Universitat d'Alacant
Universidad de Alicante

Relaxing Core Assumptions: the Impact of Data, Model and Participation Heterogeneity on Performance, Privacy and Fairness in Federated Learning

Gergely Dániel Németh

Alicante
Curso 2024 / 2025

Universitat d'Alacant
Universidad de Alicante

Relaxing Core Assumptions: the Impact of Data, Model and Participation Heterogeneity on Performance, Privacy and Fairness in Federated Learning

Gergely Dániel Németh

Thesis presented in fulfillment of the requirements
for the degree of Doctor of Philosophy by the

UNIVERSITY OF ALICANTE

With international mention

DOCTOR OF INFORMATICS

Advised by:
Nuria Oliver Ramírez, *ELLIS Alicante*
Novi Quadrianto, *University of Sussex*
Miguel Ángel Lozano Ortega, *University of Alicante*

The research presented in this thesis has been financed by the ELLIS unit Alicante Foundation with funding from the European Commission under the Horizon Europe Programme - Grant Agreement 101120237 - ELIAS, from a nominal grant from the Regional Government of Valencia in Spain (Convenio Singular signed with Generalitat Valenciana, Conselleria de Innovación, Industria, Comercio y Turismo, Dirección General de Innovación) and from a grant by the Banco Sabadell Foundation. In addition, it has been supported by funding received at the University of Sussex in part by the European Research Council under the European Union's Horizon 2020 research and innovation programme Grant Agreement no. 851538 -BayesianGDPR. Views and opinions expressed are those of the author(s) only and do not necessarily reflect those of the European Union or the European Health and Digital Executive Agency (HaDEA). Neither the European Union nor the granting authority can be held responsible for them.

Universitat d'Alacant
Universidad de Alicante

Relaxing Core Assumptions: the Impact of Data, Model and Participation Heterogeneity on Performance, Privacy and Fairness in Federated Learning

Gergely Dániel Németh

Tesis presentada para aspirar al título de doctor por la

UNIVERSIDAD DE ALICANTE

Mención de doctor internacional

DOCTORADO EN INFORMÁTICA

Dirigida por:
Nuria Oliver Ramírez, *ELLIS Alicante*
Novi Quadrianto, *University of Sussex*
Miguel Ángel Lozano Ortega, *University of Alicante*

La investigación presentada en esta tesis ha sido ha sido parcialmente financiada por la Fundación de la Comunitat Valenciana Unidad ELLIS Alicante con fondos del programa Horizon Europe de la comisión europea - Acuerdo de subvención 101120237 - ELIAS, de una subvención nominativa por parte del gobierno regional de Valencia en España (Convenio firmado con la Generalitat Valenciana, Conselleria de Innovación, Industria, Comercio y Turismo, Dirección General de Innovación) y de la fundación Banco Sabadell. Asimismo, por financiación recibida por la Universidad de Sussex como parte del Consejo Europeo de Investigación bajo el paraguas del programa Union's Horizon europeo de 2020 para la investigación e innovación mediante el acuerdo Grant nº 851538 -BayesianGDPR. Las opiniones expresadas son exclusivas del autor(es) y no reflejan necesariamente el punto de vista de la Unión Europea o de la HaDEA. La Unión Europea no otorga autoridad para ello.

This document was proudly typeset with LaTeX.

Contact Details

Gergely Dániel Németh
`neged.ng@gmail.com`

ellis | UNIT ALICANTE

ELLIS Alicante is the first Spanish unit within the **ELLIS** European network for research excellence. It is the only ELLIS unit that has been created as an independent non-profit research foundation, with the spirit of a scientific startup.

Our name, The Institute of Human-Centered AI, defines our mission: We firmly believe in the power of AI as an engine for progress and a key contributor to well-being. However, such a potential is by no means guaranteed and that's why the research of our foundation is so important. Our vision, mission and research have been awarded the 2022 Spanish Social Innovation Award by the Spanish Association of Foundations.

We aim to be a leading research lab on **ethical, responsible and human-centered AI**. We are the only ELLIS unit devoted exclusively to this topic.

At ELLIS Alicante, we address three important research areas:

- **AI to understand us**, by modeling **human behavior** using AI techniques both at the individual and aggregate levels. We focus on developing machine learning-based models of individual and aggregate human behavior. The practical applications are diverse, including the development of algorithms that generate recommendations for users or accurate and fair credit models to promote financial inclusion. At an aggregate level, we aim to model and predict human behavior on a large scale, at a country or region level, which allows addressing global challenges such as pandemics, detecting possible economic crises or responding to natural disasters. Our work during the COVID-19 pandemic is a good example of our work in this area.

- **AI that interacts with us**, via the development of intelligent, interactive systems, with a special focus on the development of smart phones, personal assistants and chatbots.

- **AI that we trust**, by tackling the ethical challenges posed by today's AI systems, such as algorithmic discrimination, violation of privacy, opacity, lack of veracity or subliminal manipulation of human behavior. Current AI algorithms are not perfect and have limitations that are important to identify and address in order to minimize the possible negative consequences of their use. In this area, we also investigate the societal and cultural impact of AI.

Abstract (EN)

Federated Learning (FL) enables decentralized training of machine learning models on distributed data while preserving privacy by design. An FL design consists of clients training models on private data and a central server aggregating a global model based on the consensus among clients. In an ideal scenario, the training data and computing resources are identically and independently distributed (i.i.d.) among clients, therefore, clients can work together in agreement to reach a global optima.

However, in realistic FL settings, heterogeneity arises between clients in terms of both data and resource availability. This research focuses on such scenarios, with a special interest on how the server can adapt the aggregation method from a simple averaging to address the clients' diversity.

The first research direction discusses existing client selection methods and proposes a novel taxonomy of FL methods where the participation of the clients is actively managed by the server to achieve a global objective with respect to the client heterogeneity. This research direction is presented in [NLQO22],

The next chapter focuses on model heterogeneity as an inclusion policy for low-resource clients. It investigates the implications of client resource constraints on privacy given a reduced model complexity in low-resource clients. This work has been presented in [NLQO25].

The final area provides a solution to the data heterogeneity problem with distribution-aware client selection. Applying this solution can mitigate spurious correlations and improve algorithmic fairness in FL. This research line has been described in [NFN+25].

[NLQO22] Németh, G. D., Lozano, M. A., Quadrianto, N., and Oliver, N. (2022). A Snapshot of the Frontiers of Client Selection in Federated Learning. *Transactions on Machine Learning Research.*

[NLQO25] Németh, G. D., Lozano, M. A., Quadrianto, N., and Oliver, N. (2025). Privacy and Accuracy Implications of Model Complexity and Integration in Heterogeneous Federated Learning. *IEEE Access*, 13, 40258-40274.

[NFN+25] Németh, G. D., Fani, E., Ng, Y. J., Caputo, B., Lozano, M. A., Oliver, N., and Quadrianto, N.(2025). FedDiverse: Tackling Data Heterogeneity in Federated Learning with Diversity-Driven Client Selection.

Resumen (ES)

El Aprendizaje Federado (FL) permite el entrenamiento descentralizado de modelos de aprendizaje automático sobre datos distribuidos preservando la privacidad por diseño. Un diseño de FL consiste en que los clientes entrenen modelos sobre datos privados y un servidor central agregue un modelo global basado en el consenso entre los clientes. En un escenario ideal, los datos de entrenamiento y los recursos informáticos se distribuyen de forma idéntica e independiente (iid) entre los clientes, por lo que éstos pueden trabajar juntos de común acuerdo para alcanzar un óptimo global.

Sin embargo, en escenarios realistas de FL, existe heterogeneidad entre los clientes tanto en términos de datos como de disponibilidad de recursos. La investigación presentada en esta tesis se centra en dichos escenarios, con especial interés en cómo el servidor puede adaptar el método de agregación de un simple promedio para hacer frente a la diversidad de los clientes.

La primera línea de investigación presenta los métodos de selección de clientes existentes y propone una nueva taxonomía de métodos de FL donde la participación de los clientes es gestionada activamente por el servidor para lograr un objetivo global con respecto a la heterogeneidad de los clientes. Esta investigación ha sido publicada en [NLQO22].

El siguiente capítulo se centra en la heterogeneidad del modelo como política de inclusión de clientes con recursos computacionales limitados. Investiga las implicaciones de las restricciones de recursos del cliente sobre la privacidad dada una complejidad reducida del modelo en clientes de bajos recursos. Esta investigación ha sido publicada en [NLQO25].

El área final ofrece una solución al problema de la heterogeneidad de los datos con una selección de clientes que tiene en cuenta la distribución de los datos. La aplicación de esta solución puede mitigar las correlaciones espurias y mejorar la equidad algorítmica en FL. Este trabajo ha sido publicado en [NFN+25].

[NLQO22] Németh, G. D., Lozano, M. A., Quadrianto, N., and Oliver, N. (2022). A Snapshot of the Frontiers of Client Selection in Federated Learning. *Transactions on Machine Learning Research.*

[NLQO25] Németh, G. D., Lozano, M. A., Quadrianto, N., and Oliver, N. (2025). Privacy and Accuracy Implications of Model Complexity and Integration in Heterogeneous Federated Learning. *IEEE Access*, 13, 40258-40274.

[NFN+25] Németh, G. D., Fani, E., Ng, Y. J., Caputo, B., Lozano, M. A., Oliver, N., and Quadrianto, N.(2025). FedDiverse: Tackling Data Heterogeneity in Federated Learning with Diversity-Driven Client Selection.

Kivonat (HU)

A Federated Learning (FL) lehetővé teszi a gépi tanulási modellek decentralizált képzését elosztott adatokon, miközben rendszer tervezéséből adódóan őrzi az adatok védelmét. Egy FL rendszer kialakításának lényege, hogy a kliensek (*clients*) modelleket tanítanak a saját privát adataikon, és egy központi szerver (*server*) a kliensek által küldött modellek konszenzusán alapuló globális modelt képez (*aggregation*). Ideális esetben a képzési adatok és a számítási erőforrások azonos és egymástól független (*indentically and independently distributed* – iid) elosztásban vannak a kliensek között, ezért a kliensek egyetértésben dolgozhatnak együtt a globális optimum elérése érdekében.

Azonban egy reális FL együttműködés esetén sokszínűség (*heterogeneity*) áll fent az ügyfelek között mind az adatok, mind az erőforrások rendelkezésre állása tekintetében. Ez a doktori értekezés ilyen forgatókönyvekre összpontosít, különös tekintettel arra, hogy a szerver hogyan tudja az aggregációs módszert az egyszerű átlagolásból az ügyfelek különbségeinek kezelésére adaptálni.

Az első kutatási irány a meglévő kliens kiválasztási módszereket (*client selection*) tárgyalja, és javaslatot tesz a FL módszerek új taxonómiájára, ahol a kliensek részvételét a szerver aktívan kezeli, hogy az ügyfelek heterogenitását figyelembe véve globális célt érjen el. [NLQO22]

A következő fejezet a modellek sokszínűségére (*model heterogeneity*) összpontosít, mint ezköz az alacsony erőforrású kliensek a rendszerbe való integrálására. Megvizsgálja a kliens erőforrás-korlátozásainak az adatvédelemre gyakorolt hatását, azaz, hogy vajon a csökkentett modell komplexitás az alacsony erőforrású kliensek esetében segít-e a privát adatainak védelmében. [NLQO25]

Az utolsó fejezetek az adatok sokszínűségének (*data heterogeneity*) problémájára nyújt megoldást azáltal, hogy a kliens kiválasztás során a szerver gondos mérlegelést végez, hogyan állíthatná párba a különböző adatokkal bíró klienseket. E megoldás alkalmazása enyhítheti a modell által tanult téves összefüggéseket (*spurious correlations*) és javíthatja az algoritmikus igazságosságot (*algorithmic fairness*) az elosztott rendszerben. [NFN+25]

[NLQO22] Németh, G. D., Lozano, M. A., Quadrianto, N., and Oliver, N. (2022). A Snapshot of the Frontiers of Client Selection in Federated Learning. *Transactions on Machine Learning Research.*

[NLQO25] Németh, G. D., Lozano, M. A., Quadrianto, N., and Oliver, N. (2025). Privacy and Accuracy Implications of Model Complexity and Integration in Heterogeneous Federated Learning. *IEEE Access*, 13, 40258-40274.

[NFN+25] Németh, G. D., Fani, E., Ng, Y. J., Caputo, B., Lozano, M. A., Oliver, N., and Quadrianto, N.(2025). FedDiverse: Tackling Data Heterogeneity in Federated Learning with Diversity-Driven Client Selection.

Acknowledgments

First and foremost I want to thank my advisors, Nuria Oliver, Novi Quadrianto and Miguel Ángel Lozano for their support and constant motivation to continue to work when I doubted myself throughout the PhD journey. I thank Nuria for acting as a role model in so many aspects of life. Among many things, she taught me how to dream big again and how to work tirelessly to achieve it. I thank Novi for the constant positivity that I have received during my Ph.D., especially during my research stay at Sussex. I also thank him for being there for me during some of the darkest hours of my life. I thank Miguel Ángel for tirelessly guiding me through the bureaucracy of university life.

I want to thank the first cohort of Ph.D. students at ELLIS Alicante, Piera Riccio, Adrián Arnaiz and Aditya Gulati for taking this journey together with me. We took a big leap of faith joining a fresh research lab and I believe none of us could have succeeded without the others. I thank Adrián for being my unofficial Spanish translator, as well as an inspiration of professional integrity. I thank Aditya for his exceptional ability to express friendship to fit one's personality. I thank Piera for being the strongest ally or enemy whenever a moral, ethical, or political question needed it. I also thank her for brightening my trips in numerous countries, including our adventure of teaching machine learning in Malawi.

I thank everyone at ELLIS Alicante who joined since the beginning. We built a thriving research lab together. Especially, I want to thank Cristina, who was probably not expecting that her job would require visiting the police numerous times.

This Ph.D. journey was not without obstacles. I was probably not expecting at the beginning that I would have to change apartments fourteen times, rebuild compute more than ten times, or visit the police twice. All these experiences gave me more confidence to deal with any obstacle life presents. The same time they also showed me how one research paper can have a direct impact on someone's life [OEK+13].

I also want to thank friends and family for keeping the connections alive, despite all the difficulties an international Ph.D. comes with. I am still learning to value the unconditional love you shower me with. Above all, my grandparents, who are not with us to see the end of my Ph.D. journey but never questioned that I would reach it. I dedicate this work to them.

Contents

Chapter 1

Introduction

Federated learning (FL) is a machine learning approach that aims to address privacy and security concerns existing in centralized machine learning. It consists of a distributed collaborative learning architecture, where a *server* communicates with many *clients* (*e.g.*mobile devices) so that clients keep their potentially sensitive private data locally and only share the processed model weights and metadata information with the server. The core task of the server is to aggregate the model parameters that have been received from the clients to learn an improved global model that is then shared back with the clients. The central server might choose different termination conditions for the training process and perform *model aggregation* using different strategies and optimizers. This learning approach has been promoted as achieving *privacy-by-design* and thus offering a promising solution in privacy-sensitive applications, such as healthcare and banking.

Since its proposal in 2017 by McMahan *et al.* [MMR+17], the field of FL has grown very rapidly. A comprehensive survey from 2021 describes advances and open problems in FL and collects more than 500 related works [KMA+21]. It analyzes the types of federated learning, describes the most crucial challenges, and summarizes the directions taken by the researchers to address these problems. This thesis follows the same notation and definitions as those presented in Kairouz *et al.* [KMA+21].

Types of FL In terms of the scope of a federated learning system, we consider two categories.

1. Horizontal vs. Vertical FL. First, depending on the nature of the data distribution, FL systems are divided into vertical and horizontal federated learning. *Vertical federated learning* consists of clients with access to *different* data about *same* individuals, who are related by means of a unique identifier. The motivation for the collaboration in this setting is to build more accurate models by including complementary information about the individuals while preserving their privacy and avoiding sending data to the central server. In contrast, *horizontal federated learning* refers to a federation where the features are the same in each client. In this case, the motivation for the collaboration is to have access to more data points thanks to the federation while keeping them privately on the clients' side. The central server learns a better performing model than the local models of any of the individual clients and sends it back to the clients so that they benefit from the federation. In this thesis, we focus on *horizontal* federated learning techniques.

2. Cross-silo vs Cross-device FL. Secondly, if we consider the nature of the clients, there

are two distinguishable types of FL architectures: *cross-silo* and *cross-device* [KMA+21]. In cross-silo federated learning, clients are expected to be reliable, available, stateful, and addressable. In contrast, in cross-device federated learning, the clients are separate and diverse individual actors that may not participate in the federation for a variety of reasons, such as a loss of connectivity or excessive energy consumption.

Challenges in FL The field of federated learning includes many important research topics and is increasingly used for different privacy-sensitive applications [LLGL24]. In the following, we summarize the most important challenges in federated learning and some commonly used strategies to address them.

1. Efficiency, where the focus is to train more accurate models faster, with less and lighter communication with the clients, as this is a major bottleneck in FL applications [KMY+16; CSP+21; DLX+24; ZZL+24]. In this category, we also include the challenge of building FL systems that are robust to heterogeneity in the data. This is often referred to as *non-IID* data, which means that data samples are not independent and identically distributed (IID) between clients [MRA+12]. A realistic example where any of these problems can be present is a company training federated learning models on the users' mobiles across the globe. The client data can depend on the demographic preferences of the user, while the online communication (on possibly metered network) and the geographic distances can motivate the use of compressed and robust communication. Strategies to address this challenge can include client selection [CWJ22; NLQO22], partially reduced model complexity for metered clients [DDT21; NLQO25], and personalization [LHBS21].

2. Protection against malicious actors. Compared to centralized machine learning, where the training is performed in a central system and the administrator can control who has access to it, federated learning clients can potentially behave maliciously, and the online communication with the server can expose the overall FL system to additional third-party curious listeners. We distinguish *privacy attacks*, where passive listeners want to extract sensitive information from the observed communication [TLC+20; NLQO25], and *adversarial attacks* where actors manipulate the FL training to achieve their objective, harming the original goal of the training [BCMC19; YDK+24]. Defenses can include restrictions on communication, such as secure multiparty computation [LZS+24] or differential privacy [TLC+20], robust optimizers [YDK+24], and encouraging greater participation of trusted clients through incentive mechanisms [STW19].

3. Fairness, which includes *fair performance*, generalized from traditional algorithmic fairness concepts applied to a distributed scenario, and *fair collaboration*, addressing heterogeneity in the contribution of clients and their motivation through rewards [SYL23; HYS+24; SACA24]. Strategies to improve fairness can include personalization [LHBS21], constrained optimization with fairness evaluation on the server side [KPDG22], and incentive mechanisms [NLQO22].

4. System challenges, including problems that arise from unreliable communication, disconnecting devices, or devices with different computational capabilities [BEG+19]. Solutions can include clustering clients with similar system capabilities [ZZWC20], reducing computational overhead for low resource clients [DDT21; NLQO25], and failure-robust aggregation [MÖB22].

Table 1 summarizes these challenges and strategies in federated learning. We highlight in blue the challenges and strategies addressed in this thesis. Our aim is to relax the core

Table 1. Challenges and common strategies in federated learning. In blue the challenges and strategies addressed in this thesis.

Challenges in FL	**Strategies to address the problems**
• Efficiency – Communication – Compression – Data heterogeneity (chapter 5) • Privacy – Differential Privacy – Curious attacks ∗ Membership inference (chapter 4) • Adversarial attacks – Model poisoning – Data poisoning • Fairness – Fair performance ∗ Group fairness (chapter 6) – Fair collaboration • System challenges – Low resource clients (chapter 4) – Device availability – Device failures • Application-specific challenges	• Modified objectives – Personalization – Clustered FL • Expanded server responsibilities – Advanced optimizers – Evaluation with server data – Shared generators • Client Participation (chapter 3) – Client selection (chapters 5 and 6) – Client weighting – Incentive mechanisms • Multi-agent games – Multi-armed bandit – Reinforcement learning • Reduced communication & computation – Homomorphic Encryption – Secure Multi-Party Computation – Differential Privacy – Model heterogeneity (chapter 4) – Knowledge distillation

assumptions of horizontal federated learning with respect to heterogeneity from three perspectives: we propose a taxonomy of client selection to allow for heterogeneity in client participation; we design methods that leverage model heterogeneity to enhance privacy; and we present a novel method to mitigate the negative effects of data heterogeneity in the clients.

Structure of the thesis In chapter 3, we focus on *client selection*. Dealing with a large number of clients was the original motivation to apply client selection in federated learning. The first FL methods [MMR+17] used uniform random selection to limit the number of clients the server has to communicate with in each round. Later works have shown that client selection methods can keep the performance of the overall model while improving the convergence rate of FL training [NY19], reducing the number of required training rounds [GMB+19], or improving fairness in the case of unbalanced data [LSBS20]. We propose a taxonomy of client selection that categorizes the methods based on characteristics such as the aim of the method, the capabilities of the clients, and the required communication. One such example is *model heterogeneity*, originally proposed to allow clients with lower computational capabilities to participate in federated learning by selecting them to train smaller neural networks [DDT21]. The core contributions of this chapter are presented in [NLQO22].

Next, in chapter 4, we design a group of methods based on *model heterogeneity* to address the question of *privacy*. FL consists of a distributed machine learning approach that enables training models without the need to transfer the raw data from different devices or locations (clients) to a central server. It is considered a privacy-preserving solution by design, as raw data never leaves clients and only the model parameters are shared with the central server , which is of great importance in many practical scenarios, such as healthcare [XGS+21; LWC+22] and finance [LTJZ20; BP20], or intelligent smartphone interfaces [APA+22]. However, recent work has shown that sensitive information about original data can be inferred by analyzing the model parameters that are shared in the communication rounds [FJR15; CLE+19]. One reason for this is that complex models overfit the training data [YGFJ18]. Thus, we investigate whether *model heterogeneity* can help mitigate the loss of privacy of participating clients by allowing clients with fewer data to train smaller models. This chapter is based on the content of the following publication [NLQO25].

In chapter 5, we address the challenge of *data heterogeneity* in the clients. In real-world FL scenarios, client data is often shaped by local factors such as different user behaviors [TYCY22], context-specific data collection environments [FMO20; YAE+18], and socioeconomic or cultural biases [BCM+18], resulting in *statistical data heterogeneity*, where data between different clients is not IID and is unbalanced. Statistical data heterogeneity hampers the generalization capabilities of the server's model across clients, slowing convergence, and reducing performance [LHY+20; CCC22]. In chapter 5, we address this issue by proposing `FedDiverse`, a novel FL *client selection* method that takes advantage of the heterogeneity of client data to reduce the spurious correlation learned by the federated model. The core concept is that if many clients own similar data, selecting clients based on their differences can reduce the unwanted correlation in the majority. This chapter is based on this publication [NFN+25].

Finally, in chapter 6, we further investigate the capabilities of FedDiverse to achieve group *fairness in FL* [SYL23]. Specifically, we adapt the *client selection* capabilities of FedDiverse to improve *group fairness* in the clients derived from bias in the training data. The rationale is to select diverse clients to increase the participation of clients with valuable data from underrepresented groups.

Chapter 2

Background

In this chapter, we present the mathematical notation used in this thesis, particularly concerning *federated learning* (we focus on horizontal federated learning in all chapters), *client selection*, and *membership inference attacks*. For chapters derived from research articles, we updated the equations to maintain consistent notation throughout the chapters [NLQO22; NLQO25; NFN+25]. We follow FL practices as well as the notation used in the "Deep Learning" book by Ian Goodfellow, Yoshua Bengio, and Aaron Courville [GBC16]. Table 2 summarizes the notation used in the thesis.

Additionally, figure 1 illustrates the challenges and strategies addressed by the thesis.

Table 2. Summary of notation in this thesis

$\mathbb{D}, n := \lvert\mathbb{D}\rvert$	dataset
$(\boldsymbol{x}, y) \in \mathbb{D}$	data sample
$\boldsymbol{x} \in \mathcal{X}, y \in \mathcal{Y}$	input, output
$\mathcal{X}, \mathcal{Y}, \mathcal{A}$	input, output, attribute space
$M(f, \boldsymbol{\theta})$	model
$f : \mathcal{X} \to \mathcal{Y}$	predictor function
$\boldsymbol{\theta} \in \Omega$	model parameters
$\boldsymbol{A}^{N \times M, l} \in \boldsymbol{\theta}$	weight matrix at layer l
$k \in \mathbb{K}, K := \lvert\mathbb{K}\rvert$	client in client set
$\mathcal{S}$	server
$t = 1..T$	traning rounds
$\mathbb{S}^t \subseteq \mathbb{K}$	selected clients for round t
$\boldsymbol{\theta}_k^t$	model parameters for the k client in round t
$\mathbb{D}_k$	dataset of client k
$\Pr[\cdot]$	probability
$\ell, \mathcal{L}$	loss, cost function
$\mathscr{A}$	3rd party attacker

2.1 Federated learning

Federated learning is a cooperation of clients, where each client $k \in \mathbb{K}, |\mathbb{K}| = K$ has access to a dataset $\mathbb{D}_k$ that is considered to be private to the client. The clients work together under

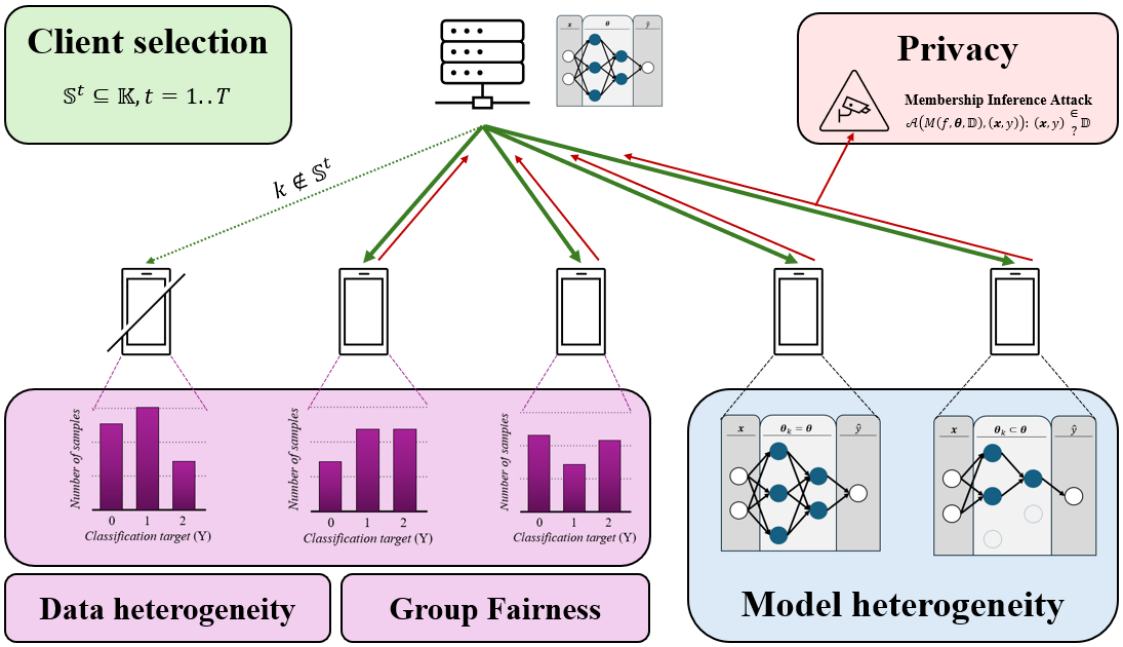


Figure 1. A summary of the research topics of this thesis: In a federated learning architecture, the central server can use **client selection** to tackle **data heterogeneity**, to reduce unwanted **spurious correlations** or to improve **algorithmic fairness**. Furthermore, the server allows the clients to use **model heterogeneity**: clients can choose to learn models of reduced complexity to improve their **privacy** against **membership inference attacks**.

a central server $\mathcal{S}$ to train a global model $f : \mathcal{X} \longrightarrow \mathcal{Y}$ with model parameters $\boldsymbol{\theta} \in \Omega$.

In *vertical FL*, the global input space is a combination of the client inputs: $\bigcup_{k\in\mathbb{K}} \mathcal{X}_k = \mathcal{X}$ with some id shared between clients $\mathcal{X}_{id} \subseteq \bigcap_{k\in\mathbb{K}} \mathcal{X}_k$. In *horizontal FL*, the clients share the same input space $\mathcal{X}_k = \mathcal{X}$. In this work, we focus on horizontal FL.

