of the Brownian Motion". In: *Phys. Rev.* 36 (5), pp. 823–841 (cit. on p. 101).

Vanhaesebrouck, Paul, Aurélien Bellet, and Marc Tommasi (2017). "Decentralized collaborative learning of personalized models over networks". In: *Artificial Intelligence and Statistics*. PMLR, pp. 509–517 (cit. on p. 116).

Voigt, Paul and Axel Von dem Bussche (2017). "The eu general data protection regulation (gdpr)". In: *A Practical Guide, 1st Ed., Cham: Springer International Publishing* 10.3152676, pp. 10–5555 (cit. on pp. 7, 78).

Wang, Hongyi, Scott Sievert, Shengchao Liu, et al. (2018). "ATOMO: Communication-efficient Learning via Atomic Sparsification". In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio, H. Wallach, H. Larochelle, et al. Vol. 31. Curran Associates, Inc. (cit. on pp. 21, 68, 201).

Wang, J., A. K. Sahu, Z. Yang, G. Joshi, and S. Kar (2019). "MATCHA: Speeding Up Decentralized SGD via Matching Decomposition Sampling". In: *2019 Sixth Indian Control Conference (ICC)*, pp. 299–300 (cit. on pp. 19, 31).

Wang, Jianyu and Gauri Joshi (2018). "Cooperative SGD: A unified Framework for the Design and Analysis of Communication-Efficient SGD Algorithms". In: arXiv: 1808.07576 (cit. on pp. 14, 18).

Wang, Jianyu, Qinghua Liu, Hao Liang, Gauri Joshi, and H. Vincent Poor (2020). "Tackling the Objective Inconsistency Problem in Heterogeneous Federated Optimization". In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (cit. on pp. 2, 5, 14, 17–19, 21, 28, 29, 31, 32, 47, 52, 56, 57, 64, 68, 78, 82, 85, 144, 146, 147, 154, 155, 163).

Wang, Jianyu, Vinayak Tantia, Nicolas Ballas, and Michael Rabbat (2020). "SlowMo: Improving Communication-Efficient Distributed SGD with Slow Momentum". In: *International Conference on Learning Representations* (cit. on pp. 16, 50).

Wang, Junxiao, Song Guo, Xin Xie, and Heng Qi (2021). *Federated Unlearning via Class-Discriminative Pruning*. arXiv: 2110.11794 [cs.CV] (cit. on pp. 79, 82).

Wang, Renjie, Wei Pan, Lei Jin, et al. (2019). "Artificial intelligence in reproductive medicine". In: *Reproduction (Cambridge, England)* 158.4, R139 (cit. on p. 1).

Wang, Shiqiang, Tiffany Tuor, Theodoros Salonidis, et al. (2019). "Adaptive Federated Learning in Resource Constrained Edge Computing Systems". In: *IEEE Journal on Selected Areas in Communications* 37.6, pp. 1205–1221. arXiv: 1804.05271 (cit. on p. 19).

Wang, Zhibo, Mengkai Song, Zhifei Zhang, et al. (2019). "Beyond Inferring Class Representatives: User-Level Privacy Leakage from Federated Learning". In: *Proceedings - IEEE INFOCOM* 2019-April, pp. 2512–2520. arXiv: 1812.00535 (cit. on pp. 6, 96).

Ward, Joe H. (1963). "Hierarchical Grouping to Optimize an Objective Function". In: *Journal of the American Statistical Association* 58.301, pp. 236–244 (cit. on p. 40).

Ward, Rachel, Xiaoxia Wu, and Leon Bottou (2019). "AdaGrad Stepsizes: Sharp Convergence Over Nonconvex Landscapes". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, pp. 6677–6686 (cit. on pp. 21, 68, 201).

Wei, Kang, Jun Li, Ming Ding, et al. (2020). "Federated learning with differential privacy: Algorithms and performance analysis". In: *IEEE Transactions on Information Forensics and Security* 15, pp. 3454–3469 (cit. on p. 6).

Woodworth, Blake, Kumar Kshitij Patel, Sebastian Stich, et al. (2020). "Is Local SGD Better than Minibatch SGD?" In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, pp. 10334–10343 (cit. on p. 28).

Wu, Chen, Sencun Zhu, and Prasenjit Mitra (2022). "Federated Unlearning with Knowledge Distillation". In: *arXiv preprint arXiv:2201.09441* (cit. on pp. 79, 82).

Wu, Wentai, Ligang He, Weiwei Lin, et al. (2020). "SAFA: A semi-asynchronous protocol for fast federated learning with low overhead". In: *IEEE Transactions on Computers* 70.5, pp. 655–668 (cit. on p. 7).

Xiao, Han, Kashif Rasul, and Roland Vollgraf (2017). "Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms". In: *CoRR* abs/1708.07747. arXiv: 1708.07747 (cit. on p. 90).

Xie, Chulin, Keli Huang, Pin-Yu Chen, and Bo Li (2019). "DBA: Distributed Backdoor Attacks against Federated Learning". In: *International Conference on Learning Representations* (cit. on pp. 6, 96).

Xu, Chenhao, Youyang Qu, Yong Xiang, and Longxiang Gao (2021). *Asynchronous Federated Learning on Heterogeneous Devices: A Survey*. arXiv: 2109.04269 [cs.DC] (cit. on pp. 47, 64).

Xu, Jie, Benjamin S Glicksberg, Chang Su, et al. (2021). "Federated learning for healthcare informatics". In: *Journal of Healthcare Informatics Research* 5, pp. 1–19 (cit. on p. 2).

Yang, Haibo, Xin Zhang, Prashant Khanduri, and Jia Liu (2022). "Anarchic federated learning". In: *International Conference on Machine Learning*. PMLR, pp. 25331–25363 (cit. on p. 69).

Yang, Qiang, Yang Liu, Tianjian Chen, and Yongxin Tong (2019). "Federated Machine Learning: Concept and Applications". In: *ACM Trans. Intell. Syst. Technol.* 10.2 (cit. on p. 46).

Yang, Timothy, Galen Andrew, Hubert Eichner, et al. (2018). "Applied federated learning: Improving google keyboard query suggestions". In: *arXiv preprint arXiv:1812.02903* (cit. on p. 118).

Yin, Dong, Yudong Chen, Kannan Ramchandran, and Peter Bartlett (2018). "Byzantine-robust distributed learning: Towards optimal statistical rates". In: *35th International Conference on Machine Learning, ICML 2018* 13, pp. 8947–8956. arXiv: 1803.01498 (cit. on pp. 6, 96).

Yu, Hao, Rong Jin, and Sen Yang (2019). "On the Linear Speedup Analysis of Communication Efficient Momentum SGD for Distributed Non-Convex Optimization". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, pp. 7184–7193 (cit. on pp. 68, 201).

Yu, Hao, Sen Yang, and Shenghuo Zhu (2019). "Parallel Restarted SGD with Faster Convergence and Less Communication: Demystifying Why Model Averaging Works for Deep Learning". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 33.01, pp. 5693–5700 (cit. on pp. 28, 68, 201).

Zafar, Muhammad Bilal, Isabel Valera, Manuel Gomez Rogriguez, and Krishna P Gummadi (2017). "Fairness constraints: Mechanisms for fair classification". In: *Artificial intelligence and statistics*. PMLR, pp. 962–970 (cit. on p. 118).

Zantedeschi, Valentina, Aurélien Bellet, and Marc Tommasi (2020). "Fully Decentralized Joint Learning of Personalized Models and Collaboration Graphs". In: *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. Ed. by Silvia Chiappa and Roberto Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, pp. 864–874 (cit. on p. 6).

Zhang, Michael, Karan Sapra, Sanja Fidler, Serena Yeung, and Jose M Alvarez (2020). "Personalized federated learning with first order model optimization". In: *arXiv preprint arXiv:2012.08565* (cit. on p. 118).

Zhao, Peilin and Tong Zhang (2015). "Stochastic Optimization with Importance Sampling for Regularized Loss Minimization". In: *Proceedings of the 32nd International Conference on Machine Learning*. Ed. by Francis Bach and David Blei. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, pp. 1–9 (cit. on p. 19).

Zinkevich, Martin, John Langford, and Alex Smola (2009). "Slow Learners are Fast". In: *Advances in Neural Information Processing Systems*. Ed. by Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta. Vol. 22. Curran Associates, Inc. (cit. on pp. 46, 66).

Appendix of Chapter 2

A

A.1 Client Sampling Schemes Calculus

In this section, we calculate for MD, Uniform, Poisson, and Binomial sampling the respective aggregation weight variance $\mathrm{Var}\left[\omega_i(S_t)\right]$, the covariance parameter α such that $\mathrm{Cov}\left[\omega_i(S_t))\right]\omega_j(S_t) = -\alpha p_i p_j$, and the variance of the sum of weights $\mathrm{Var}\left[\sum_{i=1}^n \omega_i(S_t)\right]$. We also propose statistics for the parameter N, i.e. the amount of clients the server communicates with at an iteration:

$$N = \sum_{i=1}^{n} \mathbb{I}(i \in S_t). \tag{A.1}$$

A.1.1 Property A.1

Proposition A.1. *For any client sampling, we have* $0 \leq \alpha \leq 1$ *and*

$$\mathrm{Var}\left[\sum_{i=1}^{n} \omega_i(S_t)\right] = \sum_{i=1}^{n} \mathrm{Var}\left[\omega_i(S_t)\right] - \alpha\left[1 - \sum_{i=1}^{n} p_i^2\right]. \tag{A.2}$$

Proof. **Covariance parameter**

$$\mathrm{Cov}\left[\omega_i(S_t)\right]\omega_j(S_t) = \mathbb{E}\left[\omega_i(S_t)\omega_j(S_t)\right] - p_i p_j \geq -p_i p_j. \tag{A.3}$$

Hence, we have $\alpha \leq 1$.

Aggregation Weights Sum

$$\begin{aligned}
\mathrm{Var}\left[\sum_{i=1}^{n} \omega_i(S_t)\right] &= \sum_{i=1}^{n} \mathrm{Var}\left[\omega_i(S_t)\right] + \sum_{i,j\neq i} \mathrm{Cov}\left[\omega_i(S_t)\right]\omega_j(S_t) && \text{(A.4)}\\
&= \sum_{i=1}^{n} \mathrm{Var}\left[\omega_i(S_t)\right] - \alpha \sum_{i,j\neq i} p_i p_j && \text{(A.5)}\\
&= \sum_{i=1}^{n} \left[\mathrm{Var}\left[\omega_i(S_t)\right] - \alpha p_i(1-p_i)\right] && \text{(A.6)}\\
&= \sum_{i=1}^{n} \mathrm{Var}\left[\omega_i(S_t)\right] - \alpha\left[1 - \sum_{i=1}^{n} p_i^2\right], && \text{(A.7)}
\end{aligned}$$

where we use $\sum_{i=1}^n p_i = 1$, equation (2.1), for the third and fourth equality.

Re-expressing α. Using equation (A.6), we get

$$\mathrm{Var}\left[\sum_{i=1}^{n} \omega_i(S_t)\right] = \sum_{i=1}^{n} \mathrm{Var}\left[\omega_i(S_t)\right] - \alpha\left[1 - \sum_{i=1}^{n} p_i^2\right], \tag{A.8}$$

which, with reordering, gives

$$\alpha = \frac{\sum_{i=1}^{n} \mathrm{Var}\left[\omega_i(S_t)\right] - \mathrm{Var}\left[\sum_{i=1}^{n} \omega_i(S_t)\right]}{1 - \sum_{i=1}^{n} p_i^2}. \tag{A.9}$$

□

A.1.2 No sampling scheme

When every client participate at an optimization round, we have $\omega_i(S_t) = p_i$ which gives $\mathrm{Var}_{S_t}\left[\omega_i(S_t)\right] = 0$, $\alpha = 0$, and $N = n$.

A.1.3 MD sampling

We recall equation (2.5),

$$\omega_i(S_t) = \frac{1}{m} \sum_{k=1}^{m} \mathbb{I}(l_k = i), \tag{A.10}$$

which gives

$$\mathbb{E}\left[\omega_i(S_t)\omega_j(S_t)\right] = \frac{1}{m^2} \sum_{k,l \neq k} \mathbb{E}\left[\mathbb{I}(l_k = i)\mathbb{I}(l_l = j)\right] + \frac{1}{m^2} \sum_{k=1}^{m} \mathbb{E}\left[\mathbb{I}(l_k = i)\mathbb{I}(l_k = j)\right] \tag{A.11}$$

$$= \frac{1}{m^2} \sum_{k,l \neq k} p_i p_j + \frac{1}{m^2} \sum_{k=1}^{m} \mathbb{E}\left[\mathbb{I}(l_k = i)\mathbb{I}(l_k = j)\right] \tag{A.12}$$

$$= \frac{m-1}{m} p_i p_j + \frac{1}{m} \mathbb{E}\left[\mathbb{I}(l = i)\mathbb{I}(l = j)\right] \tag{A.13}$$

Variance($i = j$). We get $\mathbb{E}\left[\mathbb{I}(l = i)\mathbb{I}(l = j)\right] = \mathbb{E}\left[\mathbb{I}(l = i)\right] = p_i$, which gives:

$$\mathrm{Var}\left[\omega_i(S_t)\right] = -\frac{1}{m} p_i^2 + \frac{1}{m} p_i \tag{A.14}$$

Covariance($i \neq j$). We get $\mathbb{E}\left[\mathbb{I}(l = i)\mathbb{I}(l = j)\right] = 0$, which gives:

$$\mathrm{Cov}\left[\omega_i(S_t)\right] \omega_j(S_t) = -\frac{1}{m} p_i p_j, \tag{A.15}$$

and by definition we get

$$\alpha = \frac{1}{m} \tag{A.16}$$

Aggregation Weights Sum. Using equation (A.14) and (A.16) with Property A.1 , we get

$$\text{Var}\left[\sum_{i=1}^{n}\omega_i(S_t)\right] = 0. \tag{A.17}$$

Amount of clients. Considering that $p(i \in S_t) = 1 - p(i \notin S_t) = 1 - (1-p_i)^m$, we get:

$$\mathbb{E}[N] = \sum_{i=1}^{n}\mathbb{P}(i \in S_t) = n - \sum_{i=1}^{n}(1-p_i)^m \leq m \tag{A.18}$$

A.1.4 Uniform Sampling

We recall equation (2.6),

$$\omega_i(S_t) = \mathbb{I}(i \in S_t)\frac{n}{m}p_i. \tag{A.19}$$

Variance. We first calculate the probability for a client to be sampled, i.e.

$$\mathbb{P}(i \in S_t) = 1 - \mathbb{P}(i \notin S_t) = 1 - \frac{n-1}{n}...\frac{n-m}{n-m+1} = 1 - \frac{n-m}{n} = \frac{m}{n}. \tag{A.20}$$

Using equation (A.20), we have

$$\text{Var}_{S_t}[\omega_i(S_t)] = \left[\frac{n}{m}p_i\right]^2 \text{Var}[\mathbb{I}(i \in S_t)] = \frac{n^2}{m^2}\frac{m}{n}(1-\frac{m}{n})p_i^2 = (\frac{n}{m}-1)p_i^2 \tag{A.21}$$

Covariance. We have

$$\begin{aligned}\mathbb{P}(\{i,j\} \in S_t) &= \mathbb{P}(i \in S_t) + \mathbb{P}(j \in S_t) - \mathbb{P}(i \cup j \in S_t) &\text{(A.22)}\\ &= \mathbb{P}(i \in S_t) + \mathbb{P}(j \in S_t) - (1 - \mathbb{P}(\{i,j\} \notin S_t)), &\text{(A.23)}\end{aligned}$$

and

$$\mathbb{P}(\{i,j\} \notin S_t) = \frac{n-2}{n}...\frac{n-m-1}{n-m+1} = \frac{(n-m)(n-m-1)}{n(n-1)}. \tag{A.24}$$

Substituting equation (A.20) and (A.24) in equation (A.23) gives

$$\begin{aligned}\mathbb{P}(\{i,j\} \in S_t) &= 2\frac{m}{n} - 1 + \frac{(n-m)(n-m-1)}{n(n-1)} &\text{(A.25)}\\ &= \frac{1}{n(n-1)}\left[2m(n-1) - n(n-1) + (n-m)(n-m-1)\right] &\text{(A.26)}\\ &= \frac{m(m-1)}{n(n-1)}. &\text{(A.27)}\end{aligned}$$

Hence, we can express the aggregation weights covariance as

$$\text{Cov}\left[\omega_i(S_t)\right]\omega_j(S_t) = \frac{n^2}{m^2}\frac{m(m-1)}{n(n-1)}p_j p_k - p_j p_k, \tag{A.28}$$

which gives

$$\alpha = \frac{n-m}{m(n-1)}. \tag{A.29}$$

Aggregation Weights Sum. Combining equation (A.21) and (A.29) with Property A.1 gives

$$\text{Var}\left[\sum_{i=1}^{n}\omega_i(S_t)\right] = \sum_{i=1}^{n}\left[\frac{n}{m}-1\right]p_i^2 - \frac{n-m}{m(n-1)}\sum_{i=1}^{n}p_i(1-p_i) = \frac{n-m}{m(n-1)}\left[n\sum_{i=1}^{n}p_i^2 - 1\right], \tag{A.30}$$

where we retrieve $\text{Var}\left[\sum_{i=1}^{n}\omega_i(S_t)\right] = 0$ for identical client importance, i.e. $\sum_{i=1}^{n}p_i^2 = \frac{1}{n}$.

Amount of Clients. $N = m$.

A.1.5 Poisson Binomial Distribution

Clients are sampled according to a Bernoulli with a probability proportional to their importance p_i, i.e.

$$\omega_i(S_t) = \frac{1}{m}\mathbb{B}(mp_i). \tag{A.31}$$

Hence, only $m \geq p_{max}^{-1}$ can be sampled and we retrieve $\mathbb{E}\left[\omega_i(S_t)\right] = \frac{1}{m}mp_i = p_i$.

Variance.

$$\text{Var}_{S_t}\left[\omega_i(S_t)\right] = \frac{1}{m^2}mp_i(1-mp_i) = \frac{1}{m}p_i(1-mp_i) \tag{A.32}$$

Covariance. Due to the independence of each stochastic weight, we also get:

$$\text{Cov}\left[\omega_i(S_t)\right]\omega_j(S_t) = 0 \tag{A.33}$$

Aggregation Weights Sum. Using Property A.1 we obtain

$$\text{Var}\left[\sum_{i=1}^{n}\omega_i(S_t)\right] = \frac{1}{m} - \sum_{i=1}^{n}p_i^2. \tag{A.34}$$

Amount of Clients.

$$\mathbb{E}[N] = m \text{ and } \operatorname{Var}[N] = m - m^2 \sum_{i=1}^{n} p_i^2. \tag{A.35}$$

A.1.6 Binomial Distribution

Clients are sampled according to a Bernoulli with identical sampling probability, i.e.

$$\omega_i(S_t) = \frac{n}{m}\mathbb{B}(\frac{m}{n})p_i. \tag{A.36}$$

Hence, we retrieve $\mathbb{E}[\omega_i(S_t)] = \frac{n}{m}\frac{m}{n}p_i = p_i$.

Variance.

$$\operatorname{Var}_{S_t}[\omega_i(S_t)] = \frac{n^2}{m^2}\frac{m}{n}(1 - \frac{m}{n})p_i^2 = \frac{n-m}{m}p_i^2. \tag{A.37}$$

Covariance. Due to the independence of each stochastic weight, we have:

$$\operatorname{Cov}[\omega_i(S_t)]\,\omega_j(S_t) = 0. \tag{A.38}$$

Aggregation Weights Sum. Using Property A.1 gives

$$\operatorname{Var}\left[\sum_{i=1}^{n}\omega_i(S_t)\right] = \frac{n-m}{m}\sum_{i=1}^{n}p_i^2. \tag{A.39}$$

Amount of Clients.

$$\mathbb{E}[N] = m \text{ and } \operatorname{Var}[N] = m - \frac{m^2}{n}. \tag{A.40}$$

A.1.7 Clustered Sampling

Clustered sampling (Fraboni, Vidal, Kameni, et al., 2021) is a generalization of MD sampling where instead of sampling m clients from the same distributions, m clients are sampled from m different distributions $\{W_k\}_{k=1}^m$ each of them privileging a different subset of clients. We denote by $r_{k,i}$ the probability of client i to be sampled in distribution k. To satisfy Definition 2.1, the original work (Fraboni, Vidal, Kameni, et al., 2021) provides the conditions:

$$\forall k \in \{1, ..., m\},\ \sum_{i=1}^{n} r_{k,i} = 1 \text{ and } \forall i \in \{1, ..., n\},\ \sum_{k=1}^{m} r_{k,i} = mp_i. \tag{A.41}$$

The clients aggregation weights remain identical to the one of MD sampling, i.e.

$$\omega_i(S_{Cl}) = \frac{1}{m}\sum_{k=1}^{K}\mathbb{I}(l_k = i), \tag{A.42}$$

where $\mathbb{I}(l_k = i)$ are still independently distributed but not identically.

We have

$$\mathbb{E}\left[\omega_i(S_t)\omega_j(S_t)\right] = \frac{1}{m^2}\sum_{k,l\neq k}\mathbb{E}\left[\mathbb{I}(l_k = i)\mathbb{I}(l_l = j)\right] + \frac{1}{m^2}\sum_{k=1}^{m}\mathbb{E}\left[\mathbb{I}(l_k = i)\mathbb{I}(l_k = j)\right] \tag{A.43}$$

$$= \frac{1}{m^2}\sum_{k,l\neq k} r_{k,i}r_{l,j} + \frac{1}{m^2}\sum_{k=1}^{m}\mathbb{E}\left[\mathbb{I}(l_k = i)\mathbb{I}(l_k = j)\right] \tag{A.44}$$

$$= p_i p_j - \frac{1}{m^2}\sum_{k=1}^{m} r_{k,i}r_{k,j} + \frac{1}{m^2}\sum_{k=1}^{m}\mathbb{E}\left[\mathbb{I}(l_k = i)\mathbb{I}(l_k = j)\right], \tag{A.45}$$

where we retrieve equation (A.13) when $r_{k,i} = p_i$.

Variance ($i = j$). We get $\mathbb{E}\left[\mathbb{I}(l_k = i)\mathbb{I}(l_k = j)\right] = \mathbb{E}\left[\mathbb{I}(l_k = i)\right] = r_{k,i}$, which gives:

$$\mathrm{Var}\left[\omega_i(S_{Cl})\right] = \frac{1}{m}p_i - \frac{1}{m^2}\sum_{k=1}^{m} r_{k,i}^2 \leq \mathrm{Var}\left[\omega_i(S_{MD})\right], \tag{A.46}$$

where the inequality comes from using the Cauchy-Schwartz inequality with equality if and only if all the m distributions are identical, i.e. $r_{k,i} = p_i$.

Covariance ($i \neq j$). We get $\mathbb{E}\left[\mathbb{I}(l_k = i)\mathbb{I}(l_k = j)\right] = 0$, which gives:

$$\mathrm{Cov}\left[\omega_i(S_{Cl})\right]\omega_j(S_{Cl}) = -\frac{1}{m^2}\sum_{k=1}^{m} r_{k,i}r_{k,j} \leq \mathrm{Cov}\left[\omega_i(S_{MD})\right]\omega_j(S_{MD}), \tag{A.47}$$

where the inequality comes from using the Cauchy-Schwartz inequality with equality if and only if all the m distributions are identical, i.e. $r_{k,i} = p_i$.

Aggregation Weights Sum

$$\mathrm{Var}\left[\sum_{i=1}^{n}\omega_i(S_{Cl})\right] = 0. \tag{A.48}$$

Algorithm 9 Federated Learning based on equation (2.3)

The server sends to the n clients the learning parameters (K, η_l, B).
for $t = 0$ **to** $T - 1$ **do**
  Sample a set of clients S_t and get their aggregation weights $d_i(t)$.
  Send to clients in S_t the current global model $\boldsymbol{\theta}^t$.
  Receive each sampled client contributions $c_i(t) = \boldsymbol{\theta}_i^{t+1} - \boldsymbol{\theta}^t$.
  Creates the new global model $\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t + \eta_g \sum_{i=1}^n d_i(t) c_i(t)$.
end for

A.1.8 Optimal Sampling

With optimal sampling (W. Chen et al., 2020), clients are sampled according to a Bernoulli distribution with probability q_i, i.e.

$$\omega_i(S_t) = \frac{p_i}{q_i} \mathbb{B}(q_i). \tag{A.49}$$

Hence, we retrieve $\mathbb{E}\left[\omega_i(S_t)\right] = \frac{p_i}{q_i} q_i = p_i$.

Variance.

$$\mathrm{Var}_{S_t}\left[\omega_i(S_t)\right] = \frac{1 - q_i}{q_i} p_i^2. \tag{A.50}$$

Covariance. Due to the independence of each stochastic weight, we have:

$$\mathrm{Cov}\left[\omega_i(S_t)\right] \omega_j(S_t) = 0. \tag{A.51}$$

Aggregation Weights Sum. Using Property A.1 gives

$$\mathrm{Var}\left[\sum_{i=1}^n \omega_i(S_t)\right] = \sum_{i=1}^n \frac{1 - q_i}{q_i} p_i^2. \tag{A.52}$$

Amount of Clients.

$$\mathbb{E}\left[N\right] = \sum_{i=1}^n q_i \text{ and } \mathrm{Var}\left[N\right] = \sum_{i=1}^n q_i(1 - q_i). \tag{A.53}$$

A.2 FL Convergence

In Table A.1, we provide the definition of the different notations used in this work. We also propose in Algorithm 9 the pseudo-code for FedAvg with aggregation scheme (2.3).

Tab. A.1.: Common Notation Summary.

Symbol	Description
n	Number of clients.
K	Number of local SGD.
η_l	Local/Client learning rate.
η_g	Global/Server learning rate.
$\tilde{\eta}$	Effective learning rate, $\tilde{\eta} = \eta_l \eta_g$.
$\boldsymbol{\theta}^t$	Global model at server iteration t.
$\boldsymbol{\theta}^*$	Optimum of the federated loss function, equation (2.1).
$\boldsymbol{\theta}_i^{t+1}$	Local update of client i on model θ^t.
$\boldsymbol{y}_{i,k}^t$	Local model of client i after k SGD ($y_{i,K}^t = \theta_i^{t+1}$ and $y_{i,0}^t = \theta^t$).
p_i	Importance of client i in the federated loss function, equation (2.1).
m	Number of sampled clients .
S_t	Set of participating clients considered at iteration t.
$\omega_i(S_t)$	Aggregation weight for client i given S_t.
α	Covariance parameter.
γ_i	cf Section 2.3
$\mathbb{E}_t[\cdot]$	Expected value conditioned on θ^t.
$\mathcal{L}(\cdot)$	Federated loss function, equation 2.1
$\mathcal{L}_i(\cdot)$	Local loss function of client i.
$g_i(\cdot)$	SGD. We have $\mathbb{E}_{\xi_i}[g_i(\cdot)] = \nabla \mathcal{L}_i(\cdot)$ with Assumption 2.4.
ξ_i	Random batch of samples from client i of size B.
L	Lipschitz smoothness parameter, Assumption 2.2.
σ^2	Bound on the variance of the stochastic gradients, Assumption 2.4.
β, κ	Assumption 2.3 parameters on the clients gradient bounded dissimilarity.

Our work is based on the one of Jianyu Wang, Q. Liu, et al. (2020). We use the developed theoretical framework they proposed to prove Theorem 2.1. The focus of our work (and Theorem 2.1) is on FEDAVG. Yet, the proof developed in this section, similarly to the one of Jianyu Wang, Q. Liu, et al. (2020), expresses a_i in such a way they can account for a wide-range of regularization method on FEDAVG, or optimizers different from Vanilla SGD. This proof can easily be extended to account for different amount of local work from the clients (Jianyu Wang, Q. Liu, et al., 2020).

Before developing the proof of Theorem 2.1 in Section A.2.5, we introduce the notation we use in Section A.2.1, some useful lemmas in Section A.2.2 and Theorem A.1 generalizing Theorem 2.1 in Section A.2.3.

A.2.1 Notations

We define by $\boldsymbol{y}_{i,k}^t$ the local model of client i after k SGD steps initialized on $\boldsymbol{\theta}^t$, which enables us to also define the normalized stochastic gradients $\boldsymbol{d}_i^t$ and the normalized gradient $\boldsymbol{h}_i^t$ defined as

$$\boldsymbol{d}_i^t = \frac{1}{a_i}\sum_{k=0}^{K-1} a_{i,k} g_i(\boldsymbol{y}_{i,k}^t) \text{ and } \boldsymbol{h}_i^t = \frac{1}{a_i}\sum_{k=0}^{K-1} a_{i,k} \nabla \mathcal{L}_i(\boldsymbol{y}_{i,k}^t), \tag{A.54}$$

where $a_{i,k}$ is an arbitrary scalar applied by the client to its kth gradient, $\boldsymbol{a}_i = [a_{i,0}, .., a_{i,K-1}]^T$, and $a_i = \|\boldsymbol{a}_i\|_1$. In the special case of FEDAVG, we have $\boldsymbol{a}_i = [1, ..., 1]$ and in the one of FEDPROX, we have $\boldsymbol{a}_i = [(1-\mu)^{K-1}, ..., 1]$ where μ is the FEDPROX regularization parameter.

With the formalism of equation (A.54), we can express a client contribution as $\boldsymbol{\theta}_i^{t+1} - \boldsymbol{\theta}^t = -\eta_l a_i \boldsymbol{d}_i^t$ and rewrite the server aggregation scheme defined in equation (2.3) as

$$\boldsymbol{\theta}^{t+1} - \boldsymbol{\theta}^t = -\eta_g \eta_l \sum_{i=1}^n \omega_i a_i \boldsymbol{d}_i^t, \tag{A.55}$$

which in expectation over the set of sampled clients S_t gives

$$\mathbb{E}_{S_t}\left[\boldsymbol{\theta}^{t+1} - \boldsymbol{\theta}^t\right] = -\tilde{\eta}\sum_{i=1}^n p_i a_i \boldsymbol{d}_i^t = -\tilde{\eta}\underbrace{\left(\sum_{i=1}^n p_i a_i\right)}_{K_{eff}} \sum_{i=1}^n \underbrace{\left(\frac{p_i a_i}{\sum_{i=1}^n p_i a_i}\right)}_{w_i} \boldsymbol{d}_i^t. \tag{A.56}$$

We define the surrogate objective $\tilde{\mathcal{L}}(\boldsymbol{x}) = \sum_{i=1}^n w_i \mathcal{L}_i(\boldsymbol{x})$, where $\sum_{i=1}^n w_i = 1$.

In what follows, the norm used for $\boldsymbol{a}_i$ can either be L1, $\|\cdot\|_1$, or L2, $\|\cdot\|_2$, For other variables, the norm is always the euclidean one and $\|\cdot\|$ is used instead of $\|\cdot\|_2$. Also, regarding the client sampling metrics, for ease of writing, we use ω_i instead of $\omega_i(S_t)$ due to the independence of the client sampling statistics with respect to the current optimization round.

A.2.2 Useful Lemmas

Lemma A.1. *Let us consider n vectors $\boldsymbol{x}_i, ..., \boldsymbol{x}_n$ and a client sampling satisfying $\mathbb{E}_{S_t}[\omega_i(S_t)] = p_i$ and $\mathrm{Cov}[\omega_i(S_t)]\,\omega_j(S_t) = -\alpha p_i p_j$. We have:*

$$\mathbb{E}_{S_t}\left[\left\|\sum_{i=1}^{n}\omega_i(S_t)\boldsymbol{x}_i\right\|^2\right] = \sum_{i=1}^{n}\gamma_i\|\boldsymbol{x}_i\|^2 + (1-\alpha)\left\|\sum_{i=1}^{n}p_i\boldsymbol{x}_i\right\|^2, \tag{A.57}$$

where $\gamma_i = \mathrm{Var}_{S_t}[\omega_i(S_t)] + \alpha p_i^2$.

Proof.

$$\mathbb{E}_{S_t}\left[\left\|\sum_{i=1}^{n}\omega_i(S_t)\boldsymbol{x}_i\right\|^2\right] = \sum_{i=1}^{n}\mathbb{E}_{S_t}\left[\omega_i(S_t)^2\right]\|\boldsymbol{x}_i\|^2 + \sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}\mathbb{E}_{S_t}\left[\omega_i(S_t)\omega_j(S_t)\right]\langle\boldsymbol{x}_i,\boldsymbol{x}_j\rangle. \tag{A.58}$$

In addition, we have:

$$\mathbb{E}_{S_t}\left[\omega_i(S_t)\omega_j(S_t)\right] = \mathrm{Cov}\left[\omega_i(S_t)\right]\omega_j(S_t) + p_i p_j = (-\alpha+1)p_i p_j, \tag{A.59}$$

where the last equality comes from the assumption on the client sampling covariance.

