UMEÅ UNIVERSITY

Personalized Models and Optimization in Federated Learning

Ali Dadras

Doctoral Thesis, February, 2025
Department of Mathematics and Mathematical Statistics
Umeå University
Sweden

Department of Mathematics
and Mathematical Statistics
Umeå University
SE-901 87 Umeå, Sweden

ali.dadras@umu.se

ISBN 978-91-8070-599-8 (Print)
978-91-8070-600-1 (digital)
ISSN 1653-0829

The papers in this thesis have been re-typeset to match the overall style of the thesis with permission granted by the copyright holders.

Printed by Scandinavian Print Group
Umeå, 2025

"The birth of the reader must be at the cost of the death of the Author."
– Roland Barthes

Abstract

The rapid increase in data generation, combined with the impracticality of centralizing large-scale datasets and the growing complexity of machine learning tasks, has driven the development of distributed learning techniques. Among these, *Federated Learning* (FL) has gained significant attention due to its privacy-preserving approach, where multiple clients collaboratively train a global model without sharing their local data. However, FL faces several key challenges, including data heterogeneity, high computational costs, and communication inefficiencies. These issues become more pronounced in real-world scenarios where client data distributions are non-IID, computational resources are limited, and communication is constrained.

This thesis addresses these challenges through the development of efficient algorithms for *Personalized Federated Learning* (pFL) and *Constrained Federated Learning*. The proposed approaches are designed to handle heterogeneous data, minimize computational overhead, and reduce communication costs while maintaining strong theoretical guarantees.

Specifically, the thesis introduces three key contributions: **(1)** $\texttt{pFL}^{\texttt{MF}}$, a novel pFL formulation based on low-rank matrix optimization, leveraging Burer-Monteiro factorization to enable personalization without relying on predefined distance metrics. **(2)** `PerMFL`, an algorithm for multi-tier pFL that introduces personalized decision variables for both teams and individual devices, enabling efficient optimization in scenarios with hierarchical client structures. **(3)** `FedFW`, a projection-free algorithm for constrained FL, which emphasizes low computational cost, privacy preservation, and communication efficiency through sparse signal exchanges.

By addressing critical issues in FL, such as data heterogeneity, computation costs, and communication bottlenecks, the proposed algorithms advance the field of Federated Learning, providing robust and scalable solutions for real-world applications.

Sammanfattning

Den snabba ökningen av datagenerering, kombinerat med omöjligheten att centralisera storskaliga datamängder och den växande komplexiteten i maskininlärningsuppgifter, har drivit utvecklingen av distribuerade inlärningstekniker. Bland dessa har *Federated Learning* (FL) fått betydande uppmärksamhet tack vare sitt integritetsskyddande tillvägagångssätt, där flera klienter tillsammans tränar en global modell utan att dela sina lokala data. Dock står FL inför flera viktiga utmaningar, inklusive dataheterogenitet, höga beräkningskostnader och kommunikationsineffektivitet. Dessa problem blir mer uttalade i verkliga scenarier där klientdatafördelningar är icke-IID, beräkningsresurser är begränsade och kommunikationen är begränsad.

Denna avhandling tar itu med dessa utmaningar genom utveckling av effektiva algoritmer för *Personalized Federated Learning* (pFL) och *Constrained Federated Learning*. De föreslagna metoderna är utformade för att hantera heterogena data, minimera beräkningsöverlagring och reducera kommunikationskostnader samtidigt som de upprätthåller starka teoretiska garantier.

Specifikt introducerar avhandlingen tre viktiga bidrag: **(1)** $\texttt{pFL}^{\texttt{MF}}$, en ny pFL-formulering baserad på låg-rank matrisoptimering, som utnyttjar Burer-Monteiro-faktorisering för att möjliggöra personalisering utan att förlita sig på fördefinierade avståndsmetrikar. **(2)** `PerMFL`, en algoritm för multi-tier pFL som introducerar personaliserade beslutvariabler för både team och individuella enheter, vilket möjliggör effektiv optimering i scenarier med hierarkiska klientstrukturer. **(3)** `FedFW`, en projiceringsfri algoritm för begränsad FL, som betonar låga beräkningskostnader, integritetsskydd och kommunikationseffektivitet genom sparsamma signalutbyten.

Genom att ta itu med kritiska problem inom FL, såsom dataheterogenitet, beräkningskostnader och kommunikationsflaskhalsar, främjar de föreslagna algoritmerna området Federated Learning och tillhandahåller robusta och skalbara lösningar för verkliga tillämpningar.

Preface

This thesis contains the following papers.

Paper I **Ali Dadras**, Sebastian U. Stich, and Alp Yurtsever. Personalized Federated Learning via Low-Rank Matrix Factorization. *Published in OPT 2024: Optimization for Machine Learning*, 2024.

Paper II Sourasekhar Banerjee, **Ali Dadras**, Alp Yurtsever and Monowar Bhuyan. Personalized Multi-tier Federated Learning. *Accepted for publication in the 31st International Conference on Neural Information Processing (ICONIP)*, 2024.

Paper III **Ali Dadras**, Sourasekhar Banerjee, Karthik Prakhya, and Alp Yurtsever. Federated Frank-wolfe Algorithm. *Published in the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, 2024.

In addition, the following works were conducted during the PhD studies but are not included as part of this thesis.

Paper I **Ali Dadras**, Klara Leffler, and Jun Yu. A Ridgelet Approach to Poisson Denoising. *arXiv preprint arXiv:2401.16099*, 2024.

Paper II Zahra Kharaghani, **Ali Dadras**, Alp Yurtsever, and Tommy Löfstedt. Fairness Regularisation in Federated Learning. *Ongoing work.*

Paper III Karlo Palenzuela, **Ali Dadras**, Tommy Löfstedt, and Alp Yurtsever. Communication-Efficient Federated Learning with Multiple Local Steps. *Ongoing work.*

This thesis was supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by Knut and Alice Wallenberg Foundation.

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Prof. **Jun Yu**, and my co-supervisor, Prof. **Alp Yurtsever**. Jun played a crucial role in guiding my thesis work and helping me develop as an independent researcher. He has been an incredible source of support during difficult times; he always made time for me, no matter how busy he was. Our conversations went far beyond research, and his insightful advice, encouragement, and trust in me meant more than I can express. I am also sincerely thankful to Alp, who played a significant role in shaping my academic journey. He guided me through complex scientific problems, helped me stay motivated, and was always available for thoughtful discussions. His dedication and approach to research, along with his perspective on life, have made him an academic role model whom I deeply respect.

I would also like to sincerely thank Prof. **Sebastian Stich** and Prof. **Monowar Bhuyan**, who were co-authors in my research papers. Prof. Stich provided me with the invaluable opportunity to visit his lab in Germany. My time there was incredibly enriching, as I gained valuable knowledge about the field of machine learning and learned what it means to conduct research at a state-of-the-art level. I deeply appreciated the unique approach of his team toward scientific problems, which broadened my perspective on research. I am thankful to Prof. Bhuyan for his insightful meetings, discussions, and thoughtful comments, which greatly contributed to improving my work. I would also like to extend my heartfelt thanks to my co-author and great friend, Dr. **Sourasekhar Banerjee**. Our excellent weekly meetings greatly contributed to deepening my understanding of numerical experiments in machine learning.

I would like to extend my heartfelt gratitude to the **Wallenberg AI, Autonomous Systems, and Software Program (WASP)** for their generous financial support, which was instrumental in enabling this research. I am also deeply appreciative of the access to **Berzelius GPUs**, which provided the computational power necessary to carry out my work. The support and resources offered by WASP have been fundamental to the progress and successful completion of my research, for which I am sincerely grateful.

Life would not have been the same without the support of my family and friends. I'm deeply thankful to my parents and my sister for always being there and believing in me through every step of this journey. A special thanks to my

friends **Hamed Haghshenas**, **Saeed Razavikia**, and **Saeideh Ghanbari Azar** for putting up with my endless complaints and somehow surviving it all. Your patience and support kept me sane during moments when abandoning everything and becoming a goat herder in the mountains felt like a solid career move!

Ali Dadras
Umeå, January, 2025

Contents

Chapter 1

Introduction

The rapid growth of data generation, the impracticality of centralizing large-scale datasets, and the increasing complexity of computational tasks have driven the Machine Learning (ML) community to develop new tools and techniques, with **Distributed Learning** at its core [1].

Recently, the demand for privacy-preserving approaches in ML has become increasingly critical. **Federated Learning (FL)** stems from the idea of division of the learning task among multiple computing agents (commonly called clients or nodes) while respecting data privacy [2]. Unlike centralized learning, where a model is trained on a unified dataset, FL must handle diverse data distributions and communication constraints. In essence, Federated Learning (FL) is particularly suitable for scenarios where clients must collaborate because it lacks a large, representative dataset that adequately captures the diversity and variability of underlying behaviors. However, this collaboration occurs in an environment where direct data sharing is restricted, often due to strict concerns and regulations regarding data privacy and storage. Consequently, FL research has been focused on designing algorithms that can solve optimization and learning problems on a network without sharing essential data but instead communicating decision variables (referred to as local models in machine learning) or other auxiliary variables like gradients or step-directions alongside privacy improvement strategies such as encryption or noise injection.