In horizontal FL, where the clients share the same input-output space, we consider $\mathbb{D} = \bigcup_{i=1}^{K} \mathbb{D}_i$ as the total dataset and $|\mathbb{D}|$ as the total number of data samples in all the clients, $(\boldsymbol{x}, y) \in \mathbb{D}_k$ as an input-output pair of training samples in the dataset of client k and ℓ is the loss function. Then, the central server aims to minimize the global cost function $\mathcal{L}(\boldsymbol{\theta})$ given by equation (1).

$$\min_{\boldsymbol{\theta}\in\Omega} \mathcal{L}(\boldsymbol{\theta}) = \frac{1}{|\mathbb{D}|} \sum_{k\in\mathbb{K}} \sum_{(\boldsymbol{x},y)\in\mathbb{D}_k} \ell(y, f(\boldsymbol{x}, \boldsymbol{\theta})) \tag{1}$$

However, the central server does not have access to the clients' private data. Thus, it broadcasts the global model parameters $\boldsymbol{\theta}$ to all clients, which are used by each client $k \in \mathbb{K}$ as the starting point to compute the local parameters $\boldsymbol{\theta}_k$ that minimize their local cost function $\mathcal{L}_k(\boldsymbol{\theta}_k, \mathbb{D}_k)$ described in equation (2).

$$\min_{\boldsymbol{\theta}_k\in\Omega} \mathcal{L}_k(\boldsymbol{\theta}_k, \mathbb{D}_k) = \frac{1}{|\mathbb{D}_k|} \sum_{(\boldsymbol{x},y)\in\mathbb{D}_k} \ell(y, f(\boldsymbol{x}, \boldsymbol{\theta}_k)) \tag{2}$$

Once the server has collected the local parameters $(\boldsymbol{\theta}_k)$, it aggregates them into the next iteration of the global parameter $\boldsymbol{\theta}$ to minimize the global cost function $\mathcal{L}(\boldsymbol{\theta})$. Each iteration of the global parameter update is called a training round and the parameters in the round

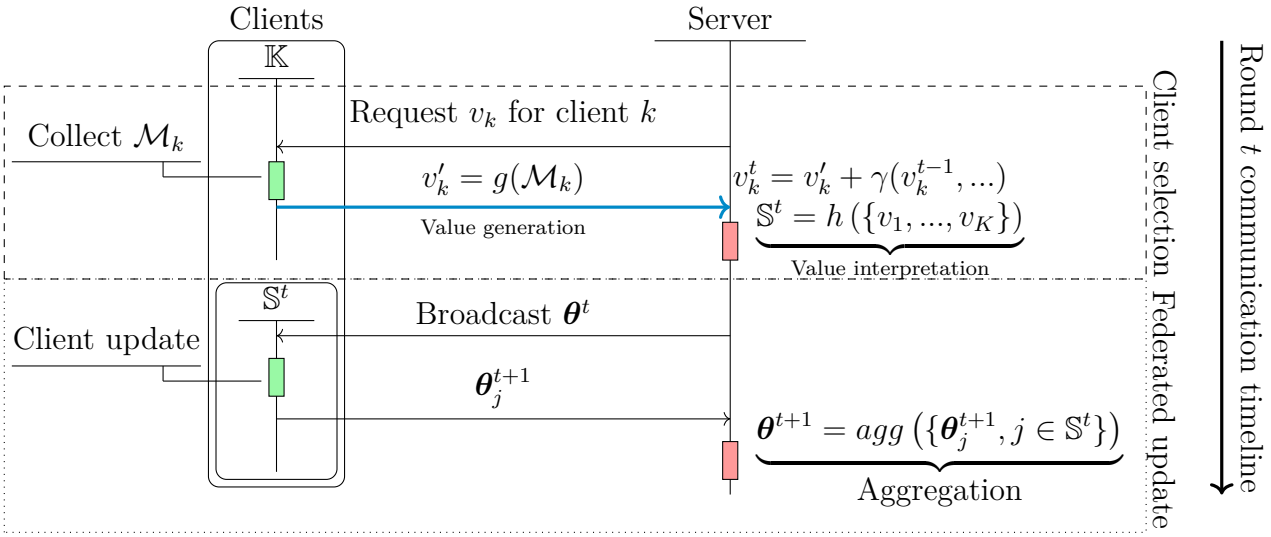


Figure 2. Diagram of a general federated learning training round with client selection. The server communicates with the client set $\mathbb{K}$ to select the participating client set $\mathbb{S}$. Only this subset of clients trains on their local data in a given round t and reports their parameters back to the server. Not all steps are required for every algorithm

t are denoted with $\boldsymbol{\theta}^t$. A naive aggregation function is to weight each client's parameter proportionate to their data sample size, given by equation (3). The federated training proceeds until reaching convergence of the global cost function $\mathcal{L}$.

$$\boldsymbol{\theta}^{t+1} = \sum_{k \in \mathbb{K}} \frac{|\mathbb{D}_k|}{|\mathbb{D}|} \boldsymbol{\theta}_k^t \tag{3}$$

The communication with each client after each local model update may generate significant overhead, particularly when the number of clients is large. To address this problem, McMahan *et al.* [MMR+17] introduced the FedAvg algorithm where the clients communicate with the server only after e local epochs. From the server's point of view, the global model parameters are updated with the clients' data in training rounds $t = 1, ..., T$. From the clients' perspective, every training round t consists of $\tau = 1, ..., e$ local epochs. Thus, the clients train a total number of eT epochs.

2.2 Client selection (participation heterogeneity)

In chapters 3 and 5, we discuss *client selection*, a FL technique where in round t instead of communicating with every client in $\mathbb{K}$, the server chooses a subset of clients ($\mathbb{S}^t$),

$$\mathbb{S}^t \subseteq \mathbb{K} \tag{4}$$

to perform the local training and update the global model based on their response.

FedAvg McMahan *et al.* [MMR+17] uses a random selection $\mathbb{S}^t$ of K' participating clients in every training round t, $\mathbb{S}^t \subseteq \mathbb{K}$ with $|\mathbb{S}^t| = K'$, such that a client k is selected with

probability $p(k \in \mathbb{S}^t) = \frac{K'}{K}$. The client update and aggregation steps are given by $\boldsymbol{\theta}_j^{t+1} = ClientUpdate(\boldsymbol{\theta}^t, \mathbb{D}_j)$ and $\boldsymbol{\theta}^{t+1} = agg(\{\boldsymbol{\theta}_j^{t+1}, j \in \mathbb{S}^t\})$, respectively.

After the seminal work of FedAvg, subsequent research works have proposed a diversity of client selection methods, most of which follow the flow illustrated in figure 2. First, the server requests a value v_k from each client, which the client computes based on its local information $\mathcal{M}_k$. Once the server has received the v_k from all the clients, it selects a subset $\mathbb{S}^t$ of the clients according to function $h(v_1, ..., v_K)$. Stateful selection methods also use previous information about the clients, calculating v_k from the client feedback v'_k and a function (γ) of previous values of the client. After this, the server broadcasts its global parameters $\boldsymbol{\theta}^t$ only to the selected clients $\mathbb{S}^t$, which update their local models and send back their updated local parameters $\boldsymbol{\theta}_k^{t+1}$. With this information, the server computes the updated global parameter $\boldsymbol{\theta}^{t+1}$.

2.3 Model heterogeneity

The previously described method follows the standard horizontal FL approach: the clients and the server share the model architecture while they keep private data locally. In chapter 4, we investigate a more generalized case, *model heterogeneity*, where some clients have a model with reduced complexity but a similar model architecture. On an abstract level, we illustrate this with

$$f_k = f \text{ and } \boldsymbol{\theta}_k \subset \boldsymbol{\theta}, \tag{5}$$

signaling that the model architecture of the client k shares the same structure as the global model, while its parameters are a subset of the server's.

We assume a heterogeneous FL architecture in a computer vision task, where both the server and clients' models are Convolutional Neural Networks (CNNs) with a different number of channels in each layer, but the same number of layers.

Let $\boldsymbol{\theta}$ denote the model parameters of the server's CNN, composed of L layers represented by a weight matrix $\boldsymbol{A}^{N \times M,l} \in \boldsymbol{\theta}$ at each layer l. In such a setting, model reduction $\boldsymbol{\theta}_k \subset \boldsymbol{\theta}$ in client k is achieved by limiting the size of each layer l in the client's CNN according to the following principle: a layer in the server represented by weight matrix $\boldsymbol{A}^{N \times M,l} \in \boldsymbol{\theta}$ is reduced to size $N_k \times M_k$, where $N_k < N$ and $M_k < M$ such that every cell $a_k^{(i_k,j_k),l}$ in the reduced matrix $\boldsymbol{A}_k^{N_k \times M_k,l}$ corresponds to a cell $a^{(i,j),l}$ in the server's matrix $\boldsymbol{A}^{N \times M,l}$:

$$\forall i_k, j_k, l : a_k^{(i_k,j_k),l} \in \boldsymbol{A}_k^l, \exists i, j : a^{(i,j),l} \in \boldsymbol{A}, a_k^{(i_k,j_k),l} = a^{(i,j),l} \tag{6}$$

In this scenario, there are two broad sets of methods proposed in the literature to perform client model integration in the server, as reflected in table 3. The first group of methods, shown in table 3(a), corresponds to algorithms that *dynamically* select the size of the clients' models but where all the clients are able to hold models of the same size as the server's model. Thus, these methods define an $Ms(\cdot)$ function that determines the $N_k^l \times M_k^l$ dimensions of the client's k model to be used of the weight matrix $\boldsymbol{A}_k^l$ in layer l of their model $\boldsymbol{\theta}_k$. Popular approaches in this category include FLADO [LGZX23] and FJORD [HLA+21]. Note that they are not applicable to settings where clients have data and computation constraints, as it is our case. Hence, they are out of the scope of this thesis.

The second group, depicted in table 3(b) and illustrated in figure 8, corresponds to methods where the clients have *fixed-size* models that are typically smaller than the server's model. As

Table 3. Two broad groups of FL methods with model size heterogeneity. **a)** FL methods with dynamic selection of the clients' model size. All clients are assumed to hold a model of the same complexity as the server's model. These methods are not applicable to settings where clients have data and computation constraints, as it is our case. Thus, they are beyond the scope of this paper. **b)** FL methods where the clients have a fixed model size, which can be smaller than the server's model size. In this case, the clients apply different strategies to select the portions from the server's model to be used in their training. The blue font corresponds to the newly proposed methods in this thesis.

(a) FL methods with dynamic client model size selection.

		Dynamic client size selection methods	
		Random	**Gradient**
Update	Each round		FLADO [LGZX23]
	Each batch	FJORD [HLA+21]	

(b) FL methods where clients have a fixed model size.

		Selection strategy	
		Resampled (S)	**Fixed (F)**
Coverage	One group (O)	OSM, OSR	HETEROFL [DDT21], OFR
	Several groups (G)	GSR	GFM, GFR
	Unique (U)	FDROPOUT [CKMT18]	UFR

we are considering CNNs, we refer to this family of methods as *channel selection* methods. The weights of each layer l in a 2D CNN are defined by an $(N, M, H, W)^l$ dimensional tensor, where M and N are the input and output channels of the layer and H and W are the height and width of the kernel, respectively. Thus, $A^{N\times M,l}$ denotes the weight matrix of each linear layer l in the server's model, where M and N are the input and output data dimensions [ZTII88], and $a^{(i,j),l}$ represents the kernel weights of the (i, j) position. In this case, $Ch(\cdot) : \boldsymbol{A}^{N\times M,l} \rightarrow \boldsymbol{A}_k^{N_k\times M_k,l}$ determines the mapping between the cells of the server's weight matrix $\boldsymbol{A}^l$ and the client's k smaller matrix $\boldsymbol{A}_k^l$ for each layer l. Without a loss of generalization, we assume that the channels are sorted.

2.4 Data heterogeneity

In chapter 5, we focus on data heterogeneity in machine learning, specifically, *spurious correlations*, *class imbalance*, and *attribute imbalance*.

Let $f : \mathcal{X} \rightarrow \mathcal{Y}$ be a predictor function parameterized by $\boldsymbol{\theta} \in \Omega$, where $\mathcal{X}$ is the feature space, $\mathcal{Y}$ is the output space, and Ω is the parameter space. We assume that the feature space consists of two subspaces: $\mathcal{X} \subseteq \mathcal{X}_y \times \mathcal{X}_a$, where $\mathcal{X}_y$ and $\mathcal{X}_a$ represent the *task-intrinsic* and the *attribute* feature spaces, respectively. The class label $y \in \mathcal{Y}$ of a sample $\boldsymbol{x} := (\boldsymbol{x}_y, \boldsymbol{x}_a)$ is determined by the discriminative feature $\boldsymbol{x}_y$, whereas the attribute label $a \in \mathcal{A}$ is determined by the attribute feature $\boldsymbol{x}_a$, where $\mathcal{A}$ is the space of attributes. The training dataset $\mathbb{D}$ consists of n feature-target sample pairs, $\mathbb{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, where each sample is identically

and independently drawn from the training distribution $\Pr_{\text{tr}}$.

Statistical data heterogeneity emerges when there is a subpopulation shift, *i.e.*, when the representation of subpopulations differs between the training $\Pr_{tr}$ and the test $\Pr_{te}$ distributions. Here, subpopulations are defined by the target labels and the attributes, $\mathcal{Y} \times \mathcal{A}$. We consider three types of statistical data heterogeneity:

2.4.1 Class imbalance (CI)

The distribution of the target labels y is different between the training and test distributions, such that certain classes are more prevalent in the training than in the test sets, *i.e.*: $\Pr_{\text{tr}}(Y = y) \gg \Pr_{\text{tr}}(Y = y')$ for some $y, y' \in \mathcal{Y}$ where $y \neq y'$. CI can yield a biased classifier that performs poorly in samples from the minority class.

2.4.2 Attribute imbalance (AI)

The probability of occurrence of a certain attribute a' in the training set is much smaller than that of other attributes $a \in \mathcal{A}$ and this disparity in prevalence does not hold in the test distribution, *i.e.*$\Pr_{\text{tr}}(A = a) \gg \Pr_{\text{tr}}(A = a')$. AI can yield a biased classifier towards the majority attribute a.

2.4.3 Spurious correlation (SC)

There is a statistical dependency between the class Y and the attribute A in the training distribution, which does not exist in the test distribution, *i.e.*, $\Pr_{\text{tr}}(Y = y \mid A = a) \gg \Pr_{\text{tr}}(Y = y) \gg \Pr_{\text{tr}}(Y = y \mid A = a')$, for some $y \in \mathcal{Y}$ and $a, a' \in \mathcal{A}$. This spurious dependency can cause a classifier to perform well on samples where the spurious relationship holds (*e.g.*, $(Y = y, A = a)$), but to perform worse where the relationship does not hold (*e.g.*, $(Y = y, A = a')$).

2.5 Privacy

In chapter 4, we measure the privacy of FL models. There are two approaches to do this: using mathematical approximations to measure *differencial privacy* [Dwo06], or measuring privacy against targeted attacks. Adversary attacks can be categorized based on objectives, such as copying the behavior of the target model or reconstructing information from the training data [LDS+21]. We use *membership inference attacks* (MIAs), where a curious attacker's target is to determine whether an input-output pair was part of the observed model's training data

$$\mathscr{A}(M(f, \boldsymbol{\theta}, \mathbb{D}), (\boldsymbol{x}, y)) : (\boldsymbol{x}, y) \underset{?}{\in} \mathbb{D}. \tag{7}$$

Furthermore, we focus on *black-box* attacks, *i.e.* attacks where the attacker has no direct access to the model's parameters $\boldsymbol{\theta}_g$ and architecture f, but it can query the model with data instances to get the model prediction $\hat{y}$. The attacker's purpose is to build an attacker model $\mathscr{A}$ that predicts, for data instance $(\boldsymbol{x}, y)$, if it was part of the training data $\mathbb{D}_k$ of model $M(f, \boldsymbol{\theta}_k, \mathbb{D}_k)$, for client $c \in \mathbb{K}$. Finally, we consider *passive* attacks where the attacker observes the behavior of a system without altering it.

Formally, the perfect attacker's model $\mathscr{A}$ is given by:

$$\mathscr{A}(f, \boldsymbol{\theta}_k, (\boldsymbol{x}, y)) = \begin{cases} 1, & \text{if } (\boldsymbol{x}, y) \in \mathbb{D}_k, M(f, \boldsymbol{\theta}_k, \mathbb{D}_k) \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

We study the performance of three different passive, black-box MIAs, which are summarized below and are described in more detail in section 4.5.

- YEOM **attack**, where the attacker chooses a global threshold ν, and selects every data instance with a loss lower than ν as a member of the training dataset in each client [YGFJ18].
- LIRA **attack,** where the attacker has access to an auxiliary dataset $\mathbb{D}_a$ and trains shadow models $M_{sw}(f, \mathbb{D}_{sw})$ on random subsets of this dataset $\mathbb{D}_{sw} \subset \mathbb{D}_a$. The data instance is predicted to be a member of the client's training set if the target model's confidence score fits into the sample's confidence score distribution in the shadow models [CCN+22].
- TMIA **attack,** which leverages knowledge distillation to collect loss trajectories to identify member and non-member instances [LZBZ22]. This attack builds on the idea that the snapshots of the loss after each training epoch (loss trajectory) can separate the member from non-member instances better than only using the final model's loss.

2.6 Fairness

Algorithmic fairness is concerned with the ethical and social implications of automated decision-making systems. As machine learning systems are increasingly deployed in sensitive domains such as healthcare, finance, and law enforcement, ensuring their fairness becomes essential to prevent preserving societal biases.

While *individual fairness* defines fairness as treating similar individuals in similar ways, the concept of *group fairness* defines groups of the population, and requires equal treatment between them [BHN23]. Groups can be defined based on protected attributes, such as *race, gender*, or *age*. These two fairness notions often come into conflict, reflecting the inherent trade-offs and tensions in quantifying fairness.

To measure the group fairness in machine learning, one has to take into account the societal environment where a trained model is applied. Given this, the literature proposed several fairness metrics [VR18]. In section 6.2.3, we review several fair FL algorithms to determine the most common fairness metrics in FL. Based on our review, these are *equal opportunity*

$$\Pr(\hat{Y} = 1 | Y = 1, A = 0) = \Pr(\hat{Y} = 1 | Y = 1, A = a), \forall a \in \mathcal{A}$$

and *demographic parity*

$$\Pr(\hat{Y} = 1 | A = 0) = \Pr(\hat{Y} = 1 | A = a), \forall a \in \mathcal{A}.$$

We use these metrics to measure the fairness of federated learning algorithms in chapter 6.

Chapter 3

Client selection: a taxonomy

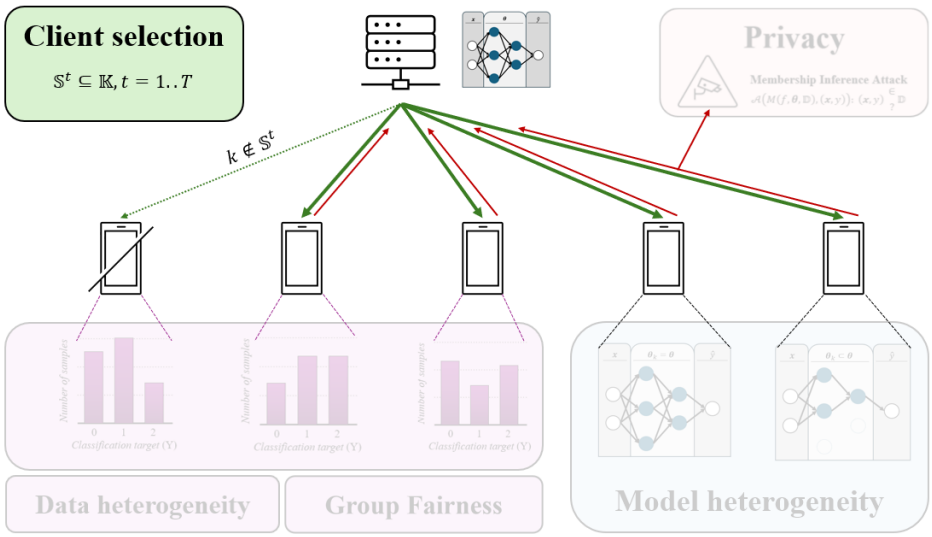


Figure 3. Content covered in his chapter in the scope of the thesis.

This chapter incorporates our research contributions presented in the paper *A Snapshot of the Frontiers of Client Selection in Federated Learning* published in Transactions on Machine Learning Research in December 2022 [NLQO22]. Modifications are applied to improve the flow of the thesis.

3.1 Introduction

In chapter 1, we described the concept of federated learning, where clients work together under a central server to train machine learning models with privacy in mind. In vanilla federated learning, all clients behave the same, they share the same goal as the server – to build a globally robust machine learning model. They are always available and train the FL model on their private data when requested. However, such limitations are often unrealistic. One challenge of FL was addresses in the very first FL paper [MMR+17]: to reduce the communication overhead of scaling FL to a large number of clients, they proposed to select a random subset of clients to train in every training round.

When a federated learning scenario involves few clients, it is feasible to incorporate their parameters in every training round. However, as the number of clients increases, so does the communication overhead, such that considering the data from all the clients becomes a challenge. At the same time, when the number of clients is large, some of the clients might have access to redundant, noisy or less valuable data than other clients. Therefore, *client selection* methods are introduced to reduce the number of working clients in each training round. Recent works have shown that client selection methods are able to keep the performance of the overall model while improving the convergence rate of the FL training [NY19], reducing the number of required training rounds [GMB+19], or enhancing fairness in case of imbalanced data [LSBS20].

While implementing sophisticated client selection methods may help to achieve these goals, they require the server to have information about the clients, such as training time [NY19] or communication stability [ZFH21]. Thus, the use of client selection methods might have privacy implications, representing a trade-off between potentially losing privacy and achieving good performance while keeping the overhead low, improving the overall utility [DLS21; WKNL20] or the fairness [MSS19; LSBS20] of the system.

In recent years, different client selection methods have been proposed in the literature [CGSY18; MSS19; ZZWC20] with a wide range of objectives, requirements, and experimental settings. Such wealth of proposals in such a short timeframe makes it hard for researchers to properly compare results and evaluate novel algorithms. Furthermore, it limits the ability of practitioners to apply the most suitable of the existing FL techniques to a specific real-world problem, because it is not clear what the state-of-the-art is in the scenario required by their application. The purpose of the research described in this chapter is to fill this gap by providing a comprehensive overview of the most significant approaches proposed to date for client selection in FL.

Our contributions are two-fold: First, we present a taxonomy of FL client selection methods that enables us to categorize existing client selection techniques and propose it as a framework to report future work in the field. Second, we identify missing gaps in existing research and outline potential lines of future work in this emergent area.

In section 2.2 we summarized the problem formulation and the mathematical notation used in the client selection chapter of this document. The structure of the rest of the chapter is as follows: Section 3.2 describes the proposed taxonomy and identifies future research directions. Commonly used benchmark datasets are presented in section 3.3, followed by our conclusions.

3.2 Proposed taxonomy of client selection methods

Our proposed taxonomy, depicted in figure 4, aims to ease the study and comparison of client selection methods. It contains six dimensions that characterize each client selection method. The top part of the figure displays the three dimensions (marked in red) that concern the server in the Federated Learning framework whereas the bottom three dimensions concern the clients (marked in green) or both the clients and the server (marked in blue). In the following, we describe in detail each of these dimensions and present the most relevant works in the client selection literature in FL according to the proposed taxonomy.

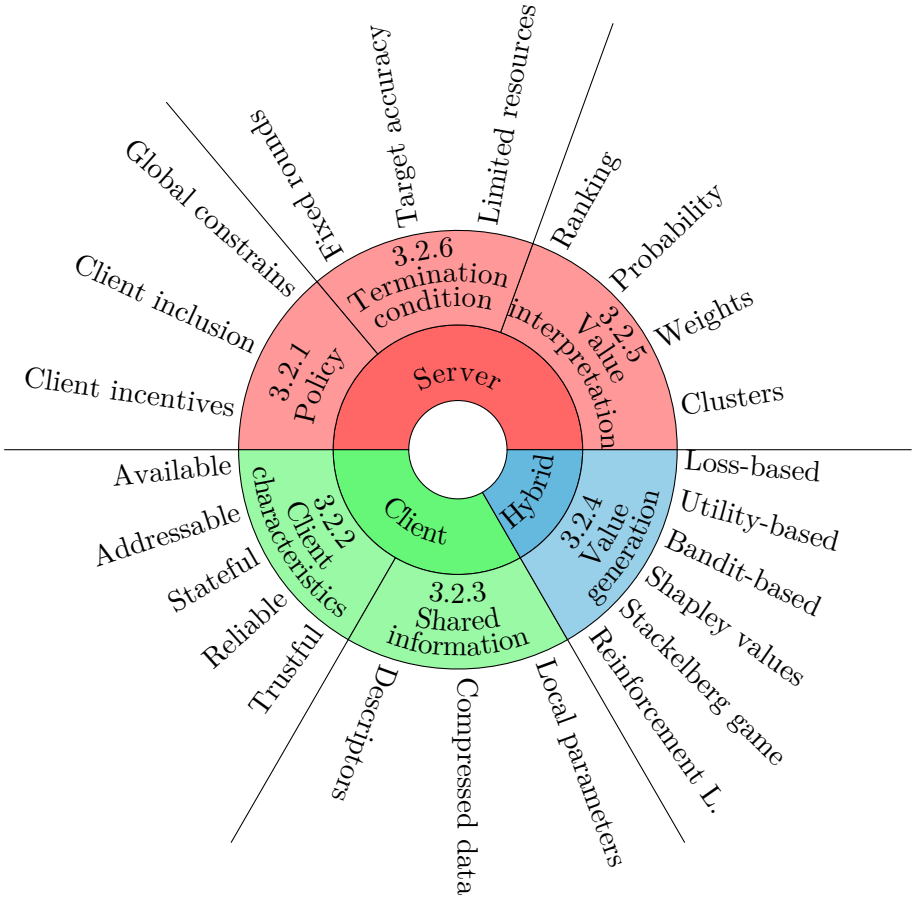


Figure 4. Proposed taxonomy of client selection methods in Federated Learning. The taxonomy has six dimensions: Policy, Termination Condition, Value Interpretation, Client Characteristics, Shared Information, and Value Generation. The top three dimensions (colored in red) capture properties on the server side, the two dimensions colored in green are on the client side, and the sixth dimension, colored in blue, corresponds to a hybrid client-server property.

3.2.1 Client selection policies

The main purpose of client selection in Federated Learning is to automatically determine the number of working clients to improve the training process. A broad range of policies has been proposed in the literature, including a variety of global constraints (e.g. improving the efficiency in the learning, limiting the amount of time or computation needed, or ensuring fairness), pre-defined client inclusion criteria, and client incentives.

We provide below a summary of the most prominent works according to their policies for client selection. The third column in table 4 provides an overview of the client selection policies of the most representative client selection methods in FL.

Global constraints

The server might define different types of global constraints to drive the client selection process.

Table 4. Selected works from the literature and their characteristics according to the proposed taxonomy. Appendix A.1 shows the detailed descriptions of the methods. Client characteristics: availability (V), addressability (D), statefulness (S), reliablity (R), and trustworthiness (T). Exp indicates an exploration term with additional randomly selected clients and ES indicates an early stopping mechanism.

Algorithm	**Policy**	**Client** characteristics	**Message**	**Value generation**	**Value** interpretation	**Termination condition**
EqRep$_{\text{baseline}}$	Selection fairness	$V\overline{DS}RT$	$\boldsymbol{\theta}$	-	Probability	-
EqW$_{\text{baseline}}$		$VD\overline{S}RT$	$\boldsymbol{\theta}, \lvert\mathbb{D}\rvert$	-	Probability	-
AFL$_{\text{G}}$	Efficiency	$\overline{V}DSRT$	$\boldsymbol{\theta}, \mathcal{L}$	Loss-based	Probability and Exp	Fixed rounds
pow-d		$\overline{V}D\overline{S}RT$	$\boldsymbol{\theta}, \lvert\mathbb{D}\rvert, \mathcal{L}$	Loss-based	Ranking	Reach accuracy
S-FedAvg		$VDSRT$	$\boldsymbol{\theta}$	Shapley values Validation set	Probability	Fixed rounds
k-FED		$VDSRT$	$\boldsymbol{\theta}, \mathbb{G}$	Clustering	Clusters	Fixed rounds
LAG		$VDSRT$	$\boldsymbol{\theta}, L_f$	Loss-based	Threshold	Fixed rounds, ES
FL-CIR		$VDSRT$	$\boldsymbol{\theta}, \mathcal{L}$	Bandit-based	Ranking	Fixed rounds
FAVOR		$VDSRT$	$\boldsymbol{\theta}$	Reinforcement Learning Validation set	Ranking	Reach accuracy
FedCS	Global constrains: Limit round time	$\overline{V}D\overline{S}RT$	$\boldsymbol{\theta}, \mathcal{T}$	Time-based	Ranking	Fixed rounds, Reach accuracy
AFL$_{\text{M}}$	Global constrains: Good-intent fairness	$VDSRT$	$\boldsymbol{\theta}, \mathcal{L}$	Loss-based	Probability	Fixed rounds
q-FFL	Global constrains: Uniform accuracy	$\overline{VDSR}\overline{T}$	$\boldsymbol{\theta}, L_f$	Loss-based	Weights	Fixed rounds
Oort	Global constrains: Limit round time	$\overline{V}DSR\overline{T}$	$\boldsymbol{\theta}, \mathcal{L}, \mathcal{T}$	Utility game	Ranking and Exp	Reach accuracy
FLAME$_{\text{C}}$	Global constrains: Limit client participation	$\overline{V}DSRT$	$\boldsymbol{\theta}, \mathcal{L}, \mathcal{T}, \mathcal{E}$	Utility game	Ranking	Limited energy
DDaBA	Global constrains: Defense against attacks	$\overline{VDS}R\overline{T}$	$\boldsymbol{\theta}$	Validation set	Weights	Fixed rounds
LT-FL	Client inclusion	$\overline{V}D\overline{SR}T$	$\boldsymbol{\theta}, \mathcal{T}$	Time-based	Clusters	Fixed rounds
FDropout		$\overline{V}D\overline{S}RT$	$\boldsymbol{\theta}$	Model compression	Weights	Fixed rounds
CI-MR	Client incentives	$VDSRT$	$\boldsymbol{\theta}, \lvert\mathbb{D}\rvert$	Shapley values	Weights	Fixed rounds
FMore		$\overline{V}D\overline{S}RT$	$\boldsymbol{\theta}, \lvert\mathbb{D}\rvert, \mathcal{T}...$	Auction Utility score	Ranking	Fixed rounds, Reach accuracy
CBIM		$\overline{V}DS\overline{RT}$	$\boldsymbol{\theta}, \mathcal{T}, \mathcal{E}...$	Contract theory Utility score	Threshold	Limited reward

Training efficiency The most common goal in client selection is to *speed up the convergence* of the training process. However, formalizing this simple goal may result in different technical solutions. Some authors [CGSY18; DLS21] focus on reducing the number of clients in a training round while maintaining the same convergence rate. Chen *et al.* [CGSY18] show that communicating only with the right clients can reduce the communication to a tenth of what it would be in a cyclic iteration of the clients. Later work by Chen, Horvath, and Richtarik [CHR20] reports that optimal client sampling may yield similar learning curves to those in a full participation scheme. Other works aim to reduce the total number of training rounds to reach the same accuracy. Cho, Wang, and Joshi [CWJ22] propose POW-D which takes half as many iterations to reach 60% accuracy on the FashionMNIST dataset than random client selection.