We also have:

$$\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}\langle p_i\boldsymbol{x}_i, p_j\boldsymbol{x}_j\rangle = \left\|\sum_{i=1}^{n}p_i\boldsymbol{x}_i\right\|^2 - \sum_{i=1}^{n}p_i^2\|\boldsymbol{x}_i\|^2, \tag{A.60}$$

Substituting equation (A.59) and equation (A.60) in equation (A.58) gives:

$$\mathbb{E}_{S_t}\left[\left\|\sum_{i=1}^{n}\omega_i(S_t)\boldsymbol{x}_i\right\|^2\right] = \sum_{i=1}^{n}\left[\mathbb{E}_{S_t}\left[\omega_i(S_t)^2\right] - (-\alpha+1)p_i^2\right]\|\boldsymbol{x}_i\|^2 + (-\alpha+1)\left\|\sum_{i=1}^{n}p_i\boldsymbol{x}_i\right\|^2, \tag{A.61}$$

Considering that we have $\mathbb{E}_{S_t}\left[\omega_i(S_t)^2\right] = \mathrm{Var}\left[\omega_i(S_t)\right] + p_i^2$, we have :

$$\mathbb{E}_{S_t}\left[\omega_i(S_t)^2\right] + (\alpha - 1)p_i^2 = \mathrm{Var}_{S_t}\left[\omega_i(S_t)\right] + \alpha p_i^2, \tag{A.62}$$

Substituting equation (A.62) in equation (A.61) completes the proof.

□

Lemma A.2 (equation (87) in Jianyu Wang, Q. Liu, et al. (2020))**.** *Under Assumptions 2.2 to 2.4, we can prove*

$$\frac{1}{2}\sum_{i=1}^{n} w_i \,\mathbb{E}\left[\left\|\nabla\mathcal{L}_i(\boldsymbol{\theta}^t) - \boldsymbol{h}_i^t\right\|^2\right] \leq \frac{1}{2}\frac{\eta_l^2 L^2\sigma^2}{1-R}\sum_{i=1}^{n} w_i\left(\|\boldsymbol{a}_i\|_2^2 - a_{i,-1}^2\right) + \frac{R\beta^2}{2(1-R)}\,\mathbb{E}\left[\left\|\nabla\tilde{\mathcal{L}}(\boldsymbol{\theta}^t)\right\|^2\right] + \frac{R\kappa^2}{2(1-R)}, \tag{A.63}$$

with $R = 2\eta_l^2 L^2 \max_i\{\|\boldsymbol{a}_i\|_1\left(\|\boldsymbol{a}_i\|_1 - a_{i,-1}\right)\}$ *with a learning rate such that* $R < 1$.

Proof. The proof is in Section C.5 of Jianyu Wang, Q. Liu, et al. (2020).

The bound here provided is slightly tighter in term of numerical constants than the one of Jianyu Wang, Q. Liu, et al. (2020). Indeed, equation (70) in Jianyu Wang, Q. Liu, et al. (2020) uses the Jensen's inequality $\|\boldsymbol{a}+\boldsymbol{b}\|^2 \leq 2\|\boldsymbol{a}\|^2 + 2\|\boldsymbol{b}\|^2$ which could instead be obtained with:

$$\mathbb{E}\left[\left\|\sum_{s=0}^{k-1} a_{i,s} g_i(\boldsymbol{y}_{i,s}^t)\right\|^2\right] = \mathbb{E}\left[\left\|\sum_{s=0}^{k-1} a_{i,s}\left(g_i(\boldsymbol{y}_{i,s}^t) - \nabla\mathcal{L}_i(\boldsymbol{y}_{i,s}^t)\right)\right\|^2\right] + \mathbb{E}\left[\left\|\sum_{s=0}^{k-1} a_{i,s}\nabla\mathcal{L}_i(\boldsymbol{y}_{i,s}^t)\right\|^2\right], \tag{A.64}$$

which uses Assumption 2.4, giving

$$\mathbb{E}\left[\langle\sum_{s=0}^{k-1} a_{i,s}\left(g_i(\boldsymbol{y}_{i,s}^t) - \nabla\mathcal{L}_i(\boldsymbol{y}_{i,s}^t\right), \sum_{s=0}^{k-1} a_{i,s}\nabla\mathcal{L}_i(\boldsymbol{y}_{i,s}^t\rangle\right] = 0 \tag{A.65}$$

with the same reasoning as for U in equation (A.82). □

Lemma A.3. *Under Assumptions 2.2 to 2.4, we can prove*

$$\sum_{i=1}^{n}\gamma_i\,\mathbb{E}\left[\left\|a_i\boldsymbol{h}_i^t\right\|^2\right] \leq \frac{1}{1-R}\sigma^2\sum_{i=1}^{n}\gamma_i\left(\|\boldsymbol{a}_i\|_2^2-(a_{i,-1}^2)\right) + 2\frac{1}{1-R}\left(\sum_{i=1}^{n}\gamma_i a_i^2\right)\left(\beta^2\,\mathbb{E}\left[\left\|\nabla\tilde{\mathcal{L}}(\boldsymbol{\theta}^t)\right\|^2\right]+\kappa^2\right), \tag{A.66}$$

where $R'=2\eta_l^2L^2\max_i\{\|a_i\|_1^2\}<1$.

Proof. Due to the definition of $\boldsymbol{h}_i^t$, we have:

$$\mathbb{E}\left[\left\|a_i\boldsymbol{h}_i^t\right\|^2\right]=a_i^2\,\mathbb{E}\left[\left\|\sum_{k=0}^{K-1}\frac{1}{a_i}a_{i,k}\nabla\mathcal{L}_i(\boldsymbol{y}_{i,k}^t)\right\|^2\right]\leq a_i^2\sum_{k=0}^{K-1}\frac{1}{a_i}a_{i,k}\,\mathbb{E}\left[\left\|\nabla\mathcal{L}_i(\boldsymbol{y}_{i,k}^t)\right\|^2\right]. \tag{A.67}$$

Using Jensen inequality, we have

$$\mathbb{E}\left[\left\|\nabla\mathcal{L}_i(\boldsymbol{y}_{i,k}^t)\right\|^2\right]\leq 2\,\mathbb{E}\left[\left\|\nabla\mathcal{L}_i(\boldsymbol{y}_{i,k}^t)-\nabla\mathcal{L}_i(\boldsymbol{\theta}^t)\right\|^2\right]+2\,\mathbb{E}\left[\left\|\nabla\mathcal{L}_i(\boldsymbol{\theta}^t)\right\|^2\right] \tag{A.68}$$

$$\leq 2L^2\,\mathbb{E}\left[\left\|\boldsymbol{y}_{i,k}^t-\boldsymbol{\theta}^t\right\|^2\right]+2\,\mathbb{E}\left[\left\|\nabla\mathcal{L}_i(\boldsymbol{\theta}^t)\right\|^2\right], \tag{A.69}$$

where the second equality comes from using Assumption 2.2.

Also, Section C.5 of Jianyu Wang, Q. Liu, et al. (2020) proves

$$\frac{1}{a_i}\sum_{k=0}^{K-1}a_{i,k}\,\mathbb{E}\left[\left\|\boldsymbol{y}_{i,k}^t-\boldsymbol{\theta}^t\right\|^2\right]\leq\frac{1}{1-R}\eta_l^2\sigma^2\left(\|\boldsymbol{a}_i\|_2^2-(a_{i,-1}^2)\right) + \frac{1}{L^2}\frac{R}{1-R}\,\mathbb{E}\left[\left\|\nabla\mathcal{L}_i(\boldsymbol{\theta}^t)\right\|^2\right]. \tag{A.70}$$

Plugging equation (A.69) and then equation (A.70) in equation (A.67), we get:

$$\mathbb{E}\left[\left\|a_i \boldsymbol{h}_i^t\right\|^2\right] \leq a_i^2 \sum_{k=0}^{K-1} \frac{1}{a_i} a_{i,k} \left[2L^2\, \mathbb{E}\left[\left\|\boldsymbol{y}_{i,k}^t - \boldsymbol{\theta}^t\right\|^2\right] + 2\, \mathbb{E}\left[\left\|\nabla \mathcal{L}_i(\boldsymbol{\theta}^t)\right\|^2\right]\right] \tag{A.71}$$

$$= 2L^2 a_i^2 \sum_{k=0}^{K-1} \frac{1}{a_i} a_{i,k}\, \mathbb{E}\left[\left\|\boldsymbol{y}_{i,k}^t - \boldsymbol{\theta}^t\right\|^2\right] + 2a_i^2\, \mathbb{E}\left[\left\|\nabla \mathcal{L}_i(\boldsymbol{\theta}^t)\right\|^2\right] \tag{A.72}$$

$$\leq 2L^2 a_i^2 \left[\frac{1}{1-R} \eta_l^2 \sigma^2 \left(\|\boldsymbol{a}_i\|_2^2 - (a_{i,-1}^2)\right) + \frac{1}{L^2} \frac{R}{1-R}\, \mathbb{E}\left[\left\|\nabla \mathcal{L}_i(\boldsymbol{\theta}^t)\right\|^2\right]\right]$$

$$+ 2a_i^2\, \mathbb{E}\left[\left\|\nabla \mathcal{L}_i(\boldsymbol{\theta}^t)\right\|^2\right] \tag{A.73}$$

$$\leq \frac{R'}{1-R} \sigma^2 \left(\|\boldsymbol{a}_i\|_2^2 - (a_{i,-1}^2)\right) + 2a_i^2 \left[\frac{R}{1-R} + 1\right] \mathbb{E}\left[\left\|\nabla \mathcal{L}_i(\boldsymbol{\theta}^t)\right\|^2\right]. \tag{A.74}$$

Multiplying by γ_i and summing over n gives

$$\sum_{i=1}^{n} \gamma_i\, \mathbb{E}\left[\left\|a_i \boldsymbol{h}_i^t\right\|^2\right] \leq \frac{R'}{1-R} \sigma^2 \sum_{i=1}^{n} \gamma_i \left(\|\boldsymbol{a}_i\|_2^2 - (a_{i,-1}^2)\right)$$

$$+ 2 \frac{1}{1-R} \sum_{i=1}^{n} \gamma_i a_i^2\, \mathbb{E}\left[\left\|\nabla \mathcal{L}_i(\boldsymbol{\theta}^t)\right\|^2\right]. \tag{A.75}$$

Using Assumption 2.3 in equation (A.75) and $R' < 1$ completes the proof.

□

A.2.3 Intermediary Theorem

Theorem A.1. *The following inequality holds:*

$$\frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\left[\left\|\nabla \tilde{\mathcal{L}}(\boldsymbol{\theta}^t)\right\|^2\right] \leq \mathcal{O}\left(\frac{1}{(1-\Omega)\tilde{\eta}\left(\sum_{i=1}^{n} p_i a_i\right) T}\right) + \mathcal{O}\left(\tilde{\eta} \frac{1}{m} A' \sigma^2\right) + \mathcal{O}\left(\eta_l^2 B' \sigma^2\right)$$

$$+ \mathcal{O}\left(\eta_l^2 C' \kappa^2\right) + \mathcal{O}\left(\tilde{\eta} D' \sigma^2\right) + \mathcal{O}\left(\tilde{\eta} E' \kappa^2\right), \tag{A.76}$$

where quantities A'-E' are defined in the following proof from equation (A.93) to equation (A.97).

Proof. Clients local loss functions are L-Lipschitz smooth. Therefore, $\tilde{\mathcal{L}}$ is also L-Lipschitz smooth which gives

$$\mathbb{E}\left[\tilde{\mathcal{L}}(\boldsymbol{\theta}^{t+1}) - \tilde{\mathcal{L}}(\boldsymbol{\theta}^{t})\right] \leq \underbrace{\mathbb{E}\left[\langle \nabla\tilde{\mathcal{L}}(\boldsymbol{\theta}^{t}), \boldsymbol{\theta}^{t+1} - \boldsymbol{\theta}^{t}\rangle\right]}_{T_1} + \frac{L}{2}\underbrace{\mathbb{E}\left[\left\|\boldsymbol{\theta}^{t+1} - \boldsymbol{\theta}^{t}\right\|^2\right]}_{T_2}, \tag{A.77}$$

where the expectation is taken over the subset of randomly sampled clients S_t and the clients gradient estimator noises ξ_i^t. Please note that we use the notation $\mathbb{E}\left[\cdot\right]$ instead of $\mathbb{E}_{\{\xi_i^t\},S_t}\left[\cdot\right]$ for ease of writing.

Bounding T_1

By conditioning on $\{\xi_i^t\}$ and using equation (A.56), we get:

$$T_1 = \mathbb{E}\left[\langle \nabla\tilde{\mathcal{L}}(\boldsymbol{\theta}^{t}), \mathbb{E}_{S_t}\left[\boldsymbol{\theta}^{t+1} - \boldsymbol{\theta}^{t}\right]\rangle\right] = -\tilde{\eta}K_{eff}\,\mathbb{E}\left[\langle \nabla\tilde{\mathcal{L}}(\boldsymbol{\theta}^{t}), \sum_{i=1}^{n} w_i \boldsymbol{h}_i^t\rangle\right], \tag{A.78}$$

which, using $2\langle a, b\rangle = \|a\|^2 + \|b\|^2 - \|a-b\|^2$ can be rewritten as:

$$T_1 = -\frac{1}{2}\tilde{\eta}K_{eff}\,\mathbb{E}\left[\left\|\nabla\tilde{\mathcal{L}}(\boldsymbol{\theta}^{t})\right\|^2 + \left\|\sum_{i=1}^{n} w_i \boldsymbol{h}_i^t\right\|^2 - \left\|\nabla\tilde{\mathcal{L}}(\boldsymbol{\theta}^{t}) - \sum_{i=1}^{n} w_i \boldsymbol{h}_i^t\right\|^2\right]. \tag{A.79}$$

Bounding T_2

$$T_2|S_t = \tilde{\eta}^2\,\mathbb{E}\left[\left\|\sum_{i=1}^{n}\omega_i a_i \boldsymbol{d}_i^t\right\|^2 |S_t\right] \tag{A.80}$$

$$= \tilde{\eta}^2\,\mathbb{E}\left[\left\|\sum_{i=1}^{n}\omega_i a_i \left(\boldsymbol{d}_i^t - \boldsymbol{h}_i^t\right) + \sum_{i=1}^{n}\omega_i a_i \boldsymbol{h}_i^t\right\|^2 |S_t\right] \tag{A.81}$$

$$= \tilde{\eta}^2\,\mathbb{E}\left[\left\|\sum_{i=1}^{n}\omega_i a_i \left(\boldsymbol{d}_i^t - \boldsymbol{h}_i^t\right)\right\|^2 |S_t\right] + \tilde{\eta}^2\,\mathbb{E}\left[\left\|\sum_{i=1}^{n}\omega_i a_i \boldsymbol{h}_i^t\right\|^2 |S_t\right]$$

$$+ 2\tilde{\eta}\underbrace{\mathbb{E}\left[\langle\sum_{i=1}^{n}\omega_i a_i \left(\boldsymbol{d}_i^t - \boldsymbol{h}_i^t\right), \sum_{i=1}^{n}\omega_i a_i \boldsymbol{h}_i^t\rangle |S_t\right]}_{U}. \tag{A.82}$$

Using Assumption 2.4, we have $\mathbb{E}\left[\langle d_i^t - h_i^t, h_j^t \rangle\right] = 0$. Hence, we get $U = 0$ and can simplify T_2 as:

$$T_2 = \tilde{\eta}^2 \sum_{i=1}^{n} \mathbb{E}\left[\omega_i^2\right] a_i^2 \, \mathbb{E}\left[\left\| \boldsymbol{d}_i^t - \boldsymbol{h}_i^t \right\|^2\right] + \tilde{\eta}^2 \, \mathbb{E}\left[\left\| \sum_{i=1}^{n} \omega_i a_i \boldsymbol{h}_i^t \right\|^2\right]. \tag{A.83}$$

Using Lemma A.1 on the second term, we get:

$$\begin{aligned} T_2 = \tilde{\eta}^2 \sum_{i=1}^{n} \mathbb{E}\left[\omega_i^2\right] a_i^2 \, \mathbb{E}\left[\left\| \boldsymbol{d}_i^t - \boldsymbol{h}_i^t \right\|^2\right] + \tilde{\eta}^2 \sum_{i=1}^{n} \gamma_i \, \mathbb{E}\left[\left\| a_i \boldsymbol{h}_i^t \right\|^2\right] \\ + \tilde{\eta}^2 (1-\alpha) \, \mathbb{E}\left[\left\| \sum_{i=1}^{n} p_i a_i \boldsymbol{h}_i^t \right\|^2\right]. \end{aligned} \tag{A.84}$$

Finally, by bounding the first term using Assumption 2.4, and noting that $p_i a_i = w_i K_{eff}$ for the second term, we get:

$$\begin{aligned} T_2 &= \tilde{\eta}^2 \sum_{i=1}^{n} \mathbb{E}\left[\omega_i^2\right] \sum_{k=0}^{K-1} a_{i,k}^2 \, \mathbb{E}\left[\left\| g_i(\boldsymbol{y}_{i,k}^t) - \nabla \mathcal{L}_i(\boldsymbol{y}_{i,k}^t) \right\|^2\right] \\ &\quad + \tilde{\eta}^2 \sum_{i=1}^{n} \gamma_i \, \mathbb{E}\left[\left\| a_i \boldsymbol{h}_i^t \right\|^2\right] + \tilde{\eta}^2 (1-\alpha) K_{eff}^2 \, \mathbb{E}\left[\left\| \sum_{i=1}^{n} w_i \boldsymbol{h}_i^t \right\|^2\right] \qquad \text{(A.85)} \\ &\leq \tilde{\eta}^2 \sum_{i=1}^{n} \mathbb{E}\left[\omega_i^2\right] \left\| \boldsymbol{a}_i \right\|_2^2 \sigma^2 + \tilde{\eta}^2 \sum_{i=1}^{n} \gamma_i \, \mathbb{E}\left[\left\| a_i \boldsymbol{h}_i^t \right\|^2\right] \\ &\quad + \tilde{\eta}^2 (1-\alpha) K_{eff}^2 \, \mathbb{E}\left[\left\| \sum_{i=1}^{n} w_i \boldsymbol{h}_i^t \right\|^2\right]. \qquad \text{(A.86)} \end{aligned}$$

Going back to equation (A.77)

Substituting equation (A.79) and equation (A.86) back in equation (A.77), we get:

$$\begin{aligned} \mathbb{E}\left[\tilde{\mathcal{L}}(\boldsymbol{\theta}^{t+1}) - \tilde{\mathcal{L}}(\boldsymbol{\theta}^t)\right] &\leq -\frac{1}{2} \tilde{\eta} K_{eff} \left\| \nabla \tilde{\mathcal{L}}(\boldsymbol{\theta}^t) \right\|^2 \\ &\quad + \frac{1}{2} \tilde{\eta} K_{eff} \, \mathbb{E}\left[\left\| \nabla \tilde{\mathcal{L}}(\boldsymbol{\theta}^t) - \sum_{i=1}^{n} w_i \boldsymbol{h}_i^t \right\|^2\right] \\ &\quad - \frac{1}{2} \tilde{\eta} K_{eff} \left[1 - L\tilde{\eta}(1-\alpha) K_{eff}\right] \mathbb{E}\left[\left\| \sum_{i=1}^{n} w_i \boldsymbol{h}_i^t \right\|^2\right] \\ &\quad + \frac{L}{2} \tilde{\eta}^2 \sum_{i=1}^{n} \mathbb{E}\left[\omega_i^2\right] \left\| \boldsymbol{a}_i \right\|_2^2 \sigma^2 + \frac{L}{2} \tilde{\eta}^2 \sum_{i=1}^{n} \gamma_i \, \mathbb{E}\left[\left\| a_i \boldsymbol{h}_i^t \right\|^2\right], \end{aligned} \tag{A.87}$$

We consider the learning rate to satisfy $1 - L\tilde{\eta}(1-\alpha)K_{eff} > 0$ such that we can simplify equation (A.87) as :

$$\frac{\mathbb{E}\left[\tilde{\mathcal{L}}(\boldsymbol{\theta}^{t+1}) - \tilde{\mathcal{L}}(\boldsymbol{\theta}^{t})\right]}{\tilde{\eta}K_{eff}} \leq -\frac{1}{2}\left\|\nabla\tilde{\mathcal{L}}(\boldsymbol{\theta}^{t})\right\|^2 + \frac{1}{2}\mathbb{E}\left[\left\|\nabla\tilde{\mathcal{L}}(\boldsymbol{\theta}^{t}) - \sum_{i=1}^{n} w_i \boldsymbol{h}_i^t\right\|^2\right] + \frac{L}{2}\tilde{\eta}\frac{1}{K_{eff}}\sum_{i=1}^{n}\mathbb{E}\left[\omega_i^2\right]\|\boldsymbol{a}_i\|_2^2\,\sigma^2 + \frac{L}{2}\tilde{\eta}\frac{1}{K_{eff}}\sum_{i=1}^{n}\gamma_i\,\mathbb{E}\left[\left\|a_i\boldsymbol{h}_i^t\right\|^2\right] \tag{A.88}$$

$$\leq -\frac{1}{2}\left\|\nabla\tilde{\mathcal{L}}(\boldsymbol{\theta}^{t})\right\|^2 + \frac{1}{2}\sum_{i=1}^{n} w_i\,\mathbb{E}\left[\left\|\nabla\mathcal{L}_i(\boldsymbol{\theta}^{t}) - \boldsymbol{h}_i^t\right\|^2\right] + \frac{L}{2}\tilde{\eta}\frac{1}{K_{eff}}\sum_{i=1}^{n}\mathbb{E}\left[\omega_i^2\right]\|\boldsymbol{a}_i\|_2^2\,\sigma^2 + \frac{L}{2}\tilde{\eta}\frac{1}{K_{eff}}\sum_{i=1}^{n}\gamma_i\,\mathbb{E}\left[\left\|a_i\boldsymbol{h}_i^t\right\|^2\right], \tag{A.89}$$

where the last inequality uses the definition of the surrogate loss function $\tilde{\mathcal{L}}$ and the Jensen's inequality.

Using Lemma A.2 and A.3, we get:

$$\begin{aligned}\frac{\mathbb{E}\left[\tilde{\mathcal{L}}(\boldsymbol{\theta}^{t+1}) - \tilde{\mathcal{L}}(\boldsymbol{\theta}^{t})\right]}{\tilde{\eta}K_{eff}} &\leq -\frac{1}{2}\left\|\nabla\tilde{\mathcal{L}}(\boldsymbol{\theta}^{t})\right\|^2 + \frac{1}{2}\frac{\eta_l^2 L^2\sigma^2}{1-R}\sum_{i=1}^{n} w_i\left(\|\boldsymbol{a}_i\|_2^2 - a_{i,-1}^2\right) \\ &+ \frac{R\beta^2}{2(1-R)}\,\mathbb{E}\left[\left\|\nabla\tilde{\mathcal{L}}(\boldsymbol{\theta}^{t})\right\|^2\right] + \frac{R\kappa^2}{2(1-R)} \\ &+ \frac{L}{2}\tilde{\eta}\frac{1}{K_{eff}}\left[\sum_{i=1}^{n}\mathbb{E}\left[\omega_i^2\right]\|\boldsymbol{a}_i\|_2^2 + \frac{1}{1-R}\sum_{i=1}^{n}\gamma_i\left(\|\boldsymbol{a}_i\|_2^2 - (a_{i,-1}^2)\right)\right]\sigma^2 \\ &+ L\tilde{\eta}\frac{1}{K_{eff}}\left[\frac{R}{1-R}+1\right]\left(\sum_{i=1}^{n}\gamma_i a_i^2\right)\left(\beta^2\,\mathbb{E}\left[\left\|\nabla\tilde{\mathcal{L}}(\boldsymbol{\theta}^{t})\right\|^2\right] + \kappa^2\right).\end{aligned} \tag{A.90}$$

If we assume that $R \leq \frac{1}{2\beta^2+1}$, and considering that $\beta^2 \geq 1$, then we have $\frac{1}{1-R} \leq 1 + \frac{1}{2\beta^2} \leq \frac{3}{2}$, $\frac{R}{1-R} \leq \frac{1}{2}$, and $\frac{R\beta^2}{1-R} \leq \frac{1}{2\beta^2+1}(1+\frac{1}{2\beta^2})\beta^2 = \frac{1}{2}$. We also define $\Omega = L\tilde{\eta}\frac{1}{K_{eff}}\frac{3}{2}\left(\sum_{i=1}^{n}\gamma_i a_i^2\right)\beta^2 \leq \frac{1}{2}$. Substituting these terms in equation (A.90) gives

$$\frac{\mathbb{E}\left[\tilde{\mathcal{L}}(\boldsymbol{\theta}^{t+1}) - \tilde{\mathcal{L}}(\boldsymbol{\theta}^{t})\right]}{\tilde{\eta}K_{eff}} \leq -\frac{1}{4}\left[1-\Omega\right]\left\|\nabla\tilde{\mathcal{L}}(\boldsymbol{\theta}^{t})\right\|^2 + \frac{3}{4}\eta_l^2 L^2 \sum_{i=1}^{n} w_i \left(\|\boldsymbol{a}_i\|_2^2 - a_{i,-1}^2\right)\sigma^2 + \frac{L}{2}\tilde{\eta}\frac{1}{K_{eff}}\left[\sum_{i=1}^{n}\mathbb{E}\left[\omega_i^2\right]\|\boldsymbol{a}_i\|_2^2 + \frac{3}{2}\sum_{i=1}^{n}\gamma_i\left(\|\boldsymbol{a}_i\|_2^2 - (a_{i,-1}^2)\right)\right]\sigma^2 + \frac{3}{2}\eta_l^2 L^2 \max_i\{a_i(a_i - a_{i,-1})\}\kappa^2 + \frac{3}{2}L\tilde{\eta}\frac{1}{K_{eff}}\left(\sum_{i=1}^{n}\gamma_i a_i^2\right)\kappa^2. \tag{A.91}$$

Averaging across all rounds, we get:

$$\frac{1-\Omega}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\left\|\nabla\tilde{\mathcal{L}}(\boldsymbol{\theta}^{t})\right\|^2\right] \leq 4\frac{\tilde{\mathcal{L}}(\theta^0) - \tilde{\mathcal{L}}(\theta^*)}{\tilde{\eta}K_{eff}T} + 3\eta_l^2 L^2 \sum_{i=1}^{n} w_i \left(\|\boldsymbol{a}_i\|_2^2 - a_{i,-1}^2\right)\sigma^2 + L\tilde{\eta}\frac{1}{K_{eff}}\left[2\sum_{i=1}^{n}\mathbb{E}\left[\omega_i^2\right]\|\boldsymbol{a}_i\|_2^2 + 3\sum_{i=1}^{n}\gamma_i\left(\|\boldsymbol{a}_i\|_2^2 - (a_{i,-1}^2)\right)\right]\sigma^2 + 6\eta_l^2 L^2 \max_i\{a_i(a_i - a_{i,-1})\}\kappa^2 + 6L\tilde{\eta}\frac{1}{K_{eff}}\left(\sum_{i=1}^{n}\gamma_i a_i^2\right)\kappa^2. \tag{A.92}$$

We define the following auxiliary variables

$$A = m\frac{1}{K_{eff}}\sum_{i=1}^{n}\mathbb{E}\left[\omega_i^2\right]\|\boldsymbol{a}_i\|_2^2 = m\frac{1}{\sum_{i=1}^{n}p_i a_i}\sum_{i=1}^{n}\left[\mathrm{Var}\left[\omega_i\right] + p_i^2\right]\|\boldsymbol{a}_i\|_2^2, \tag{A.93}$$

$$B = \sum_{i=1}^{n} w_i \left(\|\boldsymbol{a}_i\|_2^2 - a_{i,-1}^2\right) = \sum_{i=1}^{n}\frac{p_i a_i}{\sum_{j=1}^{n}p_j a_j}\left(\|\boldsymbol{a}_i\|_2^2 - a_{i,-1}^2\right), \tag{A.94}$$

$$C = \max_i\{a_i(a_i - a_{i,-1})\}, \tag{A.95}$$

$$D = \frac{1}{K_{eff}}\max_i\{a_i(a_i - a_{i,-1})\}\sum_{i=1}^{n}\gamma_i = \frac{1}{\sum_{i=1}^{n}p_i a_i}C\left(\sum_{i=1}^{n}\mathrm{Var}\left[\omega_i\right] + \alpha\sum_{i=1}^{n}p_i^2\right), \tag{A.96}$$

$$E = \frac{1}{K_{eff}}\max_i\{a_i^2\}\left(\sum_{i=1}^{n}\gamma_i\right) = \frac{1}{\sum_{i=1}^{n}p_i a_i}\max_i\{a_i^2\}\left(\sum_{i=1}^{n}\mathrm{Var}\left[\omega_i\right] + \alpha\sum_{i=1}^{n}p_i^2\right). \tag{A.97}$$

We define for A -E the respective quantities A'-E' such that $X' = \frac{1}{1-\Omega}X$. We have:

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\left\|\nabla\tilde{\mathcal{L}}(\boldsymbol{\theta}^t)\right\|^2\right] \leq 4\frac{\tilde{\mathcal{L}}(\theta^0)-\tilde{\mathcal{L}}(\theta^*)}{(1-\Omega)\tilde{\eta}\left(\sum_{i=1}^n p_i a_i\right)T} + 2L\tilde{\eta}\frac{1}{m}A'\sigma^2 + 3\eta_l^2L^2B'\sigma^2 + 6\eta_l^2L^2C'\kappa^2 + 3L\tilde{\eta}D\sigma^2 + 6L\tilde{\eta}E\kappa^2, \tag{A.98}$$

□

A.2.4 Synthesis of local learning rate η_l conditions for Theorem A.1

A sufficient bound on the local learning rate η_l for constraints on R for Lemma A.2 and equation (A.90), and constraint on R' for Lemma A.3 to be satisfied is:

$$2\left[2\beta^2+1\right]\eta_l^2L^2\max_i\{\|a_i\|_1^2\} < 1. \tag{A.99}$$

Constraints on equation (A.87) can be simplified as

$$L\eta_g\eta_l(1-\alpha)K_{eff} < 1. \tag{A.100}$$

Constraints on Ω, equation (A.90), give

$$3L\eta_g\eta_l\frac{1}{K_{eff}}\left(\sum_{i=1}^n\gamma_i a_i^2\right)\beta^2 \leq 1. \tag{A.101}$$

A.2.5 Theorem 2.1

Proof. With FEDAVG, every client performs vanilla SGD. As such, we have $a_{i,k} = 1$ which gives $a_i = K$ and $\|a_i\|_2 = \sqrt{K}$. In addition we consider a local learning rate η_l such that $\Omega \leq \frac{1}{2}$ as such we can bound A'-E' as $X' \leq 2X$.

Finally, considering that the variables A to E can be simplified as

$$A = m\sum_{i=1}^n\left[\mathrm{Var}\left[\omega_i\right]+p_i^2\right], B = (K-1), C = K(K-1), \tag{A.102}$$

$$D = (K-1)\left(\sum_{i=1}^n\mathrm{Var}\left[\omega_i\right]+\alpha\sum_{i=1}^n p_i^2\right), \text{ and } E = K\left(\sum_{i=1}^n\mathrm{Var}\left[\omega_i\right]+\alpha\sum_{i=1}^n p_i^2\right), \tag{A.103}$$

the convergence bound of Theorem A.1 can be reduced to

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\left\|\nabla\mathcal{L}(\boldsymbol{\theta}^t)\right\|^2\right] \leq \mathcal{O}\left(\frac{1}{\eta_g\eta_l KT}\right) + \mathcal{O}\left(\eta_g\eta_l\sum_{i=1}^{n}\left[\mathrm{Var}\left[\omega_i\right]+p_i^2\right]\sigma^2\right) + \mathcal{O}\left(\eta_l^2(K-1)\sigma^2\right) + \mathcal{O}\left(\eta_l^2K(K-1)\kappa^2\right) + \mathcal{O}\left(\eta_g\eta_l\left(\sum_{i=1}^{n}\mathrm{Var}\left[\omega_i\right]+\alpha\sum_{i=1}^{n}p_i^2\right)\left[(K-1)\sigma^2+K\kappa^2\right]\right), \tag{A.104}$$

which completes the proof.