The restriction of data sharing in FL introduces challenges that extend beyond the training process. Among the most significant is the difficulty in ensuring the quality and relevance of the data provided by participating clients, which is further intensified by the presence of heterogeneous data (i.e., data distributions that vary significantly across clients). This heterogeneity can heighten the challenge of building effective models[3]. Conventional FL methods that produce a single global model as the output often struggle to address these issues, leading to unstable performance due to the client drift problem. This problem arises when the single global model fails to accommodate the diverse needs and data distributions of all clients [4, 5]. To tackle these chal-

lenges, **personalized federated learning** (pFL) has emerged as a promising approach, enabling the development of models tailored to individual clients while still benefiting from collaborative training [6].

Associated with FL are significant computational and communication costs. One of the primary objectives in the field is to design optimization algorithms capable of efficiently handling a large number of parameters, particularly in complex learning tasks where parameters are subject to constraints. A portion of this thesis focuses on developing computationally and communication-efficient algorithms for **Constrained Federated Learning** problems. Furthermore, this framework provides a foundation for developing various personalized Federated Learning (pFL) algorithms, which will be discussed in detail in the subsequent chapters. The challenges addressed in this thesis can be summarized as follows:

Data Heterogeneity. A significant challenge in FL arises from data heterogeneity, as data across devices often originates from diverse, non-independent, and non-identically distributed (non-IID) distributions. Conventional FL systems assume that a single global model can serve all devices, which not only slows convergence but also compromises model accuracy in non-IID settings. This one-size-fits-all approach fails to accommodate the unique data characteristics of each client, resulting in suboptimal performance and reduced utility in personalized applications.

Computation Costs. The complexity of local computations, such as solving constrained optimization problems, underscores the need for developing efficient algorithms. In FL, clients often have limited computational resources, which can make solving complex optimization tasks impractical. Additionally, the iterative nature of FL algorithms requires frequent local updates, further emphasizing the importance of lightweight and scalable computational methods to ensure feasibility and efficiency across diverse client devices.

Communication Costs. Communication often becomes a bottleneck in real-world FL applications, particularly when the number of clients is large, the model has a high number of parameters, or network connections are unreliable. The frequent exchange of model updates between clients and the central server can result in significant delays and resource usage. To address this, FL requires communication-efficient algorithms that minimize the volume and frequency of data transfer, while maintaining convergence guarantees and model performance, even in environments with limited bandwidth.

In summary, $\texttt{pFL}^{\texttt{MF}}$ presents a novel pFL formulation grounded in low-rank matrix optimization and introduces a new pFL algorithm based on Burer-Monteiro factorization [7]. `PerMFL` presents a pFL algorithm that utilizes a multi-tier architecture to optimize and personalize local models, particularly in scenarios with predefined team structures across devices [8]. `FedFW` introduces

a projection-free algorithm for constrained FL problems, emphasizing data privacy, low per-iteration computational cost, and the communication of sparse signals [9].

Chapter 2

Basic First-order Optimization Algorithms

First-order optimization algorithms form the backbone of modern machine learning. Gradient descent (`GD`) and its variants (`SGD`, `ADAM`, and others) are the primary algorithms for solving unconstrained problems, while projected gradient descent (`PGD`) and the Frank-Wolfe (`FW`) algorithm are commonly used for constrained problems. In this chapter, we review common assumptions and definitions, as well as the convergence analyses of basic optimization algorithms in the centralized setting.

2.1 Assumptions and Definitions

Consider the following unconstrained optimization problem:

$$\min_{\mathbf{x}\in\mathbb{R}^d} F(\mathbf{x})\,. \tag{2.1}$$

We assume that the optimization problem has a non-empty solution set $\mathcal{X}^\star$, and we define the corresponding optimal function value as $F^\star := \inf_{\mathbf{x}\in\mathcal{B}} F(\mathbf{x})$. Furthermore, we assume the objective function $F\colon \mathbb{R}^d \to \mathbb{R}$ is L-smooth, as defined below. Throughout this thesis, $\langle\cdot,\cdot\rangle$ denotes the inner product, and $\|\cdot\|_2$ represents the Euclidean norm.

Definition 1 (L-smoothness)**.** *A differentiable function $F\colon \mathbb{R}^d \to \mathbb{R}$ is L-smooth if its gradient satisfies*

$$\|\nabla F(\mathbf{x}) - \nabla F(\mathbf{y})\|_2 \le L\,\|\mathbf{x}-\mathbf{y}\|_2\,, \quad \forall\,\mathbf{x},\mathbf{y}\in\mathbb{R}^d.$$

This property implies that for all $\mathbf{x},\mathbf{y}\in\mathbb{R}^d$,

$$F(\mathbf{y}) \le F(\mathbf{x}) + \langle\nabla F(\mathbf{x}),\mathbf{y}-\mathbf{x}\rangle + \frac{L}{2}\|\mathbf{y}-\mathbf{x}\|_2^2\,. \tag{smoothness}$$

The smoothness assumption provides a sufficient condition for ensuring the convergence of many first-order optimization algorithms.

In addition to the L-smoothness assumption, another common assumption on $F(\cdot)$ is μ-strong convexity, a property satisfied by many widely used objective functions. Strong convexity not only ensures that F is convex but also guarantees the existence of a unique minimum. This is formally defined as follows:

Definition 2 (μ-strong convexity)**.** *A differentiable function $F\colon \mathbb{R}^d \to \mathbb{R}$ is μ-strongly convex if there exists a constant $\mu > 0$ such that, for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$:*

$$F(\mathbf{y}) \geq F(\mathbf{x}) + \langle \nabla F(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle + \frac{\mu}{2} \|\mathbf{y} - \mathbf{x}\|_2^2 \,. \qquad \textbf{(strong-convexity)}$$

The strong convexity assumption ensures the uniqueness of the solution and enables linear convergence rates for gradient-based optimization algorithms when combined with L-smoothness.

We assume that for every point $\mathbf{x} \in \mathbb{R}^d$ we can query a stochastic gradient $\mathbf{g}(\mathbf{x})$ of $F(\mathbf{x})$, that is

$$\mathbf{g}(\mathbf{x}) := \nabla F(\mathbf{x}) + \boldsymbol{\xi}(\mathbf{x}) \,, \tag{2.2}$$

where $\boldsymbol{\xi}(\mathbf{x}) \in \mathbb{R}^d$ denotes a realization of a zero-mean random variable. We do, in general, not assume that the noise is independent of $\mathbf{x}$. For the gradient noise $\boldsymbol{\xi}(\mathbf{x})$, we assume the following

Assumption 1 ((M, σ^2)-bounded noise)**.** *There exist parameters $M \geq 0, \sigma^2 \geq 0$ such that for every gradient oracle as in (2.2) and for all $\mathbf{x} \in \mathbb{R}^d$:*

$$\mathbb{E}\left[\boldsymbol{\xi}(\mathbf{x})\right] = \mathbf{0}_d \,, \quad \mathbb{E}\left[\|\boldsymbol{\xi}(\mathbf{x})\|^2\right] \leq M \|\nabla F(\mathbf{x})\|^2 + \sigma^2 \,. \qquad \textbf{(bounded noise)}$$

For $M = 0$, we recover the uniformly bounded noise assumption.

2.2 Convergence Criteria

In the context of first-order optimization methods, convergence refers to the behavior of the optimization algorithm as it iteratively updates its parameters toward an optimal solution. Let $\mathbf{x}^t$ denote the output of the algorithm at iteration t, and $F : \mathbb{R}^d \to \mathbb{R}$ be the objective function to be minimized. Several notions of convergence are commonly analyzed in the literature:

- **Convergence in function value:** This criterion evaluates how closely the sequence of iterates approaches the minimum objective value, often measured by the difference $F(\mathbf{x}^t) - F^\star$, where $F^\star$ is the optimal value. The goal is to ensure that $F(\mathbf{x}^t) \to F^\star$ as $t \to \infty$.