Alternative definitions of efficiency include energy consumption. For example, Cho, Mathur, and Kawsar [CMK22] propose the FLAME_C FL framework that computes energy profiles for each client and simulates different energy profiles in the federation. They report that incorporating energy consumption as a global constraint in the client selection method can save up to 2.86× more clients from reaching their energy quota when compared to the same method without any energy consideration.

Other authors propose limiting the time of the local training as the main guiding principle to achieve client selection. This concept was introduced by Nishio and Yonetani [NY19] in the FEDCS method, which filters out slow clients and hence speeds up the overall training time. FEDCS requires clients to estimate $\mathcal{T}_k$, the time needed to perform their round of learning, and the server selects only clients with $\mathcal{T}_k$ lower than a certain threshold. The authors report that selecting the *right* clients according to this criterion could reduce by half the total training time to reach the desired accuracy when compared to federated average (FEDAVG) with limited local training round time.

Resilience Client selection may be used to remove malicious clients from the training process, thus increasing the resilience of the system against adversarial attacks. Rodríguez-Barroso *et al.* [RMLH22] use a server-side validation set $\mathbb{D}_V$ to measure the accuracy of each local model. Then, the models with low accuracy are given lower weights. Their results show that this method is able to keep the same level of performance even if 10 out of 50 clients send malicious model updates. An alternative approach to filter malicious clients is using *Client Incentives*. In these methods, the rewards offered to the clients are inversely proportional to the probability of being a malicious or harmful client. We direct the reader to section 3.2.1 for a description of such methods.

While client selection can be a tool to fight against attacks, it opens a new vulnerability. Malicious clients may attempt to be included in the training rounds by changing their behavior or their data to be more appealing for selection. For example, the method proposed by Blanchard *et al.* [BEGS17] which selects clients with the smallest weight update is vulnerable to adaptive attacks. It could be abused by injecting a backdoor with a very small learning rate, and therefore small weight update. The server would then be selecting this client over other benign clients.

Fairness An additional policy for client selection is to ensure **fairness** in the criteria applied to select clients. In FL, and especially in cross-device FL, algorithmic fairness definitions, particularly group fairness definitions adopted in the machine learning literature

are hard to implement because they require knowledge of sensitive attributes that are only available on the clients' side. For example, providing group fairness guarantees according to a protected attribute (e.g. race or gender), would require the server to collect data of the performance of the model on different groups according to the protected attribute. However, the privacy motivation for FL would prevent such attributes from being shared with the server. One possible way to report sensitive data with privacy protection would be to use *differential privacy* [GKN17; PDG21].

In the context of FL, different definitions of *fairness* have emerged. Work by Mohri, Sivek, and Suresh [MSS19] aims to increase good-intent fairness to minimize the maximum loss in the federated training between clients. In the experimental evaluation, the proposed AFL_M method outperforms the baseline uniform distribution in various datasets while increasing the worst-case accuracy, i.e. the performance on the client with the lowest accuracy. This principle is related to min-max notions of fairness, such as the principle of distributed justice proposed by Rawls [Raw04].

Li *et al.* [LSBS20] define fairness as the property of achieving *uniform accuracy* across all clients. This definition corresponds to the parity-based notion of fairness, as opposed to the min-max principle described above. They propose a novel FL approach called q-FFL to reduce potential accuracy differences between clients by giving a larger weight to those clients with a large error during the training process. According to their experiments, the proposed method yields an increase of the worst accuracy by 3% in a class-split Fashion-MNIST federated experiment while keeping the average accuracy the same as a state-of-the-art method (AFL_M).

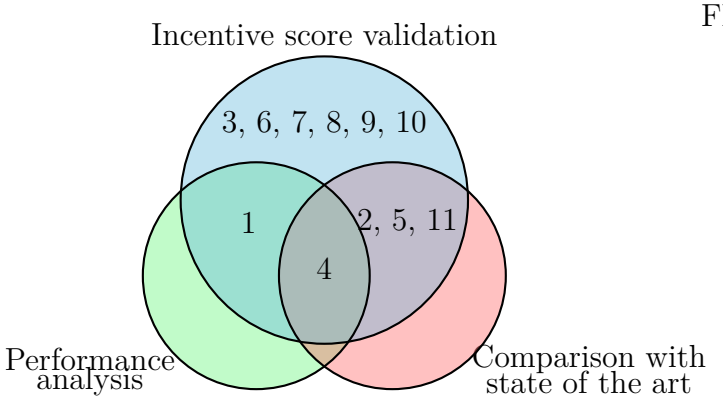
Shi, Yu, and Leung [SYL23] propose a taxonomy to categorize the fairness definitions that have been proposed in the FL literature. All the proposed fairness definitions belong to the Server Policies dimension of our taxonomy (see figure 2): *accuracy parity* and *good-intent fairness* would be categorized as Global Constraints; *selection fairness* could also be considered a Global Constraint or part of a Client Inclusion policy; finally, *contribution, regret distribution* and *expectation fairness* would belong to the Client Incentives policy.

Client inclusion policies

In some cases, the server might define specific client inclusion policies beyond the global constraints. For example, if a client has limited, but very relevant or valuable data to the problem at-hand (e.g. data from an underrepresented demographic group), the server may define client inclusion policies to ensure that such clients are selected even if they would not satisfy the global constraints.

Examples of client inclusion policies include ensuring a **loss tolerance** for communication packages. Zhou, Fang, and Hui [ZFH21] show that relaxing the global constraints for clients with poor communication channels not only improves client inclusion but also the overall performance of the FL system. Their proposed method LT-FL allows clients to discard lost packages instead of asking the server to resend them. Thus, the communication time decreases and otherwise poorly-represented clients are able to participate in the training.

An alternative client inclusion policy may be seen in the context of **model-agnostic FL**. In this case, low-resource clients have the option to learn different, simpler models. Thus, the local training may be faster and the communication requires less data transfer. One example of such an inclusion policy is Federated Dropout (FDropout) proposed by Caldas *et al.* [CKMT18], where simpler deep neural networks (with fewer neurons in the layers) are



Highlighted Incentive Mechanisms for FL Client Selection

1. Zeng *et al.* [ZZWC20]
2. Kang *et al.* [KXN+19a]
3. Sarikaya and Ercetin [SE19]
4. Zhao *et al.* [ZZJ+20]
5. Song, Tong, and Wei [STW19]
6. Wang, Dang, and Zhou [WDZ19]
7. Liu *et al.* [LAS+20]
8. Pandey *et al.* [PTB+19]
9. Hu and Gong [HG20]
10. Lim *et al.* [LXM+20]
11. Kang *et al.* [KXN+19b]

Figure 5. Experimental evaluations of proposed incentive mechanisms in the literature lack a quantitative comparison with state-of-the-art methods on benchmark datasets with a comprehensive performance and efficiency analysis.

learned in clients and the server combines them together into a larger network following the idea of dropout. Their method shows a 14× reduction in server-to-client communication, a 1.7× reduction in local computation and a 28× reduction in client-to-server communication compared to uncompressed models.

Following this work, Diao, Ding, and Tarokh [DDT21] and Liu *et al.* [LWW+22] propose methods to match weaker clients with smaller models and implement a model-agnostic FL. In HeteroFL, proposed by Diao, Ding, and Tarokh [DDT21], the smaller model's parameter matrix is a cropped version of the complete matrix. During the aggregation, the server aggregates the models with different sized hidden layers from smaller to larger. They present experimental results with 5 different sized models, the largest having 250 times as many parameters as the smallest. They report similar accuracies when setting half of the clients' models to the smallest size and half to the largest size than when all the clients have the largest model while halving the average number of parameters. Liu *et al.* [LWW+22] propose InclusiveFL where larger models have more hidden layers. They show that for more complex models (e.g. transformer models with the self-attention mechanism) InclusiveFL works better than HeteroFL on several ML benchmark datasets.

In chapter 4, we explore the use of model-agnostic FL for another objective: to improve the privacy of the client models.

Client incentives

A common assumption in Federated Learning is that all the clients are willing to participate in the training at any time. The reward for their contribution to the federation is access to an updated, global –and ideally better– model. However, clients might not see the benefit of the federation for a variety of reasons. For example, stateless clients in cross-device FL may never receive an updated model from the server after participating because the users might disconnect their devices before being selected a second time. Thus, an active research area in the literature of client selection in FL focuses on defining *incentive mechanisms*

for the clients so they participate in the federation. Note that such incentive mechanisms have been extensively studied in other multi-actor areas of science, such as economy [HS92; LNW99], environmental protection [Nic84; KND+00], mobile crowdsensing [YHSC17], and shared distributed energy sources [WZQ+19] or human resources [FDM+08].

In the context of FL, the server typically gives a reward to the participating clients according to their contribution. According to the work by Zhan *et al.* [ZZH+21], the incentive mechanism methods proposed to date may be categorized into three classes, described below. Zeng *et al.* [ZZW+21] additionally categorize the methods based on the used value generation techniques, described in section 3.2.4.

The first class of incentive mechanisms considers **client resource allocation** and gives rewards based on the computation power, energy usage, or allocated time. For example, Sarikaya and Ercetin [SE19] offer a Stackelberg game-based solution based on the clients' CPU power utilization. They study the server's strategy based on its budget and the trade-off between the number of learning rounds and the required time of a round with respect to the number of clients.

Second, the incentive mechanisms may be used to attract clients with good data quality and quantity. These incentive mechanisms are classified as **data contribution** incentives. For example, Song, Tong, and Wei [STW19] use Shapley values [Sha71] to evaluate the data contribution of clients and reward them accordingly. They define the contribution index (CI) such that (1) data distributions that do not change the model have no contribution; (2) two datasets with the same influence on the model have the same CI, and (3) if there are two disjoint test sets, the contribution index in the context of the union test set should be equal to the sum of the CI of the two original test sets. The proposed CI-MR method performs similarly to state-of-the-art methods while calculating the scores up to 5 times faster. Zeng *et al.* [ZZWC20] show that measuring the data contribution by means of a clients' utility score and applying an auction, the FMore method selects 80% of the participating clients, resulting in a $40 - 60\%$ speed-up in terms of the number of training rounds necessary to reach 90% accuracy on the MNIST dataset.

Third, the incentives may be correlated with the **reputation** of the clients. Such reputation may be computed from the usefulness of each of the client's models. In addition, several works have proposed using a blockchain to keep track of the clients' contributions and hence of their reputation [KXN+19a; ZZJ+20]. Kang *et al.* [KXN+19a] show that removing clients with a bad reputation may increase the model's final accuracy. Zhao *et al.* [ZZJ+20] report that a reputation score may punish potentially malicious clients while maintaining the global convergence of the model.

Future work

There are several areas of future work regarding the Policies of the server to perform client selection.

We refer to the lack of discussion on the privacy-efficiency trade-off more in section 3.2.3. Here we want to highlight work on privacy as a Global Constraint. Geyer, Klein, and Nabi [GKN17] defines a privacy budget for each client and proposes an experiment where clients terminate their participation if their privacy budget is reached. Thus, clients terminate in different rounds. This line of work can be expanded to more complex budgeting that reflects the information shared in messages.

While many client selection approaches have been proposed in the literature, they lack a vulnerability analysis of potential attacks directed against the approach itself. Future work in this direction would be needed before a specific client selection method is deployed in production.

Another potential research direction would focus on the clients' perspective to define the client selection strategy. While client incentive policies aim to motivate the clients' participation with fair payoffs, there is an opportunity to further study and propose FL approaches centered around the clients' interests. In this sense, we identify several relevant research questions, such as studying the trade-off between data availability and performance of the local vs federated learning models; developing models to support the clients' decisions as to when to join the federated learning scenario; and approaches that would enable clients to participate in the federation in a dynamic manner, depending on the current state of the learning model. Recent advances in model-agnostic FL in combination with incentive mechanisms suggest an interesting direction for future work. One could design a federated learning scenario, where the server would propose contracts to the clients with different model sizes and rewards. Then, the clients would be able to select the contract that matches their available resources and interests the best.

Finally, we believe that there is an opportunity to improve the empirical evaluation of existing and newly proposed incentive mechanisms for FL. There are 3 key components –depicted in figure 5– that we consider necessary to include in such an evaluation: first, a clear explanation as to why the proposed scoring satisfies the requirements of being an incentive mechanism; second, a performance and efficiency analysis on benchmark datasets; and third, a systematic comparison with other state-of-the-art client selection methods. As can be seen in the figure, most of the proposed methods in the literature fail to include one or more of these three elements. While several survey papers of incentive mechanisms in Federated Learning have been published (see e.g. Zhan *et al.* [ZZH+21] and Zeng *et al.* [ZZW+21]), to the best of our knowledge, none of them presents a thorough quantitative analysis of the topic. In section 3.3 we further discuss the scarcity of available benchmark datasets and experimental setups for Federated Learning.

3.2.2 Client characteristics

FL methods are implemented with certain assumptions regarding the characteristics that the clients should have. Thus, the Client Characteristics are the next dimension in the proposed taxonomy. The second column in table 4 provides an overview of the client characteristics required by a representative sample of the client selection methods proposed to date in the FL literature.

According to table 1 from Kairouz *et al.* [KMA+21], FL clients may have four important characteristics: availability, addressability, statefulness and reliability. We propose to add a fifth characteristic, *trustworthiness*, that should be considered when designing a client selection method in a FL scenario. These five characteristics are defined as follows:

Availability (V or $\overline{V}$): Let $\mathbb{A}$ be the active client set, i.e. the clients that are ready to participate in the next training round. In each training round, t, the server may select the participating clients $\mathbb{S}^t$ from the active client set $\mathbb{A}^t$. Generally, $\mathbb{S}^t \subseteq \mathbb{A}^t \subseteq \mathbb{K}$. Sometimes, for example in cross-silo scenarios, $\mathbb{A}^t = \mathbb{K}, \forall t = 1...T$. The assumption of client availability in a FL method will be denoted by V, otherwise $\overline{V}$.

Addressability (D or $\overline{D}$): Clients have unique identifiers, such that the server is able to individually address them and communicate with specific clients (D). Conversely, the server communicates in a broadcasting manner, sending the same message to all the clients ($\overline{D}$).

Statefulness (S or $\overline{S}$): In a stateful scenario (S), the clients are able to participate in multiple training rounds and they can refer back to the weights or parameters of previous rounds. In a stateless scenario ($\overline{S}$), each client participates only once in the learning process, so the client selection (and aggregation) policy is not able to depend on their previous performance.

Reliability (R or $\overline{R}$): Even if a client meets all the selection criteria and is selected by the server to participate in a round of training, it may fail before sending its information to the server. Federated learning methods that assume that all participating clients will successfully report back at the end of their local training make a reliability assumption, R. Otherwise, the methods are characterized as $\overline{R}$.

Trustworthiness (T or $\overline{T}$): In an ideal scenario, all clients are sharing truthful information with the server and hence they are trustworthy, T. However, clients may be malicious ($\overline{T}$) and share false information. Previous research in FL has focused on tackling this problem, such as Kang *et al.* [KXN+19a] who propose a client selection incentive that gives a bad reputation score to potentially harmful clients and Rodríguez-Barroso *et al.* [RMLH22] who gives less weight to clients with lower accuracy on a server-side validation set.

In general terms, these five characteristics of the clients are known to the server in cross-silo FL scenarios. However, in cross-device scenarios, several of these features may not be known to the server and the methods should be able to function without depending on any of them. We would like emphasize the importance of new FL client selection methods to identify their dependencies on these client characteristics such that different approaches may be compared and the reproducibility of the results is facilitated.

Future work

Client characteristics are not always considered when designing client selection policies which leads of inefficient or inconsistent FL systems. For example, cross-device FL is in many cases stateless as clients may join to the system only once during the training. However, there are reputation-based client incentives (based on the client's performance in previous rounds) proposed for cross-device FL that are impossible to implement as the server does not have access to the clients' state in a stateless context [KXN+19a]. The same problem was identified by Wang *et al.* [WCX+21] for FL in general. We would like to emphasize the importance of clearly describing the required client characteristics when proposing novel approaches to FL to ease the comparison with other methods and ensure the reproducibility of the results.

Trustworthiness in FL is often addressed at the level of optimization algorithms or model aggregation. Blanchard *et al.* [BEGS17] propose an algorithm to weigh the client models during aggregation in a way to reduce the effect of outliers. However, few research works address trustworthiness during client selection by e.g. removing harmful clients Zhao *et al.* [ZZJ+20]. We believe the addition of this feature to the client characteristics in our taxonomy may boost future research in this direction.

In addition to Client Characteristics, the proposed taxonomy includes a dimension related to the kind of information that the client shares with the server.

3.2.3 Shared information by the client

Federated learning was proposed with the motivation of preserving privacy by enabling a central server to learn from distributed client data without ever having access to the actual data. As the learning heavily depends on the information shared by the clients with the server, it is important to understand and analyze the types of messages that the clients send to the server.

Client selection requires analyzing information about the clients. Thus, there is a trade-off between a potential privacy loss from the client's perspective and the goals of implementing client selection. This section of the proposed taxonomy highlights the importance of studying and characterizing such a trade-off.

The most commonly shared client information are either the updated client weights ($\boldsymbol{\theta}_i$) or the change in the weights when compared to the previous learning round [NN21; WKNL20]. Additional methods proposed in the literature use the loss ($\mathcal{L}$) of the clients' model [GMB+19; MSS19; YWZ+21].

Moreover, clients might share additional information with the server. Examples from the literature may be categorized into two groups. Firstly, **scalar descriptors**, such as the Lipschitz-smoothness of the client loss function (L_f) [CGSY18; LSBS20], the total size of the client data ($|\mathbb{D}|$) [STW19; CWJ22; ZZWC20] or the expected training time on the client ($\mathcal{T}$) [NY19; KXN+19a; ZZWC20; ZFH21].

Secondly, a modified, **compressed version of the client data**, such as a vector $\mathbb{G}$ of cluster centers computed from the clients' data [DLS21].

The message sharing between the client and the server is denoted by $v_k = g(\mathcal{M}_k),\ k \in \mathbb{K}$, where $\mathcal{M}_k$ is the shared information by client k. To select the best clients, the server needs to assign a value (v_k) to each client. The function g generates this value from the client's shared information. This Value Generation process (described in section 3.2.4 below) may be done on either the server or the client side. Note that a server-side validation process to evaluate the clients' models reduces the server-client communication, but adds extra overhead on the server as it has to run all the client models on the validation set ($\mathbb{D}_V$) [WKNL20; NN21].

The fourth column in table 4 includes the types of Messages send by the clients to the server in each of the selected representative papers from the literature and the fifth column describes the Value Generation functions used by such papers.

To preserve the clients' privacy, several privacy-preserving methods have been used in FL frameworks, such as *differential privacy* Abadi *et al.* [ACG+16] and *secure aggregation* Bonawitz *et al.* [BIK+17]. Despite a call from Wang *et al.* [WCX+21] in a general field study to make client selection methods compatible with privacy-preserving methods, current client selection works do not discuss their compatibility. For example, client selection approaches relying on a server-side validation set (see Value generation column in table 4) cannot apply secure aggregation as they require server-side inference on the local models. Similarly, client selection strategies might not work if local differential privacy is used to disclose the clients' update Duchi, Jordan, and Wainwright [DJW13].

Future work

While current works analyze the communication in terms of performance and calculate the trade-off of including additional messages, there is a lack of research analyzing the privacy cost of sharing increased amounts of information between the clients and the server. In addi-

tion, future work is needed on the interplay between client selection and privacy-preserving methods, such as differential privacy and secure aggregation.

Using locally generated synthetic data for client selection may be another fruitful research direction. Tackling the challenge of non-identically and independently distributed (IID) distributed data problem has motivated GAN generated synthetic data sharing and progress was made in this direction [XYG+20; RMR+21], however, these methods do not leverage the client selection, only use all clients or implement random selection.

The next dimension in the proposed taxonomy is Value Generation, i.e. the process by which the server assigns a value to each client that will be used to inform the client selection process.

3.2.4 Value generation

Once the server receives each clients' shared information $\mathcal{M}_k$, it generates a relevance value (v_k) for each client. This value is used to select the set of participating clients in the next round of training ($\mathbb{S}^t$). In this section, we summarize the most significant Value Generation approaches proposed in the literature.

While the value generation typically takes place on the server side, in some cases the clients send an already processed value to the server, or the value might be even computed in both sides. We discuss how the server interprets this generated value in section 3.2.5.

A commonly used technique is to evaluate the clients directly using their local training loss. Intuitively, if a client has a large loss, the training would benefit from more rounds of the client's data. However, these **loss-based** methods need to store the results in previous rounds of the training, requiring either stateful clients [CGSY18] or clients that are able to estimate their loss in the current round. In this case and to reduce the overhead, some methods use a random selection of clients first ($\mathbb{A} \subset \mathbb{K}$) and then implement a more sophisticated client selection strategy on this subset of clients. In [CWJ22], the authors show that optimizing the size of this subset has a positive impact on the model's performance: their POW-D method outperforms $\text{EQREP}_{\text{baseline}}$ by 10% and AFL_{G} by 5% on the FMNIST dataset.

Other research suggests that the loss is only one of different metrics that one could optimize for. Lai *et al.* [LZMC21] proposes a **utility function** composed of different terms, including the loss-based statistical utility and the system's utility derived from the time needed for the training. They report a significant speed-up in training time: the proposed Oort reaches better accuracy in 10 hours than a random selection method in 30 hours on the OpenImage [KRA+20] image classification task. Cho, Mathur, and Kawsar [CMK22] incorporates a third utility term: an energy consumption-based score. They show that distributing the workload with respect to the energy usage doubles the number of training rounds without dropping clients due to reaching their energy limit. The general formulation for computing an overall utility score is given by equation (9), where $\mathbb{B}_k \subset \mathbb{D}_k$ is a subset of the kth client's data.

$$Util(k,t) = \underbrace{u_1(\mathcal{L}_k^t, \mathbb{B}_k)}_{Statistical\ utility} \times \underbrace{u_2(\mathcal{T}_k^t)}_{System\ utility} \times \underbrace{\cdots}_{Other\ utilities} \tag{9}$$

Other scholars have formulated the client selection problem as a **bandit problem**. Yang *et al.* [YWZ+21] propose a multi-armed bandit approach where the clients are the arms and the $\mathbb{S}^t$ client set is the superarm at round t to achieve reduced class imbalance between clients

and thus increase the training efficiency. Their method yields a 10% increase in accuracy when compared to random selection in a non-IID scenario of the CIFAR10 dataset.

In cooperative game theory, **Shapley values** are used to determine the contribution of clients or the value of their data [Sha51]. In FL, they are used to give a fair payoff to participating clients based on their contribution [WDZ19; STW19; LAS+20] and to identify the most valuable clients for the next training round [NN21]. Wang, Dang, and Zhou [WDZ19] use Shapley values to determine the features with the largest contribution in a vertical FL scenario. Song, Tong, and Wei [STW19] show that Shapley values are effective to compute a contribution index of the clients. Liu *et al.* [LAS+20] simulate clients with different data quality levels from the MNIST dataset and show that their Shapley value-based method gives higher scores to the clients in higher data quality group. While the above 3 works use Shapley values for incentives, Nagalapatti and Narayanam [NN21] show that removing irrelevant clients using Shapley values can increase the overall accuracy. Their S-FedAvg increases the validation accuracy from 40% to 80% when compared to FedAvg in a MNIST-based experiment.

The **Stackelberg game** is a market modeling structure with two types of actors: leaders and followers. In FL, the server acts as the leader and the clients as the followers. In Sarikaya and Ercetin [SE19], the server proposes a price for the clients' resources which the clients use to compute their utility scores that are sent to the server. Finally, the server selects the clients with the highest scores. Their experiments support that there is an optimal number of clients from an efficiency perspective that may be identified with such a client selection approach, despite the availability of more clients. Furthermore, Pandey *et al.* [PTB+19] propose to reward the clients' local accuracy and show that their method can achieve optimal utility scores in a small, 4-client scenario. Hu and Gong [HG20] propose a Stackelberg game to give incentives depending on the clients' data privacy loss. Unfortunately, these works lack both a comparative analysis of their empirical results with other state-of-the-art approaches and a performance analysis on commonly used benchmark FL datasets.

Auction and contract theory have also been proposed to assign a value to clients in FL incentive mechanisms. In this case, the server acts as the auctioneer and the clients bid with their local resources. The FMore system presented in Zeng *et al.* [ZZWC20] uses an auction to select the clients with the best utility-bid pair. The authors report a reduction of $45-68\%$ in the training rounds needed to reach a given accuracy in experiments with various datasets. Jiao *et al.* [JWN+20] demonstrate that the auction-based method can be applied in a FL scenario where clients compete for wireless channels. Deng *et al.* [DLR+21a] measure the learning quality of clients and apply an auction to select the participants. Their method is robust against attacks and achieves $50-80\%$ accuracy in tests where the FedAvg baseline method only achieves 10%.

In **contract theory**, the server proposes rewards for different client types, and the clients select the cluster they fit in according to their local resources. Kang *et al.* [KXN+19b] separates the different types of clients by their local model accuracy. Their work focuses on maximizing the profit that remains at the server after incentive payout. Lim *et al.* [LXM+20] also defines the client groups based on data quality and quantity. Their experiments show that these methods effectively reward the desired clients and motivate them to participate more than the less relevant ones.

Finally, **reinforcement learning** (RL) has also been proposed for client selection. [WKNL20] frame the client selection process as a RL task as follows: the server acts as the agent; the

state $s^t = \boldsymbol{\theta}^t, \boldsymbol{\theta}_1^t, ..., \boldsymbol{\theta}_N^t$ summarizes the current parameters in each client; the action consists of selecting a client for the next round; and the reward is the server's side accuracy. In their experiments, the proposed FAVOR method reduced the number of communication rounds by $23 - 49\%$ on baseline datasets. Deng *et al.* [DLR+21b] takes the client's data quality into account during value generation and shows that their method selects fewer and more suitable clients than a simple baseline, achieving 6% better accuracy while being $2\times$ faster. RL has also been used to implement incentive mechanisms combined with auctions or Stackelberg games. Jiao *et al.* [JWN+20] implement a RL-based auction that outperforms a greedy auction by $2 - 5\%$. Zhan *et al.* [ZLQ+20] propose using reinforcement learning to solve the Stackelberg game problem and approximate the equilibrium.

Future work

Regarding utility score-based models, there is a trend to find new, relevant descriptors of the system – such as energy in Cho, Mathur, and Kawsar [CMK22], include them in the utility function and show better results than those of previous methods. This trend can be recognized in incentive mechanisms as well, where the reward function is based on the utility of local resources. This line of work suggests a potential direction of future work by systematically collecting all the potential parameters and measuring their impact on the utility of the clients.

While some progress has been made in terms of contract theory, state-of-the-art methods to select contract groups have not investigated all the potential parameters of the clients. Kang *et al.* [KXN+19b] demonstrate that the number of contract groups impacts the performance, such that it would be important to investigate different clustering methods. Additionally, this technique does not address the issue of outliers and of clients in low-value groups which might be discriminated unfairly.

Once the clients have a value assigned to them, the server needs to interpret the values to perform the client selection. Thus, the next dimension in our taxonomy is Value Interpretation.

3.2.5 Value interpretation

At the end of the client valuation step, each client has been assigned a value on the server's side. Based on this value, the server selects the clients that will be part of the next training round.

The clients' values are typically interpreted as rankings, probabilities, or weights. When the values are interpreted as a **ranking**, the server selects the top n clients with the highest values [NY19; WKNL20; CWJ22]. If they are interpreted as a **probability** to be selected, the clients are chosen according to such a probability [GMB+19; MSS19; NN21]. Finally, the values might be considered to be **weights** that are applied to the clients' results, yielding a weighted sum of client data in the aggregation [STW19; LSBS20].

In addition to these three approaches to Value Interpretation, scholars have proposed alternative methods to interpret the clients' data. Chen *et al.* [CGSY18] propose using a **threshold** to be applied to the client values, which results in a dynamic number of selected clients in each training round. In their experiments, they report that fewer clients (and hence less communication) are needed as the training progresses, and the training reaches the same accuracy $5\times$ faster and with $10\times$ less communication than the baseline. Kang

et al. [KXN+19a] keep a reputation score of the clients and only select the clients with a reputation above a pre-defined threshold. They report that increasing this threshold and hence selecting fewer clients boosts the accuracy of the trained model.

Other scholars have proposed **clustering** the clients according to different criteria (e.g. communication costs, available resources, data characteristics) and only a few representatives from each cluster are selected and shared with the server. All the clients in a cluster are treated in the same way from the server's perspective. Examples include Dennis, Li, and Smith [DLS21] who select a reduced number of clients from each cluster of similar clients and report 35% less variance in the clients' final test accuracy when compared to random selection, indicating a possible direction towards Good-Intent Fairness; and Zhou, Fang, and Hui [ZFH21] who cluster the clients according to the communication cost with the server.

Note that several methods proposed in the literature leave space for *exploration* in the process of Value Interpretation. We denote these methods with *Exp* in table 4. In this case, the server selects a subset of the clients according to their value and leaves a predefined number of slots open to add clients which are selected randomly. For example, Goetz *et al.* [GMB+19] selects $|\mathbb{S}^t| - \epsilon$ clients based on probability and then fills the rest ϵ clients by random sampling. With a loss-based value generation function, their AFL_G method achieved a 2% area under the curve (AUC) performance increase while needing $30 - 70\%$ fewer epochs to reach this performance compared to a random selection of clients.