□

Ω is proportional to $\sum_{i=1}^{n}\gamma_i = \sum_{i=1}^{n}\mathrm{Var}\left[\omega_i\right]+\alpha\sum_{i=1}^{n}p_i^2$. With full participation, we have $\Omega = 0$. However, with client sampling, all the terms in equation (A.104) are proportional with $\frac{1}{1-\Omega}$. Yet, we provide a looser bound in equation (A.104) independent from Ω as the conclusions drawn are identical. Through Ω, $\sum_{i=1}^{n}\mathrm{Var}\left[\omega_i\right]$ and α needs to be minimized. This fact is already visible by inspection of the quantities E and F.

We note that equation (A.104) depends on client sampling through σ^2, which is an indicator of the clients SGD quality, and κ^2, which depends on the clients data heterogeneity. In the special case where clients have the same data distribution and perform full gradient descent, based on the arguments discussed in the previous paragraph, we can still provide the following bound showing the influence of client sampling on the convergence speed, while highlighting the interest of minimizing the quantities $\sum_{i=1}^{n}\mathrm{Var}\left[\omega_i\right]$ and α.

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\left\|\nabla\mathcal{L}(\boldsymbol{\theta}^t)\right\|^2\right] \leq \mathcal{O}\left(\frac{1}{(1-\Omega)\eta_g\eta_l KT}\right), \tag{A.105}$$

When setting the server learning rate at 1, $\eta_g = 1$ with client full participation, i.e. $\mathrm{Var}\left[\omega_i\right] = \mathrm{Var}\left[\sum_{i=1}^{n}\omega_i\right] = \alpha = 0$ and $m = n$, we have $E = F = 0$ and can simplify A to

$$A = n\sum_{i=1}^{n}p_i^2. \tag{A.106}$$

Therefore, the convergence guarantee we provide is $\frac{1}{\eta_l KT}+\eta_l\sum_{i=1}^{n}p_i^2\sigma^2+\eta_l^2(K-1)\sigma^2+\eta_l^2K(K-1)\kappa^2$, which is identical to the one of Jianyu Wang, Q. Liu, et al. (2020) (equation (97) in their work), where $\sum_{i=1}^{n}p_i^2$ can be replaced by $1/n$ when clients have identical importance, i.e. $p_i = 1/n$.

In the special case, where we use $\eta_l = \sqrt{m/KT}$ (Jianyu Wang, Q. Liu, et al., 2020), we retrieve their asymptotic convergence bound $\frac{1}{\sqrt{mKT}} + \sqrt{\frac{m}{KT}} \sum_{i=1}^{n} p_i^2 \sigma^2 + \frac{m}{T}\sigma^2 + \frac{m}{T} K \kappa^2$.

A.2.6 Application to Clustered Sampling

Instead of Lemma A.1 which requires $\mathrm{Cov}\left[\omega_i(S_t)\right] \omega_j(S_t) = -\alpha p_i p_j$, we propose the following Lemma for Clustered sampling expressed in function of MD sampling covariance parameter α_{MD} showing that a sufficient condition for MD sampling to perform as well as Clustered sampling is that all $\boldsymbol{x}_i$ are identical, or that all the distributions are identical, i.e. $r_{k,i} = p_i$.

Lemma A.4. *Let us consider n vectors $\boldsymbol{x}_i, ..., \boldsymbol{x}_n$ and a Clustered sampling satisfying $\mathbb{E}_{S_t}\left[\omega_i(S_t)\right] = p_i$. We have:*

$$\mathbb{E}_{S_{Cl}}\left[\left\|\sum_{i=1}^{n} \omega_i(S_{Cl})\boldsymbol{x}_i\right\|^2\right] \leq \sum_{i=1}^{n} \gamma_i(MD) \left\|\boldsymbol{x}_i\right\|^2 + (1-\alpha_{MD})\left\|\sum_{i=1}^{n} p_i \boldsymbol{x}_i\right\|^2, \quad \text{(A.107)}$$

where $\gamma_i(MD)$ and α_{MD} are the aggregation weights statistics of MD sampling. Equation (A.107) is an equality if and only if $\sum_{i=1}^{n} r_{k,i}\boldsymbol{x}_i = \sum_{j=1}^{n} r_{k,j}\boldsymbol{x}_j$.

Proof. Substituting equation (A.46) in equation (A.58) gives

$$\begin{aligned}\mathbb{E}_{S_{Cl}}\left[\left\|\sum_{i=1}^{n} \omega_i(S_{Cl})\boldsymbol{x}_i\right\|^2\right] &= \sum_{i=1}^{n} \mathbb{E}_{S_{Cl}}\left[\omega_i(S_{Cl})^2\right] \left\|\boldsymbol{x}_i\right\|^2 + \sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}}^{n} p_i p_j \langle \boldsymbol{x}_i, \boldsymbol{x}_j\rangle \\ &\quad - \frac{1}{m^2}\sum_{k=1}^{m}\sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}}^{n} r_{k,i} r_{k,j} \langle \boldsymbol{x}_i, \boldsymbol{x}_j\rangle, \end{aligned} \quad \text{(A.108)}$$

Substituting equation (A.60) in equation (A.58) gives:

$$\begin{aligned}\mathbb{E}_{S_{Cl}}\left[\left\|\sum_{i=1}^{n} \omega_i(S_{Cl})\boldsymbol{x}_i\right\|^2\right] &= \sum_{i=1}^{n} \mathbb{E}_{S_{Cl}}\left[\omega_i(S_{Cl})^2\right] \left\|\boldsymbol{x}_i\right\|^2 + \left\|\sum_{i=1}^{n} p_i \boldsymbol{x}_i\right\|^2 - \sum_{i=1}^{n} p_i^2 \left\|\boldsymbol{x}_i\right\|^2 \\ &\quad - \frac{1}{m^2}\sum_{k=1}^{m}\left[\left\|\sum_{i=1}^{n} r_{k,i}\boldsymbol{x}_i\right\|^2 - \sum_{i=1}^{n} r_{k,i}^2 \left\|\boldsymbol{x}_i\right\|^2\right]. \end{aligned} \quad \text{(A.109)}$$

With rearrangements and using equation (A.41) we get:

$$\mathbb{E}_{S_{Cl}}\left[\left\|\sum_{i=1}^{n}\omega_i(S_{Cl})\boldsymbol{x}_i\right\|^2\right] = \sum_{i=1}^{n}\left[\mathrm{Var}\left[\omega_i(S_{Cl})\right] + \frac{1}{m^2}\sum_{k=1}^{m} r_{k,i}^2\right]\|\boldsymbol{x}_i\|^2 + \left\|\sum_{i=1}^{n} p_i\boldsymbol{x}_i\right\|^2 - \frac{1}{m^2}\sum_{k=1}^{m}\left\|\sum_{i=1}^{n} r_{k,i}\boldsymbol{x}_i\right\|^2. \tag{A.110}$$

Using the expression of clustered sampling variance for the first term (equation (A.47)), and using Jensen's inequality on the third term completes the proof. Jensen's inequality is an equality if and only if $\sum_{i=1}^{n} r_{k,i}\boldsymbol{x}_i = \sum_{j=1}^{n} r_{k,j}\boldsymbol{x}_j$.

□

We adapt Theorem 2.1 to Clustered sampling. Fraboni, Vidal, Kameni, et al. (2021) prove the convergence of FL with clustered sampling by giving identical convergence guarantees to the one of FL with MD sampling. As a result, their convergence bound does not depend of the clients selection probability in the different clusters $r_{k,i}$. The authors' claim was that reducing the variance of the aggregation weights provides faster FL convergence, albeit only providing experimental proofs was provided to support this statement. Corollary A.1 here proposed extends the theory of Fraboni, Vidal, Kameni, et al. (2021) by theoretically demonstrating the influence of clustered sampling on the convergence rate. For easing the notation, Corollary A.1 is adapted to FedAvg but can easily be extended to account for any local $\boldsymbol{a}_i$ using the proof of Theorem A.1 in Section A.2.3.

Corollary A.1. *Even with no α such that $\mathrm{Cov}\left[\omega_i(S_t)\right]\omega_j(S_t) = -\alpha p_i p_j$, the bound of Theorem 2.1 still holds with B, C, and D defined as in Section A.2.3 and*

$$A = m\left[\frac{1}{m} - \frac{1}{m^2}\sum_{i=1}^{n}\sum_{k=1}^{m} r_{k,i}^2 + \sum_{i=1}^{n} p_i^2\right],\ E = \frac{1}{m}(K-1),\ \textit{and}\ F = \frac{1}{m}K, \tag{A.111}$$

where E and F are identical to the one for MD sampling and A is smaller than the one for Clustered sampling.

Proof. The covariance property required for Theorem A.1 is only used for Lemma A.1. In the proof of Theorem A.1, Lemma A.1 is only used in equation (A.84). We can instead use Lemma A.4 and keep the rest of the proof as it is in Section A.2.3. Therefore, the bound of Theorem A.1 remains unchanged for clustered sampling where E and F use the aggregation

weight statistics of MD sampling instead of clustered sampling. Statistics for MD sampling can be found in Section A.1.3 and give

$$\text{Var}\left[\sum_{i=1}^{n}\omega_i(S_{MD})\right] = 0 \text{ and } \alpha_{MD} = \frac{1}{m}, \tag{A.112}$$

while the ones of clustered sampling in Section A.1.7 give

$$\sum_{i=1}^{n}\text{Var}\left[\omega_i(S_{Cl})\right] = \frac{1}{m} - \frac{1}{m^2}\sum_{i=1}^{n}\sum_{k=1}^{m} r_{k,i}^2 \leq \sum_{i=1}^{n}\text{Var}\left[\omega_i(S_{MD})\right]. \tag{A.113}$$

□

A.2.7 Proof of Corollary 2.1

Proof. Combining equation (A.14) with equation (A.21) gives

$$\Sigma_{MD} - \Sigma_U = \left[-\frac{1}{m}\sum_{i=1}^{n} p_i^2 + \frac{1}{m}\right] - \left(\frac{n}{m} - 1\right)\sum_{i=1}^{n} p_i^2 = -\frac{1}{m}\left[(n-m+1)\sum_{i=1}^{n} p_i^2 - 1\right]. \tag{A.114}$$

Therefore, we have

$$\Sigma_{MD} \leq \Sigma_U \Leftrightarrow \sum_{i=1}^{n} p_i^2 \leq \frac{1}{n-m+1}. \tag{A.115}$$

Combining equation (A.16), (A.17), (A.29), and (A.30) gives

$$\gamma_{MD} - \gamma_U = \sum_{i=1}^{n}\text{Var}\left[\omega_i(S_{MD})\right] + \alpha_{MD}\sum_{i=1}^{n} p_i^2 - \left(\sum_{i=1}^{n}\text{Var}\left[\omega_i(S_U)\right] + \alpha_U\sum_{i=1}^{n} p_i^2\right) \tag{A.116}$$

$$= \frac{1}{m} - \frac{n-m}{m(n-1)} n \sum_{i=1}^{n} p_i^2. \tag{A.117}$$

Therefore, we have

$$\gamma_{MD} \leq \gamma_U \Leftrightarrow \sum_{i=1}^{n} p_i^2 \leq \frac{1}{n-m}\frac{n-1}{n}. \tag{A.118}$$

Noting that

$$\frac{1}{n-m+1} - \frac{1}{n-m}\frac{n-1}{n} = \frac{-m+1}{n(n-m)(n-m+1)} \leq 0, \tag{A.119}$$

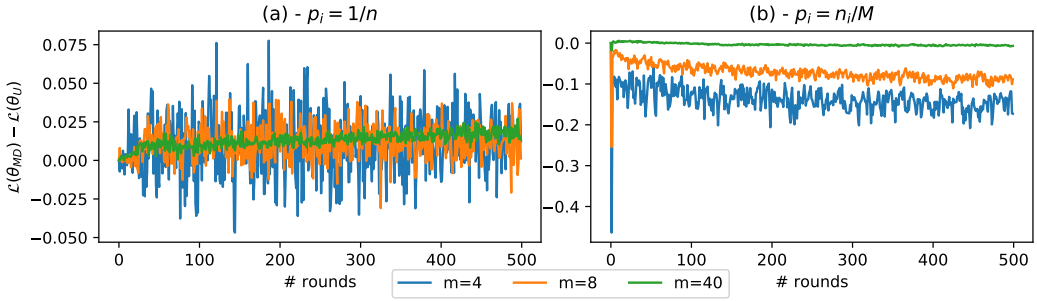

Fig. A.1.: Difference between the convergence of the global losses resulting from MD and Uniform sampling when considering $n = 80$ clients and sampling $m \in \{4, 8, 40\}$ of them while clients perform $K = 50$ SGD steps . In (a), clients have identical importance, i.e. $p_i = 1/n$. In (b), clients importance is proportional to their amount of data, i.e. $p_i = n_i/M$. Differences in global losses are averaged across 15 FL experiments with different model initialization (global losses are provided in Figure A.2).

completes the proof.

□

A.3 Additional experiments

A.3.1 Shakespeare dataset

The client local learning rate η_l is selected in $\{0.1, 0.5, 1., 1.5, 2., 2.5\}$ minimizing FEDAVG with full participation, and $n = 80$ training loss at the end of the learning process.

A.3.2 CIFAR10 dataset

We consider the experimental scenario used to prove the experimental correctness of clustered sampling in (Fraboni, Vidal, Kameni, et al., 2021) on CIFAR10 (Krizhevsky et al., n.d.). The dataset is partitioned in $n = 100$ clients using a Dirichlet distribution with parameter $\alpha = 0.1$ as proposed in Harry Hsu et al., 2019. 10, 30, 30, 20 and 10 clients have respectively 100, 250, 500, 750, and 1000 training samples, and testing samples amounting to a fifth of their training size. The client local learning rate η_l is selected in $\{0.01, 0.02, 0.05, 0.1\}$.

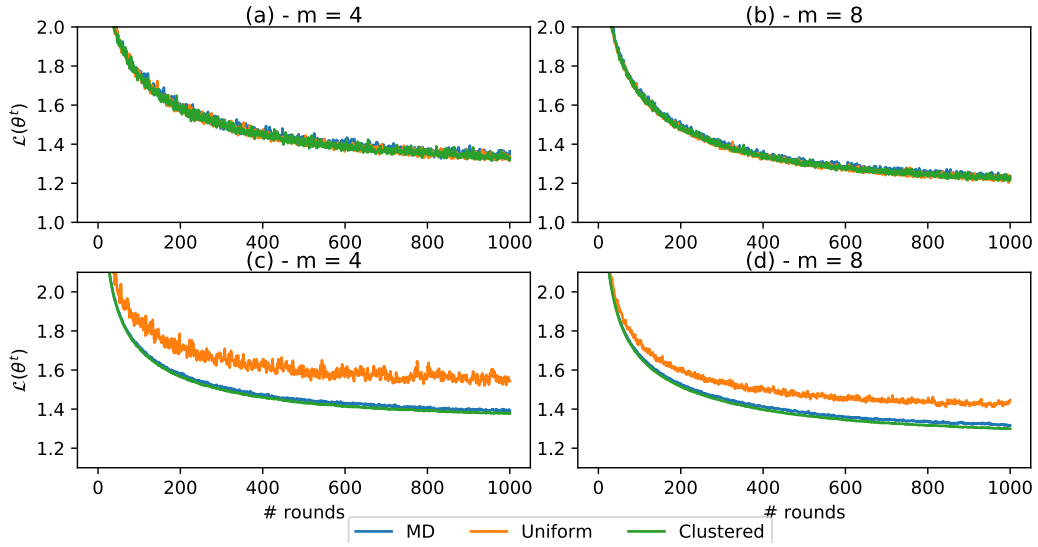

Fig. A.2.: Convergence speed of the global loss with MD sampling and Uniform sampling when considering $n = 80$ clients while sampling $m = 4$ ((a) and (c)), and $m = 8$ ((b) and (d)) while clients perform $K = 50$ SGD steps. In (a-b) , clients have identical importance, i.e. $p_i = 1/n$, and, in (d-f), their importance is proportional to their amount of data, i.e. $p_i = n_i/M$. Global losses are estimated on 15 different model initialization.

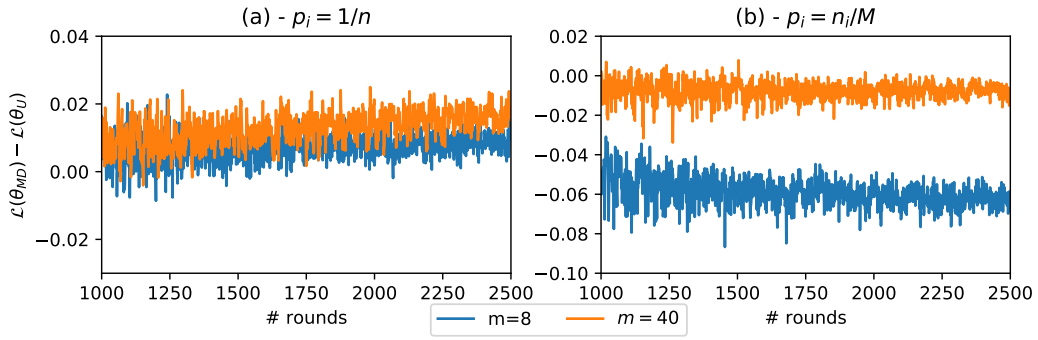

Fig. A.3.: Difference between the convergence of the global losses resulting from MD and Uniform sampling when considering $n = 80$ clients and sampling $m \in \{8, 40\}$ of them while clients perform $K = 1$ SGD step. In (a), clients have identical importance, i.e. $p_i = 1/n$. In (b), clients importance is proportional to their amount of data, i.e. $p_i = n_i/M$. Differences in global losses are averaged across 15 FL experiments with different model initialization (global losses are provided in Figure A.4).

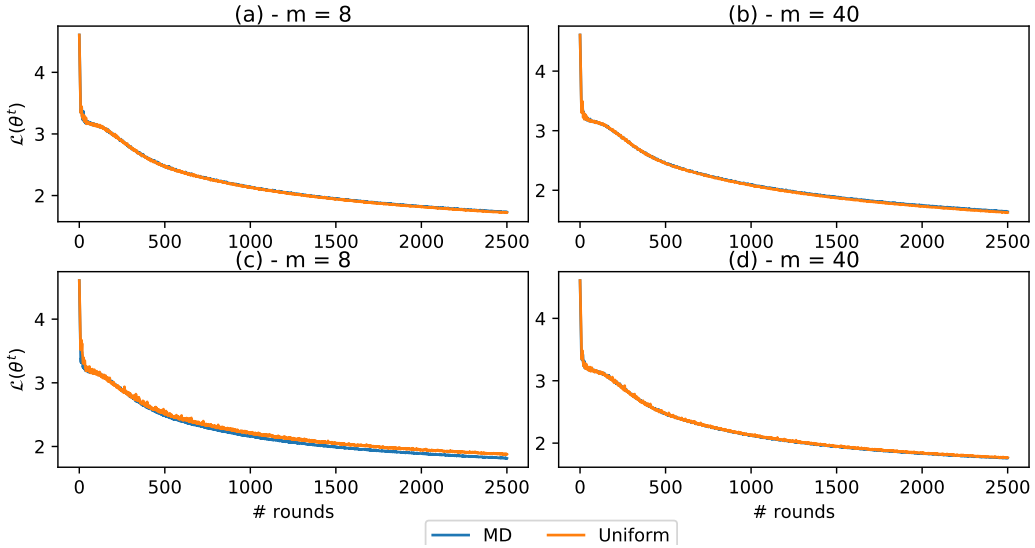

Fig. A.4.: Convergence speed of the global loss with MD sampling and Uniform sampling when considering $n = 80$ clients while sampling $m = 4$ ((a) and (d)), $m = 8$ ((b) and (e)), $m = 40$ ((c) and (f)) while clients perform $K = 1$ SGD steps. In (a-c) , clients have identical importance, i.e. $p_i = 1/n$, and, in (d-f), their importance is proportional to their amount of data, i.e. $p_i = n_i/M$. Global losses are estimated on 15 different model initialization.

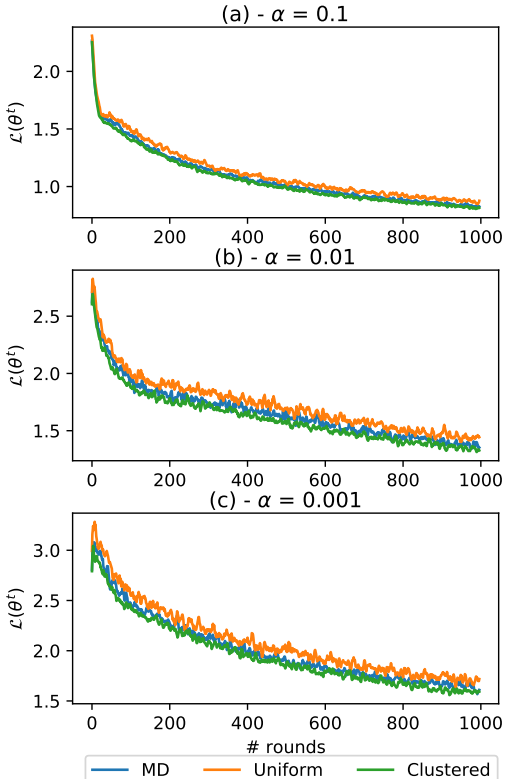

Fig. A.5.: Convergence speed of the global loss with MD sampling and Uniform sampling when considering $n = 100$ clients, while sampling $m = 10$ of them. Clients are partitioned using a Dirichlet distribution with parameter $\alpha = 0.1$ (a), $\alpha = 0.01$ (b), and $\alpha = 0.001$ (c). Global losses are estimated on 30 different model initialization.

Appendix of Chapter 3

B

B.1 Proof of Theorem 3.2

Theoretical guarantees regarding the convergence of FEDAVG were given in Jianyu Wang, Q. Liu, et al. (2020). The proof relies on Assumption 3.1 to 3.3. The full proof is provided in Jianyu Wang, Q. Liu, et al. (2020) for MD sampling where the MD sampling is shown to satisfy Lemma B.1. In Section B.1.1, we reproduce the proof provided in Jianyu Wang, Q. Liu, et al. (2020) for Lemma B.1 and, in Section B.1.2, we show that clustered sampling satisfying Proposition 3.1 also satisfies Lemma B.1. As a result, FEDAVG when sampling clients with MD or clustered sampling has identical asymptotic behavior.

Lemma B.1. *Suppose we are given $z_1, z_2, ..., z_n, x \in \mathbb{R}^d$. Let $l_1, l_2, ..., l_m$ be the index of the sampled clients and S be the set of sampled clients. We have*

$$\mathbb{E}_S\left[\frac{1}{m}\sum_{j=1}^{m} z_{l_j}\right] = \sum_{i=1}^{n} p_i z_i, \tag{B.1}$$

and

$$\mathbb{E}_S\left[\left\|\frac{1}{m}\sum_{j=1}^{m} z_{l_j}\right\|^2\right] \leq 3\sum_{i=1}^{n} p_i \left\|z_i - \nabla\mathcal{L}_i(x)\right\|^2 + 3\left\|\nabla\mathcal{L}(x)\right\|^2 + \frac{3}{m}(\beta^2\left\|\nabla\mathcal{L}(x)\right\|^2 + \kappa^2). \tag{B.2}$$

B.1.1 Proof of Lemma B.1 for Theorem 3.1

Proof. Clients are selected with MD sampling. We denote by $l_1, l_2, ..., l_m$ the m indices of the sampled clients which are iid sampled from a multinomial distribution supported on $\{1, ..., n\}$ satisfying $\mathbb{P}(l_x = i) = p_i$ and $\sum_{i=1}^{n} p_i = 1$.

By definition, MD sampling satisfies equation (B.1).

Regarding equation (B.2), we have:

$$\frac{1}{m}\sum_{j=1}^{m} z_{l_j} = \left(\frac{1}{m}\sum_{j=1}^{m} z_{l_j} - \frac{1}{m}\sum_{j=1} \nabla\mathcal{L}_{l_j}(x)\right) + \left(\frac{1}{m}\sum_{j=1}^{m} \mathcal{L}_{l_j}(x) - \nabla\mathcal{L}(x)\right) + \nabla\mathcal{L}(x). \tag{B.3}$$

Using the Jensen inequality on the $\|\cdot\|^2$ operator, we get:

$$\mathbb{E}\left[\left\|\frac{1}{m}\sum_{j=1}^{m} z_{l_j}\right\|^2\right] \leq 3\,\mathbb{E}\left[\left\|\frac{1}{m}\sum_{j=1}^{m}(z_{l_j} - \nabla\mathcal{L}_{l_j}(x))\right\|^2\right] + 3\,\mathbb{E}\left[\left\|\frac{1}{m}\sum_{j=1}^{m}\nabla\mathcal{L}_{l_j}(x) - \nabla\mathcal{L}(x)\right\|^2\right] + 3\,\|\nabla\mathcal{L}(x)\|^2 \tag{B.4}$$

Using the Jensen inequality, we get the following upper bound for the first term:

$$\mathbb{E}\left[\left\|\frac{1}{m}\sum_{j=1}^{m}(z_{l_j} - \nabla\mathcal{L}_{l_j}(x))\right\|^2\right] \leq \mathbb{E}\left[\frac{1}{m}\sum_{j=1}^{m}\left\|z_{l_j} - \nabla\mathcal{L}_{l_j}(x)\right\|^2\right] \tag{B.5}$$

$$= \sum_{i=1}^{n} p_i \left\|z_i - \nabla\mathcal{L}_i(x)\right\|^2, \tag{B.6}$$

where the equality follows from equation (B.1).

By definition, MD sampling is unbiased, i.e. $\mathbb{E}\left[\nabla\mathcal{L}_{l_j}(x)\right] = \nabla\mathcal{L}(x)$. Therefore, we get the following upper bound for the second term:

$$\mathbb{E}\left[\left\|\frac{1}{m}\sum_{j=1}^{m}\nabla\mathcal{L}_{l_j}(x) - \nabla\mathcal{L}(x)\right\|^2\right] = \mathbb{E}\left[\frac{1}{m^2}\sum_{j=1}^{m}\left\|\nabla\mathcal{L}_{l_j}(x) - \nabla\mathcal{L}(x)\right\|^2\right] \tag{B.7}$$

$$= \frac{1}{m}\sum_{i=1}^{n} p_i \|\nabla\mathcal{L}_i(x) - \nabla\mathcal{L}(x)\|^2 \tag{B.8}$$

$$= \frac{1}{m}\sum_{i=1}^{n} p_i \left\|\nabla\mathcal{L}_i(x)\right\|^2 - \frac{1}{m}\left\|\nabla\mathcal{L}(x)\right\|^2 \tag{B.9}$$

$$\leq \frac{1}{m}[(\beta^2 - 1)\left\|\nabla\mathcal{L}(x)\right\|^2 + \kappa^2] \tag{B.10}$$

$$\leq \frac{1}{m}[\beta^2\left\|\nabla\mathcal{L}(x)\right\|^2 + \kappa^2], \tag{B.11}$$

where the first inequality comes from using Assumption 3.3.

Finally, substituting equation (B.6) and (B.11) in equation (B.4) completes the proof.

□

B.1.2 Proof of Lemma B.1 for Theorem 3.2

Proof. Clients are selected with clustered sampling. The m clients indices $l_1, l_2, ..., l_m$ are still independently sampled but no longer identically. Each index l_k is sampled from a distribution W_k. Each client can be sampled with probability $\mathbb{P}(l_k = i) = r_{k,i}$.

Clustered sampling follows Proposition 3.1 and therefore satisfies equation (B.1).

Equation (B.4) holds for any sampling schemes. Therefore, we also use it to prove equation (B.2) for clustered sampling. Using the same steps as for the proof of Lemma B.1 for MD sampling, we bound the first term of equation (B.4) as:

$$\mathbb{E}\left[\left\|\frac{1}{m}\sum_{j=1}^{m}(z_{l_j} - \nabla\mathcal{L}_{l_j}(x))\right\|^2\right] \leq \sum_{i=1}^{n} p_i \left\|z_i - \nabla\mathcal{L}_i(x)\right\|^2.$$

Before bounding the second term, we define $\nabla\mathcal{L}_{W_k}(x)$ as the expected gradient of the distribution W_k with respects to the parameters x, i.e.

$$\nabla\mathcal{L}_{W_k}(x) \coloneqq \mathbb{E}_{l_k \sim W_k}\left[\nabla\mathcal{L}_{l_k}(x)\right] = \sum_{i=1}^{n} r_{k,i}\nabla\mathcal{L}_i(x) \tag{B.12}$$

Using this definition, we bound the second term as

$$\begin{aligned}
\mathbb{E}\left[\left\|\frac{1}{m}\sum_{k=1}^{m}\nabla\mathcal{L}_{l_k}(x) - \nabla\mathcal{L}(x)\right\|^2\right] &= \mathbb{E}\left[\left\|\frac{1}{m}\sum_{k=1}^{m}(\nabla\mathcal{L}_{l_k}(x) - \nabla\mathcal{L}_{W_k}(x))\right\|^2\right] && \text{(B.13)}\\
&= \frac{1}{m^2}\sum_{k=1}^{m}\mathbb{E}\left[\left\|\nabla\mathcal{L}_{l_k}(x) - \nabla\mathcal{L}_{W_k}(x)\right\|^2\right] && \text{(B.14)}\\
&= \frac{1}{m^2}\sum_{k=1}^{m}\sum_{i=1}^{n} r_{k,i}\left\|\nabla\mathcal{L}_i(x) - \nabla\mathcal{L}_{W_k}(x)\right\|^2 && \text{(B.15)}\\
&= \frac{1}{m^2}[\sum_{i=1}^{n} mp_i\left\|\nabla\mathcal{L}_i(x)\right\|^2 - \sum_{k=1}^{m}\left\|\nabla\mathcal{L}_{W_k}(x)\right\|^2] && \text{(B.16)}\\
&\leq \frac{1}{m}[\beta^2\left\|\nabla\mathcal{L}(x)\right\|^2 + \kappa^2], && \text{(B.17)}
\end{aligned}$$

where the last inequality comes from using Assumption 3.3 and equation (B.14) and (B.16) are obtained with equation (B.12).

Finally, substituting equation (B.12) and (B.17) in equation (B.4) completes the proof.

□

Equation (B.9) and (B.16) allow us to theoretically identify the convergence improvement of clustered sampling over MD sampling.

We define by $B_{MD} = \frac{1}{m}\sum_{i=1}^{n} p_i \left\|\nabla \mathcal{L}_i(x)\right\|^2 - \frac{1}{m}\left\|\nabla \mathcal{L}(x)\right\|^2$, equation (B.9), and $B_{Cl} = \frac{1}{m}\sum_{i=1}^{n} p_i \left\|\nabla \mathcal{L}_i(x)\right\|^2 - \frac{1}{m^2}\sum_{k=1}^{m}\left\|\nabla \mathcal{L}_{W_k}(x)\right\|^2$, equation (B.16). Using the Jensen inequality, we get

$$-\sum_{k=1}^{m}\frac{1}{m^2}\left\|\nabla \mathcal{L}_{W_k}(x)\right\|^2 \leq -\frac{1}{m}\left\|\sum_{k=1}^{m}\frac{1}{m}\nabla \mathcal{L}_{W_k}(x)\right\|^2 = -\frac{1}{m}\left\|\nabla \mathcal{L}(x)\right\|^2 \tag{B.18}$$

with equality if and only if $\forall k, l,\ \nabla \mathcal{L}_{W_k}(x) = \nabla \mathcal{L}_{W_l}(x)$. Thus, $B_{Cl} \leq B_{MD}$ with equality if and only if all the clients have the same data distribution or the considered clustered sampling is MD sampling.