- **Convergence in gradient norm:** This criterion focuses on the norm of the gradient, $\|\nabla F(\mathbf{x}^t)\|$, which measures the proximity of $\mathbf{x}^t$ to a stationary point. The objective is to show that $\|\nabla F(\mathbf{x}^t)\| \to 0$ as $t \to \infty$.
- **Convergence in iterates:** This criterion examines how the distance between the iterates and the optimal solution diminishes, typically measured by $\|\mathbf{x}^t - \mathbf{x}^\star\|$, where $\mathbf{x}^\star$ is an optimal solution to the problem. The objective is to establish that $\|\mathbf{x}^t - \mathbf{x}^\star\| \to 0$ as $t \to \infty$.
- **Convergence in stochastic settings:** In stochastic optimization, where the objective function is defined as an expectation over a random variable, the iterates are updated using noisy gradient estimates. Convergence in this context typically refers to ensuring that the expected function value $\mathbb{E}[f(\mathbf{x}^t)]$ approaches the optimal value $f^\star$. For non-convex optimization in stochastic settings, the notion of convergence typically involves the expected norm of the gradient. Specifically, the objective is to ensure that $\mathbb{E}[\|\nabla f(\mathbf{x}^t)\|^2] \to 0$ as $t \to \infty$. This indicates that the algorithm converges to a stationary point, which may be a local minimum, saddle point, or even a local maximum. Under mild assumptions, such as Lipschitz continuity of the gradient and bounded variance of the stochastic gradients, this criterion ensures that iterates approach a region of low gradient magnitude. Furthermore, advanced techniques like stochastic variance reduction methods and adaptive algorithms (e.g., Adam, Adagrad) can improve practical convergence behavior.
- **Convergence using the Frank-Wolfe gap:** In the context of Frank-Wolfe algorithms, convergence is often measured using the Frank-Wolfe gap. The Frank-Wolfe gap function $\text{gap}(\cdot)$ quantifies how far a given point $\mathbf{x}$ is from optimality within the feasible region $\mathcal{B}$. It is defined as

$$\text{gap}(\mathbf{x}) := \max_{\mathbf{u} \in \mathcal{B}} \langle \nabla F(\mathbf{x}), \mathbf{x} - \mathbf{u} \rangle . \tag{2.3}$$

  The Frank-Wolfe gap provides a useful stopping criterion, as it directly measures the discrepancy between the current iterate and the optimal solution. The goal is to ensure that $\text{gap}(\mathbf{x}^t) \to 0$ as $t \to \infty$, indicating convergence to a stationary point of the constrained optimization problem.
- **Convergence rate:** Depending on the problem structure (e.g., convex, strongly convex, or non-convex), different convergence rates can be established. For non-convex problems, sublinear convergence rates of the form $\mathcal{O}(1/t)$ are commonly expected. In contrast, for convex or strongly convex functions, linear convergence rates of the form $\mathcal{O}(\rho^t)$, with $\rho \in (0, 1)$, or quadratic convergence may be achievable under stronger assumptions. Here, the notation $\mathcal{O}(g(t))$ denotes an asymptotic upper bound, meaning that there exists a constant $C > 0$ and a sufficiently large t_0 such that the error is bounded by $Cg(t)$ for all $t \geq t_0$.

These convergence notions provide valuable insight into the efficiency and reliability of optimization algorithms, serving as a foundation for their theoretical

analysis and practical deployment. The next sections offer a brief overview of the convergence guarantees for GD, SGD, and FW algorithms, highlighting their behavior under various assumptions about the smoothness and convexity of the objective function. For a comprehensive treatment and detailed proofs of these results, the reader is referred to [10, 11, 12, 13, 14].

2.3 Algorithms

Gradient Descent.

The GD algorithm with a step size η is defined by the update rule:

$$\text{for } t = 0, 1, \ldots,$$
$$\mathbf{x}^{t+1} = \mathbf{x}^t - \eta \nabla F(\mathbf{x}^t), \tag{GD}$$

where $\mathbf{x}^0 \in \mathbb{R}^d$ is the initial parameter.

Theorem 1 (strongly-convex)**.** *Consider the optimization problem* (2.1) *where* F *satisfies the* **smoothness** *and* **strong-convexity** *assumptions. Then,* GD *algorithm achieves a global linear convergence rate, where the objective residual satisfies*

$$F(\mathbf{x}^t) - F^\star \leq (1 - \mu\eta)^t \big(F(\mathbf{x}^0) - F^\star\big),$$

provided that $\eta \leq \frac{1}{L}$.

Theorem 2 (non-convex)**.** *Consider the optimization problem* (2.1) *where* F *is non-convex and satisfies* **smoothness** *assumption. Then, the iterates of* GD *algorithm with a step size* $\eta \leq \frac{1}{L}$ *satisfy*

$$\frac{1}{T} \sum_{t=0}^{T-1} \|\nabla F(\mathbf{x}^t)\|^2 \leq \frac{2\big(F(\mathbf{x}^0) - F^\star\big)}{\eta T}.$$

Stochastic Gradient Descent.

The SGD algorithm with step size η_t is defined by the update rule:

$$\text{for } t = 0, 1, \ldots,$$
$$\mathbf{x}^{t+1} = \mathbf{x}^t - \eta_t \mathbf{g}(\mathbf{x}^t), \tag{SGD}$$

where $\mathbf{g}(\cdot)$ is a stochastic gradient of $F(\cdot)$, as defined in (2.2), and $\mathbf{x}^0 \in \mathbb{R}^d$ is the initial parameter.

Theorem 3 (strongly-convex)**.** *Consider problem* (2.1) *with objective function* F *satisfying* **strong-convexity**, **smoothness**, **bounded noise** *assumptions.*

Then, the iterates of SGD *algorithm with a step size of* $\eta_t = \frac{2}{\mu(t+\delta)}$ *(* $\delta > 0$ *is a constant that is chosen such that* $\eta_t \leq \frac{1}{L(M+1)}$*) satisfy*

$$\mathbb{E}\big[F(\mathbf{x}^t) - F^\star\big] \leq \frac{\nu}{t+\delta}, \quad \nu := \max\left((\delta+1)(F(\mathbf{x}^0) - F^\star), \frac{2L\sigma^2}{\mu^2}\right).$$

If instead we use a constant step size $\eta_t = \eta < \frac{1}{L(M+1)}$*, then we obtain a linear convergence rate up to a neighborhood of a solution that is proportional to* η*,*

$$\mathbb{E}\big[F(\mathbf{x}^t) - F^\star\big] \leq (1-\eta\mu)^t\big(F(\mathbf{x}^0) - F^\star\big) + \eta\frac{L}{2\mu}\sigma^2.$$

Theorem 4 (non-convex)**.** *Consider problem* (2.1) *with objective function* F *satisfying* **smoothness**, **bounded noise** *assumptions. Then, the iterates of* SGD *algorithm satisfy*

$$\frac{1}{T}\sum_{t=0}^{T-1}\|\nabla F(\mathbf{x}^t)\|^2 \leq \frac{\sqrt{8\big(F(\mathbf{x}^0) - F^\star\big)L\sigma^2}}{\sqrt{T}},$$

provided that $\eta = \frac{\eta_0}{\sqrt{T}}$ *where* $\eta_0 \leq \min\{\frac{1}{L(M+1)}, \sqrt{\frac{2(F(\mathbf{x}^0)-F^\star)}{L\sigma^2}}\}$.

Frank-Wolfe algorithm. Consider the following constrained optimization problem:

$$\min_{\mathbf{x}\in\mathcal{B}} F(\mathbf{x}), \tag{2.4}$$

where $\mathcal{B} \subseteq \mathbb{R}^d$ is a convex and compact set.

The FW algorithm with step size $\eta_t = \frac{2}{t+2}$ is defined by the update rule:

$$\begin{aligned} &\text{for } t = 0, 1, \ldots, \\ &\quad \mathbf{s}^t \in \operatorname*{argmin}_{\mathbf{x}\in\mathcal{B}} \big\langle \nabla F(\mathbf{x}^t), \mathbf{x}\big\rangle \\ &\quad \mathbf{x}^{t+1} = (1-\eta_t)\mathbf{x}^t + \eta_t\mathbf{s}^t, \end{aligned} \tag{FW}$$

where $\mathbf{x}^0 \in \mathbb{R}^d$ is the initial parameter.