Future work

The final number of participating clients has an impact on the training process. In most cases, this number is defined based on the available resources, such as energy, communication bandwidth, or incentive payouts. The Value Interpretation step provides a method to select the number of participating clients depending on such constraints. However, many Value Interpretation approaches require the definition of parameters, such as thresholds or the number of clusters. Thus, automatically identifying the optimal number of participating clients based on their values is still an open research problem.

Another direction of future work consists of investigating further the exploration-exploitation trade-off in client selection. While randomness (i.e. maximal exploration) gives a chance to include unseen clients and improves selection fairness, it also reduces the effectiveness of the selection method. Finding the right balance in the exploration-exploitation spectrum is therefore a valuable research question to pursue.

3.2.6 Termination condition

The last dimension in the proposed client selection taxonomy is the Termination condition, which is defined by the server. The termination condition determines when to end the Federated Learning training process, and hence it is an important choice of the experimental setup. The definition of the termination condition typically depends on the goal and priorities of the FL system. The seventh column in table 4 provides an overview of the termination conditions of a sample of the most representative client selection methods proposed to date in the FL literature.

We identify three main categories regarding the termination condition.

In the first category of methods, the learning process ends after a **fixed number of rounds** (T) of federated training. In simulated scenarios, this termination condition helps

to identify whether different methods reach the same accuracy. However, in some works, the methods do not reach their best performance in a predefined number of rounds [CHR20; CMK22]. In this case, it is important to motivate why the experiments ended at that point. For real-world scenarios, this termination condition is easy to implement.

In the second category of approaches, the learning process ends when a **minimum required accuracy** is achieved. This termination condition is particularly helpful to compare different federated learning methods regarding how quickly they achieve the desired accuracy. The minimum required accuracy may be determined by the performance of a baseline method or by the task at hand to solve with the federated learning system [NY19; ZZWC20; LZMC21].

Finally, alternative termination conditions may be defined from the perspective of **limiting the participation of a client**: for example, Cho, Mathur, and Kawsar [CMK22] limit the energy drain a client's device is allowed to endure. Furthermore, [KXN+19a] limit the reward clients can receive in an incentive mechanism, thus the experiment finishes when all the rewards are spent. If the server has a validation set, it may apply an early stopping (E_S) approach. Otherwise, the client may suggest that they are unable to meaningfully contribute to produce a better model. In [CGSY18], the proposed LAG-WK FL system allows the clients to evaluate whether they should send an update to the server or not, and the training stops when no client sends a new round of parameters.

Future work

In terms of future work regarding the termination condition, we highlight two potential directions. First, privacy could be considered as a key factor to define the termination of the learning process [GKN17]. The PyShift package, designed by Ziller *et al.* [ZTL+21] was implemented with this consideration in mind, giving a privacy budget to the clients which is reduced every time the server accesses information from the specific client. Beyond PyShift, we identify two potential lines of future work: an analysis of the potential privacy loss of the clients that participate in several training rounds and the inclusion of a participation budget as a global constraint to protect the privacy of the clients' data.

Second, current works give the right of termination to the server. However, in a real-life scenario, the clients could also decide to stop their participation in the training process. Incentive mechanisms aim to achieve fair client participation, but they still give the power of termination to the server and generally do not consider the clients' interests or perspectives. Client-centric termination policies could therefore be a potential line of future work.

In addition to the six previously described dimensions for client selection in FL, we would like to highlight the importance of establishing benchmark datasets and experimental frameworks to facilitate the reproducibility of the proposed models and enable the development of comparative analyses. In the next section, we provide a summary of the most commonly used datasets in the literature of client selection in FL.

3.3 Datasets and benchmark experiments

Given the distributed nature of FL and its focus on privacy protection, it is difficult to produce and share realistic open-access Federated Learning datasets.

Table 5. Commonly used datasets in client selection experiments in FL. Clients are either split by classes, or more naturally along a feature of the data– for example, by writers of social media posts.

Dataset	**Split**	**Type**	**Methods**
MNIST [LeC98]	Classes	Image	S-FedAvg, FAVOR, CI-MR, FMore, CBIM, pow-d, FAVOR, FedCS, q-FFL, AFL_M, FMore, DDaBA
FashionMNIST [XRV17]	Classes	Image	
EMNIST [CATV17]	Classes	Image	FDropout
FEMNIST [CDW+19]	Writers	Image	k-FED, DDaBA
CIFAR10 [Kri09]	Classes	Image	FL-CIR, FAVOR, FedCS, FDropout, FMore, DDaBA
Shakespeare [CDW+19]	Roles	Text	k-FED, q-FFL
Sent140 [GBH09]	Writers	Text	q-FFL
Reddit (multiple variations)	Writers	Text	q-FFL, Oort, AFL_G
UCI Adult Dataset [Bla98]	PhD or not	Tabular	q-FFL, AFL_M
London Low Carbon [MO12; SCT+15]	Households	Timeseries	[SO21; BFA21]

Thus, most of the experiments reported in the FL literature have been performed on *artificial* FL datasets, generated from well-known benchmark machine learning datasets. Table 5 includes a summary of these benchmark datasets and their characteristics.

Note that creating an *artificially distributed* dataset from an originally centralized one requires designer choices to be made. In the current client selection literature, even if two papers use the same dataset, the results are in most cases not comparable due to different choices, such as a different distribution of the data. Moreover, the models that are used (e.g. deep neural networks) vary in different evaluations, adding another layer of difficulty to make comparisons. Given the low reproducibility of current benchmarks, a fair comparison between methods requires the researchers to re-implement and tune each of the relevant past works. The taxonomy proposed in this paper helps to identify the most important methods and their characteristics to enable such a comparison.

In general terms, we find two major approaches to create *distributed* datasets for client selection in FL. The first approach generates clients *based on the target classes* of a classification dataset. With this method, researchers are able to manipulate the non-IID nature of the clients yet the dataset will inherit –potentially unknown– dependencies. For example, in the case of hand-written digits datasets, it is known that there are multiple samples from the same writer. An ideal dataset to simulate a realistic client selection experiment would need to have a large amount of non-IID clients. Unfortunately, this approach does not seem to satisfy such a condition.

In the second approach, the clients are generated *based on a feature* of the samples. In this case, the feature may be the writer of the digits in the MNIST datasets or the author

of posts in social media platform datasets – such as Sent140, Reddit listed on table 5, or StackOverflow [RCZ+20]. Cross-silo FL scenarios might be simulated by using multiple publicly available datasets and assigning one dataset to each of the clients.

In summary, generating FL datasets from known benchmark datasets does not accurately represent the FL problem at hand, yet obtaining real FL datasets is challenging. We believe that there might be an alternative path moving forward. There are fields with historically available distributed datasets where privacy concerns are emerging. In these cases, the existing data could be leveraged to propose privacy-preserving FL approaches. One of such fields is residential energy consumption. Given current global energy market trends, building accurate predictions of energy consumption will be increasingly relevant both for consumers, producers, and energy distributors. With the adoption of smart meters, it is possible to collect detailed data on the consumers' side. In fact, there are several energy datasets available, such as the London Low Carbon project data [MO12; SCT+15]. However, analyzing such sensitive data centrally has clear privacy consequences [VMFS21]. Thus, there is an increased need and interest towards FL techniques applied to this use case [SO21; BFA21]. Other domains that could also benefit from FL include self-driving cars, smart city applications, and wearable IoT devices.

Moreover, in many domains there might be several sources of data available to tackle a specific problem. Following the example of the energy consumption prediction problem, in addition to the energy consumption patterns, there are household [SCT+15; Wil14] and weather datasets. Research on model-agnostic FL would enable building models that leverage datasets of different natures across clients.

Beyond using non-FL, pre-existing datasets, there are ongoing efforts to develop specific benchmark datasets for FL. First, we would like to highlight that OpenMinded[1] has started an initiative to build a network where researchers can access distributed data for FL while preserving privacy. Second, the LEAF framework by Caldas *et al.* [CDW+19] collects 6 datasets and defines a specific split of the data designed for federated learning.

Regarding baseline models, the reviewed client selection research typically includes the FedAvg algorithm by McMahan *et al.* [MMR+17] as the baseline. While this offers a much-needed standardization in the experimental evaluations, there are more recent federated optimization algorithms that we believe should be considered as baselines, such as FedAdam and FedAdagrad [RCZ+20]. We would like to emphasize the importance for scientists to embrace an open science approach, sharing both the data and the code of newly proposed models. We would also encourage the community to leverage rapidly maturing Federated Learning frameworks – such as TFF [2], PyShift [3], and Flower [4]– to ease the reproducibility of the results and enable the integration of novel client selection strategies into other parts of the federated pipeline, such as FL optimizers or differential privacy frameworks.

[1] OpenMinded: The Medical Federated Learning Program. Accessed: 2022-06-14, `https://openmined.hubspotpagebuilder.com/medical-federated-learning-program`

[2] `https://www.tensorflow.org/federated`

[3] `https://github.com/OpenMined/PySyft`

[4] `https://flower.dev/`

3.4 Conclusion

As an emergent field, documenting, properly evaluating, and comparing novel Federated learning methods is a complex task. In this chapter, we have provided an overview of the most notable works in client selection in FL. We have proposed a taxonomy to help researchers identify and compare previous approaches and to support practitioners and engineers in finding the most suitable method for their task at hand. Our taxonomy is composed of 6 dimensions to characterize client selection methods: Policies, Termination Condition, Client Characteristics, Shared Information, Value Generation, and Value Interpretation.

The Policies, Client Characteristics, Shared Information and Value Generation help identify the most suited method to satisfy the requirements and goals in a specific scenario. Value Interpretation and the Termination Condition are important to characterize the evaluation and enable the comparison of existing methods.

We have also outlined potential lines of future research regarding each of the six dimensions of the proposed taxonomy and highlighted the need for FL benchmark datasets and models to accelerate progress and ease the comparison of proposed approaches in this field. We hope that the proposed taxonomy will prove to be helpful in this regard.

In the next chapter, we explore the model inclusion methods introduced in section 3.2.1, specifically, the privacy implications of different model integration methods in federations where the clients learn models of different complexities.

Chapter 4

Privacy: a model heterogeneity analysis

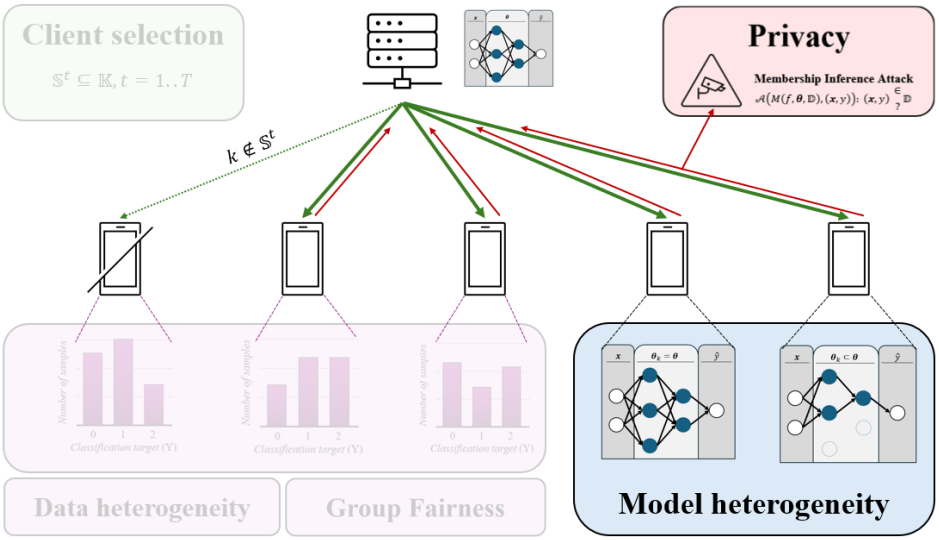


Figure 6. Content presented in this chapter in the scope of the thesis.

This chapter is based on the findings reported in the scientific publication *Privacy and Accuracy Implications of Model Complexity and Integration in Heterogeneous Federated Learning* published in IEEE Access in February 2025 [NLQO25]. Modifications are applied to improve the flow of the thesis.

4.1 Introduction

In chapter 3, we provided a taxonomy of client selection strategies in FL. One such method consists of allowing different model sizes in the clients to foster client inclusion (see section 3.2.1). [CKMT18] and [DDT21] showed that reduced model complexity for low-resource clients can be effectively used to include them in the FL process. Additionally, previous research showed that in centralized machine learning, model complexity is closely related to memorization (overfitting). [YGFJ18] demonstrated that the smaller the model, the less vulnerable it is to privacy attacks (see section 4.2). Based on these findings, in this chapter

we aim to answer the question of whether reduced model complexity in certain clients in a heterogeneous FL setting can improve the privacy of the training.

As described in chapter 1, FL comes with some core assumptions regarding the model type, the client data, and system behavior. Compared to vanilla FL, heterogeneous federated learning [YFD+23] refers to a more complex and realistic variant of FL where the participating clients have diverse conditions in terms of data, computing resources and model architectures. In this chapter, we focus on one type of heterogeneity namely *model size heterogeneity*, where different clients learn models of the same type but with varying complexities to adapt to their data and computational capabilities[5].

Several approaches have been proposed in the literature to implement FL with model size heterogeneity [CKMT18; DDT21; HLA+21; LGZX23]. However, they are generally seen as independent methods.

Furthermore, recent work discuss that regarding FL as a privacy-preserving solution by design is flawed. Research has shown that sensitive information about the original data can be inferred by analyzing the model parameters that are shared in the communication rounds [FJR15; CLE+19].

To the best of our knowledge, no study has explored the privacy implications of different heterogeneous FL methods. In this chapter, we fill this gap by providing the following 4 contributions:

(1) We are the first to frame existing heterogeneous FL methods in a novel taxonomy according to the adopted strategies to integrate the clients' models in the server's model;

(2) This taxonomy leads to the identification of seven new heterogeneous FL methods to perform client model integration in the server;

(3) We empirically evaluate the 7 proposed heterogeneous FL approaches and 2 state-of-the-art methods –namely HeteroFL and FDropout– from the perspective of server accuracy, and client accuracy and privacy on three widely used image datasets;

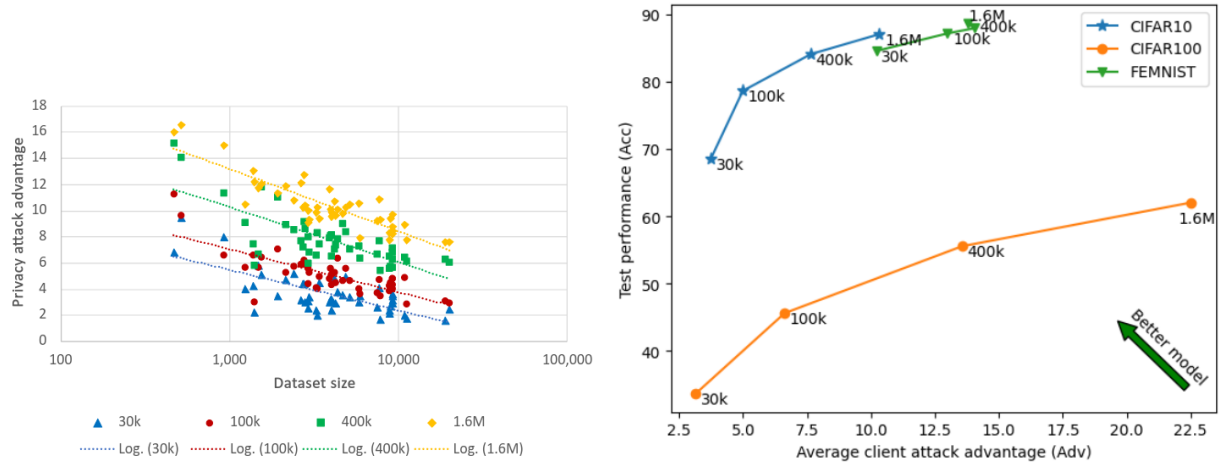
(4) We find that randomness in the strategy used to perform client model integration enhances the clients' privacy while keeping competitive performance on the server's side. In sum, our work provides the first empirical evidence on the privacy-accuracy implications of client model integration in heterogeneous FL.

4.2 Dataset size, privacy, model size and accuracy

Previous work has shown that as models get more complex, they are more vulnerable to MIAs. For example, [YGFJ18] demonstrate that their attack's accuracy increases as the model size increases on standard benchmark image datasets. In federated learning, Li et al. [LWC+22] reported that, the larger the models, the more vulnerable they are to model memorization attacks. In their case, it was a horizontal FL architecture with the same model (ResNet) both in the server and the clients. Other works have highlighted that over-parameterized models are vulnerable to membership memorization attacks [ZXS+23].

In this section, we shed further light on this topic by focusing on the privacy-accuracy trade-off in FL with respect to dataset and model size, and from the perspective of both the server and the clients. Note that prior studies have only analyzed the server's performance.

[5]In this chapter, we use the term *heterogeneous FL* to refer to heterogeneous FL methods where the clients learn models of the same architecture but different complexities than the server's model.



(a) **(b)**

Figure 7. **(a)**: Exemplary illustration of the correlation between the privacy attack advantage for the YEOM attack and the dataset size from the clients' perspective. Results for 5 repeated experiments on the CIFAR-10 dataset using the FEDAVG architecture with 10 clients having different dataset sizes, resulting in 50 client models. Each dot depicts a client in one federated training and the color represents different model complexities (CNNs), characterized by the number of parameters, ranging from 30k to 1.6 million. Note the negative correlations between the size of the clients' dataset and the attack advantage, as well as between the model's complexity and the associated attack advantage. **(b)**: Privacy-accuracy trade-off of the data depicted in (a) by averaging experiments across clients per model complexity. In addition to CIFAR-10, we also show the trade-off for the CIFAR-100 and FEMNIST datasets. The attacker's advantage and test accuracy on the clients increases as the model size increases. Observations in (a) and (b) suggest that model-agnostic federated learning could be a privacy-enhancing solution.

By means of an empirical illustrative example, we show that, for a given model and an FL scenario, there is a strong negative correlation between the size of the clients' datasets and models, and their privacy. We measure a model's privacy by its vulnerability againsts membership inference attacks (MIAs). We discuss MIAs in detail in section 4.5. We use the Yeom attack [YGFJ18] for this illustrative experiment as it requires significantly less computation than the other, more recent MIAs. This attack occurs on the last update the client sends to the server in round T, $\mathscr{A}_{\text{YEOM}}(\boldsymbol{\theta}_k^T)$.

We perform the experiments on the CIFAR-10 image dataset (see section 4.8 for a description of the dataset) with 10 homogeneous clients and a FedAvg FL architecture [MMR+17]. FedAvg follows the standard FL steps described in chapter 2 and computes the global model parameter by averaging the local model updates (see equation (2)).

To ensure a fair evaluation, the attacker's knowledge dataset $\mathbb{D}_{\mathscr{A}+}$ for the Yeom's attack is proportionate to the size of the training dataset. Specifically, we select 1%: $|\mathbb{D}_{\mathscr{A}+}| = \min(3, 0.01|\mathbb{D}_k|)$ for the attack on client k with dataset size $|\mathbb{D}_k|$. The attack test dataset $\mathbb{D}_{\text{MIA}}$ contains the same number of samples from the training set as samples from outside of the training set. If the client k has less than $5,000$ data samples, we test on all of the client data samples with non-member examples from the test set, so that $|\mathbb{D}_{\text{MIA}}| = 2\mathbb{D}_k$, otherwise it is capped at $5,000$. With such a dataset setting, a simple baseline which guesses that each MIA test data point is part of the training dataset would give a 50% accuracy. We define the attack advantage [HSS+22] as the improvement of an attack when compared to this baseline according to: $Adv(\mathscr{A}) = 2(Acc(\mathscr{A}) - 50)$, where $Acc(\mathscr{A})$ is the accuracy of the attacker's model.

Regarding the machine learning models, we adopt the architecture proposed in [DDT21]. It consists of a convolutional neural network (CNN) with 4 convolutional layers and one fully connected layer at the end. We adjust the model complexity by changing the number of channels in the convolutional layers and the number of units in the last fully connected layer. We define 4 levels of model complexity and train 5 models for each level of complexity using FedAvg with class-balanced data in each client, resulting in 50 client models. The complexity of the models is measured by the number of parameters, ranging from models with 30k to models with 1.6 million parameters.

For each model complexity, we compute the Pearson correlation coefficient between the logarithm of the clients' dataset size, $\log_{10}(|\mathbb{D}_k|)$, and the attack advantage on the clients' final update, $Adv(\mathscr{A}_{\text{YEOM}}(\boldsymbol{\theta}_k^T))$. Figure 7(a) visually illustrates the correlation between the client's dataset size and the attack advantage on the models of increasing complexity on the CIFAR-10 dataset. Note that clients with less than 400 data points are not considered in the calculation as their attack performance is not consistent through runs due to having very small (< 4) attacker knowledge. Figure 7(b) depicts the privacy-accuracy trade-off by averaging experiments across clients for each model complexity on the CIFAR-10, CIFAR-100, and FEMNIST datasets. We observe strong negative correlations between the size of the clients' dataset and the attack's advantage; and between the clients' model complexity and the corresponding attack's advantage. We also observe that both the attacker's advantage and the test accuracy on the clients increase as the model size increases. These results suggest that model-agnostic FL could enhance privacy both in the server and the clients by means of learning models in the clients that are smaller than the server's model.

4.3 Related work

In this section, we present the most relevant previous work on FL with model heterogeneity and on privacy attacks in FL.

4.3.1 Model heterogeneity in FL

In traditional FL all clients use the same model architecture as the server. However, this approach is unrealistic when clients have different computational and communication capabilities. FL with heterogeneous client models has been proposed to address this limitation as it enables training a diversity of models in the clients according to their capacities. There are two broad types of heterogeneous FL methods:

Knowledge transfer

In the first category, clients leverage a public dataset to communicate via knowledge distillation, and learn different models without sharing a global model with the server [LW19; YZJE23; WK23]. While this design enables clients to train different model architectures without limitations, its disadvantage is the lack of a competitive model in the server.

Model size heterogeneity

The second category consists methods with partial architecture sharing. For example, resource-restricted clients can learn a less complex model which is a smaller version of the server's model. In this case, both the server and client-side models are trained as part of the federation [CKMT18; DDT21; LWW+22]. In the context of deep neural networks, the model compression on the clients side can be achieved by training models with fewer [LWW+22] or with simpler [CKMT18; DDT21; HLA+21; LSL+21; LGZX23] layers. Our work focuses on heterogeneous FL methods in this category.

4.3.2 Membership inference attacks in FL

While FL was initially motivated by the desire to preserve client data, recent studies have revealed that federated systems remain vulnerable to privacy attacks, specifically in the form of membership attacks [GBX22; BRG+21; KD21; SHK+20; NHD+23]. To tackle this limitation, several privacy-preserving approaches for FL have been proposed to date, including local differential privacy [GBX22; BRG+21] and data augmentation [KD21; SHK+20]. In our work, we focus on membership inference attacks and their implications on heterogeneous FL. In MIAs, the attacker's goal is to determine whether an individual data point was part of the dataset used to train the target model. While MIAs expose less private information than other attacks, such as memorization attacks, they are still of great concern as they constitute a confidentiality violation [OV23]. Membership inference can also be used as a building block for mounting extraction attacks for existing machine learning as a service systems [CLE+19]. Several types of MIAs have been proposed in the literature [JWK+21; SSM19]. In this work, we focus on black-box attacks where the attacker does not have full access to the models but is able to query them, which is a more realistic scenario than white-box attacks that assume

full access to the models. We analyze the impact of three popular MIAs that use complementary strategies and hence offer a comprehensive evaluation of client privacy vulnerabilities in heterogeneous FL settings. Namely, the YEOM [YGFJ18], LIRA [CCN+22], and TMIA [LZBZ22] attacks. The YEOM attack is a simple, yet effective loss-based attack; LIRA is a good representative of shadow model-based techniques; and TMIA is a state-of-the-art knowledge distillation-based method to approximate the inspected model.

While it is known that the larger the complexity of a model, the higher its vulnerability against MIAs [LWC+22; YGFJ18], as illustrated by figure 7 and its corresponding section in the Appendix, we are not aware of any study of the impact on privacy of the model integration strategy adopted by the server in a heterogeneous FL setting.

4.4 A taxonomy of heterogeneous FL methods

In section 2.3, we described the current field of heterogeneous FL methods. In table 3a), we showed that methods can update the client model size each batch or each round dynamically or keep them static. In this section, we further elaborate on table 3b), and we propose a novel taxonomy of heterogenous FL methods which allows to both compare existing methods and identify seven new methods.

4.4.1 A taxonomy of heterogeneous FL methods

Figure 8 illustrates the proposed taxonomy of heterogeneous FL methods, according to three dimensions that characterize how the clients' models are integrated into the server's model. In each dimension, we label the categories with a single letter. We combine the letters to determine a method's place in the taxonomy by three letters representing the three dimensions. The figure highlights a method, OFM, that represents O (*one group*) in the first dimension, F (*fixed*) channel sets at the initialization of the training, and M (*submatrix*) policy for channel integration.

Client grouping The first dimension of the taxonomy refers to **client grouping**, classifying the methods in three classes: *one group* (O); *several groups* (G); and *unique* (U), depending on the number of channel sets used to train the models in the clients with smaller models than the server's model. In *one group*, all the clients use the same set of channels. In *several groups*, clients are clustered in groups such that clients in the same group use the same set of channels (figure 8a) shows an example with 4 groups). *Unique* corresponds to federations where every client has their set of channels selected individually, illustrated by the rectangles with different colored patterns in figure 8a).

Dynamics The second dimension (figure 8b) characterizes the **dynamics** of the channel selection approach and defines two types: *fixed* (F) methods when the channel sets are defined at the beginning of the training, and *resampled* (S) methods when the channel sets are selected in each training round, t_1 ... t_K. The figure illustrates the dynamics with 4 clients and in three training rounds t_1 to t_3. The clients' models are represented by rectangles with different colored patterns which represent the selection of channels from the server's model. In the *fixed* case, the clients get a variation of the same channels from the server in every

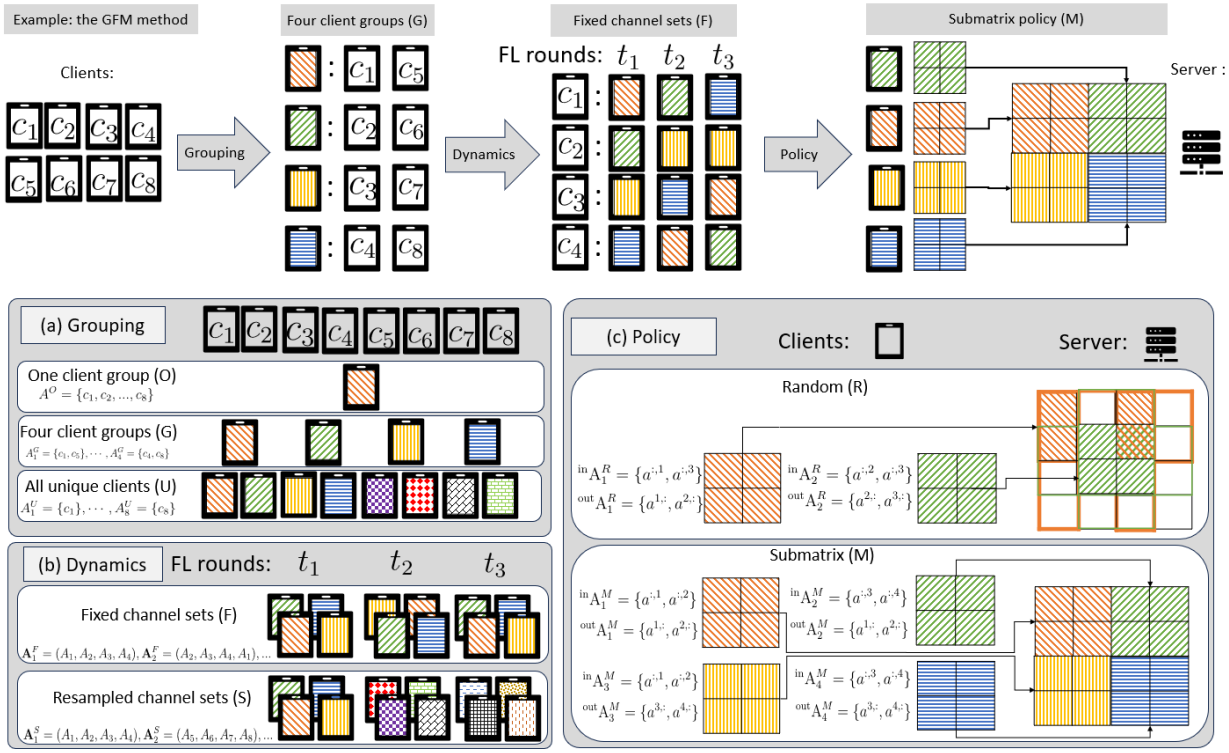


Figure 8. Proposed taxonomy of channel selection methods for heterogeneous FL architectures. The server ($\mathcal{S}$) and the clients ($c_i \in \mathbb{K}$) learn the same type of models (e.g. CNNs) but with different numbers of units. The server selects which subset (channels in the case of a CNN) of its model is used to train the clients' models. We refer to this mechanism as *client model integration*. The taxonomy considers three dimensions: **a)** The groups of clients learning from the same server channels: one group (O), four groups (G), all unique clients (U); **b)** Dynamics in channel group selection: fixed at the beginning of the training (F), sampled in each round (S); and **c)** Policy for channel selection from the server's model: according to a submatrix structure (M), randomly (R). The top of the figure illustrates the taxonomy with one type of the proposed heterogeneous FL methods, namely GFM : there are four groups of clients (G) indicated by different colors, which use fixed channel sets (F) that are integrated in the server's model as submatrices (M).

round of training (hence the patterns in the rectangles are the same in the different training rounds), while when the channel sets are *resampled*, they get a new set of channels from the server every round, and hence the patterns change in each training round t_i.