B.2 MD and Clustered Sampling Comparison

B.2.1 Client aggregation weight variance

As in Section 3.3, we denote by S_{MD} and $S_C(t)$ the random variables associated respectively to MD and clustered sampling. Also in Section 3.3, we have shown that

$$\mathrm{Var}_{S_{MD}}\left[\omega_i(S_{MD})\right] = \frac{1}{m^2} m p_i (1 - p_i), \tag{B.19}$$

and

$$\mathrm{Var}_{S_C(t)}\left[\omega_i(S_C(t)\right] = \frac{1}{m^2}\sum_{k=1}^{m} r_{k,i}^t (1 - r_{k,i}^t). \tag{B.20}$$

Hence, we get:

$$\mathrm{Var}_{S_{MD}}\left[\omega_i(S_{MD})\right] - \mathrm{Var}_{S_C(t)}\left[\omega_i(S_C(t)\right] \tag{B.21}$$

$$= \frac{1}{m^2}\left[m p_i (1 - p_i) - \sum_{k=1}^{m} r_{k,i}^t (1 - r_{k,i}^t)\right] \tag{B.22}$$

We consider an unbiased clustered sampling. Therefore, the sum of probability for client i over the m clusters satisfies $\sum_{k=1}^{m} r_{k,i}^t = mp_i$ giving:

$$\mathrm{Var}_{S_{MD}}\left[\omega_i(S_{MD})\right] - \mathrm{Var}_{S_C(t)}\left[\omega_i(S_C(t)\right] = \frac{1}{m^2}[\sum_{k=1}^{m} {r_{k,i}^t}^2 - mp_i^2] \tag{B.23}$$

Using the Cauchy-Schwartz inequality, we get: $\sum_{k=1}^{m} {r_{k,i}^t}^2 \times \sum_{k=1}^{m} 1^2 \geq \left(\sum_{k=1}^{m} r_{k,i}^t \times 1\right)^2 = (mp_i)^2$ due to the unbiased aspect of the considered clustered sampling. As such, we get:

$$\mathrm{Var}_{S_{MD}}\left[\omega_i(S_{MD})\right] - \mathrm{Var}_{S_C(t)}\left[\omega_i(S_C(t)\right] \geq 0, \tag{B.24}$$

with equality if and only if $r_{k,i}^t = p_i$.

B.2.2 Probability for a client to be sampled at least once

In Section 3.3, we have shown that

$$p(\{i \in S_{MD}\}) = 1 - (1 - p_i)^m \tag{B.25}$$

and

$$p(\{i \in S_C(t)\}) = 1 - \prod_{k=1}^{m}(1 - r_{k,i}^t). \tag{B.26}$$

Hence, we get:

$$p(\{i \in S_{MD}\}) - p(\{i \in S_C(t)\}) \tag{B.27}$$

$$= \prod_{k=1}^{m}(1 - r_{k,i}^t) - (1 - p_i)^m \tag{B.28}$$

We consider an unbiased clustered sampling. Therefore, when using the inequality of arithmetic and geometric means, we get:

$$\prod_{k=1}^{m}(1 - r_{k,i}^t) \leq \left(\frac{\sum_{k=1}^{m}(1 - r_{k,i}^t)}{m}\right)^m = (1 - p_i)^m, \tag{B.29}$$

with equality if and only if $r_{k,i}^t = p_i$. Finally, we get:

$$p(\{i \in S_{MD}\}) \geq p(\{i \in S_C(t)\}) \tag{B.30}$$

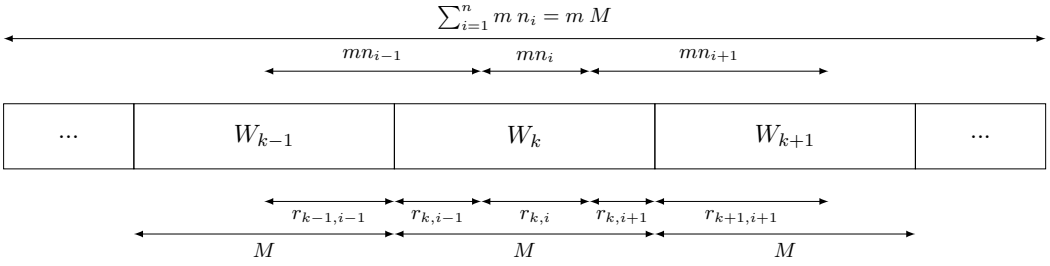

Fig. B.1.: Illustration of the clients allocation scheme of Algorithm 2. Clients are considered in decreasing importance of their number of samples and always allocate client samples to distributions that already received samples but do not yet have M of them. As a result, after allocating a client, all distributions except at most one have 0 or M samples. Client i is only sampled in W_k because every distribution with index inferior to k are filled with clients of index inferior to i, and because there is enough room in W_k to receive all the samples that need to be allocated for client i.

B.3 Explaining Algorithm 2 and 3

Algorithms 2 and 3 can be written in term of data ratio p_i instead of samples number n_i. While in both cases the algorithms would be correct, it turns out to be simpler to work with quantities of samples $n_i = p_i M$ instead which are integers. Therefore, without loss of generality, we denote by $r'_{k,i}$ the number of samples allocated by client i to distribution k. We retrieve the sampling probability of client i in distribution W_k with $r_{k,i} = \frac{r'_{k,i}}{M}$.

Also, without loss of generality, we prove Algorithms 2 and 3 at iteration t and therefore we use in the proofs $r_{k,i}$ and W_k instead of $r^t_{k,i}$ and W^t_k.

B.3.1 Algorithm 2

We illustrate in Figure B.1 the clients allocation scheme of Algorithm 2 introduced in Section 3.4, by considering how a client i is associated to the m distributions. Theorem 3.3 states that Algorithm 2 provides a sampling scheme satisfying Proposition 3.1 with complexity $\mathcal{O}(n \log(n))$ which we prove in Section 3.4 and in the following proof.

Proof. In term of complexity, the while loop for the client allocation, as illustrated in Figure B.1, either change client or distribution at every step and is thus done in complexity $\mathcal{O}(n + m)$. Sampling client is relevant if $m < n$. Therefore the allocation complexity is equivalent to $\mathcal{O}(n + m) = \mathcal{O}(n)$. Also, sorting n elements is done in complexity $\mathcal{O}(n \log(n))$. Therefore, Algorithm 2 overall complexity is $\mathcal{O}(n \log(n))$. □

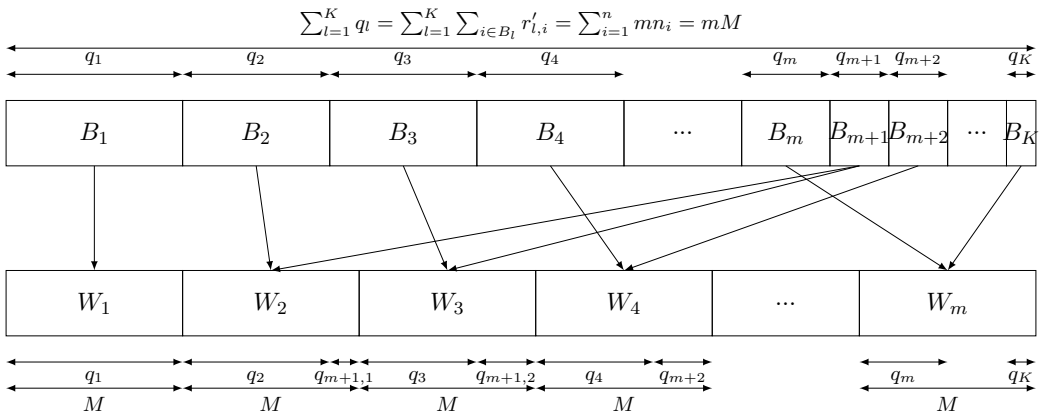

Fig. B.2.: Illustration of the clients allocation scheme of Algorithm 3. After the tree is split in K groups of clients, the groups are ordered and we consider without loss of generality that their number of samples are inversely proportional to their index. With Algorithm 3, the first m groups, i.e. B_1 to B_m, are each associated to one distribution, i.e. W_1 to W_m. The remaining groups are considered one after the other and split among the remaining slots in the groups. Each distribution has M samples from clients participating to the FL process.

B.3.2 Algorithm 3

We illustrate in Figure B.2, the clients allocation scheme of Algorithm 3 introduced in Section 3.5 by considering how a client i is associated to the m distributions. Theorem 3.4 states that Algorithm 3 provides a sampling scheme satisfying Proposition 3.1 and takes time complexity $\mathcal{O}(n^2 d + X)$. We prove these statements in Section 3.5 and the following proof.

Proof. With identical reasoning as for Algorithm 2, clients are allocated in complexity $\mathcal{O}(n)$. Computing the similarity between two clients requires d elementary operations, where d is the number of parameters in the model, and has thus complexity $\mathcal{O}(d)$. Computing the similarity matrix requires computing $\frac{n(n-1)}{2}$ client similarities and thus has total complexity $\mathcal{O}(n^2 d)$. Computing the similarity tree depends on the *clustering method* which we consider has complexity $\mathcal{O}(X)$. Transforming the tree as discussed in Section 3.5 requires going through its $n-1$ nodes and thus has time complexity $\mathcal{O}(n)$. Cutting the tree requires considering at most every nodes and has thus complexity $\mathcal{O}(n)$. Lastly, the tree is cut in at most n branches and sorting them takes therefore complexity $\mathcal{O}(n \log(n))$. Finally, combining all these time complexities gives for Algorithm 3 a time complexity of $\mathcal{O}(n^2 d + X)$.

□

In practice, the m distributions are computed at every iteration, while the server is required to compute the similarity between sampled clients and all the other clients. Therefore the similarity matrix can be estimated in complexity $\mathcal{O}(nmd)$, and Algorithm 3 has complexity $\mathcal{O}(nmd + X)$.

B.4 Additional Experiments

We describe in Section 3.6 the different datasets used for the experiments and how we use the Dirichlet distribution to partition CIFAR10 in realistic heterogeneous federated datasets. In all the experiments, we consider a batch size of 50. For every CIFAR10 dataset partition, the learning rate is selected in $\{0.001, 0.005, 0.01, 0.05, 0.1\}$ to minimize FedAvg with MD sampling training loss.

B.4.1 CIFAR10 partitioning illustration

In Figure B.3, we show the influence of α on the resulting federated dataset heterogeneity. $\alpha = 10$ provides almost an iid dataset and identical class percentages, column (a) , and same number of samples per class, column (b). With $\alpha = 0.001$, we get a very heterogeneous dataset with almost only one class per client translating into some classes much more represented than others due to the unbalanced nature aspect of the created federated dataset, cf Section 3.6.

B.4.2 Influence of the similarity measure

Figure B.4 shows the effect similarity measures (Arccos, L2, and L1) have on training global loss convergence. We retrieve that Algorithm 2 outperforms MD sampling by reducing clients aggregation weight variance. We remind that the hierarchical tree is obtained using Ward's method in this work. We notice that the tree similarity measures gives similar performances when using Algorithm 3 with Ward hierarchical clustering method.. This justifies the use of Arccos similarity for the other experiments.

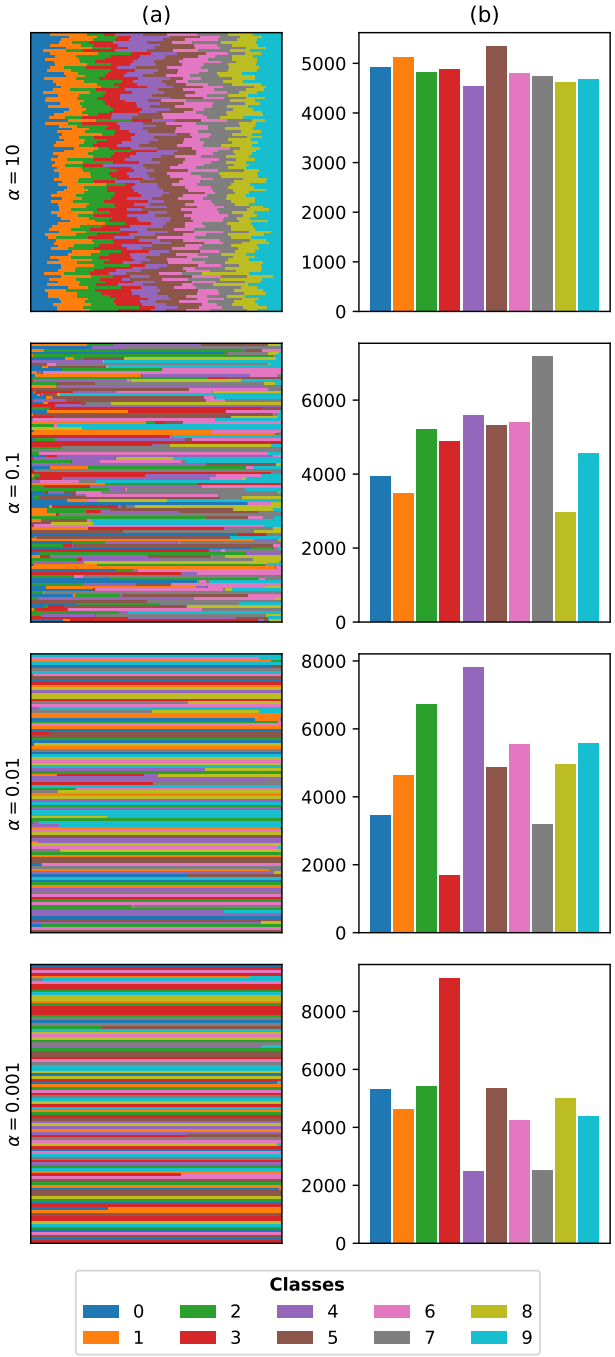

Fig. B.3.: Effect of α on the resulting clients partitioning when using a Dirichlet distribution. Plots in column (a) represent the percentage of each class owned by the clients. Plots in column (b) give for every class its total number of samples across clients. We consider in this work $\alpha \in \{0.001, 0.01, 0.1, 10\}$.

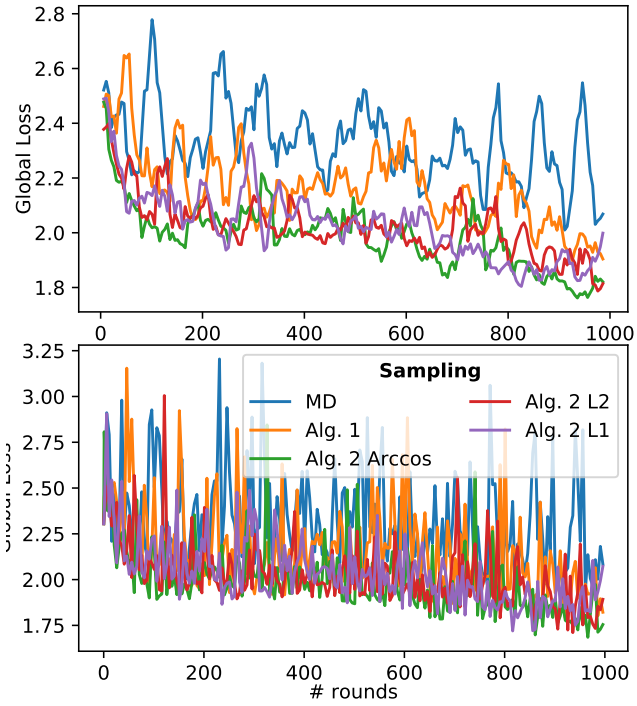

Fig. B.4.: Effect of the similarity measure chosen for Algorithm 3 on the training loss convergence. We consider the evolution of the global loss, equation (3.1), in function of the server iteration t. For clarity concerns, we plot the global loss obtained with rolling mean over 50 server iterations (top) and the raw global loss (bottom). We consider CIFAR partitioned with Dir($\alpha = 0.01$), learning rate $lr = 0.05$, $N = 100$ SGD, and $m = 10$ sampled clients.

B.4.3 More details on Figure 3.2

For sake of clarity, we note that the training loss displayed in Figures 3.2 is computed as the rolling mean over 50 iterations. In Figure B.5, we provide the raw training global loss with the testing accuracy at every server iteration.

B.4.4 Influence of m the number of sampled clients, and N the number of SGD run

We also investigates the influence the number of sampled clients m and the number of SGD run N have on the FL convergence speed and smoothness in Figure B.6. We notice that the more important the amount of local work N is, and the faster clustered sampling convergence speed is. With more local work, clients better fit their data. In non-iid dataset this translates in more forgetting on the classes and samples which are not part of the sampled clients. Regarding the amount of sampled clients m, we notice that with a smaller amount of sampled clients the improvement of clustered sampling over MD sampling is more important. We associate this result to the better data representativity of clustered sampling. For the same reason, when we increase the number of sampled clients, we see faster convergence for both MD and clustered sampling. The performance of clustered sampling is closer but still better than the one of MD sampling.

For sake of clarity, we note that the training loss displayed in Figures B.6 is computed as the rolling mean over 50 iterations. In Figure B.5, we provide the raw training global loss with the testing accuracy at every server iteration.

B.4.5 Local regularization

With FedProx (T. Li, Sahu, Zaheer, Sanjabi, Talwalkar, and Smith, 2018a), every client's local loss function is equipped with a regularization term forcing the updated model to stay close to the current global model, i.e.

$$\mathcal{L}'_i(\theta_i^{t+1}) = \mathcal{L}_i(\theta_i^{t+1}) + \frac{\mu}{2} \left\| \theta_i^{t+1} - \theta^t \right\|^2 \tag{B.31}$$

where θ_{t+1} is the updated local model of client i and θ^t is the current global model. μ is the hyperparameter monitoring the regularization and is common for all the clients. This framework enables smoother federated learning processes.

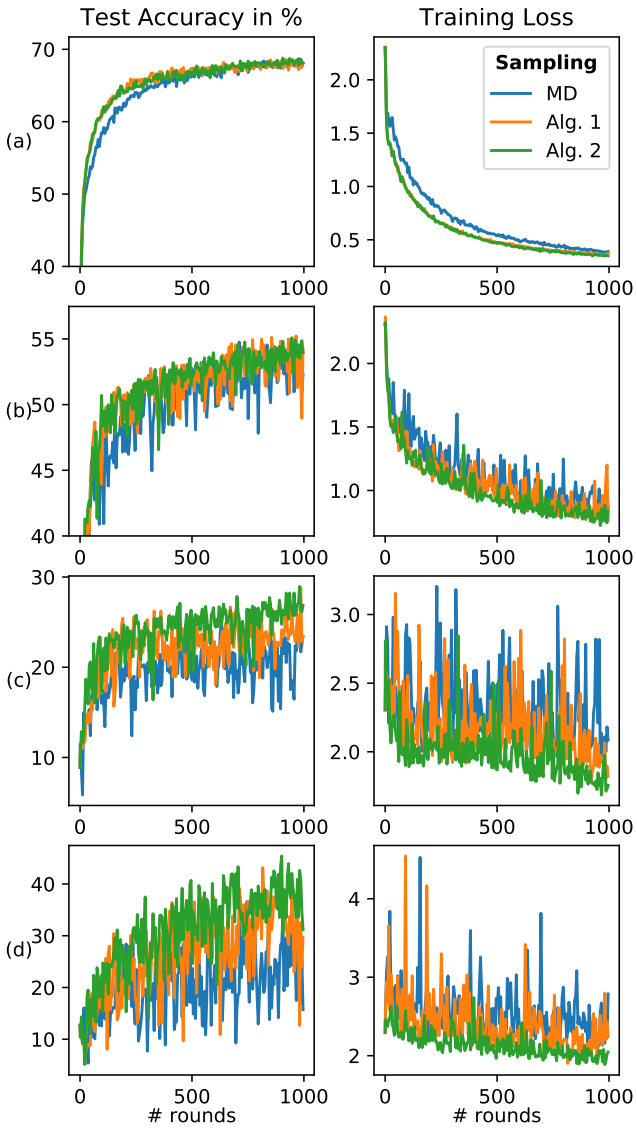

Fig. B.5.: We investigate the improvement provided by clustered sampling on federated unbalanced datasets partitioned from CIFAR10 using a Dirichlet distribution with parameter $\alpha \in \{0.001, 0.01, 0.1, 10\}$ for respective row (a), (b), (c), (d). We use $N = 100$, $m = 10$, and respective learning rate for each dataset $lr = \{0.05, 0.05, 0.05, 0.1\}$.

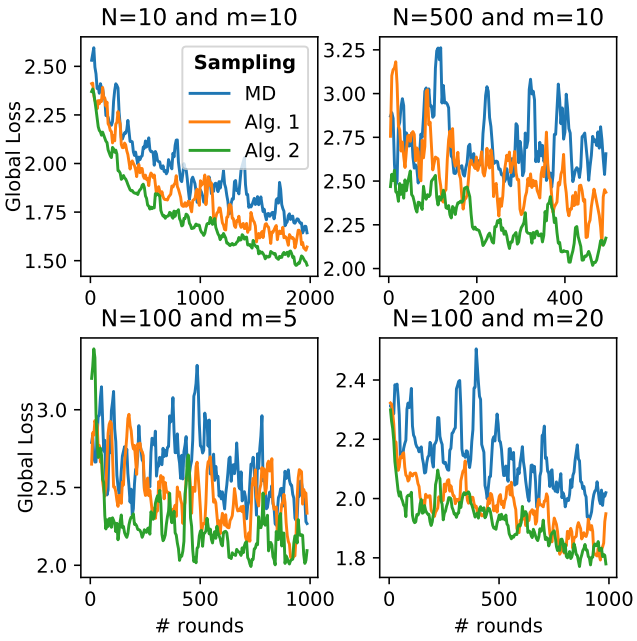

Fig. B.6.: We consider the federated dataset partitioned from CIFAR10 using a Dirichlet distribution with parameter $\alpha = 0.01$. We investigate the influence of N, the number of SGD run by each client, and m, the number of sampled clients, on the training loss convergence. For each plot, experiments in first row use respectively $lr = \{0.1, 0.05\}$ and for second row $lr = \{0.05, 0.05\}$.

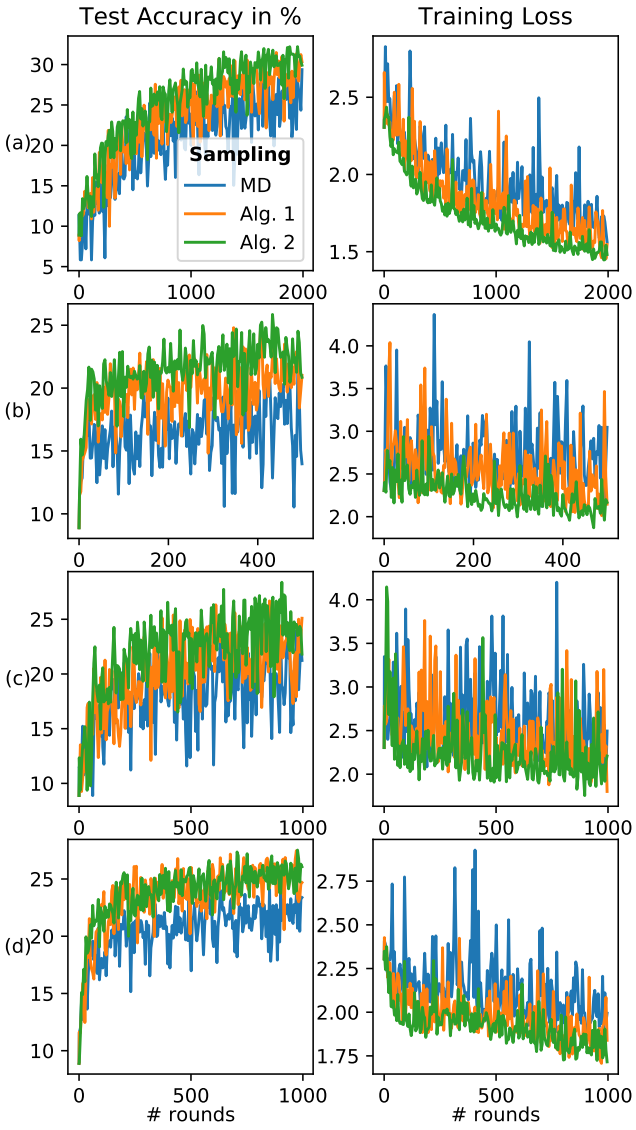

Fig. B.7.: We consider the federated dataset partitioned from CIFAR10 using a Dirichlet distribution with parameter $\alpha = 0.01$. We investigate the influence of N the number of SGD run by each client in the first two rows with $N = 10$ and $N = 500$ for $m = 10$ and the influence of sampled clients with $m = 5$ and $m = 20$ for $N = 100$ in the last two rows. For each dataset, we use respective learning rate $lr = \{0.1, 0.05, 0.05, 0.05\}$.

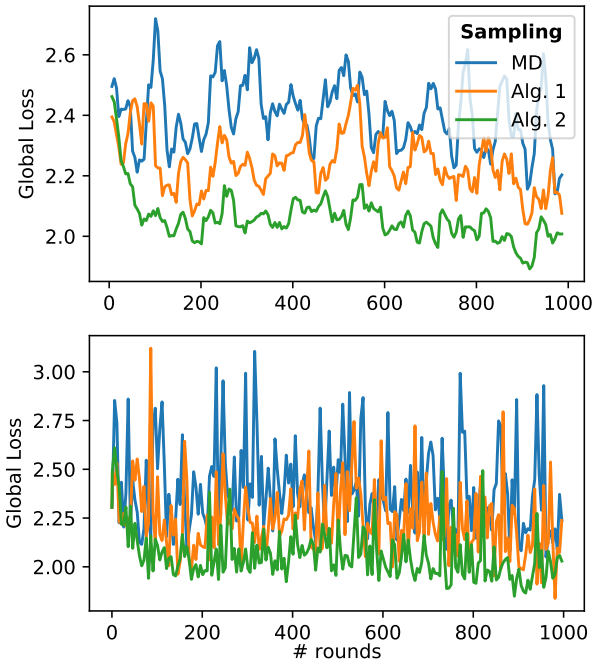

Fig. B.8.: Training loss convergence for FL with FedProx local loss function regularization ($\mu = 0.1$). We consider CIFAR10 partitioned with Dir($\alpha = 0.01$), learning rate $lr = 0.05$, $m = 10$ sampled clients, and $N = 100$ SGD.

We try a range of regularization term $\mu \in \{0.001, 0.01, 0.1, 1.\}$ and keep $\mu = 0.1$ maximizing the performances of FEDAVG with regularization and MD sampling. We notice in Figure B.8 that Algorithm 2 and 3 still outperform MD sampling.

Appendix of Chapter 4

C

C.1 Proof of Theorem 4.1

We first provide in Section C.1.2 the basic inequalities used in our proofs, and in Section C.1.3 the basic notations used to provide clearer proofs.

C.1.1 Bounding the convergence residual Σ

As defined in Section 4.2.5, the convergence residual is defined as

$$\Sigma := \sum_{i=1}^{M} q_i \, \mathbb{E}_{\boldsymbol{\xi}_i} \left[\left\| \nabla \mathcal{L}_i(\bar{\boldsymbol{\theta}}, \boldsymbol{\xi}_i) \right\|^2 \right]. \tag{C.1}$$

When considering that the clients gradient estimator are bounded by σ^2, then each client gradient estimator satisfies

$$\mathbb{E}_{\boldsymbol{\xi}_i} \left[\left\| \nabla \mathcal{L}_i(\bar{\boldsymbol{\theta}}, \boldsymbol{\xi}_i) - \nabla \mathcal{L}_i(\bar{\boldsymbol{\theta}}) \right\|^2 \right] \leq \sigma^2. \tag{C.2}$$

Under this assumption, we can bound Σ as follows

$$\Sigma \leq \sum_{i=1}^{M} q_i \left\| \nabla \mathcal{L}_i(\bar{\boldsymbol{\theta}}) \right\|^2 + q(n)\sigma^2 \tag{C.3}$$

$$\leq 2L \sum_{i=1}^{M} q_i \left[\mathcal{L}_i(\bar{\boldsymbol{\theta}}) - \mathcal{L}_i(\boldsymbol{\theta}_i^*) \right] + q(n)\sigma^2, \tag{C.4}$$

where the first inequality follows from the unbiasedness of the gradient estimator, Assumption 4.3, and the second inequality from the Lipschitz smoothness of every client loss function, Assumption 4.1. With both equation (C.3) and (C.4), the convergence guarantees of Theorem 4.1 can be extended to account for bounded gradient estimator bounded variance, an additional assumption unneeded in this work.

Tab. C.1.: Common Notation Summary (addition to Table 4.1).

Symbol	Description
M	Number of clients.
K	Number of local SGD.
η_g, η_l	Global/Local learning rate.
$\tilde{\eta}$	Effective learning rate, $\tilde{\eta} = \eta_l \eta_g$.
$\boldsymbol{\theta}^n$	Global model at server iteration n.
$\boldsymbol{\theta}_i^{n+1}$	Local update of client i on model $\boldsymbol{\theta}^n$.
$\boldsymbol{\theta}^*, \boldsymbol{\theta}_i^*$	Optimum of the federated problem (4.2)/client i.
$\boldsymbol{\theta}^{n,k}, \boldsymbol{\theta}_i^{n,k}$	Global/Local update after k SGD on global model $\boldsymbol{\theta}^n$.
α	Covariance parameter.
β	Defined in Theorem 4.1.
$\mathcal{L}(\cdot), \mathcal{L}_i(\cdot)$	Federated/local loss function.
$g_i(\cdot)$	SG. We have $\mathbb{E}_{\xi_i}[g_i(\cdot)] = \nabla \mathcal{L}_i(\cdot)$ with Assumption 4.3.
$\boldsymbol{\xi}_i$	Random batch of samples from client i of size B.
L	Lipschitz smoothness parameter, Assumption 4.1.
T_i	Computation time of client i.
t^n	Time at aggregation n.
T_i^n	Remaining computation time of client i at time t^n.
Δt^n	Time elapsed between two server aggregations.
$\rho_i(n)$	Last index at which a client i received its global model.
ρ	Highest sum of aggregation weights, i.e. $\rho \coloneqq \max(1, q(n))$

C.1.2 Basic Inequalities

We provide the following basic inequalities used in our proofs.

Let us consider f a L-Lipschitz smooth and convex function with optimum $\boldsymbol{x}^*$. For any vector $\boldsymbol{x}$ and $\boldsymbol{y}$, we have

$$\|\nabla f(\boldsymbol{x})\|^2 \leq 2L[f(\boldsymbol{x}) - f(\boldsymbol{x}^*)], \text{ and } \|\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y})\|^2 \leq L^2 \|\boldsymbol{x} - \boldsymbol{y}\|^2. \tag{C.5}$$

Let us consider g a convex function and d vectors $\{\boldsymbol{x}_k\}$ each with importance p_k such that $\sum_{k=1}^d p_k = 1$. With Jensen inequality, we have

$$g(\sum_{k=1}^d p_k \boldsymbol{x}_k) \leq \sum_{k=1}^d p_k g(\boldsymbol{x}_k). \tag{C.6}$$

Let us consider the random variable X, we have

$$\mathbb{E}\left[\|X - \mathbb{E}[X]\|^2\right] \leq \mathbb{E}\left[\|X\|^2\right]. \tag{C.7}$$

C.1.3 Additional Notation

In Table 4.1, we synthesize the different random variables associated to the clients aggregation weights. In Table C.1, we synthesize the remaining random variables.