Theorem 5 (convex)**.** *Consider the optimization problem* (2.4)*, where* F *is convex and satisfies the* **smoothness** *assumption. Then,* FW *algorithm achieves a global linear convergence rate, where the objective residual satisfies*

$$F(\mathbf{x}^t) - F^\star \leq \frac{2LR^2}{t+2},$$

where $R := \max_{\mathbf{x},\mathbf{y}\in\mathcal{B}}\|\mathbf{x}-\mathbf{y}\|$ *is the diameter of the constraint set* $\mathcal{B}$*.*

Theorem 6 (non-convex)**.** *Consider the optimization problem* (2.4)*, where F is a non-convex and satisfies the* **smoothness** *assumption. Then, the Frank-Wolfe algorithm* (FW) *achieves the following convergence rate:*

$$\min_{0\leq t\leq T-1} \operatorname{gap}(\mathbf{x}^t) \leq \frac{\max\left\{2\big(F(\mathbf{x}^0)-F^\star\big), LR^2\right\}}{\sqrt{T}}, \tag{2.5}$$

where $F^\star := \min_{\mathbf{x}\in\mathcal{B}} F(\mathbf{x})$, $R := \max_{\mathbf{x},\mathbf{y}\in\mathcal{B}} \|\mathbf{x}-\mathbf{y}\|$ is the diameter of the constraint set $\mathcal{B}$, and L is the smoothness constant of F. The function $\operatorname{gap}(\cdot)$*, referred to as the Frank-Wolfe gap, is defined as*

$$\operatorname{gap}(\mathbf{x}) := \max_{\mathbf{u}\in\mathcal{B}} \langle \nabla F(\mathbf{x}), \mathbf{x}-\mathbf{u}\rangle\,. \tag{2.6}$$

Chapter 3

Brief Overview of Federated Learning

Federated Learning (FL) is a distributed machine learning (ML) paradigm designed to enable collaborative model training across a large network of clients while restricting data sharing [2]. It is particularly valuable in scenarios where clients possess data that, individually, may be insufficient to build a robust model, but collectively can lead to significant learning improvements. However, due to privacy concerns or legal regulations, such as the Health Insurance Portability and Accountability Act (HIPPA) [15], General Data Protection Regulation (GDPR) [16], and California Consumer Privacy Act (CCPA) [17], directly sharing raw data among participants is not feasible. Instead, FL relies on exchanging local models or auxiliary information, such as gradients or step directions, to achieve learning goals without compromising data privacy.

In classical ML, data is typically collected and stored centrally for model training. By contrast, FL shifts the computational burden to edge devices, enabling them to train models locally on their private data. These local models are periodically sent to a central server, which aggregates them to form a global model. This approach not only preserves user privacy but also mitigates data transmission costs by avoiding the need to transfer large datasets [18]. Consequently, FL has emerged as an essential framework for solving large-scale optimization and learning problems across a distributed network, while simultaneously addressing privacy and communication efficiency [19].

The core idea of FL is to create a virtual, optimal model by aggregating locally trained models from multiple clients. A standard FL setup involves three key steps:

- **Local Training** – Each client trains a local model on its private dataset.
- **Model Aggregation** – The central server collects the local models, aggregates them (e.g., using weighted averaging), and updates the global model.

- **Broadcasting** – The updated global model is sent back to the clients, who repeat the process in subsequent rounds.

Unique privacy requirements and the decentralized structure of FL make it susceptible to specific challenges in real-world applications. Below, we provide a brief review of some important challenges in FL; however, this is not an exhaustive list.

Heterogeneity challenge. In real-world FL scenarios, clients are inherently heterogeneous, with variations in data volume, distribution, computational power, and network resources. We provide a brief overview of some key factors contributing to the heterogeneity issues in FL:

- **Imbalanced data.** Clients may collect data with varying quantities and label distributions, leading to imbalanced datasets across participants [20].
- **Non-IID data.** Data collected by clients may follow different underlying distributions due to varying user behaviors, device usage patterns, and environmental factors, resulting in non-IID (non-independent and identically distributed) datasets [21].
- **Massively distributed data.** FL involves a large number of geographically distributed clients, which introduces scalability challenges due to the high communication and computation overhead [22].
- **Model heterogeneity.** Due to variations in clients' computational power, storage, and network capabilities, clients may train models of different complexities, resulting in structural discrepancies across local models. These differences can hinder effective model aggregation and degrade overall performance [22]..

Communication and computation Costs. Communication is a critical bottleneck in real-world FL applications, especially when dealing with a large number of clients, high-dimensional models, or unreliable network connections. The repeated exchange of model updates between clients and the central server can lead to substantial delays and high resource consumption. Therefore, FL necessitates communication-efficient algorithms that reduce both the volume and frequency of data transmission, ensuring reliable convergence and high model performance, even in bandwidth-constrained environments [23, 24].

Privacy. Although raw data is not shared in an FL system, sensitive information about the data or clients can still be inferred through carefully designed attack methods, such as model inversion attacks [25, 26], membership inference attacks [27], and gradient-based attacks [28]. For a comprehensive review of these attack methods, see [18] and references therein.

3.1 Problem formulation

In this section, we present a formal mathematical formulation of a learning task in FL. Consider a system with n clients collaborating in the FL framework, indexed by $i = 1, \ldots, n$. Each client's data is assumed to be drawn from a specific data distribution, denoted by $\mathcal{D}_i$. The objective function for client i is defined as:

$$f_i^\natural(\mathbf{x}) := \mathbb{E}_{\xi \sim \mathcal{D}_i} \ell_i(\mathbf{x}, \xi), \tag{3.1}$$

where $\mathbf{x}$ represents the model parameters, ξ is a random variable sampled from $\mathcal{D}_i$, and $\ell_i(\mathbf{x}, \xi)$ denotes the loss function for client i. When data distributions are known, a solution to this problem can be found by minimizing $f_i^\natural(\mathbf{x}_i)$ locally:

$$\min_{\mathbf{x}_i} \; f_i^\natural(\mathbf{x}_i). \tag{3.2}$$

However, the true distribution $\mathcal{D}_i$ is unknown in practice. Instead, clients have access to an empirical sample $\mathcal{M}_i$ with the corresponding objective function:

$$f_i(\mathbf{x}) := \frac{1}{|\mathcal{M}_i|} \sum_{\xi \in \mathcal{M}_i} \ell_i(\mathbf{x}, \xi). \tag{3.3}$$

Note that, we operate under the assumption that the dataset $\mathcal{M}_i$ is not large enough for clients to accurately approximate a solution to problem (3.2) locally on their own. *Otherwise, FL would not be required.* An effective solution to this problem is possible only if the distributions $\mathcal{D}_i$ exhibit some correlation that we can exploit. At one extreme, when all distributions are the same, the standard FL template can be used, which can be formulated as

$$\min_{\mathbf{x} \in \mathbb{R}^d} F(\mathbf{x}) := \; \frac{1}{n} \sum_{i=1}^{n} \; f_i(\mathbf{x}) \,. \tag{P}$$

The finite-sum problem in (**P**) serves as a common template in many machine learning tasks. While this problem can be efficiently solved in a centralized scenario, where all data is readily accessible, obtaining a comparable solution in an FL setting requires more sophisticated and carefully designed approaches. The goal is to minimize the finite-sum problem (**P**) iteratively over n clients. By introducing n copies of the parameter $\mathbf{x}$, we can rewrite (**P**) in an equivalent form as:

$$\min_{\mathbf{x}_1, \ldots, \mathbf{x}_n} \; \frac{1}{n} \sum_{i=1}^{n} f_i(\mathbf{x}_i) \quad \text{s.t.} \quad \mathbf{x}_1 = \cdots = \mathbf{x}_n \,. \tag{3.4}$$

To solve this problem, we require the projection onto the consensus set, defined as:

$$\mathcal{C} := \{[\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{d \times n} : \mathbf{x}_1 = \mathbf{x}_2 = \cdots = \mathbf{x}_n\}.$$

This projection is equivalent to taking the average. The concept of updating parameters at the clients and averaging them at the server forms the foundation of the well-known Federated Averaging (`FedAvg`) algorithm.

3.2 Algorithm Design in FL

In this section, we review various algorithms proposed to solve (**P**) while addressing the key challenges in FL. We begin with the well-known algorithm in FL, `FedAvg` [19], which serves as the baseline for solving (**P**), and highlight some of the challenges it faces. The remainder of the section briefly reviews key algorithms that laid the foundation for the development of personalized and constrained FL, highlighting important design principles. This is not an exhaustive list but rather a selection of influential initial works. We follow the definitions and notations introduce in chapter 2.

Federated Averaging. `FedAvg` [19], also known as Local SGD [29], depicted in Algorithm 1, is one of the most popular algorithms in Federated Learning and serves as a building block for designing many others. `FedAvg` proceeds over T communication rounds. At the beginning of each round, a central server sends the current global model ($\bar{\mathbf{x}}^t$) to each of the n clients. Each client locally performs K steps of `SGD`, where the updates are given by $\mathbf{x}_i^{t,k+1} = \mathbf{x}_i^{t,k} - \eta_t \mathbf{g}_i^{t,k}$, and $\mathbf{g}_i^{t,k}$ is a stochastic gradient of $f_i(\cdot)$. After completing the local steps, each client returns its final local model ($\mathbf{x}_i^{t,K}$) to the central server. The server then averages these iterates to compute the global model for the next round ($\bar{\mathbf{x}}^{t+1}$).