Policy Finally, the third dimension (figure 9c) concerns the **policy** for channel integration of the clients' models in the server's model and consists of two kinds: *submatrix* (M) methods if the selected channels are groups of adjacent channels and *random* (R) methods if the channels are selected randomly. In the Figure, with the submatrix policy the models from each of the 4 clients are integrated in non-overlapping sections of the server's matrix whereas with the random policy different portions of the clients' models are integrated in different sections of the server's model.

4.4.2 Existing heterogenous FL algorithms

The proposed taxonomy enables us to characterize existing methods in heterogeneous FL.

FDropout

In FDropout [CKMT18], the clients learn a CNN with the same architecture but fewer parameters (smaller weight matrices) than the server, and the server randomly drops a fixed number of units from each client [SHK+14], mapping the sparse model to a dense, smaller network by removing the dropped weights.

While the original formulation of FDropout used the same model size in all the clients, an extended variation was proposed by [HLA+21] that allows clients to have different model sizes and hence falls within the heterogenous FL definition used in this chapter. In this variation, in each layer l of the server's model, randomly selected cells, $a^{i,j,l}$ and their associated rows i and columns j are dropped from the weight matrix. The size of the client's model weight matrix in each layer can be set by the number of dropped rows and columns: $|Drop(N_l, N_{k,l})| = N_l - N_{k,l}$ and $|Drop(M_l, M_{k,l})| = M_l - M_{k,l}$, where $Drop(n, m)$ selects m elements from n randomly. Therefore, for each layer l of the server's model, in FDropout:

$$a^{i,j,l} \in \boldsymbol{A}_k : i \notin Drop(N_l, N_{k,l}), j \notin Drop(M_l, M_{k,l}). \tag{10}$$

According to our taxonomy, FDropout corresponds to a USR method because each client has a unique (U), random (R) set of channels that are resampled (S) in each training round.

HeteroFL

HeteroFL [DDT21] follows a similar idea as FDropout but with two key differences when selecting the channels in the clients with smaller models than the server: 1) all the clients learn from the same portion of the server's model; and 2) instead of randomly dropping cells, the clients always keep the top-left subset of the server's weight matrix for each layer in the network. Thus, in HeteroFL, the weight matrix $\boldsymbol{A}_k^l$ of size $N_k \times M_k$ in layer l and client k corresponds to the top-left sub-matrix of the server's weight matrix $\boldsymbol{A}^l$ of size $N \times M$:

$$\forall a_k^{i,j} \in \boldsymbol{A}_k, a^{i,j} \in \boldsymbol{A} : a_k^{i,j} = a^{i,j}, i = 1..N_k, j = 1..M_k. \tag{11}$$

According to our taxonomy, HeteroFL corresponds to an OFM method as there is only one client group (O) with fixed channels (F) that correspond to a sub-matrix (M) of the server's weight matrix.

4.4.3 Newly proposed heterogeneous FL methods

In addition to HeteroFL and FDropout, our taxonomy enables us to propose seven new methods for heterogeneous FL. In the following and for simplicity, we drop the superscript l to denote the layer in the network.

1. **GFM:** In the GFM method, instead of selecting the top-left sub-matrix of the server's model (as in HeteroFL), the clients are randomly placed in n groups. In the following, we present the example where $n = 4$. Thus, the clients are assigned to one of 4 groups, O, P, Q, R. The client's channels are selected based on their group's policy,

such that each cell from the original matrix is assigned to one cell in one of the four group.

The matrix assigned to group O is the same as the HeteroFL sub-matrix: it always selects the top-left cells of the server's matrix. Clients in group R are assigned the bottom-right cells. The sub-matrices assigned to clients in groups O and P alternate between the bottom-left and the top-right cells. This is due to a restriction on how the input-output channels need to be connected. The top-right sub-matrix corresponds to selecting the second half of the input channels and the first half of the output channels. Therefore, if in layer l the client selected the top-right sub-matrix, in the next layer it has to select one of the left sub-matrices, as they are the ones with the first half of the input channels. Note that this approach can be generalized to 9, 16,... groups, depending on the number of clients and the desired model size reduction. The cell assignment in the sub-matrices of each of the four groups is summarized in equation (12) below.

The top portion of figure 9 illustrates the GFM method. As seen in the Figure, the clients are grouped into four groups (G) with fixed channel sets (F) and integrating their models as submatrices of the server's model (M).

2. **GFR:** Compared to GFM, GFR differs in the set of channels in $\boldsymbol{A}_O$, $\boldsymbol{A}_P$, $\boldsymbol{A}_Q$, and $\boldsymbol{A}_R$. Instead of selecting the first or the last N_k and M_k channels, the output channels are selected randomly, while the input channels match the output channels of the previous layer.

$$a^{(i,j),l} \in \begin{cases} \boldsymbol{A}_O, & \text{if } 1 \leq i \leq N_k, 1 \leq j \leq M_k \\ \boldsymbol{A}_P, & \text{if } 1 \leq i \leq N_k, M - M_k \leq j \leq M, l \text{ odd}, \\ & \text{or } N - N_k \leq i \leq N, 1 \leq j \leq M_k, l \text{ even} \\ \boldsymbol{A}_Q, & \text{if } N - N_k \leq i \leq N, 1 \leq j \leq M_k, l \text{ odd}, \\ & \text{or } 1 \leq i \leq N_k, M - M_k \leq j \leq M, l \text{ even} \\ \boldsymbol{A}_R, & \text{if } N - N_k \leq i \leq N, M - M_k \leq j \leq M \end{cases} \tag{12}$$

3. **GSR:** GSR is similar to GFR but in this case the set of channels are drawn randomly for each group in every round of training.

4. **OSM:** OSM generalizes HeteroFL by leveraging the channel sets $\{\boldsymbol{A}_O, \boldsymbol{A}_P, \boldsymbol{A}_Q, \boldsymbol{A}_R\}$ introduced in GFM, but in each training round all clients are using one of the 4 groups.

5. **OFR:** OFR is a variation of HeteroFL where instead of the top-left subset of channels in the server's weight matrix, the clients all get the same random set of channels for every round of training.

6. **OSR:** In OSR, the set of channels are drawn randomly in every training round, but all clients use the same set.

7. **UFR:** Finally, UFR selects K unique sets of channels from the server's model which are defined at the beginning of the training and clients have access to one of the sets according to a new permutation every round. Therefore, in a federation with N clients, the clients receive the parameters from the same set of channels every N rounds.

We are interested in shedding light on the server accuracy and client accuracy-privacy trade-off of these 9 methods to perform client model integration in FL with heterogeneous models. Specifically, we focus on *membership inference attacks* or MIAs, as they represent a critical privacy threat in federated learning. MIAs allow adversaries to determine whether a particular data point was part of a client's training set by exploiting patterns in model updates or predictions. Given the diversity in model sizes in heterogeneous FL, the susceptibility of smaller, resource-constrained models to such attacks may differ from that of more complex models. By analyzing the performance of MIAs across methods to achieve heterogeneity in FL, we aim to understand the extent the model size in the client and the model integration strategy impact both privacy and accuracy. This analysis is crucial for developing robust FL frameworks that balance privacy guarantees and model performance in real-world settings with heterogeneous devices.

4.5 Membership inference attacks in FL

In section 2.5, we defined privacy in machine learning as the level of protection against membership inference attacks (MIAs). These attacks focus on reconstructing information about the training data based on the available model ($\boldsymbol{\theta}$) by querying the $(\boldsymbol{x}, y)$ input-output samples. We introduced three MIAs, Yeom, LiRA, and tMIA. In this section we further elaborate on these three attacks and their relation to FL.

The selected attacks represent a distinct approaches to MIAs, ensuring a comprehensive evaluation and coverage of a variety of methodologies: (1) the Yeom attack is a simple, popular, loss-based and interpretable method; (2) LiRA is a good representative of shadow model-based techniques, which leverage synthetic data and advanced likelihood estimation methods to achieve high accuracy and scalability; and (3) tMIA is a state-of-the-art knowledge distillation-based method. These are passive, black-box MIAs as the attacker does not use infer the training process, and it does not use the architecture of the model to design the attack.

Additionally, in federated learning, membership inference attacks can occur on the client or the server sides. In this chapter, we focus on client attacks which occur when the attacker targets the client's model, $(f_k, \boldsymbol{\theta}_k^t)$, for client $k = \mathbb{K}$ in training round $t = 1, \ldots, T$. In a setting where all clients participate in the federation (stateful setting section 3.2.2), the attacker can collect a set of $s \leq T$ client updates $\Theta_k = \{\boldsymbol{\theta}_k^{\tau_1}, \ldots, \boldsymbol{\theta}_k^{\tau_s}\}, \tau_i \in \{1, .., T\}$; Specifically, we consider *client-side* attacks which take place on the last parameter update from the client to the server $\boldsymbol{\theta}_k^T$ where the attacker aims to identify instances of the client's dataset $\mathbb{D}_k$ for client k.

4.5.1 Yeom attack

The Yeom attack [YGFJ18] is a membership inference attack that relies on comparing the prediction loss of a model on specific data instances to a pre-defined threshold. This threshold distinguishes between instances that were likely part of the training dataset and those that were not. The underlying assumption is that data points used in training tend to have a lower loss than those that were not, because the model has learned to perform well on training instances.

Attack setup and threshold selection

The YEOM attack relies on two main components:

1. **Global threshold** (ν): The attacker sets a loss threshold ν, below which an instance is considered likely to belong to the training dataset.
2. **Attacker's knowledge**: To calculate this threshold, the attacker uses a subset of data instances with known membership. This auxiliary set includes:
   - $\mathbb{D}_{\mathscr{A}+}$: Known training instances (member data).
   - $\mathbb{D}_{\mathscr{A}-}$: Known non-training instances (non-member data).

Using $\mathbb{D}_{\mathscr{A}+}$, the attacker computes the threshold ν as the average loss on known member instances:

$$\nu = \frac{1}{|\mathbb{D}_{\mathscr{A}+}|} \sum_{(\boldsymbol{x}',y')\in\mathbb{D}_{\mathscr{A}+}} \ell(y', f(\boldsymbol{x}';\boldsymbol{\theta}))$$

where:

- $f(\boldsymbol{x}';\boldsymbol{\theta})$ is the model's prediction for input $\boldsymbol{x}'$,
- $\ell(y', f(\boldsymbol{x}';\boldsymbol{\theta}))$ is the loss for true label y' and prediction $f(\boldsymbol{x}';\boldsymbol{\theta})$.

Membership inference decision

Once the threshold ν is established, the attacker determines the membership status of a new instance $(\boldsymbol{x}, y)$ by comparing its loss to ν:

$$\mathscr{A}_{\text{YEOM}}(\hat{y}, (\boldsymbol{x}, y)) = \begin{cases} 1, & \text{if } \ell(y,\hat{y}) < \nu \\ 0, & \text{otherwise.} \end{cases}$$

Here, $\hat{y} = f(\boldsymbol{x};\boldsymbol{\theta})$ is the model's predicted label for $\boldsymbol{x}$, and $\text{loss}(y,\hat{y})$ is the computed loss for this instance.

4.5.2 LiRA attack

We use the offline version of the LIRA (Likelihood Ratio Attack) attack of [CCN+22].

Attack setup: shadow models and auxiliary dataset

The attack operates in several steps:

1. **Auxiliary dataset** ($\mathbb{D}_a$): The attacker has access to an auxiliary dataset $\mathbb{D}_a$ that is similar to the data used to train the target model, although not necessarily identical. This auxiliary data is used to simulate the behavior of the target model with respect to training and non-training instances.
2. **Shadow models** (M_{sw}): Using $\mathbb{D}_a$, the attacker trains multiple "shadow models" M_{sw} on different random subsets $\mathbb{D}_{sw} \subset \mathbb{D}_a$. Each shadow model is designed to mimic the target model's behavior, especially in terms of confidence levels for data that was and was not in the training set.

3. **Confidence scores**: For each data instance $(\boldsymbol{x}, y)$, the attacker queries each shadow model M_{sw} to obtain a confidence score, typically the model's probability prediction for the true label y. We denote the confidence score of shadow model M_{sw} on $(\boldsymbol{x}, y)$ as $\phi(M_{sw}(\boldsymbol{x}), y)$.

Modeling confidence scores with a Gaussian distribution

For the data instance $(\boldsymbol{x}, y)$, the attacker gathers confidence scores from all shadow models, yielding the set $\{\phi(M_1(\boldsymbol{x}), y), \phi(M_2(\boldsymbol{x}), y), \ldots, \phi(M_k(\boldsymbol{x}), y)\}$. This set of scores is used to model the distribution of confidence values for instances that are either "in" or "out" of the training data.

The attacker then fits a Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$ to these confidence scores, where μ and σ^2 represent the mean and variance of the scores. This Gaussian distribution captures the typical confidence score behavior of shadow models, depending on whether $(\boldsymbol{x}, y)$ was in the training data.

Calculating membership probability

Once the Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$ is fitted, the attacker uses it to assess the probability that a new confidence score $\phi(M(\boldsymbol{x}), y)$ from the target model M is characteristic of the training data.

The membership probability is calculated as:

$$1 - \Pr\left[\mathcal{N}(\mu, \sigma^2) > \phi(M(\boldsymbol{x}), y)\right]$$

This probability reflects the likelihood that $\phi(M(\boldsymbol{x}), y)$ would be a typical score under the Gaussian model. A higher probability suggests that $(\boldsymbol{x}, y)$ is likely part of the training data, whereas a lower probability indicates non-membership.

Membership inference decision

To make a final membership inference decision, the LiRA attack applies a threshold ν to determine whether the probability of membership exceeds a certain level as per the following decision rule:

$$\mathscr{A}_{\text{LiRA}}(\hat{y}, (\boldsymbol{x}, y), \mathbb{D}_a) = \begin{cases} 1, & \text{if } 1 - \Pr[\mathcal{N}(\mu, \sigma^2) > \phi(\hat{y}, y)] < \nu \\ 0, & \text{otherwise.} \end{cases} \tag{13}$$

such that

- The attack outputs 1 (indicating membership) if the membership probability exceeds the threshold $1 - \nu$.
- Otherwise, it outputs 0 (indicating non-membership).

4.5.3 Trajectory MIA attack

The TMIA attack [LZBZ22] determines the membership of a data point based on the loss trajectory of the instance over multiple training epochs. The underlying hypothesis is that the sequence of loss values (i.e., the *loss trajectory*) of a data point changes differently for training data (members) and non-training data (non-members) as the model learns. Therefore, tracking these loss values over epochs can reveal membership status.

In a black-box setting such as ours, only the final trained model is accessible, and therefore the loss trajectory throughout training is not available. To address this, the TMIA attack uses knowledge distillation to approximate the loss trajectory.

Attack setup: target and shadow models

The target model is denoted by $M^0_{tg}(f, \mathbb{D}_g)$, where f is the model function and $\mathbb{D}_g$ is the dataset used to train the model. The attack involves two key steps:

1. **Shadow model training**: A shadow model $M^0_{sw}(f, \mathbb{D}^+_{sw})$ is trained on a subset $(\mathbb{D}^+_{sw}, \mathbb{D}^-_{sw}) \subset \mathbb{D}_a$, where $\mathbb{D}^+_{sw}$ contains samples similar to the training data and $\mathbb{D}^-_{sw}$ contains non-training samples.

2. **Distillation Process**: The attacker distills both the target and shadow models on a distillation dataset $\mathbb{D}_{dl} \subset \mathbb{D}_a$. During this process, snapshots of the models are saved at each training epoch, resulting in a sequence of models:

$$\{M^0_{tg}, M^1_{tg}, \ldots, M^d_{tg}\} \quad \text{and} \quad \{M^0_{sw}, M^1_{sw}, \ldots, M^d_{sw}\}.$$

Loss trajectory calculation

For a data instance $(\boldsymbol{x}, y)$, its loss trajectory is captured by evaluating the loss of the data point at each epoch during the distillation process. This yields a sequence of losses:

$$\lambda^{(\boldsymbol{x},y)}_* = \{\ell^0_*, \ell^1_*, \ldots, \ell^d_*\}^{(\boldsymbol{x},y)}_*$$

where each ℓ^i_* is the loss at epoch i for model M^i_*, and $* \in \{tg, sw\}$ represents either the target or shadow model.

Training the attack model

The attack model M_A is trained to recognize patterns in the loss trajectories that indicate membership. To train M_A, the loss trajectories of data points in the shadow model, both from the shadow training set $\mathbb{D}^+_{sw}$ (members) and shadow non-training set $\mathbb{D}^-_{sw}$ (non-members), are used. Specifically, the training set for M_A consists of:

$$\{\lambda^{(\boldsymbol{x},y)}_{sw} \mid (\boldsymbol{x}, y) \in \mathbb{D}^+_{sw} \cup \mathbb{D}^-_{sw}\}$$

where each $\lambda^{(\boldsymbol{x},y)}_{sw}$ is the loss trajectory of $(\boldsymbol{x}, y)$ on the shadow model.

Membership inference decision

During inference, the attack model M_A predicts the membership status of a new data instance $(\boldsymbol{x}, y)$ by analyzing the loss trajectory from the target model's distillation process $\lambda_{tg}^{(\boldsymbol{x},y)}$. The attack decision rule is:

$$\mathscr{A}_{\text{TMIA}}(\hat{y}, (\boldsymbol{x}, y), \mathbb{D}_a) = \begin{cases} 1, & \text{if } M_A(\lambda_{tg}^{(\boldsymbol{x},y)}) > \nu \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

Here:

- $M_A(\lambda_{tg}^{(\boldsymbol{x},y)})$ is the output of the attack model on the loss trajectory $\lambda_{tg}^{(\boldsymbol{x},y)}$.
- ν is a threshold value; if the attack model's output exceeds this threshold, the instance is predicted as a member (1), otherwise, it is predicted as a non-member (0).

4.5.4 Performance metrics of MIAs

Three commonly used metrics to determine the performance of MIAs include:

- **AUC (Area under the curve):** This metric represents the area under the Receiver Operating Characteristic (ROC) curve, which plots the True Positive Rate (TPR) against the False Positive Rate (FPR) at various threshold settings. AUC ranges from 0.5 (random guessing) to 1.0 (perfect prediction). A higher AUC indicates better discrimination between members and non-members in the dataset and hence better performance of the MIA.
- **Attack advantage:** This measures how much better the attack model performs compared to random guessing. It is calculated as double the difference between the attack model's accuracy and 0.5 (the accuracy of a random classifier). A higher attack advantage indicates the attack is more effective at distinguishing members from non-members than a random model.
- **TPR@FPR 0.1:** This is the True Positive Rate (or recall) when the False Positive Rate is fixed at 0.1 (10%). It reflects the proportion of actual members that are correctly identified by the attack when only 10% of non-members are incorrectly classified as members. A higher TPR@FPR 0.1 indicates a more powerful attack with a low tolerance for false positives.

4.6 Privacy-accuracy trade-off

Each of the heterogeneous FL methods is expected to provide a different privacy-accuracy trade-off, depending on how the client model integration is performed on the server. We formulate below three hypotheses that we empirically validate in our experiments.

Frequency hypothesis (H1)

We hypothesize that heterogeneous FL methods where clients have access to the same set of channels more frequently perform better in terms of client-level accuracy but have worse client-level privacy. For example, in GFM the clients access the same set of channels every four rounds. Thus, compared to the two state-of-the-art heterogeneous FL methods, namely HeteroFL (same set of channels per client across rounds) and FDropout (random selection of channels per client in each round), we expect this method to yield a client privacy-performance trade-off between these two existing methods.

Based on this hypothesis, we expect:

- OSR, GSR, and USR (FDropout) to be the most resilient methods against MIAs but provide the worst client accuracy as the clients receive the parameters from a new set of channels in every round. Therefore, the same set is only repeated in every $\binom{N}{N_k}$ rounds on average for client k with client channel size N_k and server channel size N.
- OFM (HeteroFL) and OFR to be the most vulnerable against MIAs but achieve high client accuracies as the clients train using the parameters of the same set of channels in every round (1 round).

Similarity between the M and R categories (H2)

In a CNN layer, as long as the selected input channels of layer l match the output channels of layer $l-1$, the differences between variations M and R should be small. They differ only in the number of channels shared by client groups. We designed the sub-matrix category (M) to minimize the channel overlap between groups. Thus, we expect the models in the M and R categories to behave similarly regarding performance and privacy.

The differences in the privacy-accuracy trade-off between the methods decrease as the number of large clients in the federation increases (H3)

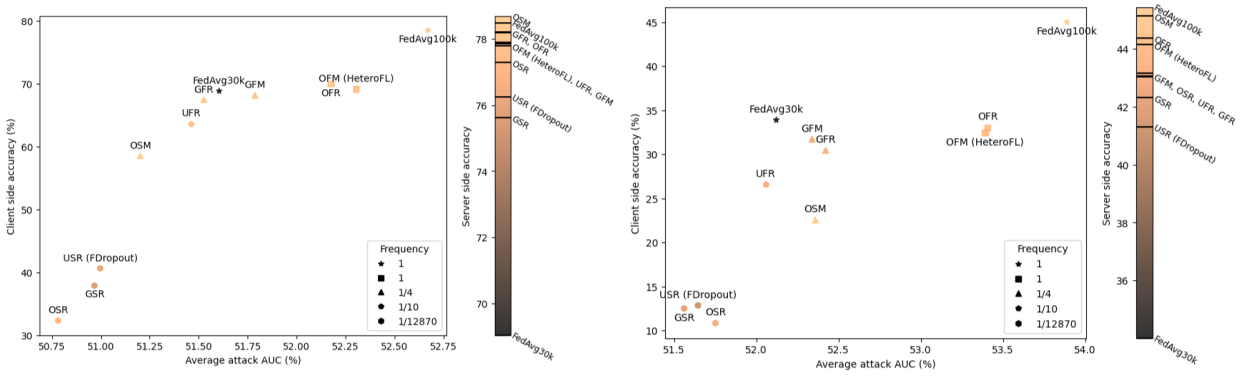
The heterogeneous FL methods discussed in this chapter are relevant when the majority of the clients learn smaller models than the server's model. Note that in cases when all the clients but one learn models of the same complexity as the server's model, the UFR and OFR methods become the same. Therefore, we expect the impact of the channel selection strategies to be larger when the majority of clients in the federation learn smaller models than the server's model.

We perform a comparative analysis of the proposed methods and empirically validate our hypotheses in experiments on commonly used vision datasets, as described next.

4.7 Methodology

4.7.1 Datasets

We perform experiments on two widely used image datasets: CIFAR-10 and CIFAR-100; given that, our work lies at the intersection of MIA techniques and heterogeneous FL, and to the best of our knowledge, these two datasets are the only common datasets in the



(a) CIFAR-10 **(b)** CIFAR-100

Figure 9. Client accuracy vs privacy trade-off and server-side accuracy of the 9 heterogeneous FL methods under study. Privacy attack performance (AUC) averaged over the 3 MIAs (Yeom, LiRA, tMIA). All heterogeneous FL architectures consist of 2 clients with large models (100k parameters, same size as the server's model) and 8 clients with small models (30k parameters). For reference, we report the performance of FedAvg30k and FedAvg100k with 10 small (30k parameters) and 10 large (100k parameters) clients, respectively. Heterogeneous FL methods with optimal client accuracy-privacy trade-off would be on the top-left corner of the graph. Heterogeneous FL methods with largest server-side accuracy are depicted at the top of the bar with gradient shading on the right-hand side of the graphs. Marker highlights repeated channel frequency.

literature from both communities(FL: [MMR+17; DDT21; HLA+21] and MIA: [YGFJ18; KD21]).

CIFAR-10 [Kri09] contains 60,000 images from 10 classes (50,000 images for training and validation and 10,000 images for testing).

CIFAR-100 [Kri09] has the same number of training and testing images as CIFAR-10 but with 100 classes and 500 training images per class.

We use a class-wise balanced, but client-wise weighted distribution. We generate a data distribution using the Dirichlet distribution $Dir(\alpha)$ once, and apply the same split for each class. This ensures that each client has the same number of images from each class while they have different dataset sizes. The dataset size imbalance is controlled by the $\alpha \in (0, \infty)$ value: the larger the α, the closer the allocation of training data to the uniform distribution and hence the closer to an IID scenario. Using $\alpha = 0.85$ this distribution generates clients with dataset sizes typically ranging from $1,000$ to $10,000$ samples. We apply *random crop* and *random flip* augmentations.

4.7.2 Machine learning model

Given the nature of the data (images), we use a sequential CNN architecture, with convolutional, batch normalization, and fully connected layers with trainable weights following [DDT21]. We control the model complexity by changing the number of channels in the convolutional layers and the number of units in the final fully connected layer. In our experiments, we increase the complexity by factors of 2: each increase in the level of model complexity entails doubling the input and output channel sizes in each inner convolutional layer and the number of units in the final fully connected layer.

The model in all our experiments is a Convolutional Neural Network, similar to the models

reported in related work [DDT21]. The layers with weight matrices consist of 2D convolutional layers with a (N, M, H, W) 4-dimensional matrix, where the first two dimensions N and M correspond to the output and input channels and the rest are the convolutional kernels. From a heterogeneous FL perspective, N and M are the dimensions that change when the clients in the federation learn models of different sizes than the server's model whereas H and W are the same as in the server. In the PyTorch implementation, the bias of the convolutional layers has a separate (N) 1-dimensional matrix. When a subset of N_k^l output channels is selected for a client k and convolutional layer l, its bias shares the same N_k^l out of N^l output channels.

After the convolutional layers in the model architecture, there are BatchNorm normalization layers with (N) 1-dimensional weight matrices with bias. Note that the BatchNorm layer $l+2$ after convolutional layer l has the same $N_k^{l+2} = N_k^l$ channels selected. The Scaler layer adapted from HeteroFL [DDT21] scales its input with respect to the model-agnostic compression rate. For $r_k = \frac{N_k}{N} = \frac{M_k}{M}$, the Scaler follows:

$$f_{\text{Scaler}}(x) = \frac{1}{r_k} x. \tag{15}$$

Finally, there is a linear layer lin with weight matrix (N, M) and bias with weight matrix size (N). Each client k shares the same N_k^{lin} output channels in this linear layer.

The complexity of the model is controlled with parameter u. Each input and output dimension of the weight matrix is a multiple of u. The model complexity levels used in this chapter –namely 30k, 100k, 400k, and 1.6M– correspond to u values of 8, 16, 32, and 64, respectively. Figure 10 illustrates the model architecture for a generic u and an example with $u = 16$.

Input-output channel dependency

In section 4.4.2, we present FDropout [CKMT18] and HeteroFL [DDT21] according to their original descriptions, which suggest that the channels of a layer l can be dropped independently from the previous and following channels. However, after extensive experiments, we observed that the client models train significantly better if the selected output channels of a convolutional layer are *the same* as the input channels of the following convolutional layer. In the FDropout adaptation of [LGZX23] the same principle is adopted: layer l only drops output channels randomly, while the selection of the input channels is inherited from the previous convolutional layer. The pseudo-code in [CCGR22] suggests that their implementation follows the original layer-independent dropout, and their results show that FDropout performs badly compared to other techniques: while the Simple Ensemble Averaging method reached the 70% accuracy of the baseline FedAvg on the FEMNIST dataset, the presented implementation of FDropout only reached 60%. In table 6, we compare FDropout (USR) and GFR with input and output channels dropped independently and with layer-wise coupling with respect to the previous and following layer. The results show that the client-side accuracy for the layer-wise methods outperforms their independent counterpart by 16% for FDropout and 7% for GFR. Based on these results, we conclude that the layer-wise dependency is necessary to achieve competitive results and follow this principle in our other experiments.

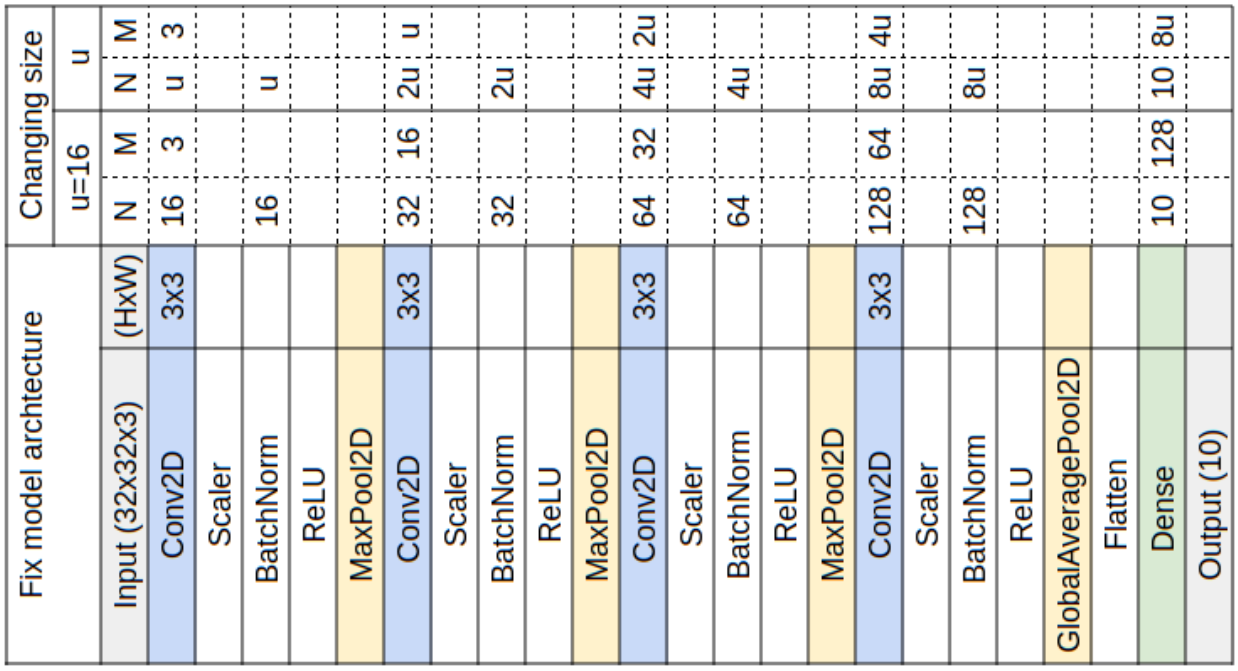


Figure 10. Model architecture and sizes of the weight matrices depending on the model complexity, controlled by the parameter u. Layer names and constant parameter dimensions on the left, varying dimensions on the right.