We introduce the following notations to provide clear and compact proofs. Whenever considering a function $f(n,k)$, we define $f(n) \coloneqq 1/K \sum_{k=0}^{K-1} f(n,k)$, and $\bar{f}(N) \coloneqq 1/N \sum_{n=0}^{N-1} f(n)$. We introduce the following quantities

$$D(\boldsymbol{x},n,k) \coloneqq \mathbb{E}\left[\langle \sum_{i=1}^{M} q_i(n) \nabla \mathcal{L}_i(\boldsymbol{\theta}_i^{\rho_i(n),k}), \boldsymbol{\theta}^{n,k} - \boldsymbol{x}\rangle\right], \quad Q(n) \coloneqq \mathbb{E}\left[\left\|\boldsymbol{\theta}^{n+1,0} - \boldsymbol{\theta}^{n,0}\right\|^2\right], \tag{C.8}$$

$$R(n,k) \coloneqq \mathbb{E}\left[\left\|\sum_{i=1}^{M} \tilde{q}_i(n) g_i(\boldsymbol{\theta}_i^{\rho_i(n),k})\right\|^2\right], \quad S(n,k) \coloneqq \sum_{i=1}^{M} \tilde{q}_i(n)\, \mathbb{E}\left[\left\|g_i(\boldsymbol{\theta}_i^{\rho_i(n),k})\right\|^2\right], \tag{C.9}$$

$$Z(n,k) = \mathcal{L}^n(\boldsymbol{\theta}^{n,k}) - \mathcal{L}^n(\bar{\boldsymbol{\theta}}^n), \quad \Delta(n,k) \coloneqq \mathbb{E}\left[\left\|\boldsymbol{\theta}^{n,k+1} - \boldsymbol{x}\right\|^2\right] - \mathbb{E}\left[\left\|\boldsymbol{\theta}^{n,k} - \boldsymbol{x}\right\|^2\right], \tag{C.10}$$

$$\phi(n,k) \coloneqq \sum_{i=1}^{M} \tilde{q}_i(n)\, \mathbb{E}\left[\left\|\boldsymbol{\theta}_i^{\rho_i(n),k} - \boldsymbol{\theta}^{n,k}\right\|^2\right], \quad \sigma_1(n) \coloneqq \sum_{i=1}^{M} \tilde{q}_i(n)\, \mathbb{E}\left[\left\|\nabla \mathcal{L}_i(\bar{\boldsymbol{\theta}}^n, \boldsymbol{\xi}_i)\right\|^2\right], \tag{C.11}$$

$$\sigma_2(n) \coloneqq \sum_{i=1}^{M} \tilde{q}_i^2(n)\, \mathbb{E}\left[\left\|\nabla \mathcal{L}_i(\bar{\boldsymbol{\theta}}^n, \boldsymbol{\xi}_i)\right\|^2\right], \text{ and } \Xi(n,k) = \mathcal{L}^n(\boldsymbol{\theta}^{n,k}) - \mathcal{L}^n(\boldsymbol{x}). \tag{C.12}$$

Finally, we define $g_i(\boldsymbol{y}) = \nabla \mathcal{L}_i(\boldsymbol{y}, \boldsymbol{\xi}_i)$ the SG of client i evaluated on model parameters $\boldsymbol{y}$ and batch $\boldsymbol{\xi}_i$. We will thus write $g_i(\boldsymbol{\theta}_{i,k}^{\rho_i(n)})$ instead of $\nabla \mathcal{L}_i(\boldsymbol{\theta}_{i,k}^{\rho_i(n)}, \boldsymbol{\xi}_{i,k}^{\rho_i(n)})$.

C.1.4 Useful Lemmas

Lemma C.1. *Let us consider n vectors $\boldsymbol{x}_i, ..., \boldsymbol{x}_n$. We have*

$$\mathbb{E}_{S_n}\left[\left\|\sum_{i=1}^{M} \omega_i(n) \boldsymbol{x}_i\right\|^2\right] = \sum_{i=1}^{M} \gamma_i(n) \left\|\boldsymbol{x}_i\right\|^2 + \alpha \left\|\sum_{i=1}^{M} q_i(n) \boldsymbol{x}_i\right\|^2, \tag{C.13}$$

where $\gamma_i(n) = \mathbb{E}_{S_n}\left[\omega_i^2(n)\right] - \alpha q_i^2(n) \geq 0$, *and* $\gamma_i(n) \leq \beta q_i(n)$ *with* $\beta \coloneqq \max\{d_i(n) - \alpha q_i(n)\}$.

Proof. We first propose the following intermediary result. For any $y \in \mathbb{R}$, we have

$$\mathbb{E}_{S_n}\left[\omega_i(n)\omega_j(n)\right] y \leq \alpha q_i(n) q_j(n) y. \tag{C.14}$$

When $y \geq 0$, equation (C.14) follows directly from Property 4.1. When $y < 0$, equation (C.14) follows from providing a lower bound to the joint probability of two Bernoullis and the fact that $\alpha \in [-1, 1]$. Indeed, in that case, we have

$$\mathbb{P}(T_i^n \leq \Delta t^n, T_j^n \leq \Delta t^n) \geq \mathbb{P}(T_i^n \leq \Delta t^n)\mathbb{P}(T_j^n \leq \Delta t^n) \geq \alpha \mathbb{P}(T_i^n \leq \Delta t^n)\mathbb{P}(T_j^n \leq \Delta t^n). \tag{C.15}$$

Going back to the stochastic sum of vectors, we have

$$\mathbb{E}_{S_n}\left[\left\|\sum_{i=1}^{M} \omega_i(n)\boldsymbol{x}_i\right\|^2\right] = \sum_{i=1}^{M} \mathbb{E}_{S_n}\left[\omega_i^2(n)\right] \|\boldsymbol{x}_i\|^2 + \sum_{i=1}^{M}\sum_{\substack{j=1 \\ j\neq i}}^{M} \mathbb{E}_{S_n}\left[\omega_i(n)\omega_j(n)\right] \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle \tag{C.16}$$

$$\leq \sum_{i=1}^{M} \mathbb{E}_{S_n}\left[\omega_i^2(n)\right] \|\boldsymbol{x}_i\|^2 + \sum_{i=1}^{M}\sum_{\substack{j=1 \\ j\neq i}}^{M} \alpha q_i(n) q_j(n) \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle, \tag{C.17}$$

where we use equation (C.14) to obtain the inequality. In addition, we have

$$\sum_{i=1}^{M}\sum_{\substack{j=1 \\ j\neq i}}^{M} \langle q_i(n)\boldsymbol{x}_i, q_j(n)\boldsymbol{x}_j \rangle = \left\|\sum_{i=1}^{M} q_i(n)\boldsymbol{x}_i\right\|^2 - \sum_{i=1}^{M} q_i^2(n) \|\boldsymbol{x}_i\|^2. \tag{C.18}$$

Substituting equation (C.18) in equation (C.17) completes the first claim.

Considering that $\mathbb{E}_{S_n}\left[\omega_i^2(n)\right] = \mathrm{Var}\left[\omega_i(n)\right] + q_i^2(n) \geq q_i^2(n)$ and $\alpha \leq 1$, we have $\gamma_i(n) \geq 0$ which completes the second claim.

Finally, the third claim follows directly from the close-form of the clients aggregation weights, equation (4.5).

Remark. We can also provide the following lower bound for equation (C.18) using Jensen inequality

$$\sum_{i=1}^{M}\sum_{\substack{j=1\\ j\neq i}}^{M}\langle q_i(n)\boldsymbol{x}_i, q_j(n)\boldsymbol{x}_j\rangle \geq \left\|\sum_{i=1}^{M} q_i(n)\boldsymbol{x}_i\right\|^2 - \frac{\max q_i(n)}{q(n)}\left\|\sum_{i=1}^{M} q_i(n)\boldsymbol{x}_i\right\|^2 \geq 0. \quad \text{(C.19)}$$

Therefore, $\mathbb{E}_{S_n}\left[\left\|\sum_{i=1}^{M}\omega_i(n)\boldsymbol{x}_i\right\|^2\right]$ is linearly proportional to α.

□

Lemma C.2. *The following equation holds for any vector $\boldsymbol{x}$:*

$$\Delta(n) \leq -2\tilde{\eta}D(\boldsymbol{x}, n) + \tilde{\eta}^2\alpha q^2(n)R(n) + \tilde{\eta}^2\beta q(n)S(n). \quad \text{(C.20)}$$

Proof. We consider S_n, the set of participating clients at optimization round n, i.e. $S_n = \{n : T_i^n \leq \Delta t^n\}$. We have

$$\mathbb{E}_{S_n}\left[\left\|\boldsymbol{\theta}^{n,k+1} - \boldsymbol{\theta}^*\right\|^2\right] = \mathbb{E}_{S_n}\left[\left\|(\boldsymbol{\theta}^{n,k+1} - \boldsymbol{\theta}^{n,k}) + (\boldsymbol{\theta}^{n,k} - \boldsymbol{\theta}^*)\right\|^2\right] \quad \text{(C.21)}$$

$$= \left\|\boldsymbol{\theta}^{n,k} - \boldsymbol{\theta}^*\right\|^2 + 2\langle\mathbb{E}_{S_n}\left[\boldsymbol{\theta}^{n,k+1} - \boldsymbol{\theta}^{n,k}\right], \boldsymbol{\theta}^{n,k} - \boldsymbol{\theta}^*\rangle + \mathbb{E}_{S_n}\left[\left\|\boldsymbol{\theta}^{n,k+1} - \boldsymbol{\theta}^{n,k}\right\|^2\right]. \quad \text{(C.22)}$$

By construction, we have $\boldsymbol{\theta}^{n,k+1} - \boldsymbol{\theta}^{n,k} = -\tilde{\eta}\sum_{i=1}^{M}\omega_i(n)g_i(\boldsymbol{\theta}_i^{\rho_i(n),k})$. Taking the expectation over S_n, we can simplify the second term of equation (C.22) with $\mathbb{E}_{S_n}\left[\boldsymbol{\theta}^{n,k+1} - \boldsymbol{\theta}^{n,k}\right] = -\tilde{\eta}\sum_{i=1}^{M} q_i(n)g_i(\boldsymbol{\theta}_i^{\rho_i(n),k})$. Finally, using Lemma C.1, we can bound the third term. Therefore, we have

$$\mathbb{E}_{S_n}\left[\left\|\boldsymbol{\theta}^{n,k+1} - \boldsymbol{\theta}^*\right\|^2\right] = \left\|\boldsymbol{\theta}^{n,k} - \boldsymbol{\theta}^*\right\|^2 + 2\tilde{\eta}\langle\sum_{i=1}^{M} q_i(n)g_i(\boldsymbol{\theta}_i^{\rho_i(n),k}), \boldsymbol{\theta}^n - \boldsymbol{\theta}^*\rangle + \tilde{\eta}^2\sum_{i=1}^{M}\gamma_i(n)\left\|g_i(\boldsymbol{\theta}_i^{\rho_i(n),k})\right\|^2 + \tilde{\eta}^2\alpha\left\|\sum_{i=1}^{M} q_i(n)g_i(\boldsymbol{\theta}_i^{\rho_i(n),k})\right\|^2. \quad \text{(C.23)}$$

Considering $\gamma_i(n) \leq \beta q_i(n)$, taking the expected value over the iteration random batches $\boldsymbol{\xi}^{\rho_i(n),k}$, and finally taking the expected value over the remaining random variables gives

$$\Delta(n,k) \leq -2\tilde{\eta}D(\boldsymbol{x}, n, k) + \tilde{\eta}^2\alpha q^2(n)R(n,k) + \tilde{\eta}^2\beta q(n)S(n,k). \quad \text{(C.24)}$$

Taking the mean over K completes the proof.

□

Lemma C.3. *Under Assumption 4.3 and 4.1, and* $D \coloneqq 6\eta_l^2(K-1)^2L^2 \leq 1/2$, *we have*

$$\phi(n) \leq 4q(n)\tau \sum_{s=1}^{\tau} Q(n-s) + 4D\frac{1}{L}q^{-1}(n)Z(n) + 6\eta_l^2(K-1)^2\sigma_1(n), \tag{C.25}$$

$$\textit{and } S(n) \leq 12q(n)L^2\tau \sum_{s=1}^{\tau} Q(n-s) + 12Lq^{-1}(n)Z(n) + 6\sigma_1(n). \tag{C.26}$$

Proof. Let us decompose the difference $\boldsymbol{\theta}_i^{\rho_i(n),k} - \boldsymbol{\theta}^{n,k}$ as

$$\boldsymbol{\theta}_i^{\rho_i(n),k} - \boldsymbol{\theta}^{n,k} = \left[\boldsymbol{\theta}^{\rho_i(n)} - \eta_l \sum_{l=0}^{k-1} g_i(\boldsymbol{\theta}_i^{\rho_i(n),l})\right] - \left[\boldsymbol{\theta}^{n} - \eta_l \sum_{l=0}^{k-1}\sum_{i=1}^{M} \tilde{q}_i(n) g_i(\boldsymbol{\theta}_i^{\rho_i(n),l})\right]. \tag{C.27}$$

Using Jensen inequality, we split the difference between the global models and the one between the gradients to get

$$\left\|\boldsymbol{\theta}_i^{\rho_i(n),k} - \boldsymbol{\theta}^{n,k}\right\|^2 \leq 2\left\|\boldsymbol{\theta}^{\rho_i(n)} - \boldsymbol{\theta}^{n}\right\|^2 + 2\eta_l^2 k \sum_{l=0}^{k-1}\left\|g_i(\boldsymbol{\theta}_i^{\rho_i(n),l}) - \sum_{i=1}^{M}\tilde{q}_i(n) g_i(\boldsymbol{\theta}_i^{\rho_i(n),l})\right\|^2. \tag{C.28}$$

Therefore, by taking the expectations of equation (C.28) and summing over M gives

$$\begin{aligned}\phi(n,k) &\leq 2\sum_{i=1}^{M}\tilde{q}_i(n)\,\mathbb{E}\left[\left\|\boldsymbol{\theta}^{\rho_i(n)} - \boldsymbol{\theta}^{n}\right\|^2\right] \\ &\quad + 2\eta_l^2 k \sum_{l=0}^{k-1}\sum_{i=1}^{M}\tilde{q}_i(n)\,\mathbb{E}\left[\left\|g_i(\boldsymbol{\theta}_i^{\rho_i(n),l}) - \sum_{i=1}^{M}\tilde{q}_i(n) g_i(\boldsymbol{\theta}_i^{\rho_i(n),l})\right\|^2\right] \qquad \text{(C.29)} \\ &\leq 2\sum_{i=1}^{M}\tilde{q}_i(n)\,\mathbb{E}\left[\left\|\boldsymbol{\theta}^{\rho_i(n)} - \boldsymbol{\theta}^{n}\right\|^2\right] + 2\eta_l^2 k \sum_{l=0}^{k-1}\sum_{i=1}^{M}\tilde{q}_i(n)\,\mathbb{E}\left[\left\|g_i(\boldsymbol{\theta}_i^{\rho_i(n),l})\right\|^2\right], \qquad \text{(C.30)}\end{aligned}$$

where we see that $S(n,l)$ appears in the second term of equation (C.30). We consider now bounding $S(n,k)$, and first note that a stochastic gradient can be bounded as follow

$$\begin{aligned}\mathbb{E}\left[\left\|g_i(\boldsymbol{\theta}_i^{\rho_i(n),k})\right\|^2\right] &\leq 3\,\mathbb{E}\left[\left\|\nabla\mathcal{L}_i(\boldsymbol{\theta}_i^{\rho_i(n),k}, \boldsymbol{\xi}_{i,k}^{\rho_i(n)}) - \nabla\mathcal{L}_i(\boldsymbol{\theta}^{n,k}, \boldsymbol{\xi}_{i,k}^{\rho_i(n)})\right\|^2\right] \\ &\quad + 3\,\mathbb{E}\left[\left\|\nabla\mathcal{L}_i(\boldsymbol{\theta}^{n,k}, \boldsymbol{\xi}_i) - \nabla\mathcal{L}_i(\bar{\boldsymbol{\theta}}^{n}, \boldsymbol{\xi}_i)\right\|^2\right] + 3\,\mathbb{E}\left[\left\|\nabla\mathcal{L}_i(\bar{\boldsymbol{\theta}}^{n}, \boldsymbol{\xi}_i)\right\|^2\right].\end{aligned} \tag{C.31}$$

When summing equation (C.31) over M, and considering the clients loss functions Lipschitz smoothness, Assumption 4.1, we have

$$S(n,k) = \sum_{i=1}^{M} \tilde{q}_i(n)\, \mathbb{E}\left[\left\|g_i(\boldsymbol{\theta}_i^{\rho_i(n),k})\right\|^2\right] \leq 3L^2\phi(n,k) + 6Lq^{-1}(n)Z(n,k) + 3\sigma_1(n). \tag{C.32}$$

We also note the following intermediary results

$$\sum_{k=0}^{K-1} k \sum_{l=0}^{k-1} x_l \leq (K-1) \sum_{k=1}^{K-1} \sum_{l=0}^{k-1} x_l \leq (K-1)^2 \sum_{k=0}^{K-2} x_k \leq (K-1)^2 \sum_{k=0}^{K-1} x_k. \tag{C.33}$$

We substitute equation (C.32) in equation (C.30) such that D appears, take the mean over K to introduce $\phi(n)$ on the two sides of the equation, and use equation (C.33). We have

$$\begin{aligned}\phi(n) \leq 2\sum_{i=1}^{M} \tilde{q}_i(n)\, \mathbb{E}\left[\left\|\boldsymbol{\theta}^{\rho_i(n)} - \boldsymbol{\theta}^n\right\|^2\right] + D\phi(n) + 2D\frac{1}{L}q^{-1}(n)Z(n) \\ + 6\eta_l^2(K-1)^2\sigma_1(n).\end{aligned} \tag{C.34}$$

Finally, reminding that $D \leq 1/2$, which gives $1 - D \geq 1/2$, and using Assumption 4.4 to bound $\mathbb{E}\left[\left\|\boldsymbol{\theta}^{\rho_i(n)} - \boldsymbol{\theta}^n\right\|^2\right]$ with Jensen inequality completes the first claim for $\phi(n)$, i.e.

$$\mathbb{E}\left[\left\|\boldsymbol{\theta}^{\rho_i(n)} - \boldsymbol{\theta}^n\right\|^2\right] \leq \tau \sum_{s=1}^{\tau} \mathbb{E}\left[\left\|\boldsymbol{\theta}^{n-s+1} - \boldsymbol{\theta}^{n-s}\right\|^2\right] = \tau \sum_{s=1}^{\tau} Q(n-s). \tag{C.35}$$

Substituting the close-form of $\phi(n)$ in equation (C.32) completes the claim for $S(n,k)$.

□

Lemma C.4. *Under Assumption 4.2 and 4.3, we have*

$$-2D(\boldsymbol{x},n) \leq -2\Xi(n) + 4Lq(n)\tau \sum_{s=1}^{\tau} Q(n-s) + 4DZ(n) + 6\eta_l^2(K-1)^2 q(n) L\sigma_1(n). \tag{C.36}$$

Proof. Follows directly from using Lemma 12 in Khaled et al., 2020a on $D(\boldsymbol{x},n,k)$, taking the mean over K, and using Lemma C.3 to bound $\phi(n)$ completes the proof.

□

Lemma C.5. *Under Assumption 4.1 and 4.3, and considering $D \leq 1/2$, we have*

$$R(n) \leq 12L^2\tau \sum_{s=1}^{\tau} Q(n-s) + 24Lq^{-1}(n)Z(n) + 3D\sigma_1(n) + 6\sigma_2(n). \tag{C.37}$$

Proof.

$$\begin{aligned} R(n,k) \leq\, & 3\,\mathbb{E}\left[\left\|\sum_{i=1}^{M} \tilde{q}_i(n)\left[g_i(\boldsymbol{\theta}_i^{\rho_i(n),k}) - \nabla\mathcal{L}_i(\boldsymbol{\theta}^{n,k}, \boldsymbol{\xi}_{i,k}^{\rho_i(n)})\right]\right\|^2\right] \\ & + 3\,\mathbb{E}\left[\left\|\sum_{i=1}^{M} \tilde{q}_i(n)\left[\nabla\mathcal{L}_i(\boldsymbol{\theta}^{n,k}, \boldsymbol{\xi}_i) - \nabla\mathcal{L}_i(\boldsymbol{\theta}^{n,k})\right]\right\|^2\right] \\ & + 3\,\mathbb{E}\left[\left\|\sum_{i=1}^{M} \tilde{q}_i(n)\nabla\mathcal{L}_i(\boldsymbol{\theta}^{n,k})\right\|^2\right]. \end{aligned} \tag{C.38}$$

We respectively call the three terms of equation (C.38), $a(n,k)$, $b(n,k)$, and $c(n,k)$. Using the local loss functions Lipschitz smoothness, Assumption 4.1, and Jensen inequality, we can bound $a(n,k)$ as

$$a(n,k) \leq 3\sum_{i=1}^{M} \tilde{q}_i(n)\,\mathbb{E}\left[\left\|g_i(\boldsymbol{\theta}_i^{\rho_i(n),k}) - \nabla\mathcal{L}_i(\boldsymbol{\theta}^{n,k}, \boldsymbol{\xi}_{i,k}^{\rho_i(n)})\right\|^2\right] \leq 3L^2\phi(n,k). \tag{C.39}$$

Using the unbiasedness of the gradient estimator, Assumption 4.3, and the local loss function Lipschitz smoothness, Assumption 4.1, we can bound $b(n,k)$ as

$$b(n,k) = 3\sum_{i=1}^{M} \tilde{q}_i^2(n)\,\mathbb{E}\left[\left\|\nabla\mathcal{L}_i(\boldsymbol{\theta}^{n,k}, \boldsymbol{\xi}_i) - \nabla\mathcal{L}_i(\boldsymbol{\theta}^{n,k})\right\|^2\right] \tag{C.40}$$

$$\leq 3\sum_{i=1}^{M} \tilde{q}_i^2(n)\,\mathbb{E}\left[\left\|\nabla\mathcal{L}_i(\boldsymbol{\theta}^{n,k}, \boldsymbol{\xi}_i)\right\|^2\right] \tag{C.41}$$

$$\leq 6\sum_{i=1}^{M} \tilde{q}_i^2(n)\left[\mathbb{E}\left[\left\|\nabla\mathcal{L}_i(\boldsymbol{\theta}^{n,k}, \boldsymbol{\xi}_i) - \nabla\mathcal{L}_i(\bar{\boldsymbol{\theta}}^{n}, \boldsymbol{\xi}_i)\right\|^2\right] + \mathbb{E}\left[\left\|\nabla\mathcal{L}_i(\bar{\boldsymbol{\theta}}^{n}, \boldsymbol{\xi}_i)\right\|^2\right]\right] \tag{C.42}$$

$$\leq 12L\max_i(\tilde{q}_i(n))\left[\tilde{\mathcal{L}}^n(\boldsymbol{\theta}^{n,k}) - \tilde{\mathcal{L}}^n(\bar{\boldsymbol{\theta}}^{n})\right] + 6\sum_{i=1}^{M} \tilde{q}_i^2(n)\,\mathbb{E}\left[\left\|\nabla\mathcal{L}_i(\bar{\boldsymbol{\theta}}^{n}, \boldsymbol{\xi}_i))\right\|^2\right]. \tag{C.43}$$

Using the Lipschitz smoothness of the local loss functions, Assumption 4.1 and Jensen inequality, we can bound $c(n,k)$ as

$$c(n,k) \leq 3\,\mathbb{E}\left[\left\|\nabla\tilde{\mathcal{L}}^n(\boldsymbol{\theta}^{n,k}) - \nabla\tilde{\mathcal{L}}^n(\bar{\boldsymbol{\theta}}^n)\right\|^2\right] \leq 6L\left[\tilde{\mathcal{L}}^n(\boldsymbol{\theta}^{n,k}) - \tilde{\mathcal{L}}^n(\bar{\boldsymbol{\theta}}^n)\right]. \tag{C.44}$$

Substituting equation (C.39), equation (C.43), and equation (C.44) in equation (C.38), considering that $\max_i(\tilde{q}_i(n)) \leq 1$, and summing over K gives

$$R(n) \leq 3L^2\phi(n) + 18Lq^{-1}(n)Z(n) + 6\sigma_2(n) \tag{C.45}$$

Using Lemma C.3 to replace $\phi(n)$, and considering that $D \leq 1/2 < 1$ completes the proof.

□

Lemma C.6. *Under Assumption 4.1 and 4.3, considering that* $\gamma_i(n) \leq \beta q_i(n)$*, and considering* $12\rho^2\left[\alpha+\beta\right]\tilde{\eta}^2K^2\tau^2L^2 \leq 1/2$*, we have*

$$\bar{Q}(N) \leq 24\rho\left[2\alpha+\beta\right]\tilde{\eta}^2K^2L\bar{Z}(N) + 6\rho^2\left[\alpha D+2\beta\right]\tilde{\eta}^2K^2\Sigma_1(N) + 12\rho^2\alpha\tilde{\eta}^2K^2\Sigma_2(N). \tag{C.46}$$

Proof. Considering the proof of Lemma C.2, using the fact that $\gamma_i(n) \leq \beta q_i(n)$, and Jensen inequality, we have

$$\begin{aligned} Q(n) &\leq q^2(n)\alpha\tilde{\eta}^2\,\mathbb{E}\left[\left\|\sum_{i=1}^{M}\tilde{q}_i(n)\sum_{k=0}^{K-1}g_i(\boldsymbol{\theta}_i^{\rho_i(n),k})\right\|^2\right] \\ &\quad + q(n)\beta\tilde{\eta}^2\sum_{i=1}^{M}\tilde{q}_i(n)\,\mathbb{E}\left[\left\|\sum_{k=0}^{K-1}g_i(\boldsymbol{\theta}_i^{\rho_i(n),k})\right\|^2\right] \qquad \text{(C.47)} \\ &\leq q^2(n)\alpha\tilde{\eta}^2K^2R(n) + q(n)\beta\tilde{\eta}^2K^2S(n) \qquad \text{(C.48)} \end{aligned}$$

Using Lemma C.5 to bound $R(n)$ and Lemma C.3 to bound $S(n)$, we can thus bound $Q(n)$ with the previous global model distances to the optimum $Q(s)$, where $\max(0,n-\tau) \leq s \leq n-1$, we thus have

$$\begin{aligned} \frac{1}{\rho\tilde{\eta}^2K^2}Q(n) &\leq 12\rho\left[\alpha+\beta\right]\tau L^2\sum_{s=1}^{\tau}Q(n-s) + 12\left[2\alpha+\beta\right]LZ(n) \\ &\quad + 3\rho\left[\alpha D+2\beta\right]\sigma_1(n) + 6\rho\alpha\sigma_2(n). \qquad \text{(C.49)} \end{aligned}$$

We can thus define $A(n)$ and $B(n)$ such that the bound of of equation (C.49) can be rewritten as in equation (C.50), with its associated implications when taking the mean over N, reordering, and considering that $\tau A(n) \leq 1/2$:

$$Q(n) \leq A(n) \sum_{s=1}^{\tau} Q(n-s) + B(n) \Rightarrow \bar{Q}(N) = \frac{1}{N} \sum_{n=0}^{N-1} Q(n) \leq 2 \frac{1}{N} \sum_{n=0}^{N-1} B(n). \tag{C.50}$$

Therefore, considering $12\rho^2 \left[\alpha + \beta\right] \tilde{\eta}^2 K^2 \tau^2 L^2 \leq 1/2$ completes the proof.

□

C.1.5 Proof of Theorem 4.1

Proof. Using Lemma C.2, we have

$$\frac{1}{\tilde{\eta}} \Delta(n) \leq -2D(\boldsymbol{x}, n) + \rho^2 \alpha \tilde{\eta} R(n) + \rho \beta \tilde{\eta} S(n) \tag{C.51}$$

Using Lemma C.4 to bound $D(\boldsymbol{x}, n)$, Lemma C.5 to bound $R(n)$, Lemma C.3 to bound $S(n)$, and $3\rho \left[\alpha + \beta\right] \tilde{\eta} L \leq 1$, we get

$$\begin{aligned} \frac{1}{\tilde{\eta}} \Delta(n) \leq & -2\Xi(n) + 8\rho\tau L \sum_{s=1}^{\tau} Q(n-s) + 4DZ(n) + 6\rho\eta_l^2 (K-1)^2 L \sigma_1(n) \\ & + 12 \left[2\alpha + \beta\right] \rho \tilde{\eta} L Z(n) + 3\rho^2 \tilde{\eta} \left[\alpha D + 2\beta\right] \sigma_1(n) + 6\rho^2 \alpha \tilde{\eta} \sigma_2(n). \end{aligned} \tag{C.52}$$

When considering the following intermediary result

$$\sum_{n=0}^{N-1} K\Delta(n) = \mathbb{E}\left[\left\|\boldsymbol{\theta}^{KN} - \boldsymbol{x}\right\|^2\right] - \left\|\boldsymbol{\theta}^0 - \boldsymbol{x}\right\|^2 \geq -\left\|\boldsymbol{\theta}^0 - \boldsymbol{x}\right\|^2, \tag{C.53}$$

reordering the terms, and taking the mean over N, we get

$$\begin{aligned} 2\bar{\Xi}(N) \leq & \frac{1}{\tilde{\eta} KN} \mathbb{E}\left[\left\|\boldsymbol{\theta}^0 - \boldsymbol{x}\right\|^2\right] + 8\rho L \tau^2 \bar{Q}(N) + 4D\bar{Z}(N) + 6\rho\eta_l^2 (K-1)^2 L \Sigma_1(N) \\ & + 12\rho \left[2\alpha + \beta\right] \tilde{\eta} L \bar{Z}(N) + 3\rho^2 \left[\alpha D + 2\beta\right] \tilde{\eta} \Sigma_1(N) + 6\rho^2 \alpha \tilde{\eta} \Sigma_2(N). \end{aligned} \tag{C.54}$$

Using Lemma C.6 to bound $\bar{Q}(N)$, and with $\nu = 16\rho L$, we have

$$
\begin{aligned}
2\bar{\Xi}(N) \leq \frac{1}{\tilde{\eta}KN}\mathbb{E}\left[\left\|\boldsymbol{\theta}^0 - \boldsymbol{x}\right\|^2\right] + 4D\bar{Z}(N) + 6\rho\eta_l^2(K-1)^2L\Sigma_1(N) \\
+ 12\rho\left[2\alpha+\beta\right]\left[\tilde{\eta}+\nu\tilde{\eta}^2K^2\tau^2\right]L\bar{Z}(N) + 3\rho^2\left[\alpha D + 2\beta\right]\left[\tilde{\eta}+\nu\tilde{\eta}^2K^2\tau^2\right]\Sigma_1(N) \\
+ 6\rho^2\alpha\left[\tilde{\eta}+\nu\tilde{\eta}^2K^2\tau^2\right]\Sigma_2(N). \qquad \text{(C.55)}
\end{aligned}
$$

We note that when $\bar{\Xi}(N) \leq 0$, the claim follows directly. Therefore, we consider $\bar{\Xi}(N) \geq 0$ for the rest of this proof. We first note that

$$
\bar{Z}(N) = \bar{\Xi}(N) + R(\{\mathcal{L}^n\}), \qquad \text{(C.56)}
$$

and consider η_l such that

$$
2 - 4D - 12\rho\left[2\alpha+\beta\right]\left[\tilde{\eta}+\nu\tilde{\eta}^2K^2\tau^2\right]L \geq 1, \qquad \text{(C.57)}
$$

which gives

$$
\begin{aligned}
\bar{\Xi}(N) \leq \frac{1}{\tilde{\eta}KN}\mathbb{E}\left[\left\|\boldsymbol{\theta}^0 - \boldsymbol{x}\right\|^2\right] + 4DR(\{\mathcal{L}^n\}) + 6\rho\eta_l^2(K-1)^2L\Sigma_1(N) \\
+ 12\rho\left[2\alpha+\beta\right]\left[\tilde{\eta}+\nu\tilde{\eta}^2K^2\tau^2\right]LR(\{\mathcal{L}^n\}) \\
+ 3\rho^2\left[\alpha D + 2\beta\right]\left[\tilde{\eta}+\nu\tilde{\eta}^2K^2\tau^2\right]\Sigma_1(N) \\
+ 6\rho^2\alpha\left[\tilde{\eta}+\nu\tilde{\eta}^2K^2\tau^2\right]\Sigma_2(N). \qquad \text{(C.58)}
\end{aligned}
$$

The 5th term can be simplified with the third one. Indeed, we consider a local learning rate such that $3\rho^2\tilde{\eta}L \leq 1$, $48\rho^3\tilde{\eta}^2K^2\tau^2L^2 \leq 1$, and we remind that $\alpha \leq 1$. We thus have

$$
\begin{aligned}
\bar{\Xi}(N) \leq \frac{1}{\tilde{\eta}KN}\mathbb{E}\left[\left\|\boldsymbol{\theta}^0 - \boldsymbol{x}\right\|^2\right] + \mathcal{O}\left(\eta_l^2(K-1)^2\left[R(\{\mathcal{L}^n\}) + \Sigma_1(N)\right]\right) \\
+ \mathcal{O}\left(\alpha\left[\tilde{\eta}+\tilde{\eta}^2K^2\tau^2\right]\left[R(\{\mathcal{L}^n\}) + \Sigma_2(N)\right]\right) \\
+ \mathcal{O}\left(\beta\left[\tilde{\eta}+\tilde{\eta}^2K^2\tau^2\right]\left[R(\{\mathcal{L}^n\}) + \Sigma_1(N)\right]\right). \qquad \text{(C.59)}
\end{aligned}
$$

With

$$
\left\|\nabla\mathcal{L}_i(\boldsymbol{\theta},\xi)\right\|^2 \leq 2\left\|\nabla\mathcal{L}_i(\boldsymbol{\theta},\xi) - \nabla\mathcal{L}_i(\bar{\boldsymbol{\theta}},\xi)\right\|^2 + 2\left\|\nabla\mathcal{L}_i(\bar{\boldsymbol{\theta}},\xi)\right\|^2, \qquad \text{(C.60)}
$$

we have

$$
\Sigma_2(N) \leq \max q_i(n)\Sigma_1(N) \leq \max q_i(n)\left[4LR(\mathcal{L}^n) + 2\Sigma\right]. \qquad \text{(C.61)}
$$

Finally, substituting equation (C.56) and (C.61) in equation (C.59) completes the proof.