Algorithm 1 `FedAvg` (Local `SGD`)

Initialize variables $\bar{\mathbf{x}}^0 = \mathbf{x}_0$ for clients $i \in \{1, 2, \ldots, n\}$
for $t = 0, 1, \ldots, T-1$ **do**
  - - Client-level training - -
  for client $i \in \mathcal{S}_t$ **do**
    $\mathbf{x}_i^{t,0} = \bar{\mathbf{x}}^t$
    for $k = 0, 1, \ldots, K-1$ **do**
      $\mathbf{x}_i^{t,k+1} = \mathbf{x}_i^{t,k} - \eta_t \, \mathbf{g}_i^{t,k}$
    end for
    $\mathbf{x}_i^{t+1} = \mathbf{x}_i^{t,K}$
    Client communicates $\mathbf{x}_i^{t+1}$ to the server.
  end for

  - - Server-level aggregation - -
  $\bar{\mathbf{x}}^{t+1} = \frac{1}{|\mathcal{S}_t|} \sum_{i \in \mathcal{S}_t} \mathbf{x}_i^{t+1}$
  Server communicates $\bar{\mathbf{x}}^{t+1}$ to the clients.

end for

- **Local updates.** `FedAvg` with a single local update ($K = 1$) is equivalent to `SGD` algorithm applied on $F(\cdot)$

$$\bar{\mathbf{x}}^{t+1} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i^{t+1} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i^{t,1} = \frac{1}{n}\sum_{i=1}^{n} \left(\mathbf{x}_i^{t,0} - \eta_t \mathbf{g}_i^{t,0}\right) = \bar{\mathbf{x}}^t - \eta_t \mathbf{g}(\bar{\mathbf{x}}^t)\,.$$

  On the other hand, consider the case where the number of local updates is very large ($K \approx \infty$). In this scenario, each client finds its own individual local minimizer (assume objective functions are smooth and strongly convex), and the global model becomes the average of these individual local minima, which is a fixed point of `FedAvg` and can be far from the minimizer of (**P**). Another scenario arises when different clients perform different numbers of local updates (K_i). In this case, it has been shown that `FedAvg` converges to a stationary point of a surrogate of the function $F(\cdot)$ in (**P**) [30]. While taking multiple local steps per client helps reduce communication frequency [31], it can introduce client drift—where the client models diverge significantly from the global model due to differences in local data distributions.

- **Partial Participation.** A key challenge in `FedAvg` is addressing random device participation schedules. Unlike classical distributed optimization schemes, in most FL applications, clients have a certain level of autonomy and are not fully controlled by the server. Due to various factors, such as network congestion or resource constraints, clients may participate intermittently in the training process. Consequently, in each round, only a subset of all clients, denoted by $\mathcal{S}_t \subseteq \{1, 2, \ldots, n\}$, participates [32, 33].

- **Client drift.** Another important line of analysis focuses on evaluating the quality of the solution obtained by `FedAvg`. It has been shown that, even in the homogeneous case (i.e., when data distributions are identical across clients, $\mathcal{D}_i \equiv \mathcal{D}$ in (3.1)), `FedAvg` may drift away from the optimum, even if initialized at the optimal solution of problem (**P**); see [34]. This phenomenon, commonly referred to as *client drift*, can significantly degrade the performance of `FedAvg`. In essence, the average of the local models may deviate significantly from the global optimal solution, leading to suboptimal performance. This issue becomes even more critical when dealing with heterogeneous data [35].

- **Step size.** The client drift effect is closely linked to step size: larger step sizes can exacerbate drift, especially when K is large, as clients may overfit to their local objectives and stray far from the global solution [30]. Conversely, smaller step sizes help control drift by limiting each client's movement per step, though they may also slow convergence [36]. Balancing step size and the number of local steps is thus crucial in `FedAvg`; common strategies include using a larger number of local steps with a reduced step size [23] or employing adaptive techniques to dynamically adjust the step size [32], helping

to stabilize the learning process and improve convergence efficiency across heterogeneous clients.

- **Convergence analysis.** Convergence guarantees for the `FedAvg` algorithm can be derived under various assumptions such as bounded dissimilarity ($\exists \sigma > 0, \frac{1}{n}\sum_{i=1}^{n} \|\nabla f_i(\mathbf{x}) - \nabla F(\mathbf{x})\|_2^2 \leq \sigma^2$, for all $\mathbf{x} \in \mathbb{R}^d$) [37], bounded gradients ($\exists G > 0, \frac{1}{n}\sum_{i=1}^{n} \|\nabla f_i(\mathbf{x})\|_2^2 \leq G^2$, for all $\mathbf{x} \in \mathbb{R}^d$) [29, 38], and generic smoothness and convexity assumptions [39, 40]. For further discussions and extensions, see also [41, 34, 42]. These bounds show the convergence of `FedAvg` to a minimizer or a stationary point by *upper bounding* the *objective residual* or the *norm of the gradient*, respectively, see section 2.2.

Addressing heterogeneity via regularization. Among the earliest approaches to address the low performance of FL methods in heterogeneous scenarios is the idea of penalizing local models. This can be formulated as

$$f_i^{\text{reg}}(\mathbf{x}_i) = f_i(\mathbf{x}_i) + \psi_i(\mathbf{x}_i, \mathbf{x}), \tag{3.5}$$

where $\mathbf{x}$ denotes the global (server) model, and $\mathbf{x}_i$ represents the local model used to compute gradients. Note that $\psi(\cdot,\cdot)$ *can be designed such that the functions* $f_i^{\text{reg}}(\cdot)$ *have the same stationary points as* $F(\cdot)$ *in* (**P**).

- `FedProx` [43]**.** One possible option for the regularizer $\psi_i(\cdot,\cdot)$ is the proximal term $\psi_i(\mathbf{x}_i, \mathbf{x}) := \frac{\mu}{2}\|\mathbf{x}_i - \mathbf{x}\|^2$, where $\mu \geq 0$ is a hyperparameter of the algorithm. This regularizer encourages each local model $\mathbf{x}_i$ to remain close to the global model $\mathbf{x}$.

- `FedPD` [44] and `FedDyn` [45]**.** These algorithms incorporate the augmented Lagrangian of $f_i(\cdot)$, using the following regularizer:

$$\psi_i(\mathbf{x}, \bar{\mathbf{x}}^t) := \langle \boldsymbol{\theta}_i, \mathbf{x}_i - \mathbf{x} \rangle + \frac{\mu}{2}\|\mathbf{x}_i - \mathbf{x}\|^2 ,$$

  where $\boldsymbol{\theta}_i$ is an auxiliary local variable, updated at the end of each global iteration as $\boldsymbol{\theta}_i^{t+1} = \boldsymbol{\theta}_i^t + \mu(\mathbf{x}_i^{t,K} - \mathbf{x}^t)$, and $\mu > 0$ is a hyperparameter.

- `FedDANE` [46]**.** This work draws inspiration from the `DANE` algorithm [47] and uses the following regularizer:

$$\psi_i(\mathbf{x}, \mathbf{x}) := \left\langle \frac{1}{|\mathcal{S}_t|}\sum_{j \in \mathcal{S}_t} \nabla f_j(\mathbf{x}) - \nabla f_i(\mathbf{x}), \mathbf{x}_i - \mathbf{x} \right\rangle + \frac{\mu}{2}\|\mathbf{x}_i - \mathbf{x}\|^2 ,$$

  where $\mu > 0$. The original algorithm requires two communication rounds per global iteration. A variant that requires only one communication round is introduced in this paper.

Addressing Client Drift via Control Variates. Another line of research addresses client drift by employing the following techniques: **(1)** Momentum techniques such as `VRLSGD` [48] and `MIME` [49], **(2)** Cross-client variance reduction techniques including `scaffold` [23], `Scaffnew` [24], and `Scafflix` [50], and **(3)** Operator splitting techniques such as `FedSplit` [51]. A detailed discussion of these methods is beyond the scope of this thesis; see [52] for a more comprehensive treatment.

3.3 Personalized Federated Learning

Personalization in FL refers to an approach that leverages a global model to derive customized local models tailored to each individual client. Several methods have been proposed, categorized into different approaches for achieving personalization in FL, including:

Fine-tuning. Fine-tuning is a simple yet powerful technique that involves using the global model as an initialization point and further training it on each client's local data. This approach allows local models to better fit the specific data distribution of individual clients [53, 54].

- `MAML` [55]**.** This algorithm finds a global model $\mathbf{x}$ by solving the following optimization problem:

$$\min_{\mathbf{x}\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x} - \gamma\nabla f_i(\mathbf{x}))\,,$$

  where γ is the step size. Each client then uses the global model $\mathbf{x}$ as an initialization and performs a few gradient descent (`GD`) steps on its local data.