Table 6. Input-output channels selected independently and with respect to the previous layers in the CNN. FDropout (USR) and GFR experiments on CIFAR-10 with 2 large clients out of 10 clients in total, repeated 3 times. Client-side performance is significantly better when the channel selection is structured layer-wise compared to their independent counterparts. Privacy evaluated with the Yeom attack.

Name	Server		Client average	
	Acc ↑	Adv↓	Acc↑	Adv↓
USR independent	75.44 ± 1.75	2.57 ± 1.62	23.56 ± 0.29	**1.74** ± 0.44
USR layerwise	**76.38** ± 1.36	**2.45** ± 1.34	**39.99** ± 1.11	2.32 ± 1.52
GFR indepentent	76.41 ± 1.73	**2.87** ± 0.78	55.82 ± 1.61	**3.00** ± 0.50
GFR layerwise	**77.11** ± 1.52	3.34 ± 0.98	**62.80** ± 1.43	3.20 ± 0.24

Table 7. Correlations of the three performance attack metrics (AUC, Adv, TPR@FPR 0.1%) on the three MIAs (YEOM, LIRA, TMIA) on the CIFAR-10 dataset. Note how AUC has the largest correlation across the three MIAs. Hence, we use in the experiments the average AUC over the three MIAs as the privacy performance metric.

	Privacy Attack Metrics: AUC / Adv / TPR@FPR 0.1%		
	TMIA	LIRA	YEOM
TMIA	1.00 / 1.00 / 1.00	**0.83** / 0.70 / 0.00	**0.98** / 0.78 / 0.46
LIRA	**0.83** / 0.70 / 0.00	1.00 / 1.00 / 1.00	**0.85** / 0.68 / 0.02
YEOM	**0.98** / 0.78 / 0.46	**0.85** / 0.68 / 0.02	1.00 / 1.00 / 1.00

4.7.3 Experimental setup

In all experiments, we define a heterogeneous FL architecture with 10 clients which are trained with the Adam optimizer, a learning rate of 0.001 for one local epoch, a batch size of 128, and 150 rounds of FL. Experiments are repeated 3 times and we report mean values and standard deviation. The server learns a *large* model, which corresponds to a CNN network with 100k parameters. The clients learn a model with either the same complexity as the server's model or one complexity level below with 30k parameters (*small* model). All models are built in PyTorch [PGC+17] with the Flower federated framework [BTM+20]. FL clients are simulated in parallel on 2 AMD EPYC 7643 48-Core CPUs with 252GB RAM.

We train the 9 heterogeneous FL methods in a FL architecture with 10 clients, of which 2, 5, or 8 clients learn small models and the rest learn models of the same complexity as the server's model. Clients with a smaller dataset size are selected first to learn smaller models.

We also train as baselines two FEDAVG baselines, FedAvg30k and FedAvg100k, where the server and the clients learn models with 30k and 100k parameters, respectively. Thus, we train 29 different FL models for each data distribution.

While we do not perform experiments with heterogeneous FL architectures that include more than two levels of model complexity, we expect our results to extrapolate to other configurations in a similar way as reported in [DDT21].

Membership inference attacks

Each client is subject to the 3 previously described MIAs. For LIRA and TMIA, the auxiliary dataset is drawn from the datasets of the rest of the clients $\mathbb{D}_a^k = \{\mathbb{D}_1, ..., \mathbb{D}_K\} \setminus \mathbb{D}_k$. We use the same shadow models to attack models from the same experiment. We train 16 shadow models for LIRA and use 25 distillation epochs for TMIA.

Performance metrics

We report three performance metrics: (1) client-side and (2) server-side accuracies on the test sets; and (3) the average AUC of the 3 MIAs. We select AUC because it is the metric that exhibited the largest correlation across MIAs on the evaluation datasets. Table 7 depicts the correlation of the three performance metrics (AUC, attack advantage and TPR@FPR 0.1) of the three MIAs (TMIA, LIRA, YEOM) on the CIFAR-10 dataset for illustration.

Table 8. Pearson correlation coefficient of dataset size and attack advantage for different model sizes in class-balanced heterogeneous data distribution.

# Model parameters	30k	100k	400k	1.6M
CIFAR-10	-0.62	-0.65	-0.57	-0.87
CIFAR-100	-0.54	-0.70	-0.85	-0.90
FEMNIST	-0.71	-0.65	-0.63	-0.75

4.8 Results

4.8.1 Model size vs attack advantage

Table 8 shows the Pearson correlation coefficient between the client dataset sizes and their vulnerability against client-side Yeom membership inference attacks. Numbers correspond to running experiments 5 times in a federation with 10 clients. Clients with less than 400 data samples are excluded from the analysis, resulting in the exclusion of 3 clients in the 5 runs with the CIFAR-10 and CIFAR-100 datasets. All values in the table exceed the critical value of non-significant correlation for the given sample size.

4.8.2 Privacy – performance trade-off

Figure 9 depicts the three performance measures of the study, namely client-side accuracy (Y-axis), average attack AUC of the 3 MIAs (X-axis) and server-side accuracy (bar with gradient shading), of the nine heterogeneous FL methods in a federation with 2 *large* clients on the CIFAR-10 (a) and CIFAR-100 (b) datasets. The complete set of results can be found in tables 24 and 25 in the Appendix.

The results corroborate our first hypothesis *H1* related to the accuracy-privacy trade-off. From a client perspective, methods GFM and GFR achieve similar accuracies as HeteroFL but with better privacy protection ($0.5-1.0\%$ AUC). Their overall accuracy-privacy trade-off is similar to that of FedAvg30k yet they achieve significantly better server-side accuracy (**77.89**% and **78.19**% over 69.04% for CIFAR-10 and **43.05**% and **43.17**% over 34.01% for CIFAR-100). FDropout, and the GSR and OSR methods perform well in terms of client privacy, but their client accuracy is significantly lower when compared to the rest of the methods.

Supporting our *H2* hypothesis, methods GFR and GFM, and methods OFR and OFM yield similar results in all three measures on the two datasets, with OFM (HeteroFL) and OFR on CIFAR-100 being the closest with differences of only $0.2\%, 0.6\%$, and 0.02% on the server accuracy, client accuracy, and attack AUC, respectively.

Interestingly, while the OSM method performs as expected on the client side, it outperforms every other FL method on its server-side accuracy, providing the best server accuracy - client privacy trade-off of all the studied methods. We hypothesize that two factors contribute to this result: 1) in category O, all clients train on the same channels in each round, thus the global model does not "forget" the samples of clients not seen in the current round; 2) the training sees the same 4 sets of channels in every 4 rounds, allowing more information to be stored than in only learning on one set (such as in OFM), but still keeping a repeated structure compared to OSR. We leave to future work the investigation of the reasons behind the excellent performance of OSM regarding server accuracy.

Table 9. Absolute differences in performance between the best and worse performing methods in federations with 2, 5, and 8 *large* clients. As the ratio of clients with the same model size as the server increases, the differences in performance between the methods decreases corroborating our third hypothesis.

	CIFAR-10	CIFAR-100
Server Acc (2 clients)	**Δ3.09** (75.61-78.70)	**Δ3.82** (41.31-45.13)
Server Acc (5 clients)	Δ0.56 (78.44-79.00)	Δ1.13 (44.57-45.70)
Server Acc (8 clients)	Δ0.72 (78.38-79.10)	Δ1.69 (44.85-46.54)
Client Acc (2 clients)	**Δ37.65** (32.30-69.95)	**Δ22.18** (10.85-33.02)
Client Acc (5 clients)	Δ20.61 (51.41-72.02)	Δ11.88 (24.27-36.08)
Client Acc (8 clients)	Δ6.63 (67.32-73.96)	Δ3.25 (37.03-40.06)
Attack AUC (2 clients)	**Δ1.52** (50.78-52.30)	**Δ1.85** (51.56-53.41)
Attack AUC (5 clients)	Δ0.87 (51.69-52.56)	Δ1.32 (52.61-53.85)
Attack AUC (8 clients)	Δ0.65 (52.40-53.06)	Δ0.99 (52.91-53.90)

4.8.3 Impact of the number of clients with small model complexity

To evaluate hypothesis *H3*, we perform experiments with heterogeneous federations with 10 clients, of which 2, 5, or 8 clients learn models of the same model complexity as the server's model. Table 9 summarizes the difference in performance between the best and the worst performing methods in each of these federations. Note how the difference in performance between the best and worst performing models in a federation with 2 *large* clients vs a federation with 8 *large* clients is **3x** for the server-side accuracy and attack AUC, and over **6x** for the client-side accuracy. These results support hypothesis *H3*.

4.8.4 Non-IID data

We study the impact of non-IID (non-independent and identically distributed) data using the Federated EMNIST or FEMNIST [CDW+19] dataset, which is an image dataset of handwritten characters. We select this dataset because it is common in both the MIA and FL literature: while the original MNIST dataset is frequently used to evaluate MIAs, [YGFJ18] its federated version (FEMNIST) is commonly used to evaluate FL methods [YBR20; ANA+24]. It consists of 62 classes with a long-tail data distribution. In its federated version, the images are distributed by the ID of the writer whose handwriting they are. Following the official sub-sampling method, we select 20% of the data, keeping only writers with at least 300 samples and splitting into train-test datasets where the test dataset corresponds to images by unseen writers. This results in approximately 165 writers in the train set. We distribute the data among 10 clients following the standard practice in the literature [YBR20]. We do not apply data augmentation on this dataset.

While the previously formulated hypotheses hold in the case of non-IID data on the client-side, the server-side accuracy significantly drops when using heterogeneous FL methods: 2.2 points for FL with 2 *large* clients, and 0.9 points for 5 *large clients*. Furthermore, the FedAvg100k baseline outperforms several of the studied methods regarding client privacy. These results shed light on the limitations of the studied model integration methods in heterogeneous FL and suggest that further research is needed to develop novel heterogeneous FL methods that consider the spurious correlations within the clients [ZMMN21]. The

Appendix contains more details about the experiments with non-IID data, including the results of applying two popular approaches to mitigate the challenges associated with non-IID data. The results, summarized in table 10, illustrate how most methods improve both in accuracy and privacy protection, yet there is no single method that yields the best overall performance, highlighting the importance of considering the model integration strategy in heterogeneous FL settings.

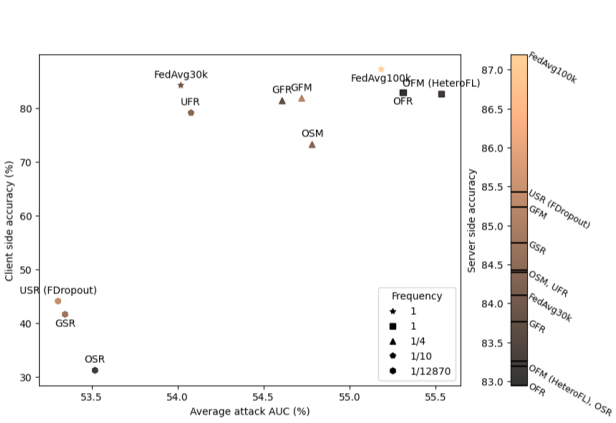
Figure 11 depicts the performance of the 9 model-agnostic FL methods on the FEMNIST dataset for a federation with 2 and 5 *large* (100k parameters) clients and the same experimental setup as that described in the main chapter.

In the case of a federation with 2 large clients (figure 11(a)), we observe significant differences in the server's performance when compared to the results obtained with the CIFAR-10 and CIFAR-100 datasets. Contrary to the CIFAR-x datasets, methods GFR, OSR, OFM, and OFR underperform in terms of server-side accuracy when compared to the FedAvg30k baseline, while methods OFR and OFM underperform in terms of privacy compared to the FedAvg100k baseline. These results suggest that heterogeneous FL architectures where some of the clients learn smaller models can lead to higher privacy risks, highlighting the importance of analyzing the impact of model integration in those settings, as we do in this chapter.

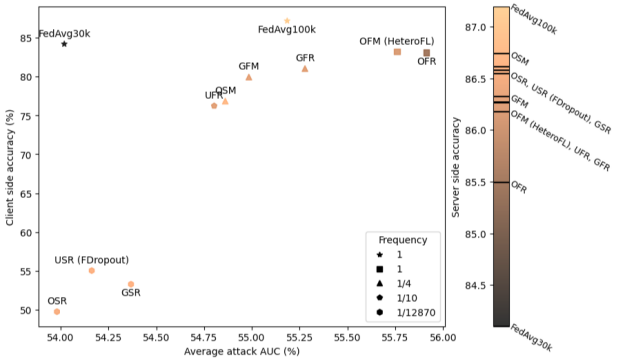
In a federation with 5 large clients (figure 11(b)), the server-side results are more similar to those obtained on the CIFAR-x datasets: method OSM yields the best server-side accuracy and all the methods outperform FedAvg30k. Interestingly and contrary to the behavior on the CIFAR-x datasets, FDropout is competitive with the other methods on its server-side accuracy, yet it yields poor client-side accuracy.

To mitigate the observed decrease in performance and privacy of the models in the case of non-IID data, we have integrated two commonly used methods to tackle non-IIDness in federated learning, namely FedProx [LSZ+20], and FedAvgM [HQB19]. FedProx applies a proximal term in the client loss based on the client model's distance from the server model, which regularizes the client model updates, reducing the differences between client updates and therefore improving client privacy. FedAvgM uses momentum on the server side. It has been shown to be a simple yet effective method to improve performance in non-IID settings. Table 10 summarizes the performance of all the heterogeneous FL approaches with non-IID data after applying FedProx and FedAvgM. As seen in the Table, there is an increase in the server and client accuracy and an improvement in client privacy for most of the heterogeneous FL methods. In addition, other approaches that have been proposed in the literature to mitigate the challenges of non-IID data could also be used, including contrastive loss [LHS21; MSC+23], distillation [ZLF+24], and representation learning [WXX+24]. We leave to future work a more in-depth exploration of such methods as it is out of the main scope of this chapter.

In future work, we also plan to further study the behavior of model-agnostic FL methods on non-IID data. We speculate that these differences in behavior might be due to spurious correlations, which are nonexistent in the CIFAR-x datasets yet present in the FEMNIST dataset, as the writer's style might be correlated to certain classes and clients [ZMMN21].



(a) Performance of the 9 model-agnostic FL methods and baselines on the FEMNIST dataset with 2 *large* (100k parameters) clients. The server-side accuracy of several model-agnostic FL methods is inferior to FedAvg30k's accuracy. Contrary to the results obtained on the CIFAR-x datasets, FDropout is the model-agnostic FL method with the best server-side accuracy.



(b) Performance of the 9 model-agnostic FL methods and baselines on the FEMNIST with 5 *large* (100k parameters) and 5 *small* (30k parameters) clients. The non-IID dataset requires more *large* clients in the federation to outperform the FedAvg30k baseline on the server-side accuracy. The most privacy-sensitive model-agnostic approaches, such as OFM (HeteroFL) yield worse client accuracy-privacy trade-off than the FedAvg100k baseline in this non-IID setting.

Figure 11. Performance of the 9 model-agnostic methods and baselines on the FEMNIST dataset with 2 and 5 *large* (100k parameters) clients. These results suggest that more sophisticated model-agnostic approaches that take into account spurious correlations beyond channel selection strategies are needed.

Table 10. Performance and change in performance when FedProx or FedAvgM are applied to the Federated Learning methods. Improvements are highlighted in bold and the best performing results are underlined. As seen in the Table, most methods improve both in accuracy and privacy protection yet there is no single method that yields the best performance. The table illustrates the impact of the server model integration strategy in heterogeneous FL settings. As observed with IID data, randomness in the channel selection yields better privacy protection and competitive server accuracy at the expense of client accuracy.

FL Method	FedProx($\mu = 0.01$)			FedAvgM ($\eta = 0.9, \beta_1 = 0.3$)		
	↑ Server Acc	↑ Client Acc	↓ MIA (Yeom) AUC	↑ Server Acc	↑ Client Acc	↓ MIA (Yeom) AUC
FedAvg100k	87.37 (**0.17**)	87.46 (**0.28**)	56.19 (0.21)	87.53 (**0.33**)	87.50 (**0.32**)	56.08 (0.10)
FedAvg30k	84.40 (**0.30**)	84.27 (**0.07**)	55.13 (**-0.28**)	84.22 (**0.12**)	84.20 (**0.01**)	54.91 (**-0.51**)
OFM (HeteroFL)	83.09 (-0.10)	82.82 (**0.23**)	56.68 (**-0.06**)	82.97 (-0.22)	82.61 (**0.02**)	56.60 (**-0.14**)
OFR	82.49 (-0.46)	83.08 (**0.16**)	56.92 (0.43)	83.08 (**0.14**)	82.59 (-0.33)	56.26 (**-0.23**)
OSM	84.28 (-0.12)	73.68 (**0.50**)	55.77 (**-0.56**)	84.33 (-0.07)	73.95 (**0.77**)	56.18 (**-0.15**)
GFM	85.40 (**0.17**)	81.82 (**0.01**)	55.89 (**-0.29**)	85.33 (**0.09**)	81.82 (**0.02**)	55.86 (**-0.32**)
GFR	83.92 (**0.15**)	81.59 (**0.26**)	55.89 (**-0.21**)	84.04 (**0.28**)	82.01 (**0.69**)	55.84 (**-0.26**)
UFR	84.67 (**0.24**)	79.58 (**0.46**)	55.45 (**-0.09**)	85.11 (**0.69**)	79.05 (-0.07)	55.55 (0.01)
OSR	83.98 (**0.72**)	32.07 (**0.91**)	54.90 (**-0.24**)	83.71 (**0.46**)	28.93 (-2.24)	55.52 (0.38)
GSR	84.79 (**0.01**)	44.75 (**3.16**)	54.96 (0.01)	84.39 (-0.39)	44.34 (**2.75**)	55.21 (0.26)
USR (FDropout)	85.62 (**0.19**)	46.90 (**2.85**)	54.98 (0.03)	85.29 (-0.14)	45.03 (**0.98**)	54.98 (0.02)

4.9 Conclusion

In this chapter, we have proposed a novel taxonomy of heterogeneous FL methods that not only frames existing approaches into the same family of methods, but also enables the proposal of 7 new methods. In extensive empirical evaluations with the CIFAR-10 and CIFAR-100 datasets, we have studied the server-side accuracy and the client accuracy-privacy trade-off of these approaches when subjected to three commonly used membership inference attacks. Our results show that heterogeneous FL models can be used to mitigate the vulnerability against such attacks. Moreover, the strategy adopted to integrate the clients' models into the server's model impacts both the accuracy and privacy of the federation. By establishing a comprehensive taxonomy and introducing novel methodologies, we pave the way for enhanced privacy of sensitive data within federated learning environments. In the next chapter, we develop a more robust method to consider non-IID data in the clients in FL settings, particularly when there might be spurious correlations present in the datasets.

Chapter 5

Data heterogeneity: FedDiverse, a diversity-driven client selection

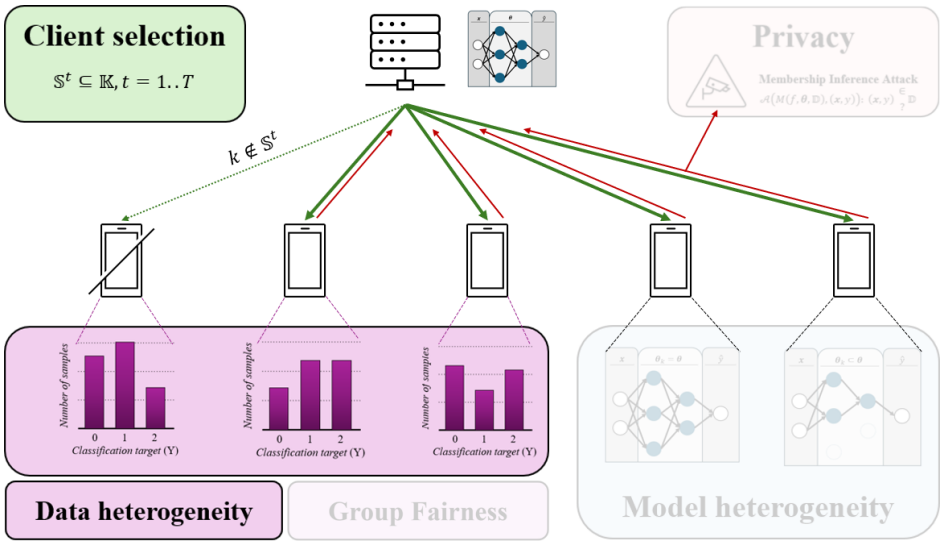


Figure 12. Content of this chapter in the scope of the thesis.

This chapter is based on the research presented in *FedDiverse: Tackling Data Heterogeneity in Federated Learning with Diversity-Driven Client Selection* [NFN+25]. Modifications are applied to improve the flow of the thesis.

5.1 Introduction

The findings of chapter 4 illustrate that the hypotheses based on centralized learning hold in federated learning with IID data. However, for non-IID data, the unexpected behavior suggests that there is an under-explored correlation between the data distribution, model performance, and privacy. To address this, in this chapter, we investigate the concept of spurious correlations – when the model learns misleading correlations from the training data – in the federated setting. To mitigate the effects of spurious correlation in federated learning, we draw inspiration from chapter 3 and propose a client selection method that leverages diversity in the client data to build a more robust FL method.

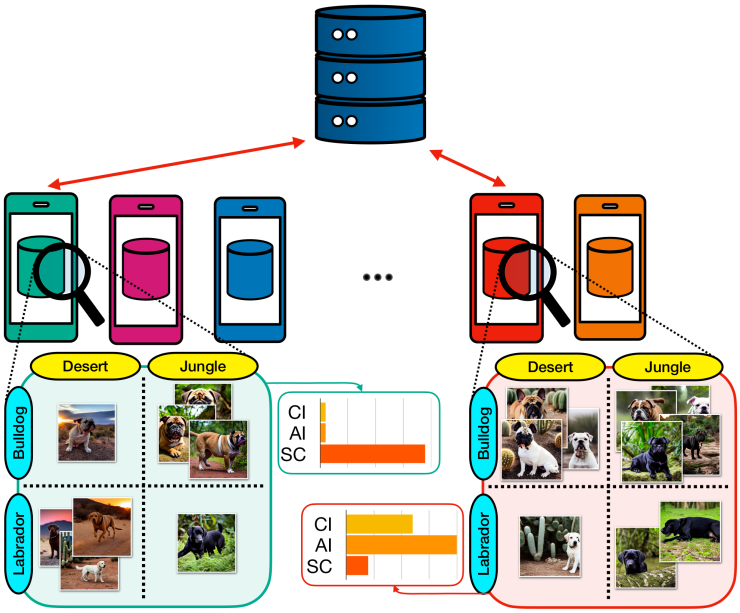


Figure 13. Visual representation of FEDDIVERSE. CI, AI, and SC stand for Class Imbalance, Attribute Imbalance, and Spurious Correlation in the clients' data distributions. Observe the statistical data heterogeneity in the selected clients (turquoise and red). FEDDIVERSE automatically selects clients with a diversity of local statistics to learn a global model that is resilient to statistical data heterogeneity.

In real-world FL scenarios, client data is often shaped by local factors such as differing user behaviors [TYCY22], context-specific data collection environments [FMO20; YAE+18], and socio-economic or cultural biases [BCM+18], resulting in *statistical data heterogeneity*, where data across different clients is non-independent and identically distributed (non-IID) and imbalanced. Statistical data heterogeneity hampers the generalization capabilities of the server's model across clients, slowing convergence and reducing performance [LHY+20; CCC22].

Previous studies in FL have addressed statistical data heterogeneity from an algorithmic perspective, providing convergence theorems, analyzing computational costs, and proposing solutions to mitigate its effects [KKM+20; LHY+20; AZM+21; LSZ+20]. However, there is a lack of fine-grained analyses of this problem. In this chapter, we address this gap and propose decomposing the *attribute-target label* relationships to identify three types of statistical data heterogeneity: (1) *class imbalance* (CI), when target labels have asymmetric distributions; (2) *attribute imbalance* (AI), when attributes exhibit imbalanced distributions; and (3) *spurious correlations* (SC), that emerge when the model learns misleading correlations between a non-discriminative attribute, such as the background, and the target label (see section 2.4). These three types of data heterogeneity pose a challenge both in centralized [YLCT20; YZKG23] and federated [KMA+21; MBB24] learning.

Prior work in centralized machine learning has shown that CI, AI, and SC often arise when data is limited or lacks sufficient diversity [YZC+24; GJM+20]. Thus, a typical solution consists of using an additional and diverse yet unlabeled dataset –called "validation", "target" or "deployment" dataset– to do self-training (e.g. [LHC+21; CWKM20]) or to learn a representation that is invariant to attributes (e.g. [TCK+21]).

In FL, the *diversity* of client data can be leveraged to devise client selection methods that mitigate the effects of CI, AI, and SC. By prioritizing clients with complementary data distributions, the server's model is exposed to diverse training patterns without accessing raw data, enhancing generalization while preserving privacy.

In this chapter, we leverage this idea and address the challenge of statistical data heterogeneity in FL by proposing a novel client selection algorithm called FedDiverse that takes advantage of diversity in client data distributions. We empirically evaluate FedDiverse on 7 computer vision datasets that exhibit varying levels of CI, AI and SC, leading to the following **contributions**:

(1) We propose a fine-grained analysis of statistical data heterogeneity in FL by means of 6 metrics;

(2) We introduce and share 7 FL datasets for binary and multiclass image classification tasks that cover a broad range of statistical data heterogeneity;

(3) We present FedDiverse, illustrated in figure 13, a novel client selection method that is designed to mitigate the impact of statistical data heterogeneity (CI, AI and SC) in FL training while ensuring the privacy of clients and respecting the resource-constrained nature of each client.

5.2 Related work

5.2.1 Data heterogeneity in federated learning

Statistical heterogeneity or non-IID data is a major concern in FL because it can hinder the training process, leading to poor generalization and slow and unstable convergence [KMA+21]. Various methods have been proposed to address this issue [MBB24]. Some approaches add regularization terms to align local updates with the global model, such as FedDyn [AZM+21] and FedProx [LSZ+20], while other methods aim to reduce variance between client updates, such as SCAFFOLD [KKM+20], MOON [LHS21], and FedFM [YNX+23]. In other approaches, the clients share additional information with the server that reveals information about their statistical data heterogeneity. In pow-d [CWJ22], clients share the average loss of the previous global model applied to their local data; in IGPE [ZWL+24] they share averaged network embeddings; and in FedAF [WFK+24] they share synthetic data. Finally, optimization-based server-side methods, such as FedAvgM [HQB19], MIME [KJK+20], and FedOpt [RCZ+20], employ adaptive learning rates at the server to manage statistical diversity among clients.

However, none of these strategies explicitly address the challenge posed by spurious correlations in client data, leaving room for improvement.

5.2.2 Spurious correlations in centralized ML

Spurious correlations can significantly hinder robustness and generalization in machine learning [YZC+24; GJM+20; NAN20]. Proposed solutions to this problem fall into two main categories. The first category [SKHL20; ABGL19; YWL+22] unrealistically assumes that spurious attributes are known or partially labeled, enabling models to reduce reliance on these attributes by re-weighting samples or modifying training processes. These methods

often require that data groups or environments be explicitly defined to minimize spurious dependencies.

The second category does not assume prior knowledge of spurious attributes. Instead, models are designed to automatically distinguish meaningful patterns from spurious ones, often using techniques such as adversarial training [KKK+19; CYZ19] or counterfactual data augmentation [KHL19; WZY+19]. For example, LfF trains two models concurrently: a biased model to capture dataset biases and a debiased one trained on re-weighted samples influenced by the biased model's predictions [NCA+20]; and Just-Train-Twice initially identifies "failure" cases where the model misclassifies, then increases the weights of these cases in a second training phase to improve robustness against spurious features [LHC+21].

Even though spurious correlations have been sparsely studied in the FL literature, recent research has begun to explore this challenge. To the best of our knowledge, [WZNK24] is the first piece of work to tackle spurious correlations in FL by investigating personalization such that models are tailored to the individual clients' data. In contrast, we aim to learn a single global model that remains robust to spurious correlations across all client distributions, achieving strong generalization performance for all clients.

5.2.3 Client selection and weighting in FL

Client selection and client weighting are two primary strategies in FL to manage client contributions during training and mitigate the challenges posed by heterogeneous data [NLQO22]. In client selection, which is especially relevant in resource-constrained settings, only a subset of clients participates in each training round to reduce communication and resource demands, improving training efficiency. Conversely, client weighting includes all clients in each round but adjusts their influence on the global model by means of a weight, aiming to accelerate convergence and performance [CGSY18; DLS21; CWJ22]. Both strategies support fairness [ZFH21; CKMT18] and security [RMLH22; BEGS17], mitigating effects from clients with unreliable or adversarial data.