□

C.1.6 Simplifying the constraint on the learning rate

The constraints on the learning rate can be summarized as $D = 6\eta_l^2(K-1)^2L^2 \leq 1/2$ (Lemma C.3), $12\rho^2\left[\alpha+\beta\right]\tilde{\eta}^2K^2\tau^2L^2 \leq 1/2$ (Lemma C.6), $3\rho\left[\alpha+\beta\right]\tilde{\eta}L \leq 1$ (Theorem 4.1), $2-4D-12\rho\left[2\alpha+\beta\right]\left[\tilde{\eta}+\nu\tilde{\eta}^2K^2\tau^2\right]L \geq 1$ (Theorem 4.1), $3\rho^2\tilde{\eta}L \leq 1$ (Theorem 4.1), and $48\rho^3\tilde{\eta}^2K^2\tau^2L^2 \leq 1$ (Theorem 4.1).

We note that $\alpha \leq 1$, and $\beta \leq 1$. We thus propose the following sufficient conditions to satisfy the conditions above

$$48\eta_l^2(K-1)^2L^2 \leq 1, 144\rho^2\tilde{\eta}L \leq 1, \text{ and } 2304\rho^3\tilde{\eta}^2K^2\tau^2L^2 \leq 1, \tag{C.62}$$

which can further be simplified with

$$\eta_l \leq \frac{1}{48KL}\min\left(1, \frac{1}{3\rho^2\eta_g(\tau+1)}\right). \tag{C.63}$$

C.2 Proof of Theorem 4.2

In this proof, we consider $\tilde{\mathcal{L}}^n = q^{-1}(n)\mathcal{L}^n$.

C.2.1 Useful Lemma

Lemma C.7. *The difference between the gradients of $\mathcal{L}(\boldsymbol{\theta})$ and $\mathcal{L}(\boldsymbol{\theta})$ can be bounded as follow*

$$\begin{aligned}\left\|\nabla\mathcal{L}(\boldsymbol{\theta}) - \nabla\tilde{\mathcal{L}}^n(\boldsymbol{\theta})\right\|^2 \leq 4L\chi_n^2[\tilde{\mathcal{L}}^n(\boldsymbol{\theta}) - \sum_{j\in W_n}\tilde{s}_j(n)\mathcal{L}_j(\boldsymbol{\theta}_j^*)] \\ + 4L\sum_{j\notin W_n} r_j[\mathcal{L}_j(\boldsymbol{\theta}) - \mathcal{L}_j(\boldsymbol{\theta}_j^*)],\end{aligned} \tag{C.64}$$

where $W_n = \{j : s_j(n) > 0\}$ and $\chi_n^2 = \sum_{j\in W_n}(r_j - \tilde{s}_j(n))^2/\tilde{s}_j(n)$.

Proof. We have $\sum_{j=1}^{J} s_j(n) = \sum_{i=1}^{M} q_i(n) = q(n)$. Hence, by definition of $\mathcal{L}(\boldsymbol{\theta})$ and $\mathcal{L}^n(\boldsymbol{\theta})$, we have

$$\nabla\mathcal{L}(\boldsymbol{\theta}) - \nabla\tilde{\mathcal{L}}^n(\boldsymbol{\theta}) = \sum_{j=1}^{J}(r_j - \tilde{s}_j(n))\nabla\mathcal{L}_j(\boldsymbol{\theta}) \tag{C.65}$$

$$= \sum_{j\in W_n} \frac{r_j - \tilde{s}_j(n)}{\sqrt{\tilde{s}_j(n)}}\sqrt{\tilde{s}_j(n)}\nabla\mathcal{L}_j(\boldsymbol{\theta}) + \sum_{j\notin W_n} r_j\nabla\mathcal{L}_j(\boldsymbol{\theta}). \tag{C.66}$$

Applying Jensen and Cauchy-Schwartz inequality gives

$$\left\|\nabla\mathcal{L}(\boldsymbol{\theta}) - \nabla\tilde{\mathcal{L}}^n(\boldsymbol{\theta})\right\|^2 \leq 2\left\|\sum_{j\in W_n} \frac{r_j - \tilde{s}_j(n)}{\sqrt{\tilde{s}_j(n)}}\sqrt{\tilde{s}_j(n)}\nabla\mathcal{L}_j(\boldsymbol{\theta})\right\|^2$$

$$+ 2\left\|\sum_{j\notin W_n} r_j\nabla\mathcal{L}_j(\boldsymbol{\theta})\right\|^2 \tag{C.67}$$

$$\leq 2\left[\sum_{j\in W_n} \frac{(r_j - \tilde{s}_j(n))^2}{\tilde{s}_j(n)}\right]\sum_{j=1}^{J}\tilde{s}_j(n)\left\|\nabla\mathcal{L}_j(\boldsymbol{\theta})\right\|^2$$

$$+ 2\left[\sum_{j\notin W_n} r_j\right]\sum_{j\notin W_n} r_j\left\|\nabla\mathcal{L}_j(\boldsymbol{\theta})\right\|^2 \tag{C.68}$$

Considering the Lipschitz smoothness of the clients loss function, and $\sum_{j\notin W_n} r_j \leq 1$ completes the proof.

□

C.2.2 Proof of Theorem 4.2

Proof. Using Jensen inequality and Lemma C.7 gives

$$\|\nabla\mathcal{L}(\boldsymbol{\theta})\|^2 \leq 2\left\|\nabla\mathcal{L}(\boldsymbol{\theta}) - \frac{1}{q(n)}\nabla\mathcal{L}^n(\boldsymbol{\theta})\right\|^2 + 2\left\|\frac{1}{q(n)}\nabla\mathcal{L}^n(\boldsymbol{\theta})\right\|^2 \tag{C.69}$$

$$\leq 4L\left[\chi_n^2\frac{1}{q(n)} + \frac{1}{q^2(n)}\right][\mathcal{L}^n(\boldsymbol{\theta}) - \mathcal{L}^n(\bar{\boldsymbol{\theta}}^n)]$$

$$+ \chi_n^2\frac{1}{q(n)}4L[\mathcal{L}^n(\bar{\boldsymbol{\theta}}^n) - \sum_{j\in W_n} s_j(n)\mathcal{L}_j(\boldsymbol{\theta}_j^*)]$$

$$+ 4L\sum_{j\notin W_n} r_j[\mathcal{L}_j(\boldsymbol{\theta}) - \mathcal{L}_j(\boldsymbol{\theta}_j^*)] \tag{C.70}$$

We take the maximum of χ_n^2 and $q(n)$, the mean over the KN serial SGD steps, and use Theorem 4.1 to complete the proof .

□

C.3 Applying Theorem 4.3

This section extends Section 4.4, where we apply Theorem 4.3 to centralized learning (Section C.3.1) and synchronous FEDAVG with unbiased and biased client sampling (Section C.3.2 and C.3.3 respectively).

C.3.1 Centralized Learning

In this setting, one client, i.e. $M = 1$, learns a predictive model on its own data. In this case, we always have $\tilde{q}_1(n) = 1$, and the resulting optimization problem is always proportional to $\mathcal{L} = \mathcal{L}_1$ which thus gives $R(\{\mathcal{L}^n\}) \leq R(\mathcal{L}) = 0$. There is no gradient delay ($\tau = 1$), while the clients always participate at each optimization round ($\alpha = 1$ and $\beta = 0$), while the global learning rate is redundant with the local learning rate ($\eta_g = 1$). The server performs KN SGD steps. All these considered elements give

$$\epsilon = \mathcal{O}\left(\frac{\left\|\boldsymbol{\theta}^0 - \boldsymbol{\theta}^*\right\|^2}{\eta_l KN}\right) + \mathcal{O}\left(\eta_l \, \mathbb{E}_{\boldsymbol{\xi}}\left[\left\|\nabla \mathcal{L}(\boldsymbol{x}, \boldsymbol{\xi})\right\|^2\right]\right). \tag{C.71}$$

With equation (C.71), we retrieve standard convergence guarantees for centralized ML derived in Bottou et al., 2016.

C.3.2 Unbiased client sampling ($q_i(n) = p_i$)

We define by S_n the set of sampled clients performing their local work at optimization step n. Setting $\Delta t^n = \max_{i \in S_n} T_i$, with $T_i = \infty$ for the clients that are not sampled, and thus not in S_n, gives $\mathbb{P}(T_i \leq \Delta t^n) = \mathbb{P}(i \in S_n)$. S_n is independent from the clients hardware capabilities and is decided by the server. This allows to pre-compute $\mathbb{P}(T_i \leq \Delta t^n)$ and to allocate to each client the aggregation weight d_i such that $q_i = p_i$.

Standard unbiased client sampling schemes include sampling m clients uniformly without replacement (Xiang Li et al., 2020) or sampling m clients according to a Multinomial distribution (T. Li, Sahu, Zaheer, Sanjabi, Talwalkar, and Smith, 2018a). Fraboni, Vidal, Kameni, et al. (2022a) shows that both Uniform and MD sampling satisfy Property 4.1. In

particular, in those setting, the term $\alpha \leq 1$ is proportional to m, the amount of sampled clients, while $1 \geq \beta > 0$ is inversely proportional to m. We get

$$\epsilon = \mathcal{O}\left(\frac{1}{\eta_g \eta_l K N}\right) + \mathcal{O}\left(\eta_g \eta_l \alpha \frac{1}{M} \Sigma\right) + \mathcal{O}\left(\eta_l^2 (K-1)^2 \Sigma\right) + \mathcal{O}\left(\eta_g \eta_l \beta \Sigma\right). \quad \text{(C.72)}$$

The second term, proportional to α/M, is reduced at the expense of the introduction of a fourth term proportional to β. In turn, it still provides faster optimization rounds with $\Delta t^n = \max_{i \in S_n} T_i$ and $N = \mathcal{O}\left(T / \mathbb{E}\left[\max_{i \in S_n} T_i\right]\right)$. FedAvg with client sampling generalizes FedAvg with full client participation ($\alpha = 1$ and $\beta = 0$).

C.3.3 Biased client sampling ($q_i(n) \neq p_i$)

The condition $q_i(n) = p_i$ imposes the design of new client sampling based on the clients data heterogeneity. Nevertheless, we show convergence of biased client samplings where m clients are selected according to a deterministic criterion, e.g. when selecting the m clients with the highest loss (Cho et al., 2020), or when selecting the m clients with the most available computation resources (Nishio and Yonetani, 2019). In this case, $\mathbb{P}(i \in S_n) = 0/1$, with 1 if a client satisfies the criterion and 0 otherwise. In this case, no weighting scheme can make an optimization round unbiased. We also have $\mathbb{P}(\{i, j\} \in S_n) = \mathbb{P}(i \in S_n)\mathbb{P}(j \in S_n)$, which gives $\alpha = 1$ with $\beta = 0$. Without modification, this client sampling cannot satisfy the relaxed sufficient conditions of Theorem 4.3 and thus converges to a suboptimum point. This drawback can be mitigated by allocating a part of time in the window W to sample clients according to the criterion, and the rest of the window to consider clients such that $q_i = p_i$ is satisfied over W optimization rounds. By denoting ϵ_{FedAvg} the convergence guarantees (4.33), we have

$$\epsilon = \epsilon_{\text{FedAvg}} + \mathcal{O}\left(\eta_g \eta_l (W-1) K\right). \quad \text{(C.73)}$$

We note that equation (C.73) provides a looser bound than equation (4.33) in term of optimization rounds N. Still, this bound is informative and shows that, with minor changes, biased clients sampling based on a deterministic criterion can be proven to converge to the FL optimum.

C.4 Additional Experiments

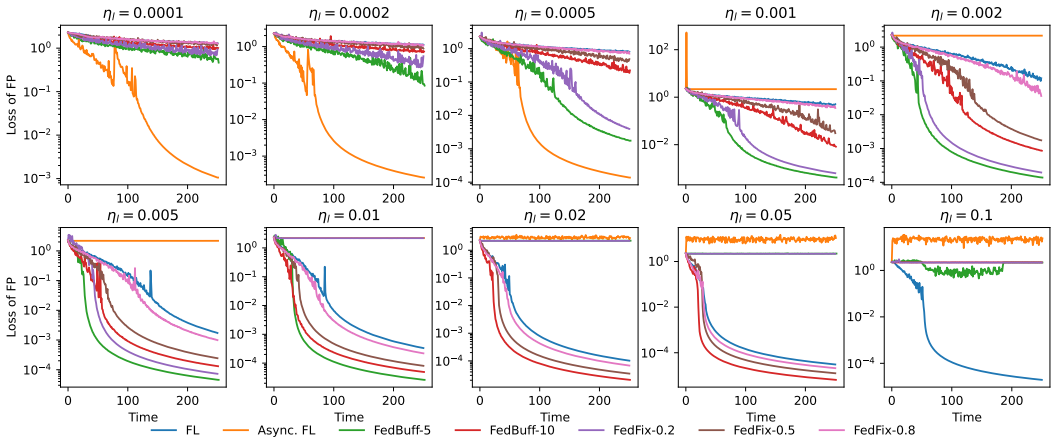

Fig. C.1.: Evolution of federated problem (4.2) (FP) loss for CIFAR10 and time scenario $F80$, with $M = 20$ and $K = 10$.

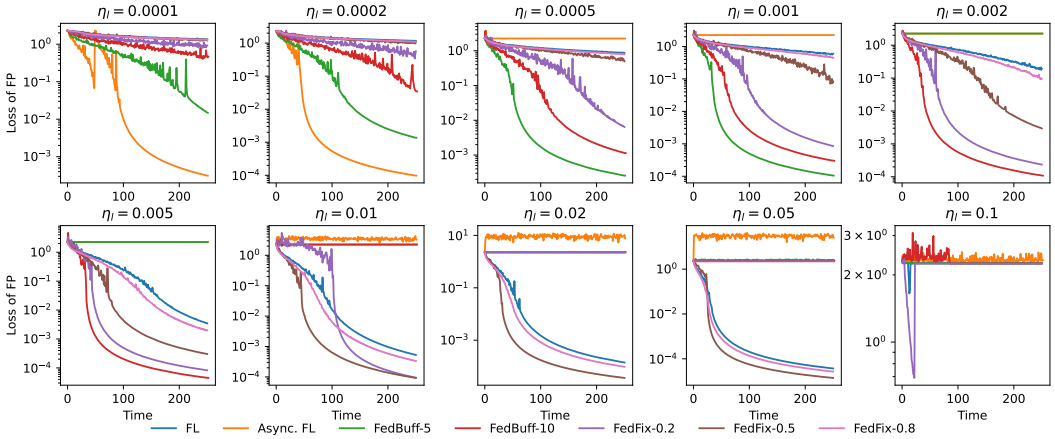

Fig. C.2.: Evolution of federated problem (4.2) (FP) loss for CIFAR10 and time scenario $F80$, with $M = 50$ and $K = 10$.

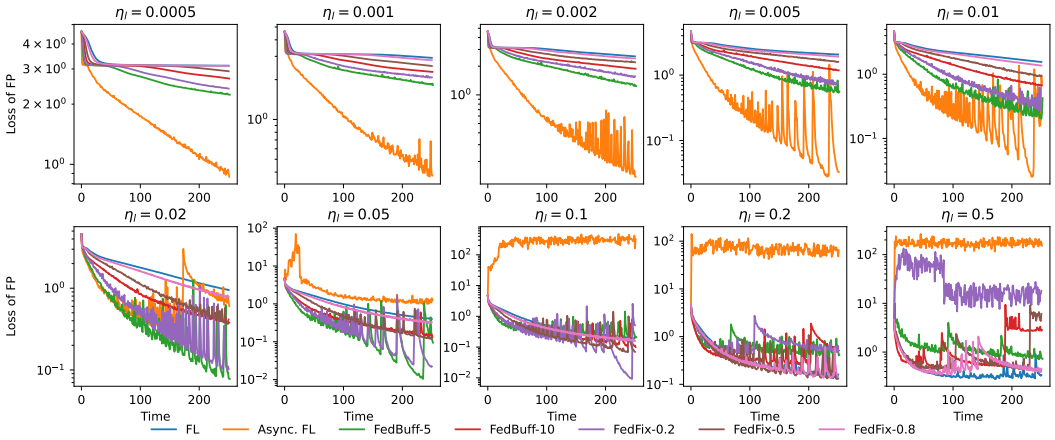

Fig. C.3.: Evolution of federated problem (4.2) (FP) loss for Shakespeare and time scenario $F80$, with $M = 20$ and $K = 10$.

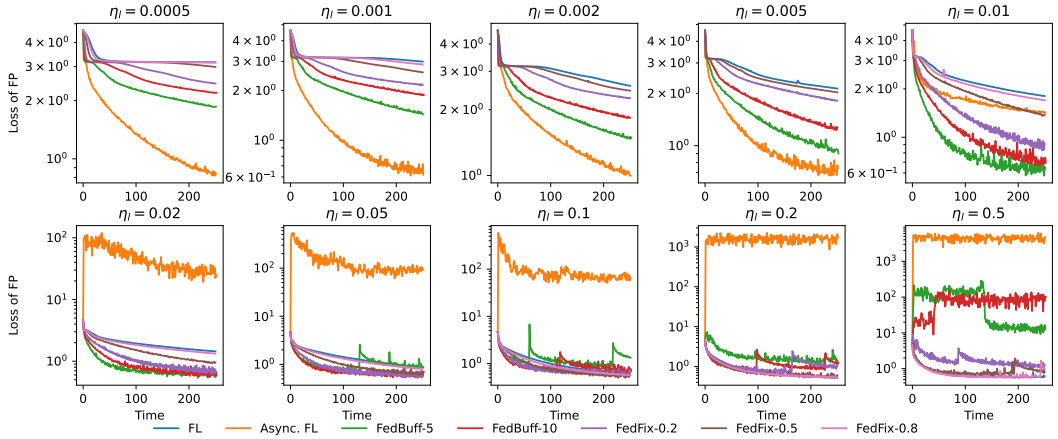

Fig. C.4.: Evolution of federated problem (4.2) (FP) loss for Shakespeare and time scenario $F80$, with $M = 50$ and $K = 10$.

Appendix of Chapter 5

D

D.1 When fine tuning does not guarantee unlearning: example on linear regression

Let us consider a linear regression optimization, with feature matrix $\boldsymbol{X}$ and predictions $\boldsymbol{y}$ such that the loss function f is defined as

$$f(\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{\theta}) = \frac{1}{2} \left[\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}\right]^T \left[\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}\right]. \tag{D.1}$$

In this example, we assume there are more features than data samples, which makes $\boldsymbol{X}^T\boldsymbol{X}$ a singular matrix. While f is convex, f has more than one global optimum. Any model with parameter $\boldsymbol{\theta}^*$ such that

$$\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\theta}^* = \boldsymbol{X}^T\boldsymbol{y} \tag{D.2}$$

is a global optimum. When $\boldsymbol{X}^T\boldsymbol{X}$ is non-singular, we retrieve the unique optimum in close-form $\boldsymbol{\theta}^* = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y}$. We show with this simple example that, upon unlearning a data sample, no amount of fine-tuning on the model $\boldsymbol{\theta}^*$ can lead to the same model obtained when retraining from a random initial model. We differentiate between $(\boldsymbol{X}, \boldsymbol{y})$ and $(\boldsymbol{X}_{-1}, \boldsymbol{y}_{-1})$ our data with and without a given data point.

Optimizing f, as defined in equation (D.1), with N steps of gradient descent, learning rate η, and initial model $\boldsymbol{\theta}_0$ gives model parameters $\boldsymbol{\theta}^N$ defined as

$$\boldsymbol{\theta}^N = \underbrace{\left[I - \eta\boldsymbol{X}^T\boldsymbol{X}\right]^N}_{A(\boldsymbol{X},N)}\boldsymbol{\theta}^0 + \underbrace{\eta\sum_{n=0}^{N-1}\left[I - \eta\boldsymbol{X}^T\boldsymbol{X}\right]^n\boldsymbol{X}^T\boldsymbol{y}}_{B(\boldsymbol{X},\boldsymbol{y},N)}. \tag{D.3}$$

We first note that we retrieve the standard form for the global optimum of linear regression when $\boldsymbol{X}^T\boldsymbol{X}$ is non-singular as $\lim_{n\to\infty} A(\boldsymbol{X}, n) = 0$ and $\lim_{n\to\infty} B(\boldsymbol{X}, \boldsymbol{y}, n) = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y}$. In the general form accounting for the singular case, at least one eigenvalue of $A(\boldsymbol{X}, N)$ is equal to 1 independently from the amount of gradient descent steps N. Hence, the parameters of the model obtained with gradient descent optimization always depend from the ones of the initial model $\boldsymbol{\theta}^0$. Hence, when unlearning our data sample from $\boldsymbol{\theta}^N$, the resulting trained model still depends of that data samples. Indeed, if we compare

the model $\boldsymbol{\theta}_{-1}^{\tilde{N}}$ trained on the data samples $(\boldsymbol{X}_{-1}, \boldsymbol{y}_{-1})$, to the model $\phi_{-1}^{\tilde{N}}$ obtained after fine-tuning the model $\boldsymbol{\theta}^N$ with $\tilde{N}$ server aggregations, we have

$$\phi_{-1}^{\tilde{N}} - \boldsymbol{\theta}_{-1}^{\tilde{N}} = A(\boldsymbol{X}_{-1}, \tilde{N})A(\boldsymbol{X}, N)\boldsymbol{\theta}^0 + A(\boldsymbol{X}_{-1}, \tilde{N})B(\boldsymbol{X}, \boldsymbol{y}, N). \tag{D.4}$$

D.2 Forgetting a Single Client with IFU, Proof of Theorem 5.1

We first consider the case where clients perform $K = 1$ SGD in Section D.2.1 before considering the general case $K \geq 1$ in Section D.2.2.

D.2.1 Proof of Theorem 5.1 for $K = 1$

Proof. We define by $\boldsymbol{\theta}^N = \text{FEDAVG}(I, N)$ and $\boldsymbol{\phi}^N = \text{FEDAVG}(I_{-c}, N)$ the models trained with FEDAVG on $\boldsymbol{\theta}_0$ with respectively all the clients, i.e. I, and all the clients but client c, i.e. I_{-c}, performing $K = 1$ GD step.

When clients perform $K = 1$ GD step, two consecutive global models can be related, when training with clients in I as a simple GD step, i.e.

$$\boldsymbol{\theta}^{n+1} = \boldsymbol{\theta}^n - \eta \nabla f_I(\boldsymbol{\theta}^n). \tag{D.5}$$

By considering the same process for I_{-c} and with Assumption 5.1, we get

$$\begin{aligned} \boldsymbol{\phi}^{n+1} - \boldsymbol{\theta}^{n+1} &= \boldsymbol{\phi}^n - \boldsymbol{\theta}^n - \eta \left[\nabla f_{I_{-c}}(\boldsymbol{\phi}^n) - \nabla f_I(\boldsymbol{\theta}^n)\right] && \text{(D.6)} \\ &= \left[I - \eta H_{I_{-c}}(\boldsymbol{\theta}^n)\right] \left[\boldsymbol{\phi}^n - \boldsymbol{\theta}^n\right] - \eta \left[\nabla f_{I_{-c}}(\boldsymbol{\theta}^n) - \nabla f_I(\boldsymbol{\theta}^n)\right]. && \text{(D.7)} \end{aligned}$$

$H_{I_{-c}}(\boldsymbol{\theta}^n)$ is semi-positive, Assumption 5.1. Let us define $\sigma_{\max}(H_{I_{-c}}(\boldsymbol{\theta}^n))$ the highest eigenvalue of $H_{I_{-c}}(\boldsymbol{\theta}^n)$. When consider that $\eta \leq 1/\sigma_{\max}(H_{I_{-c}}(\boldsymbol{\theta}^n))$, and due to the subadditivity of the norm, we get the following recurrent inequality

$$\left\|\boldsymbol{\phi}^{n+1} - \boldsymbol{\theta}^{n+1}\right\|_2 \leq \eta \left\|\nabla f_I(\boldsymbol{\theta}^n) - \nabla f_{I_{-c}}(\boldsymbol{\theta}^n)\right\|_2 + \left\|\boldsymbol{\phi}^n - \boldsymbol{\theta}^n\right\|_2, \tag{D.8}$$

which when developed completes the proof when clients perform $K = 1$ GD. □

D.2.2 Proof of Theorem 5.1 for $K \geq 1$

Proof. We maintain the definitions of $\boldsymbol{\theta}^n$ and $\boldsymbol{\phi}^n$ introduced in Section D.2.1. To account for the amount of local work K, we introduce $\boldsymbol{\theta}_i^{n,k}$ the model of client i after k GD steps performed on global model $\boldsymbol{\theta}^n$. We apply a similar reasoning for $\boldsymbol{\phi}_i^{n,k}$.

With Assumption 5.1, we have

$$\nabla f_i(\boldsymbol{\phi}_i^{n,k}) = \nabla f_i(\boldsymbol{\theta}_i^{n,k}) + H_i(\boldsymbol{\theta}_i^{n,k}) \left(\boldsymbol{\phi}_i^{n,k} - \boldsymbol{\theta}_i^{n,k}\right), \tag{D.9}$$

which gives

$$\begin{aligned}
\boldsymbol{\phi}_i^{n,k+1} - \boldsymbol{\theta}_i^{n,k+1} &= \left(\boldsymbol{\phi}_i^{n,k+1} - \boldsymbol{\phi}_i^{n,k}\right) - \left(\boldsymbol{\theta}_i^{n,k+1} - \boldsymbol{\theta}_i^{n,k}\right) + \left(\boldsymbol{\phi}_i^{n,k} - \boldsymbol{\theta}_i^{n,k}\right) && \text{(D.10)}\\
&= -\eta \left[\nabla f_i\left(\boldsymbol{\phi}_i^{n,k}\right) - \nabla f_i\left(\boldsymbol{\theta}_i^{n,k}\right)\right] + \left(\boldsymbol{\phi}_i^{n,k} - \boldsymbol{\theta}_i^{n,k}\right) && \text{(D.11)}\\
&= \left[I - \eta H_i(\boldsymbol{\theta}_i^{n,k})\right] \left(\boldsymbol{\phi}_i^{n,k} - \boldsymbol{\theta}_i^{n,k}\right) && \text{(D.12)}\\
&= \left[\prod_{r=0}^{k} \left[I - \eta H_i(\boldsymbol{\theta}_i^{n,r})\right]\right] (\boldsymbol{\phi}^n - \boldsymbol{\theta}^n), && \text{(D.13)}
\end{aligned}$$

where the third equality follows from equation (D.9), and the fourth from expanding the recurrent equation. For the rest of this work, we define $Q_i^n = \prod_{k=0}^{K-1} \left[I - \eta H_i(\boldsymbol{\theta}_i^{n,k})\right]$.

Using equation (D.13), we relate the difference between two global models with every client in I and in I_c. When removing client c the clients' importance changes. We consider importance p_i when training with I. Instead, when training with clients in I_c, we consider the regularized importance $q_i = p_i/(1-p_c)$ for the remaining clients and $q_c = 0$. We have

$$\begin{aligned}
\boldsymbol{\phi}^{n+1} - \boldsymbol{\theta}^{n+1} &= \sum_{i=1}^{M} q_i \left(\boldsymbol{\phi}_i^{n+1} - \boldsymbol{\phi}^n\right) - \sum_{i=1}^{M} p_i \left(\boldsymbol{\theta}_i^{n+1} - \boldsymbol{\theta}^n\right) && \text{(D.14)}\\
&= \sum_{i=1}^{M} q_i \left[\left(\boldsymbol{\phi}_i^{n+1} - \boldsymbol{\theta}_i^{n+1}\right) + \left(\boldsymbol{\theta}_i^{n+1} - \boldsymbol{\theta}^n\right)\right] - \sum_{i=1}^{M} p_i \left(\boldsymbol{\theta}_i^{n+1} - \boldsymbol{\theta}^n\right) && \text{(D.15)}\\
&= \left(\sum_{i=1}^{M} q_i Q_i^n\right) (\boldsymbol{\phi}^n - \boldsymbol{\theta}^n) + \Delta(I_{-c}, \boldsymbol{\theta}^n) - \Delta(I, \boldsymbol{\theta}^n). && \text{(D.16)}
\end{aligned}$$

We consider a learning rate η such that $\eta \leq 1/\sigma_{\max}(H_i(\boldsymbol{\theta}^{n,k}))$. Hence, $\|Q_i^n\|_2 \leq 1$. With equation (D.16), we get the following inequality

$$\left\|\boldsymbol{\phi}^{n+1} - \boldsymbol{\theta}^{n+1}\right\|_2 \leq \|\boldsymbol{\phi}^n - \boldsymbol{\theta}^n\|_2 + \|\Delta(I, \boldsymbol{\theta}^n) - \Delta(I_{-c}, \boldsymbol{\theta}^n)\|_2, \tag{D.17}$$

which expansion completes the proof.

□

D.2.3 Local Loss Functions' Regularization and Strong Convexity, Proof of Corollary 5.1

Proof. Under L2 regularization, every client's regularized loss function F_i is expressed as

$$F_i(\boldsymbol{\theta}) = f_i(\boldsymbol{\theta}) + \frac{\lambda}{2}\|\boldsymbol{\theta}\|^2 \text{ and } \nabla F_i(\boldsymbol{\theta}) = \nabla f_i(\boldsymbol{\theta}) + \lambda\boldsymbol{\theta}. \tag{D.18}$$

When clients perform $K = 1$ GD step, equation (D.16) reduces to

$$\boldsymbol{\phi}^{n+1} - \boldsymbol{\theta}^{n+1} = \eta\left[\nabla f_I(\boldsymbol{\theta}^n) - \nabla f_{I_{-c}}(\boldsymbol{\theta}^n)\right] + \left[(1-\eta\lambda)I - \eta H_{I_{-c}}(\theta^n)\right](\boldsymbol{\phi}^n - \theta^n), \tag{D.19}$$

which, if $\eta \leq 1/(\lambda + \sigma_{\max}(H_i(\boldsymbol{\theta}^n))$, gives

$$\left\|\boldsymbol{\phi}^{n+1} - \boldsymbol{\theta}^{n+1}\right\|_2 \leq \eta\left\|\nabla f_I(\boldsymbol{\theta}^n) - \nabla f_{I_{-c}}(\boldsymbol{\theta}^n)\right\|_2 + (1-\eta\lambda-\eta\mu)\left\|\boldsymbol{\phi}^n - \boldsymbol{\theta}^n\right\|_2. \tag{D.20}$$

When clients perform $K \geq 1$ GD steps, we have $\boldsymbol{\phi}_i^{n+1} - \boldsymbol{\theta}_i^{n+1} = Q_i^n\left[\boldsymbol{\phi}^n - \boldsymbol{\theta}^n\right]$ with

$$Q_i^n = \prod_{r=0}^{K-1}\left[(1-\eta\lambda)I - \eta H_i(\boldsymbol{\theta}_i^{n,r})\right]. \tag{D.21}$$

Hence, we retrieve equation (D.16). We consider the local learning rate satisfy $\eta \leq 1/(\lambda + \sigma_{\max}(H_i(\boldsymbol{\theta}^n)))$. Hence, considering that Q_i^n can be bounded with the μ-strong convexity of the Hessian, we get

$$\left\|\boldsymbol{\phi}^{n+1} - \boldsymbol{\theta}^{n+1}\right\|_2 \leq \eta\left\|\Delta(I,\boldsymbol{\theta}^n) - \Delta(I_{-c},\boldsymbol{\theta}^n)\right\|_2 + (1-\eta\lambda-\eta\mu)^K\left\|\boldsymbol{\phi}^n - \boldsymbol{\theta}^n\right\|_2. \tag{D.22}$$

Developing this recurrent equation completes the proof.