Distance-based methods. Many existing pFL methods are designed by relaxing the equality constraint in (3.4). *The goal is to find the local models* $\mathbf{x}_i$ *that are close to the global model* $\bar{\mathbf{x}}^t$ *and at the same time local objectives* $f_i(\mathbf{x}_i)$ *are small enough.*

- `pFedMe` [56]**.** This algorithm employs Moreau envelope smoothing and minimizes the following new objective function:

$$\min_{\mathbf{x}\in\mathbb{R}^d} \tilde{F}(\mathbf{x}) := \frac{1}{n}\sum_{i=1}^{n} \tilde{f}_i(\mathbf{x})\,, \tag{3.6}$$

  where $\tilde{f}_i(\mathbf{x}) = \min_{\mathbf{u}\in\mathbb{R}^d}\left\{f_i(\mathbf{u}) + \frac{1}{2\alpha}\left\|\mathbf{u}-\mathbf{x}\right\|^2\right\}$ and $\alpha > 0$. Note that as $\alpha\to\infty$, $\tilde{f}_i(\mathbf{x}) \to f_i^\star$ becomes independent of $\mathbf{x}$, and as $\alpha\to 0$, $\tilde{f}_i(\mathbf{x}) \to f_i(\mathbf{x})$ which reduces (3.6) to (**P**). Therefore, α controls the trade-off between the proximity of personalized models to the global model.

- **Interpolating Methods.** These methods consider personalized models as an interpolation between the local models $\mathbf{x}_i$ and the global model $\mathbf{x}$, given by

$$\mathbf{x}_i^{\text{per}} = \alpha_i \mathbf{x} + (1 - \alpha_i)\mathbf{x}_i \,,$$

where $0 \leq \alpha_i \leq 1$. Different algorithms have been proposed based on this interpolation.

  - `APFL` [57] This algorithm solves a new optimization problem for both the local models $\mathbf{x}_i$ and the global model $\mathbf{x}$ using alternating gradient descent:

$$\min_{\mathbf{x}, \mathbf{x}_i \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} f_i(\alpha_i \mathbf{x} + (1 - \alpha_i)\mathbf{x}_i) \,.$$

  - `FLIX` [58]**.** Each client first finds its individual minimizer $\mathbf{x}_i^\star$, and then the following optimization problem is solved:

$$\min_{\mathbf{x} \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} f_i(\alpha_i \mathbf{x} + (1 - \alpha_i)\mathbf{x}_i^\star) \,.$$

Model De-coupling Methods. The main idea behind these methods is to decouple each local model into two components: a feature extractor block and a classifier block. The feature extractor block is communicated and aggregated across clients, while the classifier block is trained locally by each client. It has been shown that selecting the last layer of a neural network as the personalization layer can significantly improve performance [59, 60, 61].

- `FedPer` [60]**.** Consider that each client's model can be partitioned into two parts $(\mathbf{x}, \mathbf{x}_i^{\text{per}})$, where $\mathbf{x}$ represents the part learned collaboratively across clients (commonly referred to as the feature extractor layer in neural networks), and $\mathbf{x}_i^{\text{per}}$ represents the personalized part (commonly referred to as the classifier layer) tailored to suit each client's specific data. This problem can be formalized as

$$\min_{\mathbf{x}, \mathbf{x}_i^{\text{per}} \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} f_i(\mathbf{x}, \mathbf{x}_i^{\text{per}}) \,.$$

  In this setting, each client locally updates its model and only shares the feature extractor layer (the initial layers of the network), while keeping the classifier layer (typically the final layers of the network) private.

- `FedRep` [62]**.** This method leverages all of the data stored across clients to learn a global low-dimensional representation using gradient-based updates. Further, it enables each client to compute a personalized, low-dimensional classifier, which we term as the client's head, which accounts for the unique labeling of each client's local data.

Client clustering methods. This set of algorithms clusters clients into groups based on similarities in their data distributions/models and trains a separate model for each group [63, 64, 65, 66, 67].

3.4 Constrained Federated Learning

Constrained FL refers to the setting where the objective is to minimize a global loss function while ensuring that the solution lies within a feasible set defined by constraints. This can be formulated as the following optimization problem:

$$\min_{\mathbf{x}\in\mathcal{B}} \quad F(\mathbf{x}) := \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x}), \tag{constrained P}$$

where $f_i(\mathbf{x})$ represents the local objective function at client i, and $\mathcal{B} \subseteq \mathbb{R}^d$ denotes the feasible region common to all clients. The constraints in $\mathcal{B}$ can arise from physical, operational, or privacy-related limitations specific to real-world applications. Most of the FL algorithms are designed for smooth objectives, **smoothness** assumption, which is not satisfied by **constrained P**. Therefore, solving FL problems with non-smooth objectives requires developing specialized algorithms, such as Proximal algorithms [68], projected gradient methods [69] or Frank-Wolfe variants [70].

- `FedDualAvg` [71]**.** Inspired by the primal-dual averaging method [72], this algorithm solves a Federated Composite Optimization (FCO) problem:

$$\min_{\mathbf{x}\in\mathbb{R}^d} F(\mathbf{x}) := \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x}) + \Psi(\mathbf{x})\,, \tag{FCO}$$

  where $\Psi(\cdot)$ is a general closed convex (possibly non-smooth) function [73]. Defining $\Psi(\mathbf{x})$ as the indicator function of the set $\mathcal{B}$, we have

$$\Psi(\mathbf{x}) = \begin{cases} 0, & \text{if } \mathbf{x} \in \mathcal{B}, \\ +\infty, & \text{otherwise}\,, \end{cases}$$

  in which case (**FCO**) reduces to (**constrained P**) as a special case. Each client performs the dual averaging algorithm, but only the dual states are aggregated.

- `FedDR` [74]**.** This algorithm aims to solve the (**FCO**) problem by combining the nonconvex Douglas-Rachford (DR) splitting technique with a randomized block-coordinate method.

Chapter 4

Summary of Contributions

This chapter lists the papers included in this thesis and outlines the contributions of the authors to each paper.

4.1 Paper I

Ali Dadras, Sebastian U Stich, and Alp Yurtsever, "Personalized Federated Learning via Low-Rank Matrix Factorization," *OPT 2024: Optimization for Machine Learning*, 2024.

Contributions. This paper introduces a novel formulation for personalized Federated Learning (pFL) based on low-rank matrix optimization:

$$\min_{\mathbf{X}\in\mathbb{R}^{d\times n}} \; F(\mathbf{X}) := \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x}_i) \quad \text{s.t.} \quad \operatorname{rank}(\mathbf{X}) \leq r\,, \tag{4.1}$$

where $\mathbf{X} \in \mathbb{R}^{d\times n}$ denotes the system-level decision variable obtained by concatenating clients' decision variables as $\mathbf{X} := [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]$, and r is a problem-specific tuning parameter. In this method, *personalized models are characterized by their membership in a low-dimensional subspace rather than their proximity according to a distance metric*; see section 3.3 for a discussion on distance-based pFL methods.

We adopt the well-known Burer-Monteiro (BM) factorization technique and replace the system-level decision variable $\mathbf{X} \in \mathbb{R}^{d\times n}$ with its factorized form $\mathbf{X} = \mathbf{U}\mathbf{V}^\top$. This leads to the following optimization problem:

$$\min_{\mathbf{U}\in\mathbb{R}^{d\times r},\, \mathbf{V}\in\mathbb{R}^{n\times r}} \; \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{U}\mathbf{v}_i)\,, \tag{4.2}$$

where $\mathbf{V}^\top := [\mathbf{v}_1, \cdots, \mathbf{v}_n] \in \mathbb{R}^{r\times n}$. In this notation, personalized model parameters are computed as $\mathbf{x}_i = \mathbf{U}\mathbf{v}_i \in \mathbb{R}^d$. The proposed algorithm, $\texttt{pFL}^{\texttt{MF}}$, uses

alternating stochastic gradient descent and can handle multiple local steps for feature extractors $\mathbf{v}_i$ as well as partial client participation with probability p. $\texttt{pFL}^{\texttt{MF}}$ achieves a convergence rate of $\mathcal{O}(1/\sqrt{T})$ in the stochastic gradient setting and $\mathcal{O}(1/T)$ in the deterministic gradient setting. We evaluated the proposed algorithm in various scenarios and compared its performance with state-of-the-art pFL methods.

Authors' contributions. The main idea of the paper was proposed by Ali Dadras and refined through discussions with Alp Yurtsever. Ali Dadras contributed to the theoretical analysis, conducted numerical experiments, and prepared the manuscript. Alp Yurtsever and Sebastian Stich provided continuous guidance, offering insightful feedback and suggestions for improving the analysis and revising the manuscript.

4.2 Paper II

Sourasekhar Banerjee, **Ali Dadras**, Alp Yurtsever, and Monowar Bhuyan, "Personalized Multi-Tier Federated Learning," *accepted for publication in the 31st International Conference on Neural Information Processing (ICONIP)*, 2024.