Client selection or client weighting strategies address the challenge of statistical heterogeneity in FL by prioritizing or scaling the client contributions based on data quality and relevance. In the client selection category, methods like FedPNS [WW22] and pow-d [CWJ22] prioritize clients that are expected to contribute significantly to model accuracy, either through gradient similarity to the average model gradient or by selecting clients whose data produces high loss on the server's model. Fed-CBS aims to reduce the class imbalance by selecting the clients that will generate a more class-balanced grouped dataset [ZLT+23].

Client clustering is a common technique for selecting clients that represent groups that share similar data distributions[6]. Server-side clustering methods typically consider the similarity of the client gradient updates as a proxy of the similarity between their data distributions (e.g., FCCPS [XZLD22]) or their projection into a lower dimension for compression (e.g. HCSFed [SSG+23]). In addition, clients can send metrics that describe the statistical heterogeneity of their local data, such as entropy in HiCS-FL [CV25]. Sharing the full characteristics of the client data distribution with the server has also been investigated [PLY23; WSK+22], yet it could be considered a privacy violation [CV25], and it is

[6]We exclude works referred as clustered federated learning (FL), where each client cluster trains a separate model personalized to the data distribution of that cluster [GTL23; HSF+23], as our aim is to train one robust model shared by all clients.

typically unknown for spurious correlations. Finally, client weighting methods, such as CI-MR [STW19], FMore [ZZWC20] and FedNova [WLL+20], reward clients with high-value data or normalize updates to counter statistical heterogeneity.

Although most existing methods address non-IID data in FL through class imbalance, we study other types of statistical heterogeneity, such as attribute imbalance and spurious correlations, as described next.

5.3 A framework of data heterogeneity in FL

In section 2.4, we introduced the concept of *spurious correlations*, *class imbalance*, and *attribute imbalance* in a dataset $(\boldsymbol{x}, y) \in \mathbb{D}$ given predictor function $f : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{X}$ consists of *task-intrinsic* ($\mathcal{X}_y$) and *attribute* ($\mathcal{X}_a$) features, where class label $y \in \mathcal{Y}$ and sample $\boldsymbol{x} := (\boldsymbol{x}_y, \boldsymbol{x}_a)$ has a discriminative feature $\boldsymbol{x}_y$ and an attribute feature $\boldsymbol{x}_a$, with attribute label $a \in \mathcal{A}$ determined by $\boldsymbol{x}_a$[7]. In this chapter, we propose a way to measure spurious correlations, class imbalance, and attribute imbalance in centralized and federated datasets.

5.3.1 Data heterogeneity metrics

Centralized metrics. To measure the degree of statistical data heterogeneity in dataset $\mathbb{D}$, we adopt the metrics proposed in [YZKG23]:

$$\Delta_{\mathrm{CI}}(\mathbb{D}) = 1 - H(Y)/\log|\mathcal{Y}| \tag{16}$$

$$\Delta_{\mathrm{AI}}(\mathbb{D}) = 1 - H(A)/\log|\mathcal{A}| \tag{17}$$

$$\Delta_{\mathrm{SC}}(\mathbb{D}) = 2I(Y;A)/(H(Y) + H(A)) \tag{18}$$

where

$$H(Y) = -\sum_{y\in\mathcal{Y}} \Pr(y)\log(\Pr(y)) \tag{19}$$

and

$$I(Y;A) = \sum_{y\in\mathcal{Y}}\sum_{a\in\mathcal{A}} \Pr(y,a)\log\left(\frac{\Pr(y,a)}{\Pr(y)\Pr(a)}\right) \tag{20}$$

are the entropy and mutual information with respect to the empirical distribution of the dataset, respectively. Each metric is bounded within $[0, 1]$.

Federated learning metrics. We present six metrics – three global and three local – that characterize statistical data heterogeneity in FL, expanding the previously presented metrics for centralized learning.

Global FL metrics. In the FL context, when the metrics in equations (16) to (18) are computed on the union of the clients' datasets, *i.e.*$\mathbb{D} = \bigcup_{k\in\mathbb{K}} \mathbb{D}_k$, they provide a global understanding of the severity of CI, AI and SC, namely:

$$\textbf{Global class imbalance: } GCI = \Delta_{\mathrm{CI}}(\mathbb{D}) \tag{21}$$

$$\textbf{Global attribute imbalance: } GAI = \Delta_{\mathrm{AI}}(\mathbb{D}) \tag{22}$$

$$\textbf{Global spurious correlation: } GSC = \Delta_{\mathrm{SC}}(\mathbb{D}) \tag{23}$$

[7] In this work, we assume that the labeling of the attribute is not available in the training set.

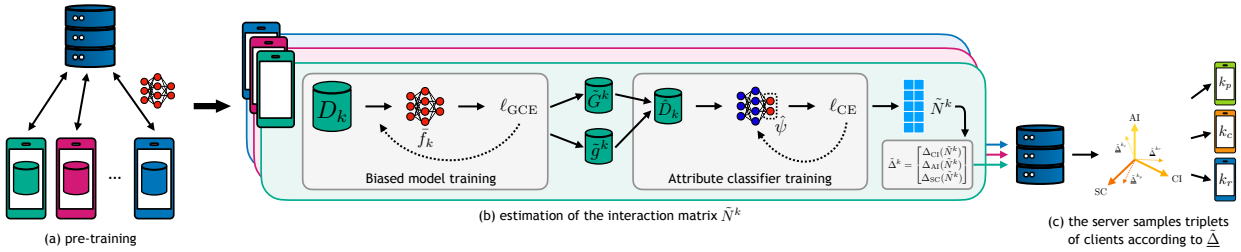


Figure 14. Main steps of FEDDIVERSE. First (a), there is a phase of standard federated model pre-training. Second (b), the clients estimate their interaction matrices and, from them, their data heterogeneity triplets, which they share with the server. Finally, (c), the server uses the received triplets to perform client selection. Learnable parameters are marked in red, while fixed parameters are in blue.

Client FL metrics. The global FL metrics fail to capture the heterogeneity present in the datasets of individual clients. To this end, we propose three additional client metrics, where the local values of CI, AI, and SC are averaged across all the K clients:

$$\textbf{Client class imbalance: } CCI = \frac{1}{K}\sum_{k\in\mathbb{K}} \Delta_{\text{CI}}(\mathbb{D}_k) \tag{24}$$

$$\textbf{Client attribute imbalance: } CAI = \frac{1}{K}\sum_{k\in\mathbb{K}} \Delta_{\text{AI}}(\mathbb{D}_k) \tag{25}$$

$$\textbf{Client spurious correlation: } CSC = \frac{1}{K}\sum_{k\in\mathbb{K}} \Delta_{\text{SC}}(\mathbb{D}_k) \tag{26}$$

In practice, data heterogeneity often consists of a mixture of CI, AI and SC in the data distributions of different clients, as shown in figure 13 where Bulldog/Labrador is the target label and Desert/Jungle as the non-discriminative attribute in the image classification task.

5.4 Client selection via FedDiverse

The proposed FEDDIVERSE method consists of two components, illustrated in figure 14 and described next. First, an approach to estimate the statistical data heterogeneity in the clients, characterized by their local CI, AI, and SC (section 5.4.1). Second, a client selection strategy designed to include diverse clients in each round from the perspective of their statistical data heterogeneity (section 5.4.2).

5.4.1 Estimation of the statistical data heterogeneity

Preliminaries. The global *interaction matrix* $\boldsymbol{R}$ represents the count of samples in a global dataset $\mathbb{D}$ by class $\mathcal{Y}$ and attribute $\mathcal{A}$. For each client k, a local interaction matrix $\boldsymbol{R}^k$ captures its own non-normalized joint distribution of classes and attributes in their dataset $\mathbb{D}_k$, such that $\boldsymbol{R} = \sum_{k\in\mathbb{K}} \boldsymbol{R}^k$. Although clients cannot access the full distribution of their interaction matrices due to unknown attribute distributions, each can compute a marginal interaction vector $\boldsymbol{r}^k \in \mathbb{N}^{|Y|} : \boldsymbol{R}^k_y = \sum_{a\in\mathcal{A}} \boldsymbol{R}^k_{ya}$, where $\boldsymbol{R}^k_{ya}$ are the number of samples belonging to class $y \in Y$ and attribute $a \in A$ in the client's dataset $\mathbb{D}_k$. Therefore, $\boldsymbol{r}^k_y$ contains the distribution of the classes in the dataset $\mathbb{D}_k$.

Table 11. Models in FedDiverse statistical data heterogeneity estimation

	Model name	Training data	Inference
f	Federated model	$\bigcup_{k\in\mathbb{K}} \mathbb{D}_k$	The model trained in FL.
$\bar{f}_k$	Biased model	$\mathbb{D}_k$	Generating $\mathbb{G}^k$ and $\mathbb{g}^k$ if $\lvert\mathcal{Y}\rvert = 2$.
$\bar{f}_k^y$	One-vs-rest model $\forall y \in \mathcal{Y}$	$\{\boldsymbol{x}, y'\} : y' = \begin{cases} 1, & \text{if } \lambda = y \\ 0, & \text{otherwise} \end{cases}, \{\boldsymbol{x}, \lambda\} \in \mathbb{D}_k$	$\{\boldsymbol{x}, y\}$ in $\tilde{\mathbb{G}}_y^k$ if $\bar{f}_k^y$ learns the sample.
$\hat{\psi}$	Attribute classifier	$\{\boldsymbol{x}, \tilde{a}\}, \tilde{a} = \begin{cases} 1, & \text{if } \{\boldsymbol{x}, y\} \in \tilde{\mathbb{G}}_y^k \\ 0, & \text{if } \{\boldsymbol{x}, y\} \in \tilde{\mathbb{g}}_y^k \end{cases}, \{\boldsymbol{x}, y\} \in \mathbb{D}_k, y = \hat{y}$	Generates $\tilde{\boldsymbol{R}}^k$ given "pivot class" $\hat{y}$

The interaction matrix reflects the precise, non-normalized distributions of classes and attributes. In cases with strong spurious correlations, local models may rely on an attribute a which is the most correlated with the class y instead of intrinsic class features. For each client, the *majority group* for a class y, denoted as $\mathbb{G}_y^k$, includes the samples where the attribute a has the highest count in $\boldsymbol{R}^k$, and the *minority group*, $\mathbb{g}_y^k$, includes samples where a has the lowest count. By aggregating these for each class, we define the majority group for client k as $\mathbb{G}^k$ and the minority group as $\mathbb{g}^k$. Because clients do not fully know the attribute function, they estimate these groups.

Finally, under the assumption that there are two attributes ($|\mathcal{A}| = 2$), the attribute set can be defined as $A = \{a_0, a_1\}$. This structure allows clients to infer the minority group attributes for a class y once they know the majority group attribute. Note that most datasets addressing SC or AI problems typically contain only two attributes (see table 2 in [YZKG23]).

Estimation of the interaction matrices. Each client k approximates $\boldsymbol{R}^k$ as $\tilde{\boldsymbol{R}}^k$ and uses this estimated matrix to compute its data heterogeneity triplet (DHT)[8]. To preserve privacy, the clients only share the triplet with the server, which uses these triplets to select clients, as explained in section 5.4.2

$$\tilde{\Delta}^k = [\Delta_{\text{CI}}(\tilde{\boldsymbol{R}}^k), \Delta_{\text{AI}}(\tilde{\boldsymbol{R}}^k), \Delta_{\text{SC}}(\tilde{\boldsymbol{R}}^k)]^\top. \tag{27}$$

In the following, we outline the three-step method adopted by the clients to estimate their interaction matrices $\tilde{\boldsymbol{R}}^k$ and hence their data heterogeneity triplets $\tilde{\Delta}^k$. Note that this estimation is only performed once at the beginning of the FL training process. Table 11 summarizes the models trained in this pipeline.

1. Pre-training: A global pre-training phase is carried out for a small number of rounds T_0 using the FedAvg algorithm, resulting in the global parameters $\boldsymbol{\theta}^{T_0}$.

2. Learning a biased model: After pre-training, each client receives $\boldsymbol{\theta}^{T_0}$ and overfits a local model called a *biased model* $\bar{f}_k$ to its own data using the generalized cross-entropy loss function ℓ_{GCE} [ZS18]. This loss function encourages the model to rely more heavily on easy-to-learn patterns, which are often associated with spurious correlations [NCA+20]. As a result, each client can distinguish between a majority group $\mathbb{G}^k$ (where the majority of correctly predicted samples will belong) and a minority group $\mathbb{g}^k$ (where the incorrectly predicted samples will mainly belong). The predicted majority and minority groups for class y are denoted by $\tilde{\mathbb{G}}_y^k$ and $\tilde{\mathbb{g}}_y^k$, $\forall y \in \mathcal{Y}$, respectively. Given the nature of the ℓ_{GCE} loss, for $|\mathcal{Y}| > 2$, we train one-vs-rest binary classifiers $\bar{f}_k^y$ for each $y \in \mathcal{Y}$ to determine $\tilde{\mathbb{G}}_y^k$ from the correctly predicted samples.

[8]The metrics in equations (16) to (18) can be equivalently calculated using the interaction matrix, as it fully describes the non-normalized joint distribution of classes and attributes.

3. Attribute classifier: Using the biased model, clients label samples in the majority and minority groups, even though they lack information about the exact attribute labels. They identify a "pivot class" which has the smallest difference in sample size between the predicted majority and minority groups, *i.e.* $\hat{y} = \arg\min_{y \in \mathcal{Y}} \left| \left| \tilde{\mathbb{G}}_y^k \right| - \left| \tilde{\mathbb{g}}_y^k \right| \right|$. This class forms a new dataset $\hat{\mathbb{D}}_k$, which contains all the samples in $\mathbb{D}_k$ whose class is $\hat{y}$. Each client then trains an *attribute classifier* $\hat{\psi}$ locally on $\hat{\mathbb{D}}_k$ using cross-entropy loss to predict the attribute labels. This classifier yields an approximate interaction matrix $\tilde{\boldsymbol{R}}^k$ by predicting the attributes according to the attribute labels in $\hat{\mathbb{D}}_k$. Finally, each client computes their approximate DHT $\tilde{\Delta}^k$ and sends it to the server $\mathcal{S}$.

The server collects all the triplets sent by the clients in the *approximate data heterogeneity matrix* $\tilde{\Delta} \in [0,1]^{3 \times K}$ where each column corresponds to one client k and each row corresponds to the CI, AI, and SC components of the clients' $\tilde{\Delta}^k$.

Note that the final values of the scores in $\tilde{\Delta}^k$ are the same independently of the specific labeling choice for the $\hat{\mathbb{D}}_k$ dataset, *i.e.*clients could equivalently assign the attribute label 1 to the majority group samples and 0 to the minority group samples. Moreover, sharing the $\tilde{\Delta}^k$ does not disclose private information from the clients and only incurs negligible additional communication costs. Thus, this approach is suitable for resource-constrained scenarios.

Generalization for multiclass problems

In this section, we further discuss the generalization of the interaction matrix predictor for multiclass problems. Following [NCA+20], we train a biased classifier to identify the strongest correlation with a binary attribute present in the dataset. They used the generalized cross-entropy loss ℓ_{CE}, introduced in [ZS18] as:

$$\ell_{GCE}(\Pr(\boldsymbol{x};\boldsymbol{\theta}), y) = \frac{1 - p \Pr_y(\boldsymbol{x};\boldsymbol{\theta})^q}{q}, \tag{28}$$

where, for $\lim_{q \to 0} \frac{1-\Pr^q}{q} = -\log p$ we get the standard binary cross-entropy

$$\ell_{BCE}(\Pr(\boldsymbol{x};\boldsymbol{\theta}), y) = -(y \log \Pr(\boldsymbol{x};\boldsymbol{\theta}) + (1-y) \log(1 - \Pr(\boldsymbol{x};\boldsymbol{\theta}))). \tag{29}$$

[NCA+20] successfully used ℓ_{GCE} to amplify the bias for binary classification problems. However, for multiclass classification, categorical cross-entropy (ℓ_{CCE}) is used instead of ℓ_{BCE}. ℓ_{CCE} is given by equation (30), where C is the set of positive classes for the sample. Instead of generalizing the categorical cross-entropy ℓ_{CCE}, similar to ℓ_{BCE} being generalized with ℓ_{GCE} by [NCA+20], we chose to keep the biased classifier binary and leverage the demonstrated bias-amplifying ability of ℓ_{GCE}.

$$\ell_{CCE}(\Pr(\boldsymbol{x};\boldsymbol{\theta}), y) = \frac{1}{|M|} \sum_{y \in C} -\log \left(\frac{e^{\Pr_y(\boldsymbol{x},\boldsymbol{\theta})}}{\sum_{j \in Y} e^{\Pr_j(\boldsymbol{x},\boldsymbol{\theta})}} \right), \tag{30}$$

This means that for the biased classifier instead of using one multiclass classification, we train $|Y|$ binary classification models ($\bar{f}_k^b$) such that for the $b \in Y$ selected class

$$y^* = \begin{cases} 1, & \text{if } y = b \\ 0, & \text{otherwise} \end{cases} \tag{31}$$

We construct the $\tilde{\mathbb{G}}_k$ and $\tilde{\mathbb{g}}_k$ class-by-class: if the binary classifier $\bar{f}_k^b$ classified the sample from class b correctly, it counts for the majority ($\tilde{\mathbb{G}}$), otherwise the minority ($\tilde{\mathbb{g}}$).

5.4.2 Client selection

The rationale of FedDiverse is to sample clients with different types of statistical data heterogeneity (CI, AI, and SC) in each round, leveraging it to achieve better generalization and robustness to real-world shifts [GTL23; ZLT+23; PLY23; HSF+23].

FedDiverse's client selection is achieved by leveraging the information in the triplet $\tilde{\Delta}^k$ received from each client and sampling clients to ensure diversity in the three dimensions of the triplets, *i.e.*, selecting clients whose datasets exhibit a variety of CI, AI, and SC. The client selection consists of the following three steps.

1. Probabilistic selection (SC): The first criterion for selecting a client is based on the presence of spurious correlations. The probability distribution p_{SC} over all clients, based on the SC dimension of the data heterogeneity triplet (DHT) $\tilde{\Delta}^k$ is given by: $p_{\text{SC}} = \frac{\tilde{\Delta}_3}{\left\|\tilde{\Delta}_3\right\|_1}, \quad p_{\text{SC}} \in [0,1]^K$, where $\tilde{\Delta}_3$ is a vector composed of the SC values of all clients. The probability of selecting each client is proportional to its corresponding value in p_{SC}.

2. Complementary selection (AI or CI): After selecting a client based on SC, the next step ensures that the next selected client exhibits complementary data heterogeneity. To do so, the server computes the row-normalized matrix $\underline{\tilde{\Delta}}$, where $\underline{\tilde{\Delta}}_i^k = \frac{\tilde{\Delta}_i^k}{\sum_{i=1}^3 \tilde{\Delta}_i^k}, \quad \forall i \in \{1,2,3\}, \forall k \in \mathbb{K}$. Using $\underline{\tilde{\Delta}}$, the server selects the client whose normalized triplet is the least aligned (i.e. has the smallest dot product) with the normalized triplet of the already selected client. Formally, this is computed as: $k_c = \arg\min_{k \in \mathbb{K} \setminus \{k_p\}} \left\langle \underline{\tilde{\Delta}}^{k_p}, \underline{\tilde{\Delta}}^{k} \right\rangle$ where k_p denotes the already selected client and $\langle \cdot, \cdot \rangle$ represents the dot product.

3. Orthogonal selection (CI or AI): The next client k_r is chosen to complement the heterogeneity profile of the data of the clients already selected. To achieve this, the server selects the client whose DHT aligns the most with the vector perpendicular to the DHTs of the two previously selected clients (which represent SC and either CI or AI). Formally, this is computed as: $k_r = \arg\max_{k \in \mathbb{K} \setminus \{k_p, k_c\}} \left\langle \underline{\tilde{\Delta}}^{k_p} \times \underline{\tilde{\Delta}}^{k_c}, \underline{\tilde{\Delta}}^{k} \right\rangle$ where $(\cdot \times \cdot)$ is the cross product, ensuring that the selected client exhibits heterogeneity in the remaining dimension.

This client selection approach leverages all three dimensions of the DHT by selecting clients with different types of data heterogeneity. The server repeats the steps above iteratively until the desired number of clients has been selected, excluding clients already chosen in the current round. To enhance variability, the order in which dimensions (SC, CI, AI) are prioritized is rotated every three clients.

As illustrated in the experimental section, FedDiverse's client selection can be applied in conjunction with any FL optimization approach.

5.5 Experiments

5.5.1 Datasets

We perform the experimental evaluation taking as a basis three computer vision datasets that are commonly used for benchmarking algorithms in the presence of statistical data

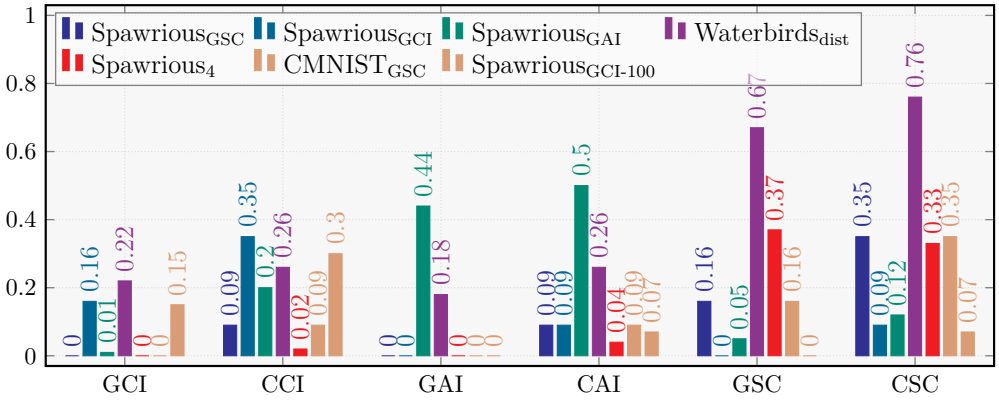


Figure 15. Global and client statistical data heterogeneity metrics of each of the proposed datasets. Note how each dataset has different values of class imbalance, attribute imbalance and spurious correlations both globally and in the clients.

heterogeneity. From these three base datasets, we create 7 different datasets that cover a wide variety of CI, AI, and SC both globally and in the clients, as explained next and reflected in figure 15.

WaterBirds The WaterBirds dataset [WBW+11] is an image classification dataset with two classes (*waterbirds* and *landbirds*), and two background attributes (*water* and *land*). In the training set, there is a spurious correlation where waterbirds are more often found on water backgrounds, and landbirds are more often seen on land backgrounds. We follow the original train/test split and distribute the training data over 30 clients as follows: 3 clients predominantly have CI; 2 clients have mostly AI; and the rest of the clients are impacted largely by the same SC as the global dataset.

Spawrious The Spawrious dataset [LDKS23] consists of 4 dog breeds (target labels y) on 6 backgrounds (attributes a) groups generated with Stable Diffusion v1.4 [RBL+22]. There are 6,336 images for each (y, a) pair, making it the largest vision dataset where the level of spurious correlation is adjustable [YZC+24]. We save 10% of the data to create a balanced test set and use the remaining data to generate 5 federated datasets with various levels of statistical data heterogeneity. We identify and use the 2 hardest background groups (namely *beach* and *snow*) together with 2 (*labrador* and *dachshund*) or 4 (*labrador*, *dachshund*, *bulldog*, and *corgi*) dog breed classes.

While the WaterBirds dataset contains CI, AI and SC (see figure 15), we create 5 Spawrious datasets with different data distributions to investigate the impact of CI, AI, and SC individually:

First, we create 3 datasets where only one type of data heterogeneity is present globally: spurious correlation in Spawrious$_{GSC}$; class imbalance in Spawrious$_{GCI}$; and attribute imbalance in Spawrious$_{GAI}$. Second, we create Spawrious$_4$ which contains high levels of spurious correlation and 4 classes. Third, we create Spawrious$_{\text{GCI-100}}$ with class imbalance and 100 clients.

CMNIST The CMNIST dataset [ABGL19] is generated based on the binarized MNIST dataset, with labels $y = 0$ for digits less than five and $y = 1$ otherwise. The attribute is given by the foreground color, $\mathcal{A} = \{\text{red}, \text{green}\}$. We use the same data distribution as for Spawrious$_{GSC}$ with 2 classes. Hence, in this dataset, there is a high level of global and client spurious correlations.

5.5.2 Data distributions

In this section, we include additional details on the proposed FL datasets. To construct each dataset, we first define the global interaction matrix (given for WaterBirds) such that a centralized ERM training has at least 3.2% drop between worst group and average accuracy. This ensures that the statistical data heterogeneity will have an impact on the training. Table 12 summarizes the results on the centralized version of the datasets.

In table 13, we show the data distribution between clients. Each cell of the table shows a type of client: $m \times \boldsymbol{R}^k, \Delta^k$, where m is the number of clients of that type, $\boldsymbol{R}^k$ is the interaction matrix of that type of client and Δ^k is the data heterogeneity triplet (DHT) of that type. Note that the different datasets are designed for federations with different numbers of clients such that the overall imbalance in the dataset size among clients remains small.

5.5.3 Experimental setup

We simulate a federated learning scenario with a total of 24 to 100 clients depending on the dataset [9] on a machine with 3 Nvidia A100-80G GPUs using both the Flower [BTM+20] and PyTorch [PGM+19] frameworks. Our code is available at `https://github.com/Erosinho13/SpuriousFL`.

The server and the clients trained a MobileNet v2 [How17] model, where batch normalization layers were replaced with group normalization layers and initial weights were pre-trained on Imagenet. We applied the *categorical crossentropy* loss function with 0.001 learning rate and a batch size of 28. Unless otherwise noted, we used $T = 200$ rounds of federated training with equally weighted clients. In experiments without client selection, all clients (24 to 100) participate in the federation in every round. In the cases where client selection is performed, the server selects 9 clients to participate in the federation in each round, except for $\text{Spawrious}_{\text{GCI-100}}$ where 12 clients are selected.

We performed all experiments on the previously described datasets. We report *worst-group accuracy* [SKHL20] and its standard deviation, defined as

$$\min_{(y,a)\in\mathcal{Y}\times\mathcal{A}} \mathbb{E}[\mathbb{1}\{y = f(x;\boldsymbol{\theta})\} \mid Y = y, A = a] \tag{32}$$

over 3 runs using a balanced global test dataset.

5.5.4 Baselines

We compare FedDiverse's client selection strategy with 6 baselines, described below. All the methods are implemented using server-side momentum FedAvgM [HQB19].

1. Uniform random selection, where clients are randomly selected according to a uniform distribution.

2. Round robin selection, where the server keeps track of how many times R_k a client k has been selected such that the client cannot participate again while $\exists j \neq k, R_j < R_k$.

[9] The federations with the $\text{Spawrious}_{\text{GSC}}$, $\text{Spawrious}_{\text{GCI}}$ and $\text{CMNIST}_{\text{GSC}}$ datasets have 24 clients; $\text{Spawrious}_{\text{GAI}}$ and Spawrious_4 have 25 clients; WaterBirds_{dist} has 30 clients; and $\text{Spawrious}_{\text{GCI-100}}$ has 100 clients.

Table 12. Details of the global data distributions of the datasets. Performance measured by training MobileNetV2 for 10 epochs with SGD

Dataset	(1) $\text{Spawrious}_{\text{GSC}}$, (2) $\text{CMNIST}_{\text{GSC}}$	$\text{Spawrious}_{\text{GCI}}$	$\text{Spawrious}_{\text{GAI}}$	$\text{WaterBirds}_{\text{dist}}$	Spawrious_4
Interaction matrix ($\boldsymbol{R}$)	$\begin{bmatrix}1760 & 640\\ 640 & 1760\end{bmatrix}$	$\begin{bmatrix}1760 & 1760\\ 640 & 640\end{bmatrix}$	$\begin{bmatrix}2000 & 500\\ 2000 & 100\end{bmatrix}$	$\begin{bmatrix}3498 & 184\\ 56 & 1057\end{bmatrix}$	$\begin{bmatrix}2000 & 200\\ 2000 & 200\\ 200 & 2000\\ 200 & 2000\end{bmatrix}$
Average accuracy (%)	(1) 93.15, (2) 96.35	92.92	93.9	82.37	93.07
Worst group accuracy(%)	(1) 87.82, (2) 93.15	87.62	89.09	55.09	85.88
Class imbalance (CI)	0	0.16	0.01	0.22	0
Attribute imbalance (AI)	0	0	0.44	0.18	0
Spurious correlation (SC)	0.16	0	0.05	0.67	0.37

Table 13. Client interaction matrices and ground-truth triplets for the proposed FL data distributions. Each cell of the table contains a client interaction matrix in the middle, the number of clients with that matrix on the left, and the CI, AI, and SC values of the matrix on the right.