□

D.2.4 Generalization

The proof of Theorem 5.1 can be also extended to account for FL regularization methods (T. Li, Sahu, Zaheer, Sanjabi, Talwalkar, and Smith, 2018a; T. Li, Sahu, Zaheer, Sanjabi,

Talwalkar, and Smithy, 2019; Acar et al., 2021), other SGD solvers (Kingma and Ba, 2014; R. Ward et al., 2019; Xiaoyu Li and Orabona, 2019; Yu, Jin, et al., 2019; Yu, S. Yang, et al., 2019; Haddadpour, Kamani, Mahdavi, and V. Cadambe, 2019), client sampling (T. Li, Sahu, Zaheer, Sanjabi, Talwalkar, and Smith, 2018a; Xiang Li et al., 2020; Fraboni, Vidal, Kameni, et al., 2022a) and/or gradient compression/quantization (Reisizadeh et al., 2020; Basu et al., 2019; H. Wang et al., 2018).

D.2.5 Calculus simplification with uniform importance

For computation purposes, we propose the following expression to estimate a client bounded sensitivity, equation (5.8. When removing client c, each client has new importance $q_i = p_i/(1-p_c)$ for the remaining clients and $q_c = 0$. Hence, we have

$$\|\Delta(I, \boldsymbol{\theta}^n) - \Delta(\boldsymbol{\theta}^n, \mathcal{D}_{-c})\|_2 = \left\| \left[\boldsymbol{\theta}^{n+1} - \boldsymbol{\theta}^n\right] - \left[\sum_{i=1}^{M} q_i \boldsymbol{\theta}_i^{n+1} - \boldsymbol{\theta}^n\right] \right\|_2 \tag{D.23}$$

$$= \left\| \boldsymbol{\theta}^{n+1} - \frac{1}{1-p_c}\left[\boldsymbol{\theta}^{n+1} - p_c \boldsymbol{\theta}_i^{n+1}\right] \right\|_2 \tag{D.24}$$

$$= \frac{p_c}{1-p_c} \left\| \boldsymbol{\theta}_i^{n+1} - \boldsymbol{\theta}^{n+1} \right\|_2 \tag{D.25}$$

In the special case where clients have identical importance, we have $p_c/(1-p_c) = 1/(M-1)$.

D.3 Convergence of SIFU, Theorem 5.3

D.3.1 Intermediate results

Property D.1. *If there exists ν, s, u such that $s < u$, $(\nu, t_s) \in O(s)$ and $(\nu, t_u) \in O(u)$, then $t_s \geq t_u$.*

Proof. We first assume that s and u satisfy $u = s + 1$. Considering that $(\nu, t_s) \in O(s)$ and $(\nu, t_u) \in O(u)$, we have, by definition of ζ_u in equation (5.17), $\nu \leq \zeta_u$.

- $\zeta_u > \nu$. Considering that $u = s + 1$, we have $t_s = t_u$, equation (5.19).
- $\zeta_u = \nu$. Considering that $(\nu, t_s) \in O(s)$ and $(\nu, t_u) \in O(u)$, then we have $\nu \leq s - 1$. Therefore, by definition of ζ_u, we have $\Psi_{\zeta_u}(t_s, W_u) > \Psi^*$. By construction of T_u, equation (5.18), we have $t_u = T_u < t_s$.

When considering the more general case where there exists an integer k such that $u = s + k$ while $(\nu, t_s) \in O(s)$ and $(\nu, t_u) \in O(u)$, then it is sufficient to consider iteratively an integer j ranging from 1 to k. Considering $(\nu, t_u) \in O(u)$, there exists t_{s+j} such that $(\nu, t_{s+j}) \in O(s + j)$. In that case, using the same reasoning as for $k = 1$, we have $t_s \leq t_{s+1} \leq \ldots \leq t_{s+k-1} \leq t_u$.

□

D.3.2 Proof of Theorem 5.3

Proof. Proving that $\boldsymbol{\theta}_r^{N_r}$ (ϵ, δ)-unlearns every client in F_r, equation (5.20), reduces to proving that $\boldsymbol{\theta}_r^0$ (ϵ, δ)-unlearns every client in F_r, equation (5.20). Indeed, the data of clients in F_r are not used on the noised perturbed model $\boldsymbol{\theta}_r^0 = \boldsymbol{\theta}_{\zeta_r}^{T_r} + \mathcal{N}(0, \sigma^2 \boldsymbol{I}_{\boldsymbol{\theta}})$.

We prove by induction that $\boldsymbol{\theta}_r^0$ (ϵ, δ)-unlearns every client in F_r, equation (5.20). The initialization ($r = 1$) directly follows from IFU, Algorithm 6, with Theorem 5.2. We now assume that for every s such that $s \leq r - 1$, $\boldsymbol{\theta}_s^0$ (ϵ, δ)-unlearns every client in F_s and prove that $\boldsymbol{\theta}_r^0$ (ϵ, δ)-unlearns every client in F_r.

- $\underline{s \leq \zeta_r}$. Using the induction property, $\boldsymbol{\theta}_{\zeta_r}^0$ (ϵ, δ)-unlearns every clients in W_s. Clients in W_s are not used for training on $\boldsymbol{\theta}_{\zeta_r}^0$. Hence, $\boldsymbol{\theta}_{\zeta_r}^{T_r}$ and $\boldsymbol{\theta}_r^0$ also (ϵ, δ)-unlearns every client in W_s.

- $\underline{s = r.}$ By definition of ζ_r, equation (5.17), the noise perturbations for every model in $O(r)$ is such that $\boldsymbol{\theta}_{\zeta_r}^0$ (ϵ, δ)-unlearns every client in W_r. Hence, by definition of T_r on the bounded sensitivity of clients in W_r at unlearning request ζ_r, equation (5.18), the noised perturbed model $\boldsymbol{\theta}_r^0$ (ϵ, δ)-unlearns every client in W_r, Theorem 5.2.

- $\underline{\zeta_r < s \leq r - 1.}$ The successive update of the oracle, equation (5.19), from $O(\zeta_r)$ to $O(s)$ gives, by construction, that there exists t_s such that the coordinates (ζ_r, t_s) are in $O(s)$. Hence, by definition of ζ_s, equation (5.17), we have $\zeta_s \geq \zeta_r$ and the successive noise perturbations to obtain $\boldsymbol{\theta}_{\zeta_r}^0$ (ϵ, δ)-unlearns every client in W_s. Also, while we have the coordinates (ζ_r, t_s) in $O(s)$, we also have the coordinates (ζ_r, T_r) in $O(r)$, equation (5.19). Therefore, using property D.1, we have $t_s \geq T_r$. Hence, we have $\Psi_{\zeta_r}(T_r, W_s) \leq \Psi^*$. Therefore, with the noise perturbation of SIFU, clients in W_s are (ϵ, δ)-unlearned in $\boldsymbol{\theta}_r^0$.

□

D.4 Experiments

For every benchmark, we consider the number of SGD steps K, batch size B, number of clients M, the number of sampled clients m, the standard deviation σ of the noise perturbation, and the local learning rate η given in Table D.1. Also, for our unlearning scheme SIFU, DP, and LAST, we consider an unlearning budget of $\epsilon = 10$ and $\delta = 0.01$. The unlearning budget plays the important role of identifying in the training history the global model to perturb. Theorem 5.2 shows that ϵ and σ are linearly related. Hence, to unlearn a client c from a global model c, a smaller σ can be considered, but at the cost of a lower unlearning budget (ϵ, δ), Definition 5.1. Also, for fair comparison of DP with other FU schemes, we select the best clipping value C, in a range from 0.001 to 1, for which the global model reaches the target accuracy in the smallest amount of aggregation rounds. Finally, for FashionMNIST, CIFAR10, CIFAR100, and CelebA, we consider model architectures composed of three convolutional layers followed by two fully connected layers, with implementation at URL.

Tab. D.1.: Hyperparameters used for our different unlearning benchmarks described in Section 5.5.1.

Dataset	K	B	M	m	σ	η	C
MNIST	10	100	100	10	0.05	0.01	0.5
FashionMNIST	5	20	100	10	0.1	0.02	0.5
CIFAR10	5	20	100	5	0.05	0.01	0.2
CIFAR100	5	20	100	5	0.05	0.02	0.2
CelebA	10	20	100	20	0.1	0.01	0.5

The training and retraining depends on the initial model $\boldsymbol{\theta}_0^0$ and the clients' batches of data used at every aggregation to compute their local work. Hence, we replicate each unlearning scenario on 10 different seeds and plot in Figure 5.2 to D.3 their averaged results. For the unlearning benchmarks described in Section 5.5.1 and used in Figure 5.2, D.2, and D.3, the stopping accuracies considered are 93% for MNIST, 90% for FashionMNIST, CIFAR10, and CIFAR100, and 99.9% for CelebA. For Figure 5.3 and D.1 with unlearning benchmark described in Section 5.5.3, the stopping accuracies considered are instead 99.9% for MNIST, FashionMNIST, CIFAR10, and CelebA, and 99% for CIFAR100. Reaching such accuracies is easier with the backdoored datasets because the clients' data heterogeneity is only due to their watermark, Section 5.5.3.

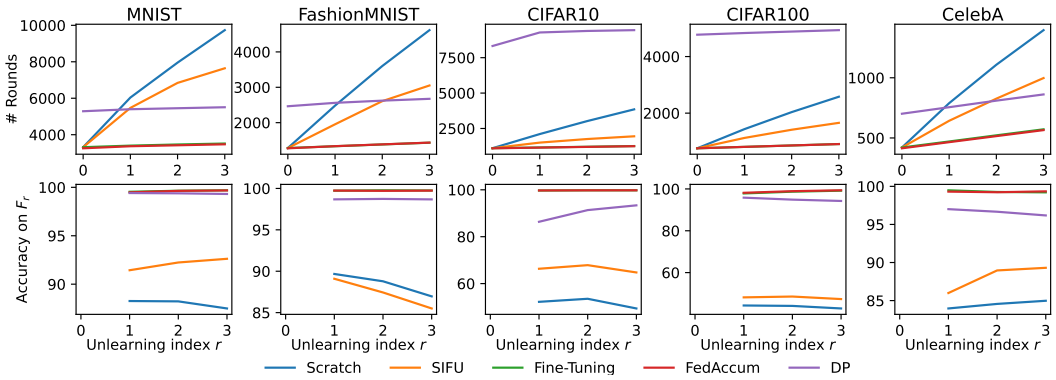

Fig. D.1.: Total amount of aggregation rounds (1st row) and model accuracy of unlearned clients (2nd row) for the unlearning of watermarked data from MNIST, FashionMNIST, CIFAR10, CIFAR100, and CelebA (the lower the better).

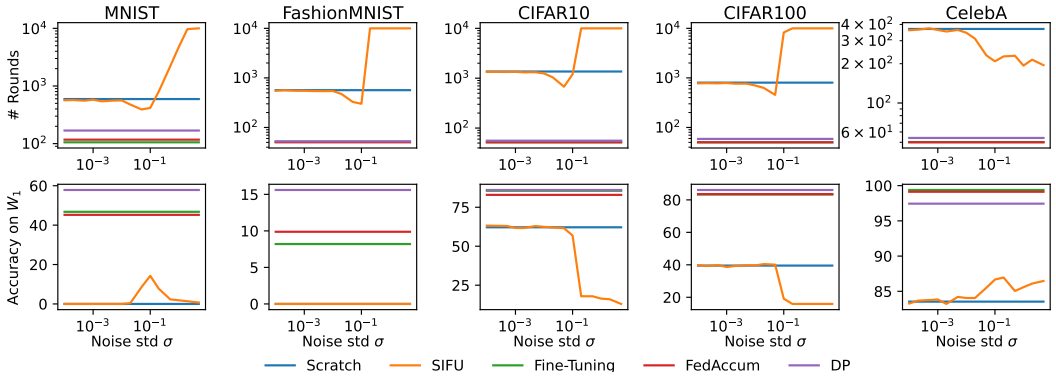

Fig. D.2.: Impact of the noise standard deviation σ when unlearning with SIFU versus SCRATCH. Total amount of aggregation rounds (1st row) and model accuracy of unlearned clients (2nd row) for MNIST, FashionMNIST, CIFAR10, CIFAR100, and CelebA (the lower the better).

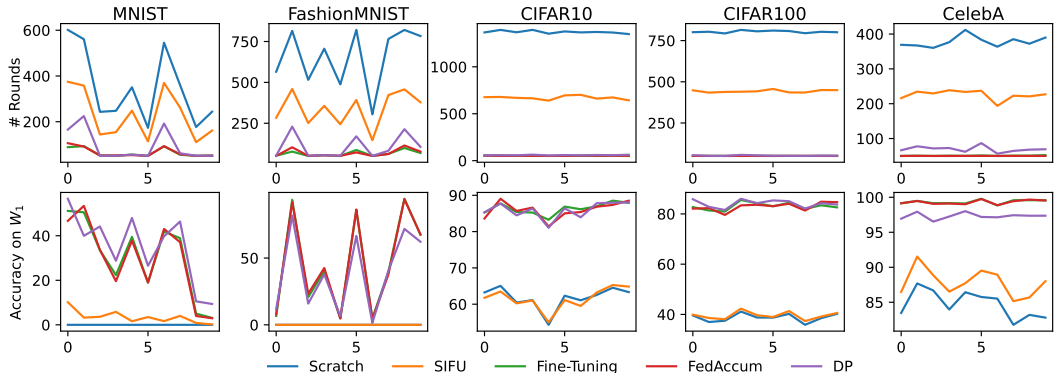

Fig. D.3.: Total amount of aggregation rounds (1st row) and model accuracy of unlearned clients (2nd row) for MNIST, FashionMNIST, CIFAR10, CIFAR100, and CelebA (the lower the better). This figure displays the unlearning capabilities of the unlearning benchmarks introduced in Section 5.5.1 after training on clients in I and unlearning $|W_1| = 10$ clients. For each integer on the x-axis, a different set of clients to unlearn is considered. Each unlearning request is composed of 10 random clients for CIFAR10, CIFAR100, and CelebA. For MNIST and FashionMNIST, each unlearning request $|W_1|$ has 10 clients of the same class such that the x-axis is the class forgotten. The integers on the x-axis corresponds to the class of the clients to unlearn.

Appendix of Chapter 6

E

E.1 Complete Proofs for FedAvg

E.1.1 Proof of Theorem 6.1

We prove with a reasoning by induction that:

$$\tilde{\theta}^t - \theta^t = \sum_{i=0}^{t-1} \left(\epsilon + \frac{M_K}{N}\right)^{t-i-1} f(\theta^i) + \sum_{i=0}^{t-1} \left(\epsilon + \frac{M_K}{N}\right)^{t-i-1} (\tilde{\nu}_i - \nu_i), \tag{E.1}$$

with $f(\theta^t) = \frac{M_K}{N}\left[\theta^t - \sum_{j\in J} \frac{M_j}{N-M_K}[\eta_j(\theta^t - \theta_j^*) + \theta_j^*]\right]$, $\epsilon = \sum_{j\in J} \frac{M_j}{N}\eta_j$, $\nu_t = \sum_{j\in J} \frac{M_j}{N-M_K}\rho_j\zeta_{j,t}$ and $\tilde{\nu}_t = \sum_{j\in J} \frac{M_j}{N}\rho_j\tilde{\zeta}_{j,t}$. By definition of θ^{t+1}, $\mathbb{E}\left[f(\theta^t)\right] = \frac{M_K}{N}\left[\mathbb{E}\left[\theta^t\right] - \mathbb{E}\left[\theta^{t+1}\right]\right]$.

Proof. **Server iteration** $t = 1$

Using the fair clients local model parameters evolution of Section 6.2.3 and the server aggregation process expressed in equation (6.10), the global model can be written as

$$\theta^1 = \sum_{j\in J} \frac{M_j}{N - M_K}\left[\eta_j\left(\theta^0 - \theta_j^*\right) + \theta_j^*\right] + \nu_0. \tag{E.2}$$

Similarly, the global model for federated learning with plain free-riders can be expressed as

$$\tilde{\theta}^1 = \sum_{j\in J} \frac{M_j}{N}\left[\eta_j\left(\theta^0 - \theta_j^*\right) + \theta_j^*\right] + \frac{M_K}{N}\theta^0 + \tilde{\nu}_0. \tag{E.3}$$

By subtracting equation (E.2) to equation (E.3), we obtain:

$$\tilde{\theta}^1 - \theta^1 = -\frac{M_K}{N}\sum_{j\in J} \frac{M_j}{N - M_K}\left[\eta_j\left(\theta^0 - \theta_j^*\right) + \theta_j^*\right] + \frac{M_K}{N}\theta^0 + \tilde{\nu}_0 - \nu_0 \tag{E.4}$$

Hence, $\tilde{\theta}_1 - \theta_1$ follows the formalization.

From t to $t+1$

We suppose the property true at a server iteration t. Hence, we get:

$$\tilde{\theta}^t - \theta^t = \sum_{i=0}^{t-1} \left(\epsilon + \frac{M_K}{N}\right)^{t-i-1} f(\theta^i) + \sum_{i=0}^{t-1} \left(\epsilon + \frac{M_K}{N}\right)^{t-i-1} (\tilde{\nu}_i - \nu_i), \tag{E.5}$$

With the same reasoning as for $t = 1$, we get:

$$\theta^{t+1} = \sum_{j \in J} \frac{M_j}{N - M_K} \left[\eta_j \left(\theta^t - \theta_j^*\right) + \theta_j^*\right] + \nu_t \tag{E.6}$$

and

$$\tilde{\theta}^{t+1} = \sum_{j \in J} \frac{M_j}{N} \left[\eta_j \left(\tilde{\theta}^t - \theta_j^*\right) + \theta_j^*\right] + \frac{M_K}{N} \tilde{\theta}^t + \tilde{\nu}_t \tag{E.7}$$

By using equation (E.5) for equation (E.7), we get:

$$\begin{aligned}
\tilde{\theta}^{t+1} = & \sum_{j \in J} \frac{M_j}{N} \left[\eta_j \left(\theta^t - \theta_j^*\right) + \theta_j^*\right] \\
& + \epsilon \sum_{i=0}^{t-1} \left(\epsilon + \frac{M_K}{N}\right)^{t-i-1} f(\theta^i) \\
& + \epsilon \sum_{i=0}^{t-1} \left(\epsilon + \frac{M_K}{N}\right)^{t-i-1} (\tilde{\nu}_i - \nu_i) \\
& + \frac{M_K}{N} \theta^t \\
& + \frac{M_K}{N} \sum_{i=0}^{t-1} \left(\epsilon + \frac{M_K}{N}\right)^{t-i-1} f(\theta^i) \\
& + \frac{M_K}{N} \sum_{i=0}^{t-1} \left(\epsilon + \frac{M_K}{N}\right)^{t-i-1} (\tilde{\nu}_i - \nu_i) \\
& + \tilde{\nu}_t
\end{aligned} \tag{E.8}$$

which can be rewritten as:

$$\begin{aligned}\tilde{\theta}^{t+1} = &\sum_{j\in J}\frac{M_j}{N}\left[\eta_j\left(\theta^t-\theta_j^*\right)+\theta_j^*\right]\\ &+[\epsilon+\frac{M_K}{N}]\sum_{i=0}^{t-1}\left(\epsilon+\frac{M_K}{N}\right)^{t-i-1}f(\theta^i)\\ &+[\epsilon+\frac{M_K}{N}]\sum_{i=0}^{t-1}\left(\epsilon+\frac{M_K}{N}\right)^{t-i-1}(\tilde{\nu}_i-\nu_i)\\ &+\frac{M_K}{N}\theta^t+\tilde{\nu}_t,\end{aligned} \tag{E.9}$$

leading to

$$\begin{aligned}\tilde{\theta}^{t+1} = &\sum_{j\in J}\frac{M_j}{N}\left[\eta_j\left(\theta^t-\theta_j^*\right)+\theta_j^*\right]\\ &+\sum_{i=0}^{t-1}\left(\epsilon+\frac{M_K}{N}\right)^{t-i}f(\theta^i)\\ &+\sum_{i=0}^{t-1}\left(\epsilon+\frac{M_K}{N}\right)^{t-i}(\tilde{\nu}_i-\nu_i)\\ &+\frac{M_K}{N}\theta^t+\tilde{\nu}_t\end{aligned} \tag{E.10}$$

By subtracting equation (E.10) to equation (E.6), we obtain:

$$\begin{aligned}\tilde{\theta}^{t+1}-\theta^{t+1} = &-\frac{M_K}{N}\sum_{j\in J}\frac{M_j}{N-M_K}\left[\eta_j\left(\theta^t-\theta_j^*\right)+\theta_j^*\right]\\ &+\sum_{i=0}^{t-1}\left(\epsilon+\frac{M_K}{N}\right)^{t-i}f(\theta^i)\\ &+\sum_{i=0}^{t-1}\left(\epsilon+\frac{M_K}{N}\right)^{t-i}(\tilde{\nu}_i-\nu_i)\\ &+\frac{M_K}{N}\theta^t+\tilde{\nu}_t-\nu_t\end{aligned} \tag{E.11}$$

Given that $-\frac{M_K}{N}\sum_{j\in J}\frac{M_j}{N-M_K}\left[\eta_j\left(\theta^t-\theta_j^*\right)+\theta_j^*\right]+\frac{M_K}{N}\theta^t = f(\theta^t)$, we get:

$$\tilde{\theta}^{t+1}-\theta^{t+1} = \sum_{i=0}^{t}\left(\epsilon+\frac{M_K}{N}\right)^{t-i}f(\theta^i)+\sum_{i=0}^{t}\left(\epsilon+\frac{M_K}{N}\right)^{t-i}(\tilde{\nu}_i-\nu_i). \tag{E.12}$$

□

E.1.2 Proof of Theorem 6.2

Proof. **Expected Value**

Let us first have a look at the expected value. By definition, a sum of Gaussian distributions with 0 mean, $\mathbb{E}[\nu_i] = 0$ and $\mathbb{E}[\tilde{\nu}_i] = 0$. We also notice that $\mathbb{E}\left[f(\theta^t)\right] = \frac{M_K}{N}\left[\mathbb{E}\left[\theta^t\right] - \mathbb{E}\left[\theta^{t+1}\right]\right]$. Hence, we obtain

$$\mathbb{E}\left[\tilde{\theta}^t - \theta^t\right] = \frac{M_K}{N}\sum_{i=0}^{t-1}\left(\epsilon + \frac{M_K}{N}\right)^{n-i-1}\mathbb{E}\left[\theta^t - \theta^{t+1}\right]. \tag{E.13}$$

We consider that federated learning is converging, hence $|\,\mathbb{E}\left[\theta^t\right] - \mathbb{E}\left[\theta^{t+1}\right]\,| \xrightarrow{t\to+\infty} 0$, and for any positive α, there exists N_0 such that $|\,\mathbb{E}\left[\theta^t - \theta^{t+1}\right]\,| < \alpha$. Since $\eta_j \in]0,1[$, we have $\epsilon \in]0, \frac{N-M_K}{N}[$ and $\epsilon + \frac{M_K}{N} \in]0,1[$. Thus, we can rewrite equation (E.13) as

$$|\,\mathbb{E}\left[\tilde{\theta}^t - \theta^t\right]\,| \leq \sum_{i=0}^{N_0-1}\left(\epsilon + \frac{M_K}{N}\right)^{t-i-1}|\,\mathbb{E}\left[\theta^t\right] - \mathbb{E}\left[\theta^{t+1}\right]\,| + \sum_{i=N_0}^{t-1}\left(\epsilon + \frac{M_K}{N}\right)^{t-i-1}\alpha. \tag{E.14}$$

We define by $R_\alpha = \max_{i\in[1,N_0]}|\,\mathbb{E}\left[\theta^t\right] - \mathbb{E}\left[\theta^{t+1}\right]\,|$, and get:

$$|\,\mathbb{E}\left[\tilde{\theta}^t - \theta^t\right]\,| \leq \underbrace{\sum_{i=0}^{N_0-1}\left(\epsilon + \frac{M_K}{N}\right)^{t-i-1}}_{A} R_\alpha + \underbrace{\sum_{i=N_0}^{t-1}\left(\epsilon + \frac{M_K}{N}\right)^{t-i-1}}_{B}\alpha. \tag{E.15}$$

- **Expressing A.**

$$A = \sum_{i=0}^{N_0-1}\left(\epsilon + \frac{M_K}{N}\right)^{t-i-1} \tag{E.16}$$

$$= \left(\epsilon + \frac{M_K}{N}\right)^{t-1}\frac{1 - \left(\epsilon + \frac{M_K}{N}\right)^{-N_0}}{1 - \left(\epsilon + \frac{M_K}{N}\right)^{-1}} \tag{E.17}$$

$$\xrightarrow{t\to+\infty} 0 \tag{E.18}$$

- **Expressing B.**

$$B = \sum_{i=N_0}^{t-1} \left(\epsilon + \frac{M_K}{N}\right)^{t-i-1} \tag{E.19}$$

$$= \left(\epsilon + \frac{M_K}{N}\right)^{t-N_0-1} \frac{1 - \left(\epsilon + \frac{M_K}{N}\right)^{-(t-N_0)}}{1 - \left(\epsilon + \frac{M_K}{N}\right)^{-1}} \tag{E.20}$$

$$= \frac{1 - \left(\epsilon + \frac{M_K}{N}\right)^{t-N_0}}{1 - \left(\epsilon + \frac{M_K}{N}\right)} \tag{E.21}$$

$$\xrightarrow{t \to +\infty} \frac{1}{1 - \left(\epsilon + \frac{M_K}{N}\right)} > 0 \tag{E.22}$$

Using equation (E.18) and (E.22) in equation (E.15), we get:

$$\forall \alpha \lim_{t \to +\infty} |\, \mathbb{E}\left[\tilde{\theta}^t - \theta^t\right] \,| \leq B\alpha, \tag{E.23}$$

which is equivalent to

$$\lim_{t \to +\infty} \mathbb{E}\left[\tilde{\theta}^t - \theta^t\right] = 0. \tag{E.24}$$

Variance

The Wiener processes, ν_i and $\tilde{\nu}_i$ are independent from the server models parameters θ^i. Also, each Wiener process is independent with the other Wiener processes. Hence, we get:

$$\begin{aligned} \mathrm{Var}\left[\tilde{\theta}^t - \theta^t\right] = & \underbrace{\mathrm{Var}\left[\sum_{i=0}^{t-1} \left(\epsilon + \frac{M_K}{N}\right)^{t-i-1} f(\theta^i)\right]}_{E} \\ & + \sum_{i=0}^{t-1} \left(\epsilon + \frac{M_K}{N}\right)^{2(t-i-1)} \underbrace{\mathrm{Var}\left[\tilde{\nu}_i - \nu_i\right]}_{F}, \end{aligned} \tag{E.25}$$

Expressing E. Before getting a simpler expression for E, we need to consider $\mathrm{Cov}\left[f(\theta^l), f(\theta^m)\right]$. To do so, we first consider $f(\theta^t) - \mathbb{E}\left[f(\theta^t)\right]$.

$$f(\theta^t) - \mathbb{E}\left[f(\theta^t)\right] = \underbrace{\frac{M_K}{N}\left[1 - \sum_{j \in J} \frac{M_j}{N - M_K} \eta_j\right]}_{G} [\theta^t - \mathbb{E}\left[\theta^t\right]], \tag{E.26}$$

We can prove with a reasoning by induction that $\theta^t - \mathbb{E}\left[\theta^t\right] = \sum_{i=0}^{n-1} \left(\sum_{j\in J} \frac{M_j}{N-M_K}\eta_j\right)^{t-i-1} \nu_i = \sum_{k=0}^{n-1} \epsilon^{t-i-1}\nu_i$. All the ν_i are independent across each others and have 0 mean, hence:

$$\mathrm{Cov}\left[f(\theta_l), f(\theta_m)\right] = G^2 \sum_{i=0}^{\min\{l-1,m-1\}} \epsilon^{l+m-2i-2}\, \mathbb{E}\left[\nu_i^2\right] \tag{E.27}$$

Considering that $\mathbb{E}\left[\nu_i^2\right] = \mathrm{Var}\left[\nu_i\right] = \sum_{j\in J}\left(\frac{M_j}{N-M_K}\rho_j\right)^2$, we get:

$$\mathrm{Cov}\left[f(\theta^l), f(\theta^m)\right] = G^2 \sum_{j\in J}\left(\frac{M_j}{N-M_K}\rho_j\right)^2 \sum_{i=0}^{\min\{l-1,m-1\}} \epsilon^{t-i-1} \tag{E.28}$$

We define $G' = G^2 \sum_{j\in J}\left(\frac{M_j}{N-M_K}\rho_j\right)^2$. Given that $\epsilon \in]0,1[$, we get the following upper bound on E:

$$\mathrm{Cov}\left[f(\theta^l), f(\theta^m)\right] \leq G' \min\{l,m\} \tag{E.29}$$

By denoting $H = \epsilon + \frac{M_K}{N}$, we can rewrite E as:

$$E = \sum_{l=0}^{t-1}\sum_{m=0}^{t-1} H^{2(t-1)-l-m}\, \mathrm{Cov}\left[f_l(\theta^l), f(\theta^m)\right] \tag{E.30}$$

$$\leq \sum_{l=0}^{t-1}\sum_{m=0}^{t-1} H^{2(t-1)-l-m} G' \min\{l,m\} \tag{E.31}$$

Considering that $\min\{l,m\} \leq l$, we get:

$$E \leq G' \sum_{l=0}^{t-1}\sum_{m=0}^{t-1} H^{2(t-1)-l-m} l \tag{E.32}$$

$$= G' H^{2(t-1)} \sum_{l=0}^{t-1} H^{-l} l \sum_{m=0}^{t-1} H^{-m} \tag{E.33}$$

$$= G' H^{2(t-1)} \sum_{l=0}^{t-1} H^{-l} l \frac{1-H^{-n}}{1-H^{-1}} \tag{E.34}$$

$$= G' H^{2(t-1)} \frac{1-H^{-n}}{1-H^{-1}} \sum_{l=0}^{t-1} H^{-l} l \tag{E.35}$$

Considering the power series $\sum_{k=0}^{+\infty} nx^n = \frac{x}{(1-x)^2}$, we get that $\sum_{l=0}^{t-1} H^{-l} l = \frac{H^{-1}}{(1-H^{-1})^2}$. Hence, E's upper bound goes to 0. Given that E is non-negative, we get

$$E \xrightarrow{t\to+\infty} 0. \tag{E.36}$$

Expressing F. Let us first consider the noise coming from the SGD steps. All the $\tilde{\nu}_i$ are independent with ν_i. Hence, we have

$$F = \text{Var}\left[\tilde{\nu}_i\right] - \text{Var}\left[\nu_i\right] \tag{E.37}$$

$$= \text{Var}\left[\sum_{j\in J} \frac{M_j}{N}\rho_j\tilde{\zeta}_{j,i} - \sum_{j\in J}\frac{M_j}{N-M_K}\rho_j\zeta_{j,i}\right] \tag{E.38}$$

$$= [\frac{1}{N^2} + \frac{1}{(N-M_K)^2}]\sum_{j\in J}(M_j\rho_j)^2 \tag{E.39}$$

Replacing (E.39) in equation (E.25), we can express the variance as

$$\text{Var}\left[\tilde{\theta}^t - \theta^t\right] = E + F\sum_{i=0}^{t-1} H^{2(t-i-1)} \tag{E.40}$$

$$= E + FH^{2(t-1)}\sum_{i=0}^{t-1} H^{-2i} \tag{E.41}$$

$$= E + FH^{2(t-1)}\frac{1-H^{-2t}}{1-H^{-2}} \tag{E.42}$$

$$= E + F\frac{1-H^{2t}}{1-H^2} \tag{E.43}$$

By replacing F and H with their respective expression, we can conclude that

$$\text{Var}\left[\tilde{\theta}^t - \theta^t\right] \xrightarrow{t\to+\infty} \frac{[\frac{1}{N^2} + \frac{1}{(N-M_K)^2}]\sum_{j\in J}(M_j\rho_j)^2}{1-\left(\epsilon + \frac{M_K}{N}\right)^2} \tag{E.44}$$

□

Note 1: The asymptotic variance is strictly increasing with the number of data points declared by the free-riders M_K.

While M_j and ρ_j are constants and independent from the number of free-riders and from their respective number of data points, N and ϵ depend on the total number of free-riders' samples M_K. We first rewrite $\epsilon = \frac{1}{N}\alpha$ with $\alpha = \sum_{j\in J} M_j\eta_j$ not depending on M_K and we get:

$$\epsilon + \frac{M_K}{N} = \frac{1}{N}[\alpha + M_K]. \tag{E.45}$$

By defining $M_J = \sum_{j\in J} M_j$, we get:

$$1 - \left(\epsilon + \frac{M_K}{N}\right)^2 = \frac{1}{N^2}[M_J^2 + 2M_K[M_J - \alpha] - \alpha^2], \tag{E.46}$$

with $M_J - \alpha > 0$ because $\eta_j \in]0, 1[$.