Contributions. This paper addresses multi-tier pFL by formulating an optimization problem that introduces personalized decision variables for both teams and individual devices, using squared Euclidean distance regularization:

$$\min_{\mathbf{x}\in\mathbb{R}^d} \min_{\mathbf{w}_i\in\mathbb{R}^d} \min_{\boldsymbol{\theta}_{i,j}\in\mathbb{R}^d} \frac{1}{M}\sum_{i=1}^{M}\frac{1}{N_i}\sum_{j=1}^{N_i}\left(f_{i,j}(\boldsymbol{\theta}_{i,j}) + \frac{\lambda}{2}\|\boldsymbol{\theta}_{i,j}-\mathbf{w}_i\|^2 + \frac{\gamma}{2}\|\mathbf{w}_i-\mathbf{x}\|^2\right), \tag{4.3}$$

where $f_{i,j}(\cdot)$ represents the loss function of the j^{th} device from the i^{th} team, and $\mathbf{w}_i$ and $\boldsymbol{\theta}_{i,j}$ are decision variables for each team and device, respectively. The hyperparameters $\gamma \geq 0$ and $\lambda \geq 0$ control the degree of personalization impact at the team and device levels, respectively.

The goal is to find a global model representing a rough average, capturing common characteristics across all teams and devices; personalized team models that are close to the global model but can deviate to capture shared features within each team; and personalized device models that resemble the team model but can deviate to accommodate the unique characteristics of each device.

We propose an algorithm, `PerMFL`, to solve this problem, which flexibly accommodates local objectives while jointly training both global and personalized models. We conducted extensive experiments across various scenarios and compared our method with state-of-the-art algorithms in pFL.

Authors' contributions. The main idea of the paper emerged through discussions among the authors. Ali Dadras conducted the convergence analysis

of the algorithm, while Sourasekhar Banerjee performed the numerical experiments and contributed to the manuscript preparation. Alp Yurtsever provided guidance on the analytical steps. Both Alp Yurtsever and Monowar Bhuyan offered valuable feedback on the manuscript and contributed to its revision.

4.3 Paper III

Ali Dadras, Sourasekhar Banerjee, Karthik Prakhya, and Alp Yurtsever, "Federated Frank-Wolfe Algorithm," *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2024.

Contributions. This paper introduces `FW`-type methods, `FedFW`, for solving the Constrained FL problem:

$$\min_{\mathbf{x}\in\mathcal{B}} \quad F(\mathbf{x}) := \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x})\,, \tag{4.4}$$

where $\mathcal{B} \subseteq \mathbb{R}^p$ is a convex and compact set. As mentioned in section 3.4, the objective function (4.4) is non-smooth, and therefore `FW` cannot be applied directly to this problem. By combining the smoothing technique from [75], variants of the `FW` algorithm, and taking into account the privacy aspect of FL, we introduced three `FW`-type algorithms for FL. The vanilla version uses modified `FW` step directions:

$$\mathbf{s}_i^t \in \operatorname*{argmin}_{\mathbf{x}\in\mathcal{B}} \left\langle \frac{1}{n}\nabla f_i(\mathbf{x}_i^t) + \lambda_t(\mathbf{x}_i^t - \bar{\mathbf{x}}^t), \mathbf{x} \right\rangle, \tag{4.5}$$

at the client level, where $\lambda_t = \lambda_0\sqrt{t+1}$ for any $\lambda_0 > 0$. The local step directions $\mathbf{s}_i^t$ are then used to update the local models $\mathbf{x}_i$:

$$\mathbf{x}_i^{t+1} = (1-\eta_t)\mathbf{x}_i^t + \eta_t \mathbf{s}_i^t, \tag{4.6}$$

where $\eta_t = \frac{2}{t+1}$ is the step size. `FedFW` provides better privacy and communication properties by sharing (sparse) step directions instead of gradients. `FedFW` guarantees $\mathcal{O}(t^{-1/2})$ convergence rates when the objective function is smooth and convex. If the convexity assumption is removed, the rate reduces to $\mathcal{O}(t^{-1/3})$. With access to only stochastic gradients, `FedFW` achieves an $\mathcal{O}(t^{-1/3})$ convergence rate in the convex setting. Additionally, we proposed an empirically faster version of `FedFW` by incorporating an augmented Lagrangian dual update. We evaluated the proposed algorithm in various scenarios and compared its performance with state-of-the-art pFL methods.

- `personalized FedFW.` An extension of `FedFW` to the pFL framework can be achieved by noting the impact of λ_t. Increasing λ_t forces all local models to become closer to each other over iterations. This can be treated as a design parameter to control the level of proximity of the local models. In other words, a fixed λ allows us to control the closeness of the local models $\mathbf{x}_i^t$ to the global model $\bar{\mathbf{x}}^t$.

Authors' contributions. The main idea of the paper was proposed by Alp Yurtsever. Ali Dadras derived the theoretical results, contributed to writing the manuscript, and assisted with preliminary numerical experiments. Karthik Prakhya conducted the initial numerical experiments. Sourasekhar Banerjee developed the Federated Learning benchmark used in the advanced numerical experiments. Alp Yurtsever provided continuous guidance, feedback, and comments on the theoretical analysis, manuscript preparation, and experimental work.

Bibliography

[1] Ron Bekkerman, Mikhail Bilenko, and John Langford. *Scaling up machine learning: Parallel and distributed approaches*. Cambridge University Press, 2011.

[2] Jakub Konečnỳ et al. "Federated optimization: Distributed machine learning for on-device intelligence". In: *arXiv preprint arXiv:1610.02527* (2016).

[3] Adrian Nilsson et al. "A performance evaluation of federated learning algorithms". In: *Proceedings of the second workshop on distributed infrastructures for deep learning*. 2018, pp. 1–8.

[4] Yue Zhao et al. "Federated learning with non-iid data". In: *arXiv preprint arXiv:1806.00582* (2018).

[5] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. "Measuring the effects of non-identical data distribution for federated visual classification". In: *arXiv preprint arXiv:1909.06335* (2019).

[6] Alysa Ziying Tan et al. "Towards personalized federated learning". In: *IEEE transactions on neural networks and learning systems* 34.12 (2022), pp. 9587–9603.

[7] Ali Dadras, Sebastian U Stich, and Alp Yurtsever. "Personalized Federated Learning via Low-Rank Matrix Factorization". In: *OPT 2024: Optimization for Machine Learning*.

[8] Sourasekhar Banerjee et al. "Personalized multi-tier federated learning". In: *arXiv preprint arXiv:2407.14251* (2024).

[9] Ali Dadras et al. "Federated frank-wolfe algorithm". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2024, pp. 58–75.

[10] Arkadi Nemirovski et al. "Robust stochastic approximation approach to stochastic programming". In: *SIAM Journal on optimization* 19.4 (2009), pp. 1574–1609.

[11] Saeed Ghadimi and Guanghui Lan. "Stochastic first-and zeroth-order methods for nonconvex stochastic programming". In: *SIAM journal on optimization* 23.4 (2013), pp. 2341–2368.

[12] Lam M Nguyen, Trong Nghia Hoang, and Pin-Yu Chen. *Federated Learning: Theory and Practice*. Elsevier, 2024.

[13] Martin Jaggi. "Revisiting Frank-Wolfe: Projection-free sparse convex optimization". In: *International conference on machine learning*. PMLR. 2013, pp. 427–435.

[14] Simon Lacoste-Julien. "Convergence rate of frank-wolfe for non-convex objectives". In: *arXiv preprint arXiv:1607.00345* (2016).

[15] U.S. Congress. *Health Insurance Portability and Accountability Act (HIPAA)*. `https://www.hhs.gov/hipaa/for-professionals/index.html`. Accessed: January 10, 2025. 1996.

[16] European Commission. *General Data Protection Regulation (GDPR)*. `https://gdpr-info.eu/`. Accessed: January 10, 2025. 2018.

[17] California State Legislature. *California Consumer Privacy Act (CCPA)*. `https://oag.ca.gov/privacy/ccpa`. Accessed: January 10, 2025. 2018.

[18] Viraaji Mothukuri et al. "A survey on security and privacy of federated learning". In: *Future Generation Computer Systems* 115 (2021), pp. 619–640.

[19] Brendan McMahan et al. "Communication-efficient learning of deep networks from decentralized data". In: *Artificial intelligence and statistics*. PMLR. 2017, pp. 1273–1282.

[20] Haibo He and Edwardo A Garcia. "Learning from imbalanced data". In: *IEEE Transactions on knowledge and data engineering* 21.9 (2009), pp. 1263–1284.

[21] Qinbin Li et al. "Federated learning on non-iid data silos: An experimental study". In: *2022 IEEE 38th international conference on data engineering (ICDE)*. IEEE. 2022, pp. 965–978.

[22] Mang Ye et al. "Heterogeneous federated learning: State-of-the-art and research challenges". In: *ACM Computing Surveys* 56.3 (2023), pp. 1–44.

[23] Sai Praneeth Karimireddy et al. "Scaffold: Stochastic controlled averaging for federated learning". In: *International conference on machine learning*. PMLR. 2020, pp. 5132–5143.