Also, considering that

$$\frac{1}{N^2} + \frac{1}{(N - M_K)^2} = \frac{1}{N^2}[\frac{M_K^2}{M_J^2} + 2\frac{M_K}{M_J} + 2], \tag{E.47}$$

we can rewrite

$$\frac{\frac{1}{N^2} + \frac{1}{(N-M_K)^2}}{1 - \left(\epsilon + \frac{M_K}{N}\right)^2} = \frac{\frac{M_K^2}{M_J^2} + 2\frac{M_K}{M_J} + 2}{M_J^2 + 2M_K[M_J - \alpha] - \alpha^2} \tag{E.48}$$

As the numerator is a polynomial of order 2 in M_K and the denominator is a polynomial of order 1 in M_K, the asymptotic variance is increasing with M_K.

Note 2: When considering that the SGD noise variance is different for federated learning with and without free-riders, we get:

$$F = \frac{1}{N^2}\sum_{j \in J}(M_j \tilde{\rho}_j)^2 + \frac{1}{(N - M_K)^2}\sum_{j \in J}(M_j \rho_j)^2 \tag{E.49}$$

E.1.3 Proof of Theorem 6.3

Proof. **Relation between federated learning with and without free-riders global model**

With a reasoning by induction similar to Proof E.1.1, we get:

$$\tilde{\theta}^t - \theta^t = \sum_{i=0}^{t-1}\left(\epsilon + \frac{M_K}{N}\right)^{t-i-1} f(\theta^i) \tag{E.50}$$

$$+ \sum_{i=0}^{t-1}\left(\epsilon + \frac{M_K}{N}\right)^{t-i-1} (\tilde{\nu}_i - \nu_i) \tag{E.51}$$

$$+ \sum_{i=0}^{t-1}\left(\epsilon + \frac{M_K}{N}\right)^{t-i-1} \frac{M_K}{N}\varphi\epsilon_t, \tag{E.52}$$

Expected value

ϵ_t is a delta-correlated Gaussian White noise which implies that $\mathbb{E}[\epsilon_t] = 0$. Following the same reasoning steps as in Proof E.1.2, we get:

$$\lim_{t \to +\infty} \mathbb{E}\left[\tilde{\theta}^t - \theta^t\right] = 0. \tag{E.53}$$

Variance

All the ϵ_t are independent Gaussian white noises implying $\mathrm{Var}\left[\epsilon_t\right] = 1$. Following the same reasoning steps as in Proof E.1.2, we get:

$$\mathrm{Var}\left[\sum_{i=0}^{t-1}\left(\epsilon + \frac{M_K}{N}\right)^{t-i-1}\frac{M_K}{N}\varphi\epsilon_t\right]$$

$$= \sum_{i=0}^{t-1}\left(\epsilon + \frac{M_K}{N}\right)^{2(t-i-1)}\frac{M_K^2}{N^2}\varphi^2 \tag{E.54}$$

$$= \left(\epsilon + \frac{M_K}{N}\right)^{2(t-1)}\frac{1-\left(\epsilon+\frac{M_K}{N}\right)^{-2t}}{1-\left(\epsilon+\frac{M_K}{N}\right)^{-2}}\frac{M_K^2}{N^2}\varphi^2 \tag{E.55}$$

$$= \frac{1-\left(\epsilon+\frac{M_K}{N}\right)^{2t}}{1-\left(\epsilon+\frac{M_K}{N}\right)^{2}}\frac{M_K^2}{N^2}\varphi^2 \tag{E.56}$$

$$\xrightarrow{t\to+\infty} \frac{1}{1-\left(\epsilon+\frac{M_K}{N}\right)^{2}}\frac{M_K^2}{N^2}\varphi^2 \tag{E.57}$$

As for equation (E.25), all the ϵ_t are independent from ν_t, from $\tilde{\nu}_t$, and from the global model parameters θ^t. Hence, for one disguised free-rider we get the following asymptotic variance:

$$\mathrm{Var}\left[\tilde{\theta}^t - \theta^t\right] \xrightarrow{t\to+\infty} \frac{\left[\frac{1}{N^2} + \frac{1}{(N-M_K)^2}\right]\sum_{j\in J}\left(M_j\rho_j\right)^2}{1-\left(\epsilon+\frac{M_K}{N}\right)^{2}} + \frac{1}{1-\left(\epsilon+\frac{M_K}{N}\right)^{2}}\frac{M_K^2}{N^2}\varphi^2. \tag{E.58}$$

□

E.1.4 Proof of Corollary 6.1

Proof. **Relation between federated learning with and without free-riders global model**

With a reasoning by induction similar to Proof E.1.1, we get:

$$\tilde{\theta}^t - \theta^t = \sum_{i=0}^{t-1}\left(\epsilon + \frac{M_K}{N}\right)^{t-i-1} f(\theta^i) \tag{E.59}$$

$$+ \sum_{i=0}^{t-1}\left(\epsilon + \frac{M_K}{N}\right)^{t-i-1}(\tilde{\nu}_i - \nu_i) \tag{E.60}$$

$$+ \sum_{k\in K}\sum_{i=0}^{t-1}\left(\epsilon + \frac{M_K}{N}\right)^{t-i-1}\frac{M_k}{N}\varphi_k\epsilon_{k,t}, \tag{E.61}$$

Expected value

$\epsilon_{k,t}$ are delta-correlated Gaussian White noises which implies that $\mathbb{E}\left[\epsilon_{k,t}\right] = 0$. Following the same reasoning steps as in Proof E.1.2, we get:

$$\lim_{t \to +\infty} \mathbb{E}\left[\tilde{\theta}^t - \theta^t\right] = 0. \tag{E.62}$$

Variance

All the $\epsilon_{k,t}$ are independent Gaussian white noises over server iterations t and free-riders indices k implying $\mathrm{Var}\left[\epsilon_t\right] = 1$. Following the same reasoning steps as in Proof E.1.2, we get:

$$\mathrm{Var}\left[\sum_{i=0}^{t-1}\left(\epsilon + \frac{M_K}{N}\right)^{t-i-1}\frac{M_k}{N}\varphi_k\epsilon_{k,t}\right] \xrightarrow{t \to +\infty} \frac{1}{1-\left(\epsilon + \frac{M_K}{N}\right)^2}\frac{M_k^2}{N^2}\varphi_k^2 \tag{E.63}$$

Like for equation (E.25), all the $\epsilon_{k,t}$ are independent from ν_t, $\tilde{\nu}_t$ and the global model parameters θ^t. Hence, for multiple disguised free-rider we get the following asymptotic variance:

$$\begin{aligned}\mathrm{Var}\left[\tilde{\theta}^t - \theta^t\right] \xrightarrow{t \to +\infty} & \frac{\left[\frac{1}{N^2} + \frac{1}{(N-M_K)^2}\right]\sum_{j \in J}\left(M_j\rho_j\right)^2}{1-\left(\epsilon + \frac{M_K}{N}\right)^2} \\ & + \frac{1}{1-\left(\epsilon + \frac{M_K}{N}\right)^2}\sum_{k \in K}\frac{M_k^2}{N^2}\varphi_k^2.\end{aligned} \tag{E.64}$$

□

E.1.5 Proof of Corollary 6.2

Proof. **Relation between federated learning with and without free-riders global model**

The relation remains the same for Theorem 6.2, Theorem 6.3, and Corollary 6.1 by replacing η_j with $\eta_j(t) = \sum j \in J\frac{M_j}{N}\rho_j(t)$ and φ_k by $\varphi_k(t)$ for disguised free-riding.

Expected value

With ρ_j^t and $\varphi(t)$ the properties for $\tilde{\nu}_t$, ν_t, ϵ_t and $\epsilon_{k,t}$ remain identical. Hence, they still are delta-correlated Gaussian White noises implying that $\mathbb{E}\left[\tilde{\nu}_t\right] = \mathbb{E}\left[\nu_t\right] = \mathbb{E}\left[\epsilon_t\right] = \mathbb{E}\left[\epsilon_{k,t}\right] = 0$. Hence, for Theorem 6.2, Theorem 6.3, and Corollary 6.1, we get:

$$\lim_{t \to +\infty} \mathbb{E}\left[\tilde{\theta}^t - \theta^t\right] = 0. \tag{E.65}$$

Variance

Variance asymptotic behaviour proven in Proof E.1.2, E.1.3, and E.1.4 can be reduced to the one in Proof E.1.2. Hence, F, equation (E.39), need to be reexpressed to take into account $\rho_j(t)$. All the $\tilde{\nu}_i$ are still independent with ν_i. Hence, we have:

$$F = \mathrm{Var}\left[\tilde{\nu}_i(t) - \nu_i(t)\right] = \mathrm{Var}\left[\sum_{j \in J} \frac{M_j}{N} \rho_j^t \tilde{\zeta}_{j,i} - \sum_{j \in J} \frac{M_j}{N - M_K} \rho_j^t \zeta_{j,i}\right] \tag{E.66}$$

Considering that $\rho_j^t \xrightarrow{t \to +\infty} 0$, we get:

$$F \xrightarrow{t \to +\infty} 0 \tag{E.67}$$

Using the same reasoning as the one used for the expected value convergence in Proof E.1.2, we get that the SGD noise contribution linked to F goes to 0 at infinity.

For the disguised free-riders, $\epsilon_{k,t}$ are still independent Gaussian white noises implying $\mathrm{Var}\left[\epsilon_{k,t}\right] = 1$. Hence, following a reasoning similar to the on in Proof E.1.2, we get:

$$\mathrm{Var}\left[\sum_{i=0}^{t-1} \left(\epsilon + \frac{M_K}{N}\right)^{t-i-1} \frac{M_K}{N} \varphi_k(t) \epsilon_{k,t}\right] = \sum_{i=0}^{t-1} \left(\epsilon + \frac{M_K}{N}\right)^{2(t-i-1)} \frac{M_K^2}{N^2} \varphi_k^2(t) \tag{E.68}$$

Considering that $\varphi_k(t) \xrightarrow{t \to +\infty} 0$, by using the same reasoning as for the proof of the expected value for free-riders, Section XX, we get:

$$\mathrm{Var}\left[\sum_{i=0}^{t-1} \left(\epsilon + \frac{M_K}{N}\right)^{t-i-1} \frac{M_K}{N} \varphi_k(t) \epsilon_{k,t}\right] \xrightarrow{t \to +\infty} 0 \tag{E.69}$$

Hence, we can conclude that

$$\mathrm{Var}\left[\tilde{\theta}^t - \theta^t\right] \xrightarrow{t \to +\infty} 0. \tag{E.70}$$

□

E.2 Complete Proofs for FedProx

FedProx is a generalization of FedAvg. As such, we use the proof done for FedAvg to prove convergence of free-riders attack using FedProx as an optimization solver. The L2 norm monitored by μ changes the gradient as $g_j(\theta_j) \simeq r_j[\theta_j - \theta_j^*] + \mu[\theta_j - \theta^t]$.

Using equation (6.7), we then get:

$$\mathrm{d}\theta_j = -\lambda \left[r_j[\theta_j - \theta_j^*] + \mu[\theta_j - \theta^t] \right] + \frac{\lambda}{\sqrt{S}} \sigma_j(\theta_j) \mathrm{d}W_j, \tag{E.71}$$

leading to

$$\begin{aligned} \theta_j(u) = e^{-\lambda[r_j+\mu]u}\theta_j(0) + \frac{r_j\theta_j^* + \mu\theta^t}{r_j+\mu}[1 - e^{-\lambda(r_j+\mu)u}] \\ + \frac{\lambda}{\sqrt{S}} \int_{x=0}^{u} e^{-\lambda(r_j+\mu)(u-x)} \sigma_j(\theta_j) \mathrm{d}W_x. \end{aligned} \tag{E.72}$$

considering that $\theta_j(0) = \theta^t$, $\theta_j(\frac{EM_j}{S}) = \theta_j^{t+1}$, and $\sigma_j(\theta_j) = \sigma_j^t$, we get:

$$\theta_j^{t+1} = \gamma_j\theta^t + \frac{r_j\theta_j^* + \mu\theta^t}{r_j+\mu}[1-\gamma_j] + \frac{\lambda}{\sqrt{S}} \int_{x=0}^{\frac{EM_j}{S}} e^{-\lambda(r_j+\mu)(\frac{EM_j}{S}-x)} \sigma_j^t \mathrm{d}W_x, \tag{E.73}$$

where $\gamma_j = e^{-\lambda[r_j+\mu]\frac{EM_j}{S}}$. We can reformulate this as

$$\theta_j^{t+1} = [\gamma_j + \mu\frac{1-\gamma_j}{r_j+\mu}]\theta^t + \frac{r_j}{r_j+\mu}[1-\gamma_j]\theta_j^* \tag{E.74}$$

$$+ \frac{\lambda}{\sqrt{S}} \int_{x=0}^{\frac{EM_j}{S}} e^{-\lambda(r_j+\mu)(\frac{EM_j}{S}-x)} \sigma_j^t \mathrm{d}W_x, \tag{E.75}$$

The SGD noise variance between two server iterations for FedProx is:

$$\mathrm{Var}\left[\theta_j^{t+1}|\theta^t\right] = \underbrace{\frac{\lambda}{S}{\sigma_j^t}^2 \frac{1}{2(r_j+\mu)} \left[1 - e^{-2\lambda(r_j+\mu)\frac{EM_j}{S}}\right]}_{{\rho_j^t}^2}, \tag{E.76}$$

We also define $\eta_j' = \gamma_j + \mu\frac{1-\gamma_j}{r_j+\mu}$ and $\delta_j = \frac{r_j}{r_j+\mu}[1-\gamma_j]$. For FedAvg, $\mu = 0$, we get $\eta_j' = \eta_j$ and $\delta_j = 1 - \eta_j$. By property of the exponential, $\gamma_j \in]0,1[$. As r_j and μ are non negative, then $\eta_j' \in]0,1[$ like η_j for FedAvg.

Theorem 6.1 for FedProx

We consider ${\rho_j'}^2 = \frac{\lambda}{S}{\sigma_j}^2 \frac{1}{2(r_j+\mu)} \left[1 - e^{-2\lambda(r_j+\mu)\frac{EM_j}{S}}\right]$

Using the same reasoning by induction as in Proof E.1.1, we get:

$$\tilde{\theta}^t - \theta^t = \sum_{i=0}^{t-1} \left(\epsilon' + \frac{M_K}{N}\right)^{t-i-1} g(\theta^i) + \sum_{i=0}^{t-1} \left(\epsilon' + \frac{M_K}{N}\right)^{t-i-1} (\tilde{\nu}'_i - \nu'_i), \tag{E.77}$$

with $g(\theta^t) = \frac{M_K}{N}\left[\theta^t - \sum_{j\in J} \frac{M_j}{N-M_K}[\eta'_j\theta^t + \delta_j\theta^*_j]\right]$, $\epsilon' = \sum_{j\in J} \frac{M_j}{N}\eta'_j$, $\nu'_t = \sum_{j\in J} \frac{M_j}{N-M_K}\rho'_j\zeta_{j,t}$ and $\tilde{\nu}'_t = \sum_{j\in J} \frac{M_j}{N}\rho'_j\tilde{\zeta}_{j,t}$.

Theorem 6.2 for FedProx

Like for FedAvg, we make the assumption that federated learning without free-riders using FedProx converge. In addition, $\tilde{\nu}'_t$ and ν'_t are also independent delta-correlated Gaussian white noises. Following the same proof as in Proof E.1.2, we thus get:

$$\lim_{t\to+\infty} \mathbb{E}\left[\tilde{\theta}^t - \theta^t\right] = 0. \tag{E.78}$$

and

$$\mathrm{Var}\left[\tilde{\theta}^t - \theta^t\right] \xrightarrow{t\to+\infty} \frac{[\frac{1}{N^2} + \frac{1}{(N-M_K)^2}]\sum_{j\in J}\left(M_j\rho'_j\right)^2}{1 - \left(\epsilon' + \frac{M_K}{N}\right)^2} \tag{E.79}$$

The asymptotic variance still strictly increases with M_K.

Note: We introduce $x = \lambda(r_j + \mu)\frac{EM_j}{S}$. By taking the partial derivative of ρ'_j with respect to μ, we get:

$$\frac{\delta\rho'_j}{\delta\mu} = \frac{\lambda}{2S}\sigma_j^2 \frac{1}{(r_j+\mu)^2}[-1 + (1+2x)e^{-2x}], \tag{E.80}$$

which is strictly negative for a positive μ considering that all the other constants are positive. Hence, the SGD noise variance ρ'_j is inversely proportional with the regularization factor μ.

Similarly, for ϵ', by considering that η'_j can be rewritten as $\eta'_j = \gamma_j\frac{r_j}{r_j+\mu} + \frac{\mu}{r_j+\mu}$, the partial derivative of η'_j with respect to μ can be expressed as:

$$\frac{\delta\eta'_j}{\delta\mu} = \frac{r_j}{(r_j+\mu)^2}[1 - (1-x)e^{-x}], \tag{E.81}$$

which is strictly positive. Hence η'_j is strictly increasing with the regularization μ and so is ϵ'.

Considering the behaviours of ϵ' and ρ'_j with respect to the regularization term μ, the more regularization is asked by the server and the smaller the asymptotic variance is, leading to more accurate free-riding attacks.

Theorem 6.3 for FedProx

The free-riders mimic the behaviour of the fair clients. Hence, we get:

$$\varphi_k{}'^2 = \frac{\lambda}{S}{\sigma_k}^2 \frac{1}{2(r_j+\mu)} \left[1 - e^{-2\lambda(r_k+\mu)\frac{EM_j}{S}}\right] \tag{E.82}$$

leading to

$$\mathrm{Var}\left[\tilde{\theta}^t - \theta^t\right] \xrightarrow{t\to+\infty} \frac{[\frac{1}{N^2} + \frac{1}{(N-M_K)^2}]\sum_{j\in J}\left(M_j\rho'_j\right)^2}{1-\left(\epsilon' + \frac{M_K}{N}\right)^2} + \frac{1}{1-\left(\epsilon' + \frac{M_K}{N}\right)^2}\frac{M_K^2}{N^2}\varphi'^2. \tag{E.83}$$

For disguised free-riders, the variance is also inversely proportional to the regularization parameter μ.

Corollary 6.1 for FedProx

Similarly, for many free-riders, we get:

$$\begin{aligned}\mathrm{Var}\left[\tilde{\theta}^t - \theta^t\right] \xrightarrow{t\to+\infty} & \frac{[\frac{1}{N^2} + \frac{1}{(N-M_K)^2}]\sum_{j\in J}\left(M_j\rho'_j\right)^2}{1-\left(\epsilon' + \frac{M_K}{N}\right)^2} \\ & + \frac{1}{1-\left(\epsilon' + \frac{M_K}{N}\right)^2}\frac{M_K^2}{N^2}\sum_{k\in K}\varphi'^2_k.\end{aligned} \tag{E.84}$$

E.3 Additional experimental results

E.3.1 Accuracy Performances

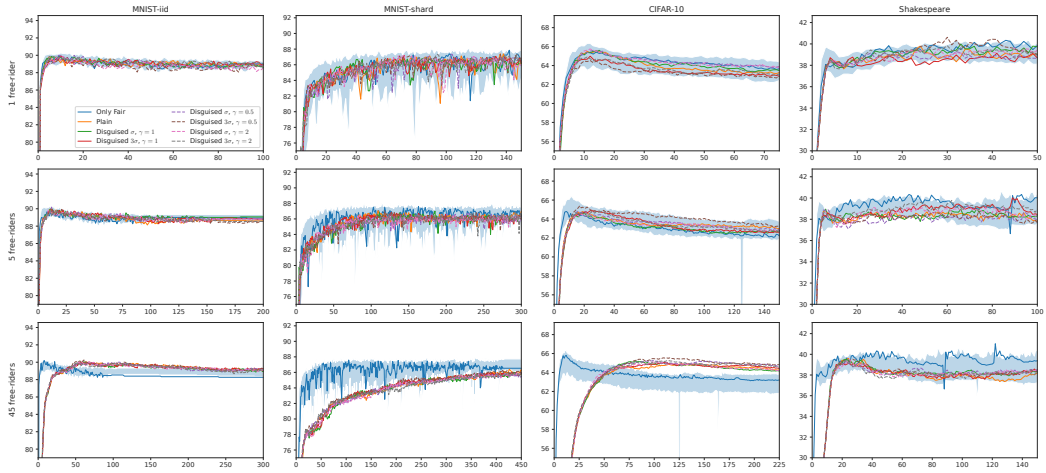

Fig. E.1.: Accuracy performances for FedAvg and 20 epochs in the different experimental scenarios.

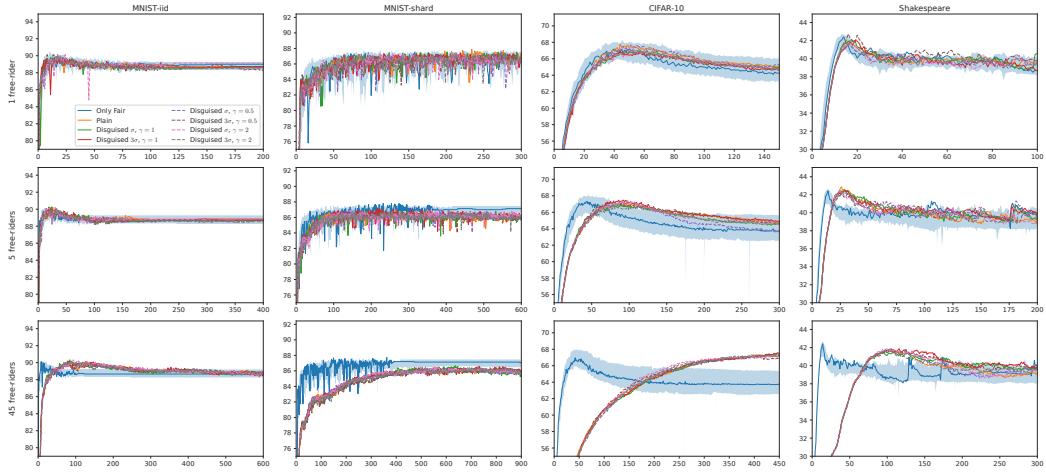

Fig. E.2.: Accuracy performances for FedAvg and 5 epochs in the different experimental scenarios.

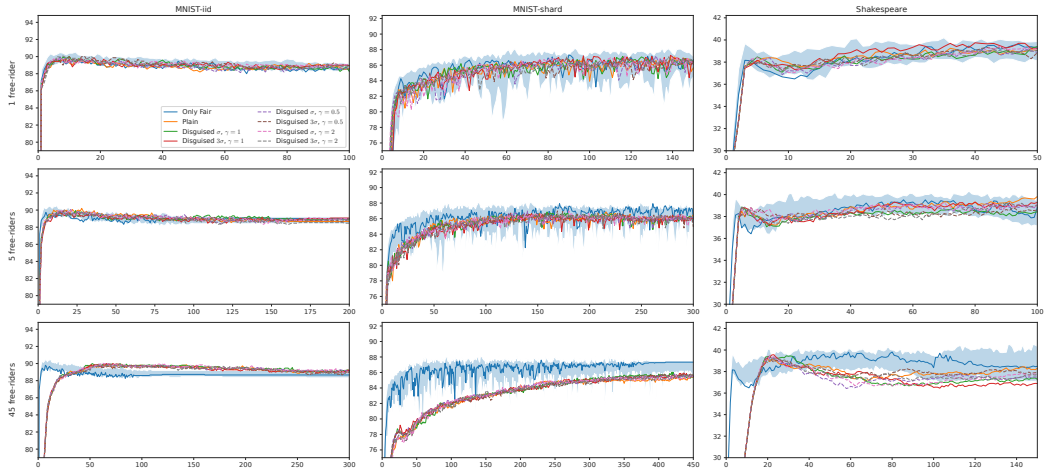

Fig. E.3.: Accuracy performances for FedProx and 20 epochs in the different experimental scenarios.

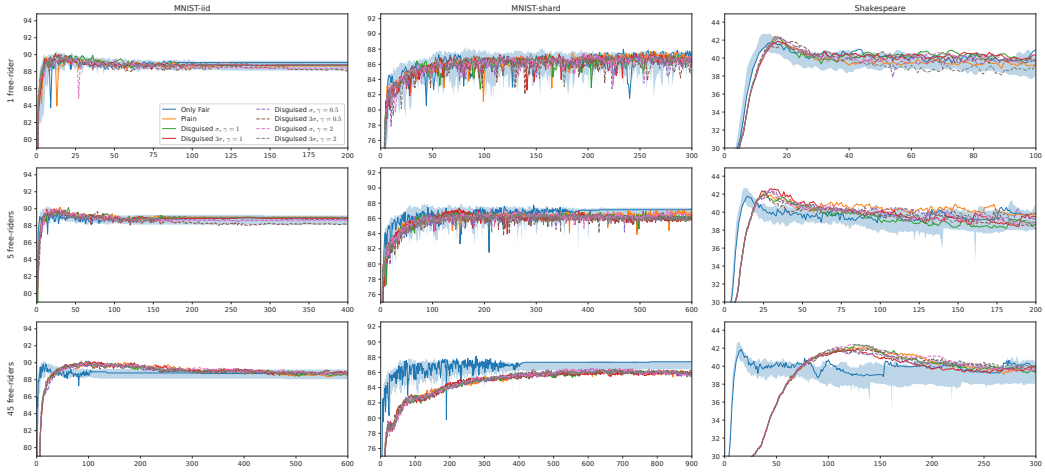

Fig. E.4.: Accuracy performances for FedProx and 5 epochs in the different experimental scenarios.

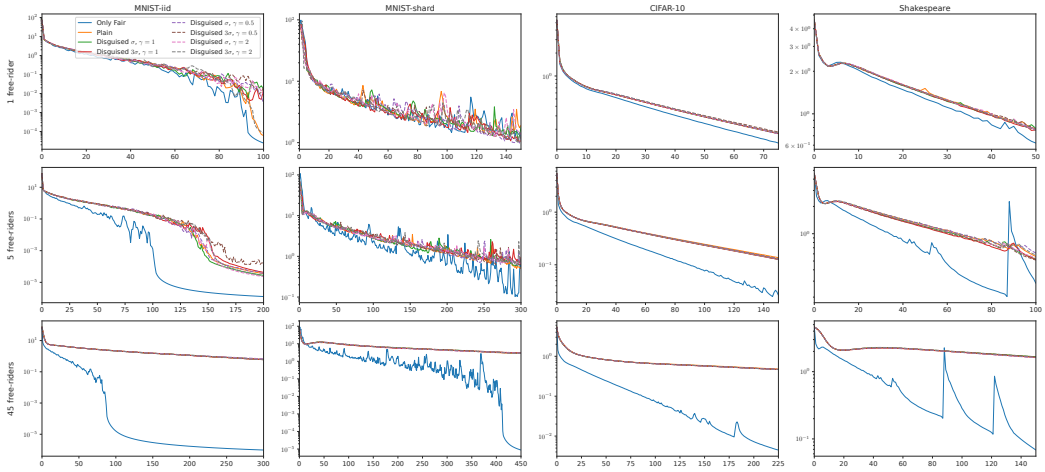

Fig. E.5.: Loss performances for FedAvg and 20 epochs in the different experimental scenarios.

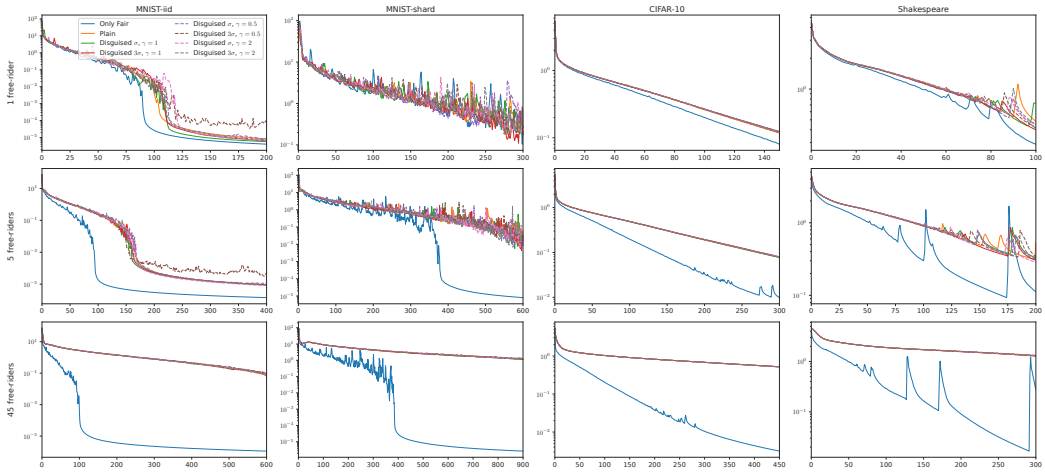

Fig. E.6.: Loss performances for FedAvg and 5 epochs in the different experimental scenarios.

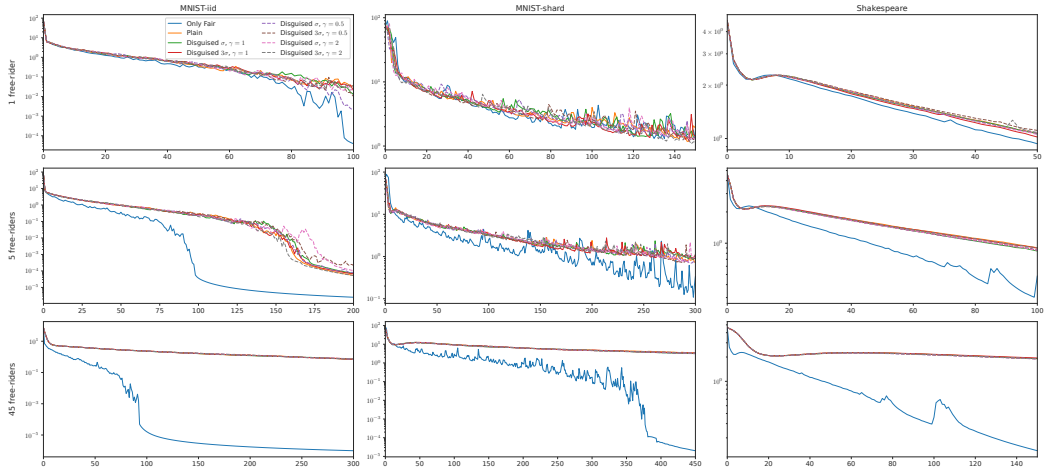

Fig. E.7.: Loss performances for FedProx and 20 epochs in the different experimental scenarios.

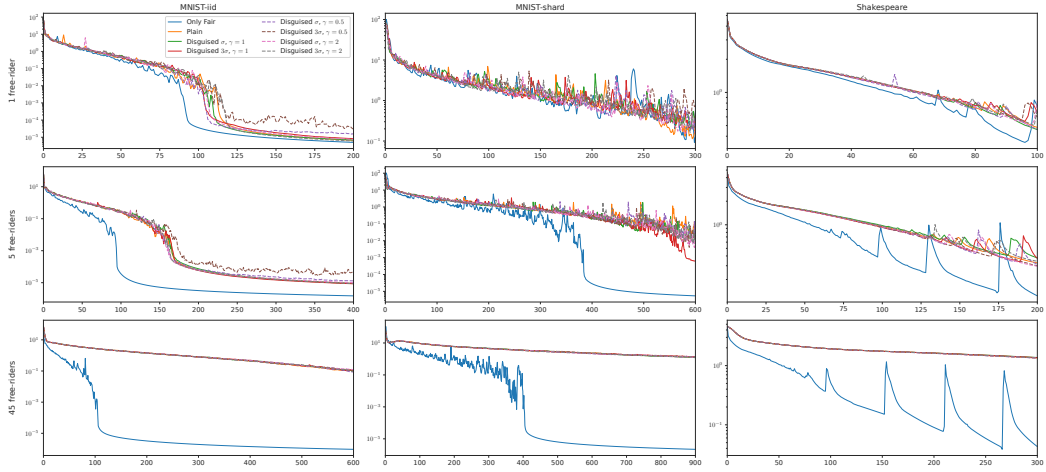

Fig. E.8.: Loss performances for FedProx and 5 epochs in the different experimental scenarios.