[24] Konstantin Mishchenko et al. "Proxskip: Yes! local gradient steps provably lead to communication acceleration! finally!" In: *International Conference on Machine Learning*. PMLR. 2022, pp. 15750–15769.

[25] Ligeng Zhu, Zhijian Liu, and Song Han. "Deep leakage from gradients". In: *Advances in neural information processing systems* 32 (2019).

[26] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. "Model inversion attacks that exploit confidence information and basic countermeasures". In: *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. 2015, pp. 1322–1333.

[27] Reza Shokri et al. "Membership inference attacks against machine learning models". In: *2017 IEEE symposium on security and privacy (SP)*. IEEE. 2017, pp. 3–18.

[28] Wenqi Wei et al. "A framework for evaluating gradient leakage attacks in federated learning". In: *arXiv preprint arXiv:2004.10397* (2020).

[29] Sebastian U Stich. "Local SGD converges fast and communicates little". In: *arXiv preprint arXiv:1805.09767* (2018).

[30] Jianyu Wang et al. "Tackling the objective inconsistency problem in heterogeneous federated optimization". In: *Advances in neural information processing systems* 33 (2020), pp. 7611–7623.

[31] Blake Woodworth et al. "Is local SGD better than minibatch SGD?" In: *International Conference on Machine Learning*. PMLR. 2020, pp. 10334–10343.

[32] Sashank Reddi et al. "Adaptive federated optimization". In: *arXiv preprint arXiv:2003.00295* (2020).

[33] Yae Jee Cho, Jianyu Wang, and Gauri Joshi. "Client selection in federated learning: Convergence analysis and power-of-choice selection strategies". In: *arXiv preprint arXiv:2010.01243* (2020).

[34] Margalit R Glasgow, Honglin Yuan, and Tengyu Ma. "Sharp bounds for federated averaging (local sgd) and continuous perspective". In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2022, pp. 9050–9090.

[35] Grigory Malinovskiy et al. "From local SGD to local fixed-point methods for federated learning". In: *International Conference on Machine Learning*. PMLR. 2020, pp. 6692–6701.

[36] Zachary Charles and Jakub Konečnỳ. "On the outsized importance of learning rates in local update methods". In: *arXiv preprint arXiv:2007.00878* (2020).

[37] Farzin Haddadpour and Mehrdad Mahdavi. "On the convergence of local descent methods in federated learning". In: *arXiv preprint arXiv:1910.14425* (2019).

[38] Xiang Li et al. "On the convergence of fedavg on non-iid data". In: *arXiv preprint arXiv:1907.02189* (2019).

[39] Ahmed Khaled, Konstantin Mishchenko, and Peter Richtárik. "First analysis of local GD on heterogeneous data". In: *arXiv preprint arXiv:1909.04715* (2019).

[40] Ahmed Khaled, Konstantin Mishchenko, and Peter Richtárik. "Tighter theory for local SGD on identical and heterogeneous data". In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2020, pp. 4519–4529.

[41] Blake E Woodworth, Kumar Kshitij Patel, and Nati Srebro. "Minibatch vs local sgd for heterogeneous distributed learning". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 6281–6292.

[42] Fan Zhou and Guojing Cong. "On the convergence properties of a K-step averaging stochastic gradient descent algorithm for nonconvex optimization". In: *arXiv preprint arXiv:1708.01012* (2017).

[43] Tian Li et al. "Federated optimization in heterogeneous networks". In: *Proceedings of Machine learning and systems* 2 (2020), pp. 429–450.

[44] Xinwei Zhang et al. "Fedpd: A federated learning framework with adaptivity to non-iid data". In: *IEEE Transactions on Signal Processing* 69 (2021), pp. 6055–6070.

[45] Durmus Alp Emre Acar et al. "Federated learning based on dynamic regularization". In: *arXiv preprint arXiv:2111.04263* (2021).

[46] Tian Li et al. "Feddane: A federated newton-type method". In: *2019 53rd Asilomar Conference on Signals, Systems, and Computers*. IEEE. 2019, pp. 1227–1231.

[47] Ohad Shamir, Nati Srebro, and Tong Zhang. "Communication-efficient distributed optimization using an approximate newton-type method". In: *International conference on machine learning*. PMLR. 2014, pp. 1000–1008.

[48] Xianfeng Liang et al. "Variance reduced local sgd with lower communication complexity". In: *arXiv preprint arXiv:1912.12844* (2019).

[49] Sai Praneeth Karimireddy et al. "Mime: Mimicking centralized stochastic algorithms in federated learning". In: *arXiv preprint arXiv:2008.03606* (2020).

[50] Kai Yi, Laurent Condat, and Peter Richtárik. "Explicit personalization and local training: Double communication acceleration in federated learning". In: *arXiv preprint arXiv:2305.13170* (2023).

[51] Reese Pathak and Martin J Wainwright. "FedSplit: An algorithmic framework for fast federated optimization". In: *Advances in neural information processing systems* 33 (2020), pp. 7057–7066.

[52] Jianyu Wang et al. "A field guide to federated optimization". In: *arXiv preprint arXiv:2107.06917* (2021).

[53] Shanshan Wu et al. "Motley: Benchmarking heterogeneity and personalization in federated learning". In: *arXiv preprint arXiv:2206.09262* (2022).

[54] Li Daliang and Wang Junpu. "FedMD: Heterogenous federated learning via model distillation". In: *arXiv preprint arXiv:1910.03581* 2 (2019).

[55] Chelsea Finn, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks". In: *International conference on machine learning*. PMLR. 2017, pp. 1126–1135.

[56] Canh T Dinh, Nguyen Tran, and Josh Nguyen. "Personalized federated learning with moreau envelopes". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 21394–21405.

[57] Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. "Adaptive personalized federated learning". In: *arXiv preprint arXiv:2003.13461* (2020).

[58] Elnur Gasanov et al. "FLIX: A simple and communication-efficient alternative to local methods in federated learning". In: *arXiv preprint arXiv:2111.11556* (2021).

[59] Konstantin Mishchenko et al. "Partially Personalized Federated Learning: Breaking the Curse of Data Heterogeneity". In: *arXiv preprint arXiv:2305.18285* (2023).

[60] Manoj Ghuhan Arivazhagan et al. "Federated learning with personalization layers". In: *arXiv preprint arXiv:1912.00818* (2019).

[61] Jaehoon Oh, Sangmook Kim, and Se-Young Yun. "Fedbabu: Towards enhanced representation for federated image classification". In: *arXiv preprint arXiv:2106.06042* (2021).

[62] Liam Collins et al. "Exploiting shared representations for personalized federated learning". In: *International conference on machine learning*. PMLR. 2021, pp. 2089–2099.

[63] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. "Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints". In: *IEEE transactions on neural networks and learning systems* 32.8 (2020), pp. 3710–3722.

[64] Avishek Ghosh et al. "An efficient framework for clustered federated learning". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 19586–19597.

[65] Moming Duan et al. "Fedgroup: Efficient federated learning via decomposed similarity-based clustering". In: *2021 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom)*. IEEE. 2021, pp. 228–237.

[66] Christopher Briggs, Zhong Fan, and Peter Andras. "Federated learning with hierarchical clustering of local updates to improve training on non-IID data". In: *2020 international joint conference on neural networks (IJCNN)*. IEEE. 2020, pp. 1–9.

[67] Jiahao Tan et al. "pFedSim: Similarity-Aware Model Aggregation Towards Personalized Federated Learning". In: *arXiv preprint arXiv:2305.15706* (2023).

[68] Neal Parikh, Stephen Boyd, et al. "Proximal algorithms". In: *Foundations and trends® in Optimization* 1.3 (2014), pp. 127–239.

[69] Teng Zhang and Xing Fan. "Projected Gradient Descent Algorithm for Low-Rank Matrix Estimation". In: *arXiv preprint arXiv:2403.02704* (2024).

[70] Marguerite Frank and Philip Wolfe. "An algorithm for quadratic programming". In: *Naval research logistics quarterly* 3.1-2 (1956), pp. 95–110.

[71] Honglin Yuan, Manzil Zaheer, and Sashank Reddi. "Federated composite optimization". In: *International Conference on Machine Learning*. PMLR. 2021, pp. 12253–12266.

[72] Yurii Nesterov. "Primal-dual subgradient methods for convex problems". In: *Mathematical programming* 120.1 (2009), pp. 221–259.

[73] Yu Nesterov. "Gradient methods for minimizing composite functions". In: *Mathematical programming* 140.1 (2013), pp. 125–161.

[74] Quoc Tran Dinh et al. "FedDR–randomized Douglas-Rachford splitting algorithms for nonconvex federated composite optimization". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 30326–30338.

[75] Yu Nesterov. "Smooth minimization of non-smooth functions". In: *Mathematical programming* 103 (2005), pp. 127–152.