$\textbf{Spawrious}_{\textbf{GSC}}$, $\textbf{CMNIST}_{\textbf{GSC}}$	$\Delta_{\substack{\text{CI}\\ \text{AI}\\ \text{SC}}}$	$\textbf{Spawrious}_{\textbf{GCI}}$	$\Delta_{\substack{\text{CI}\\ \text{AI}\\ \text{SC}}}$	$\textbf{Spawrious}_{\textbf{GAI}}$	$\Delta_{\substack{\text{CI}\\ \text{AI}\\ \text{SC}}}$	$\textbf{WaterBirds}_{\textbf{dist}}$	$\Delta_{\substack{\text{CI}\\ \text{AI}\\ \text{SC}}}$	$\textbf{Spawrious}_{\textbf{4}}$	$\Delta_{\substack{\text{CI}\\ \text{AI}\\ \text{SC}}}$	$\textbf{Spawrious}_{\textbf{GCI-100}}$	$\Delta_{\substack{\text{CI}\\ \text{AI}\\ \text{SC}}}$
$2\times\begin{bmatrix}90 & 90\\ 10 & 10\end{bmatrix}$	0.53 0.00 0.00	$15\times\begin{bmatrix}90 & 90\\ 10 & 10\end{bmatrix}$	0.53 0.00 0.00	$1\times\begin{bmatrix}120 & 5\\ 20 & 10\end{bmatrix}$	0.29 0.54 0.15	$1\times\begin{bmatrix}23 & 23\\ 10 & 110\end{bmatrix}$	0.15 0.28 0.18	$2\times\begin{bmatrix}20 & 20\\ 20 & 20\\ 5 & 5\\ 5 & 5\end{bmatrix}$	0.14 0.00 0.00	$64\times\begin{bmatrix}68 & 68\\ 10 & 10\end{bmatrix}$	0.45 0.00 0.00
$2\times\begin{bmatrix}10 & 10\\ 90 & 90\end{bmatrix}$	0.53 0.00 0.00	$1\times\begin{bmatrix}10 & 10\\ 90 & 90\end{bmatrix}$	0.53 0.00 0.00	$1\times\begin{bmatrix}120 & 40\\ 5 & 10\end{bmatrix}$	0.58 0.14 0.07	$1\times\begin{bmatrix}110 & 23\\ 10 & 23\end{bmatrix}$	0.28 0.15 0.18	$2\times\begin{bmatrix}5 & 5\\ 5 & 5\\ 20 & 20\\ 20 & 20\end{bmatrix}$	0.14 0.00 0.00	$4\times\begin{bmatrix}10 & 10\\ 68 & 68\end{bmatrix}$	0.45 0.00 0.00
$2\times\begin{bmatrix}90 & 10\\ 90 & 10\end{bmatrix}$	0.00 0.53 0.00	$2\times\begin{bmatrix}90 & 10\\ 90 & 10\end{bmatrix}$	0.00 0.53 0.00	$2\times\begin{bmatrix}170 & 5\\ 5 & 5\end{bmatrix}$	0.70 0.70 0.24	$1\times\begin{bmatrix}89 & 39\\ 1 & 29\end{bmatrix}$	0.30 0.01 0.27	$2\times\begin{bmatrix}20 & 5\\ 20 & 5\\ 20 & 5\\ 20 & 5\end{bmatrix}$	0.00 0.28 0.00	$8\times\begin{bmatrix}68 & 10\\ 68 & 10\end{bmatrix}$	0.00 0.45 0.00
$2\times\begin{bmatrix}10 & 90\\ 10 & 90\end{bmatrix}$	0.00 0.53 0.00	$2\times\begin{bmatrix}10 & 90\\ 10 & 90\end{bmatrix}$	0.00 0.53 0.00	$2\times\begin{bmatrix}5 & 5\\ 170 & 5\end{bmatrix}$	0.70 0.70 0.24	$1\times\begin{bmatrix}29 & 39\\ 1 & 89\end{bmatrix}$	0.01 0.30 0.27	$2\times\begin{bmatrix}5 & 20\\ 5 & 20\\ 5 & 20\\ 5 & 20\end{bmatrix}$	0.00 0.28 0.00	$8\times\begin{bmatrix}10 & 68\\ 10 & 68\end{bmatrix}$	0.00 0.45 0.00
$15\times\begin{bmatrix}90 & 10\\ 10 & 90\end{bmatrix}$	0.00 0.00 0.53	$2\times\begin{bmatrix}90 & 10\\ 10 & 90\end{bmatrix}$	0.00 0.00 0.53	$2\times\begin{bmatrix}10 & 30\\ 120 & 10\end{bmatrix}$	0.21 0.21 0.38	$1\times\begin{bmatrix}81 & 35\\ 9 & 31\end{bmatrix}$	0.18 0.02 0.14	$1\times\begin{bmatrix}5 & 20\\ 5 & 20\\ 20 & 5\\ 20 & 5\end{bmatrix}$	0.00 0.00 0.19	$8\times\begin{bmatrix}68 & 10\\ 10 & 68\end{bmatrix}$	0.00 0.00 0.45
$1\times\begin{bmatrix}10 & 90\\ 90 & 10\end{bmatrix}$	0.00 0.00 0.53	$2\times\begin{bmatrix}10 & 90\\ 90 & 10\end{bmatrix}$	0.00 0.00 0.53	$2\times\begin{bmatrix}80 & 80\\ 20 & 2\end{bmatrix}$	0.47 0.01 0.08	$9\times\begin{bmatrix}126 & 1\\ 1 & 31\end{bmatrix}$	0.28 0.28 0.87	$7\times\begin{bmatrix}119 & 5\\ 119 & 5\\ 5 & 119\\ 5 & 119\end{bmatrix}$	0.00 0.00 0.50	$8\times\begin{bmatrix}10 & 68\\ 68 & 10\end{bmatrix}$	0.00 0.00 0.45
				$14\times\begin{bmatrix}80 & 15\\ 90 & 2\end{bmatrix}$	0.00 0.56 0.06	$16\times\begin{bmatrix}127 & 1\\ 1 & 31\end{bmatrix}$	0.28 0.28 0.87	$9\times\begin{bmatrix}118 & 5\\ 118 & 5\\ 5 & 118\\ 5 & 118\end{bmatrix}$	0.00 0.00 0.50		
				$1\times\begin{bmatrix}110 & 5\\ 85 & 8\end{bmatrix}$	0.01 0.66 0.01						

3. FedNova [WLL+20], a client weighting approach by means of importance weighting. The parameter aggregation is given by $\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t - \tau_{eff} \sum_k \frac{|\mathbb{D}_k|}{|\mathbb{D}|} \cdot \beta \nabla_k^{t+1}$, where β is the same momentum as in FedAvgM and τ_{eff} is the effective iteration step and it is computed from the client's steps.

4. pow-d [CWJ22], a loss-based selection method. First, the server $\mathcal{S}$ selects $\kappa_{br} : \kappa < \kappa_{br} < K$ clients randomly to broadcast the model parameters $\boldsymbol{\theta}^t$. All $k \in \mathbb{S}_{\kappa_{br}}$ clients compute $\ell(\boldsymbol{\theta}^t, \mathbb{D}_k)$ and report it back to the server. Then, the server sorts the clients such that for $i, j \in \{1, \ldots, K\}, i < j \rightarrow \ell(\boldsymbol{\theta}^t, \mathbb{D}_i) < \ell(\boldsymbol{\theta}^t, \mathbb{D}_j)$ and selects the first κ clients to participate in the computation of $\boldsymbol{\theta}^{t+1}$.

5. FedPNS [WW22] identifies clients that negatively impact the aggregated gradient change by comparing a client's gradient change ∇_k^{t+1} with the overall gradient change excluding that client, $\nabla^{t+1} - \nabla_k^{t+1}$. If a client slows down the aggregated gradient, as indicated by $\langle \nabla^{t+1}, (\nabla^{t+1} - \nabla_k^{t+1}) \rangle$, the client is flagged. Flagged clients are less likely to be selected in subsequent rounds, while non-flagged clients and those not sampled in round t are more likely to be selected.

6. HCSFed [SSG+23] clusters the clients based on the compressed gradients after the first round of training. We use 3 clusters and randomly select clients from each cluster.

5.5.5 Communication and computation overhead

The baselines have varying levels of communication and computation overhead reported in table 14. FedNova performs client weighting instead of selection, hence, all the clients participate in the federation in each round. While pow-d performs client selection, the server needs $\ell(\boldsymbol{\theta}^t, \mathbb{D}_k)$ from all clients to determine which clients to select in each round. FedPNS requires no additional work from the clients, but the server calculates the similarity between the client gradient updates in every round, which can result in significant overhead for complex models and large number of clients. HCSFed addresses this issue by compressing the model gradients and organizing the clients into clusters after the first training round and minimizing the overhead for subsequent rounds. Uniform random, Round robin and FedDiverse are the only three client selection methods where **only the participating clients** perform computations and communicate with the server in each round. FedDiverse's additional communication overhead is limited to just 3 scalar values per client while the client-side computational overhead occurs only in a single training round. The only recurring overhead is the server-side selection, which involves sorting clients based on their DHT values.

5.5.6 Results

Table 15 depicts the worst group accuracies for FedDiverse and all the baselines on the 7 datasets. Note how client selection with FedDiverse is the *only method* that yields competitive performance across all datasets.

5.5.7 Benchmarking FedDiverse with FL methods

We evaluate FedDiverse's ability to improve the robustness of existing FL optimization algorithms when combined with them. We aim to (1) evaluate the ability of FedDiverse's

Table 14. Communication and computation overhead for FedDiverse and the baselines where $K = 24..100, r = 10^{-5}, |\boldsymbol{\theta}| = 2.23 \cdot 10^6, n_k = 10^2..10^3$

Method	Frequency	Communication Overhead	Computation Overhead Client	Computation Overhead Server ($\forall t$)
FedDiverse	$t = 1$	3	$\forall k \in \mathbb{K} : \lvert Y \rvert O(n_k \lvert \boldsymbol{\theta} \rvert)$	$O(K)$
Round Robin	0	0	0	$O(1)$
FedNova	$\forall t$	$3 + \forall k \notin \mathbb{K} : \boldsymbol{\theta}_k^t$	$O(1) + \forall k \notin \mathbb{K} : O(n_k \lvert \boldsymbol{\theta} \rvert)$	$O(K)$
pow-d	$\forall t$	$1 + \forall k \notin \mathbb{K} : \boldsymbol{\theta}_k$	$\forall k \notin \mathbb{K} : O(n_k \lvert \boldsymbol{\theta} \rvert)$	$O(K \log K)$
FedPNS	0	0	0	$O(K^2 \lvert \boldsymbol{\theta} \rvert^2)$
HCSFed	$t = 1$	$r\boldsymbol{\theta}_k$	$r \lvert \boldsymbol{\theta} \rvert^2$	$t = 1 : O(K \cdot r \lvert \boldsymbol{\theta} \rvert)$ $t \neq 1 : O(K)$

Table 15. Worst group accuracies (mean and std) over three experiments of FedDiverse and the baselines in a federation with 24 to 100 clients, with 9 clients selected every round, and FedAvgM as the FL optimization algorithm. The best-performing method is highlighted with **bold**, and the second best is underlined. (*): 12 clients selected from 100. (**): Not scalable due to excessive computational cost.

Client Selection algorithm	**Dataset** Spawrious$_{\text{GSC}}$	Spawrious$_{\text{GCI}}$	Spawrious$_{\text{GAI}}$	WaterBirds$_{\text{dist}}$	Spawrious$_4$	CMNIST$_{\text{GSC}}$	Spawrious$_{\text{GCI-100}}$*
FedDiverse	<u>88.01</u> ± 0.96	**89.91** ± 1.91	**87.28** ± 1.61	<u>54.10</u> ± 2.03	**86.06** ± 0.58	**94.01** ± 0.98	**91.22** ± 1.61
Uniform random	86.27 ± 1.12	87.59 ± 2.00	85.86 ± 2.56	42.42 ± 0.59	84.02 ± 0.63	92.00 ± 1.61	86.96 ± 1.28
Round robin	87.12 ± 0.87	87.64 ± 0.90	86.17 ± 2.65	41.23 ± 2.18	83.54 ± 1.83	<u>93.51</u> ± 0.49	85.54 ± 0.40
FedNova	87.49 ± 0.73	88.52 ± 1.49	<u>87.22</u> ± 0.47	42.83 ± 0.71	84.65 ± 0.64	93.23 ± 0.34	87.33 ± 0.18
pow-d	**89.12** ± 0.32	<u>89.01</u> ± 1.18	86.91 ± 1.52	**56.75** ± 2.49	83.54 ± 2.01	92.85 ± 0.47	<u>89.85</u> ± 1.00
FedPNS	85.75 ± 1.34	85.02 ± 9.12	82.22 ± 6.94	48.75 ± 12.14	84.35 ± 1.45	91.49 ± 1.42	N/A**
HCSFed	86.80 ± 0.86	87.17 ± 0.27	85.96 ± 2.70	41.66 ± 1.80	<u>85.59</u> ± 0.66	91.45 ± 1.11	85.49 ± 0.78

Table 16. Worst group accuracies (mean and std) over three experiments of FedDiverse combined with four FL optimization methods on the proposed datasets vs the default random selection. The best-performing client selection method is highlighted in bold and the best-performing combination is underlined. Note how all the FL optimization algorithms improve their performance when doing client selection with FedDiverse vs random selection across all datasets.

FL algorithm	Spawrious$_{\text{GSC}}$		Spawrious$_{\text{GCI}}$		Spawrious$_{\text{GAI}}$		WaterBirds$_{\text{dist}}$		Spawrious$_4$		CMNIST$_{\text{GSC}}$	
	Random	FedDiverse	Random	FedDiverse	Random	FedDiverse	Random	FedDiverse	Random	FedDiverse	Random	FedDiverse
FedAvg	85.09 ± 1.00	**85.90** ± 1.62	85.65 ± 3.85	**89.43** ± 0.63	80.49 ± 0.52	**84.33** ± 1.51	31.72 ± 3.05	**46.47** ± 1.31	81.07 ± 1.29	**83.86** ± 1.20	87.58 ± 2.38	**91.01** ± 0.58
FedAvgM	86.27 ± 1.12	**88.01** ± 0.96	87.59 ± 2.00	**89.91** ± 1.91	85.86 ± 2.56	**<u>87.28</u>** ± 1.61	42.42 ± 0.59	**<u>54.10</u>** ± 2.03	**<u>84.02</u>** ± 0.63	**86.06** ± 0.58	92.00 ± 1.61	**<u>94.01</u>** ± 0.98
FedProx	84.43 ± 1.91	**86.33** ± 1.49	82.91 ± 4.89	**87.30** ± 3.01	81.39 ± 2.12	**83.81** ± 2.46	31.57 ± 2.87	**43.51** ± 0.70	80.44 ± 1.55	**83.64** ± 0.74	91.36 ± 0.95	**91.49** ± 1.86
FedAvgM + FedProx	85.41 ± 1.67	**<u>87.85</u>** ± 1.26	88.38 ± 1.42	**<u>90.48</u>** ± 1.61	**85.65** ± 3.76	85.17 ± 1.79	44.29 ± 1.26	**53.84** ± 0.90	82.97 ± 0.69	**<u>86.12</u>** ± 0.97	92.42 ± 0.71	**93.24** ± 0.38

client selection method to improve performance across a variety of datasets and FL optimization algorithms; and (2) shed light on which method yields the best performance. The algorithms benchmarked in this section are:

1. FedAvg [MMR+17], which serves as the baseline FL method where in each round the global model is replaced by the average of the client models.

2. FedAvgM [HQB19], which includes server-level momentum, inspired by the momentum algorithm [Nes13]. It is designed to improve non-IID convergence. The momentum parameter is set to $\beta = 0.95$.

3. FedProx [LSZ+20], where the client loss contains a proximal term derived from the difference between server and client weights to stabilize the convergence:

$$\ell_{prox}(f_k(\boldsymbol{x};\boldsymbol{\theta}_k), y) = \ell(f_k(\boldsymbol{x};\boldsymbol{\theta}_k), y) + \frac{\mu}{2}||\boldsymbol{\theta} - \boldsymbol{\theta}_k||_2 \tag{33}$$

where μ is a parameter set to 0.1 in our experiments.

As FedAvgM changes the server aggregation method, FedProx the local loss function, and FedDiverse the client selection policy, we can use any combination of the 3 methods to mitigate statistical data heterogeneity. As reflected in table 16, FedDiverse improves the performance over random selection when combined with every FL method and in all datasets. The combination of FedDiverse with FedAvgM yields very competitive performance and hence we opt for FedAvgM as the FL optimization method to be used in all of the experiments.

5.6 Ablation study

In this section, we study the performance of FedDiverse on the WaterBirds dataset and under different configurations, reflected in table 17. We compare 3 scenarios:

1. Our realistic setup, where the interaction matrix $\tilde{\boldsymbol{R}}^k$ and the data heterogeneity triplets $\tilde{\Delta}^k$ are estimated;

2. An ideal –yet unrealistic– scenario where the interaction matrix $\boldsymbol{R}^k$ and therefore the triplets Δ^k are known to the server; and

3. A method where the full interaction matrix $\boldsymbol{R}^k$ is sent to the server instead of the triplets. In this case, the server first computes the client weights ω_k that minimize the variance of the matrix $\boldsymbol{S} = \sum_{k\in\mathbb{K}} \omega_k \boldsymbol{R}^k$, *i.e.*, $\min \text{Var}(\boldsymbol{S}) = \frac{1}{|Y||A|} \sum_{y\in Y} \sum_{a\in A} (s_{y,a} - \nu)^2$, where ν is the average number of samples per (y, a) groups. We solve it as a convex optimization problem and use the ω_k weight as the probability to sample client k. Note that this method would raise privacy concerns. We call this method FedRK to highlight that the full interaction matrix ($\boldsymbol{R}^k$) has to be shared with the server.

Furthermore, we evaluate the impact of increasing the number of pre-training steps and compare FedDiverse when combined with FedAvg and FedAvgM.

As seen in the table, perfect knowledge of $\boldsymbol{R}^k$ could yield an increase of up to 5.71 and 3.74 points in worst group accuracy with FedAvg and FedAvgM, respectively. Communicating the true (typically unknown) interaction matrix instead of the triplets could add up to 4.15 and 4.78 points to the worst-group accuracy with FedAvg and FedAvgM, respectively. Increasing the number of pre-training steps is only helpful with FedAvgM, yet the performance gains are not significant.

5.6.1 Comparison with different architectures

In table 18, we report experiments with ResNet50 [HZRS16] on the WaterBirds dataset. Note that computer vision models used in FL scenarios are typically smaller than a ResNet50 [HQB19; LSZ+20]. However, the spurious correlation literature in centralized machine learning uses this model in the reported benchmarks [YZKG23]. Thus, we include this experiment for completeness. As we can observe, FedDiverse improves the performance also on the ResNet50.

5.6.2 Pre-training with a different number of rounds

In table 19 we summarize the experimental results obtained when increasing the number of pre-training rounds before using the FedDiverse algorithm to determine the values of the clients' DHTs. Note how using $T_0 = 1$ yields similar results to using more pre-training rounds (with full participation). Thus, we keep $T_0 = 1$ in all experiments to reduce the computation and communication costs.

5.6.3 Sensitivity analysis of the hyper-parameters of the FedDiverse algorithm

To determine the right hyper-parameters for FedDiverse, we conducted an experiment on the $\text{Spawrious}_{\text{GSC}}$ dataset by changing the following 3 hyper-parameters: the training steps of the *biased model* $\tau_{biased} = \{5, 25, 50, 75, 100\}$, the training steps of the *attribute classifier* $\tau_{attr} = \{5, 25, 50, 75, 100\}$, and the q value of the *generalized cross-entropy loss* $q = \{0.1, 0.3, 0.5, 0.7, 0.9\}$. We performed an exhaustive grid search on these values. Figure 16 summarizes the results of this sensitivity analysis. We report the Euclidean distance between the predicted and true DHT values, thus the best parameters correspond to the smallest distance

$$\min_{\tau_{biased}, \tau_{attr}, q} \operatorname*{avg}_{k \in 1..K} ||\tilde{\Delta}^k - \Delta^k|| \tag{34}$$

In conclusion, we use $\tau_{biased} = 50$, $\tau_{attr} = 10$, and select $q = 0.3$ as the best q value for the given τ parameters.

5.6.4 Analysis of FedDiverse's sampling strategy

Figure 17 show the distribution of clients sampled per round as per the FedDiverse client selection algorithm (as specifically described in algorithms 5 and 6) on the $\text{Spawrious}_{\text{GSC}}$, $\text{Spawrious}_{\text{GCI}}$, $\text{Spawrious}_{\text{GAI}}$, Spawrious_{4} and $\text{WaterBirds}_{\text{dist}}$ datasets, respectively. Note that the simulation conducted on figure 17 a) can equivalently be considered as for the $\text{CMNIST}_{\text{GSC}}$ dataset, since both $\text{CMNIST}_{\text{GSC}}$ and $\text{Spawrious}_{\text{GSC}}$ have the same clients distributions.

The figures are based on a simulation where 9 clients are sampled per round over 20 training rounds, following the setup described in the main chapter. Each client is assigned a type –CI, AI, or SC– based on the highest values of their corresponding metrics, as detailed in table 13. Clients are then sorted by type, with CI clients having the lowest IDs, followed by AI and SC clients. The background color in the figures represents the client type while the percentage in the background indicates the average selection rate for that type across all 20

Table 17. Ablation study of FedDiverse with different configurations on the WaterBirds dataset.

Method	Pre-training (T_0)	Interaction matrix	Message	Worst group accuracy (%) FedAvg	Worst group accuracy (%) FedAvgM
FedDiverse	20	predicted	DHT($\tilde{\Delta}^k$)	44.03 ± 0.32	55.04 ± 3.09
	1	predicted	DHT($\tilde{\Delta}^k$)	46.47 ± 1.31	54.10 ± 2.03
	20	known	DHT(Δ^k)	48.08 ± 3.27	58.00 ± 0.77
	1	known	DHT(Δ^k)	50.62 ± 3.00	58.88 ± 2.57
FedRK	20	known	$\boldsymbol{R}^k$	49.74 ± 2.30	58.41 ± 2.04
	1	known	$\boldsymbol{R}^k$	51.82 ± 3.79	57.84 ± 0.48

Table 18. Study of FedDiverse combined with other non-IID mitigation techniques using different machine learning models on the $\text{Waterbirds}_{\text{dist}}$ dataset.

FL algorithm	MobileNet		ResNet50	
	Random	FedDiverse	Random	FedDiverse
FedAvg	31.72 ± 3.05	**46.47** ± 1.31	59.97 ± 2.47	**65.47** ± 2.31
FedAvgM	42.42 ± 0.59	**54.10** ± 2.03	62.56 ± 0.78	**67.81** ± 2.87
FedProx	31.57 ± 2.87	**43.51** ± 0.70	62.36 ± 0.94	**69.11** ± 1.92
FedAvgM + FedProx	44.29 ± 1.26	**53.84** ± 0.90	64.54 ± 3.29	**66.77** ± 4.16

Table 19. Study on the effect of pre-training rounds on the final worst group accuracy and determining the DHT values of the FedDiverse algorithm in $\text{WaterBirds}_{\text{dist}}$ dataset using FedAvg and FedAvgM algorithms for server-side aggregation.

Pre-training(T_0)	Worst group accuracy(%) FedAvg	Worst group accuracy(%) FedAvgM	DHT prediction error $\|\|\tilde{\Delta}^k - \Delta^k\|\|$ FedAvg	DHT prediction error $\|\|\tilde{\Delta}^k - \Delta^k\|\|$ FedAvgM
1	**46.47** ± 1.31	54.10 ± 2.03	0.50 ± 0.03	0.50 ± 0.10
5	42.47 ± 1.56	52.80 ± 2.45	0.49 ± 0.07	0.54 ± 0.01
10	42.16 ± 4.29	51.97 ± 4.43	**0.47** ± 0.01	0.52 ± 0.02
15	41.20 ± 3.73	53.12 ± 4.54	**0.47** ± 0.06	0.52 ± 0.05
20	<u>44.03</u> ± 0.32	**55.04** ± 3.09	<u>0.48</u> ± 0.02	<u>0.49</u> ± 0.01
30	42.16 ± 2.39	<u>54.36</u> ± 2.45	0.49 ± 0.02	**0.47** ± 0.04

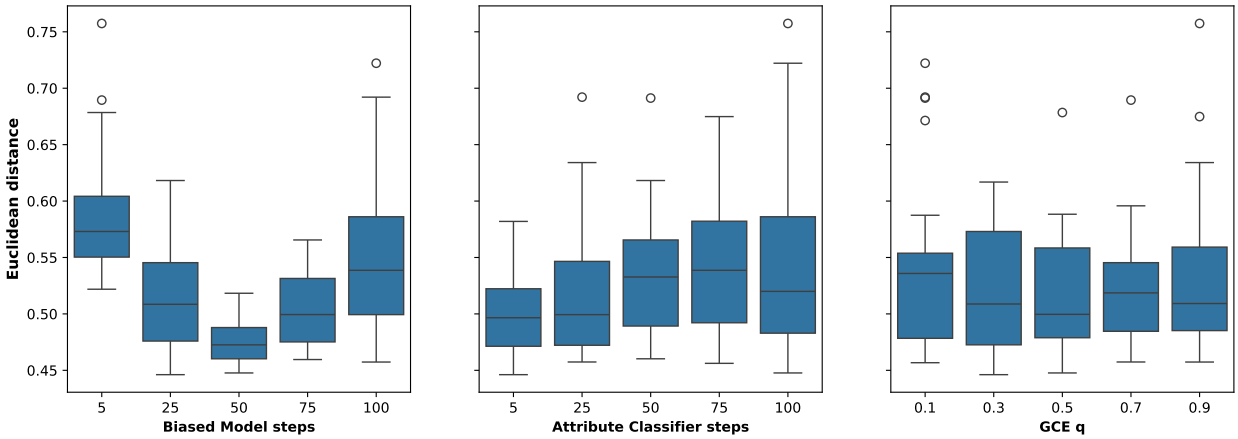


Figure 16. FedDiverse's sensitivity to the biased model's training steps τ_{biased}, the attribute classifier training steps τ_{attr} and the generalized cross-entropy loss' q value (q). We use $\tau_{biased} = 50$, $\tau_{attr} = 10$, and selected $q = 0.3$ as best q value for the given τ parameters.

rounds. In this simulation, we assume that the server has full knowledge of the clients' true data heterogeneity triplets.

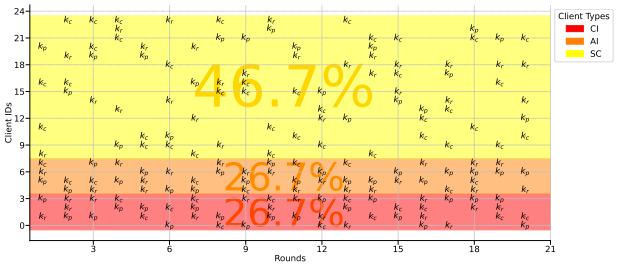
These plots illustrate how FedDiverse samples the clients in a much more uniform way within each type, as the percentages are close to 33.3%. Interestingly, figure 17 e) shows that the 3 clients of type CI are sampled in each round, proving how FedDiverse effectively succeeds in sampling clients with different types of data heterogeneity.

5.7 Limitations and future work

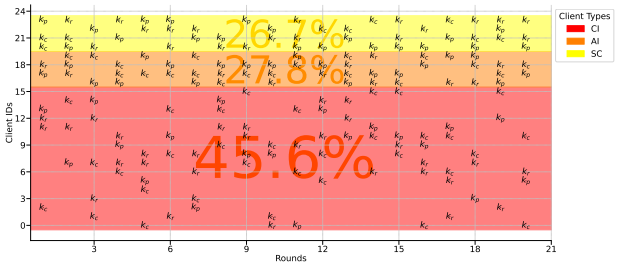
To the best of our knowledge, FedDiverse is the first algorithm specifically designed to address the issue of spurious correlations in federated learning. As noted by [WZNK24], the heterogeneity among clients can help mitigate learning shortcuts that arise from these spurious correlations. Unlike their approach, which leverages spurious features to create a Personalized FL solution, the goal of FedDiverse is to develop a single global model that is resilient to spurious correlations. Furthermore, FedDiverse leverages various types of statistical data heterogeneity in the clients during each sampling round to enhance the generalization capabilities of the model and reduce the overall impact of data heterogeneity.

However, FedDiverse is not exempt from limitations. First, it has been designed and evaluated specifically for image classification tasks. In future work, we plan to extend FedDiverse to other computer vision tasks, such as semantic segmentation for FL, a topic of growing interest in the community [DZC+23; YGQ+22; SFT+23; FCC23]. The presence of spuriously correlated features in these tasks could pose security risks by leading models to rely on misleading patterns, potentially compromising their performance in safety-critical applications.

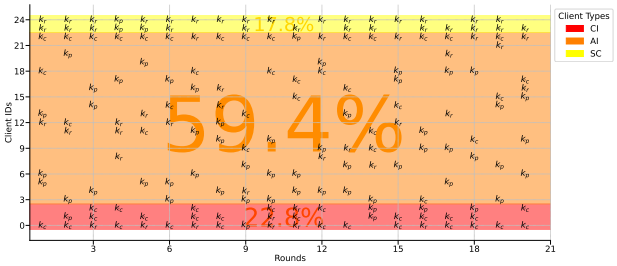
Furthermore, FedDiverse could be improved by addressing scenarios with multiple spurious attributes, each with more than two possible values. Future versions could be designed to approximate multi-dimensional interaction tensors rather than the current bi-dimensional interaction matrices $\boldsymbol{R}^k$. These tensors would link the true ground-truth label with various spurious attributes, accommodating more complex attribute interactions.



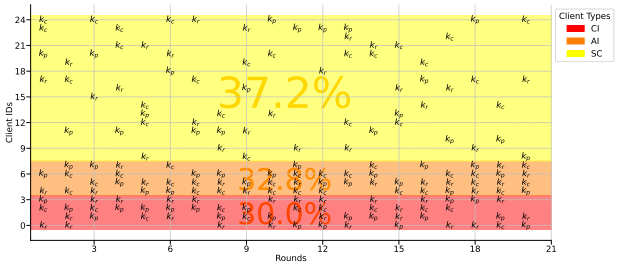
(a) Simulation of the sampling strategy of FedDiverse on the $\text{Spawrious}_{\text{GSC}}$ dataset. With uniform random sampling, clients belonging to each specific type would have been sampled with the following proportions: CI=16.7%, AI=16.7%, SC=66.6%. This simulation can be equivalently interpreted as if the dataset is $\text{CMNIST}_{\text{GSC}}$ since the two datasets have the same client distributions.



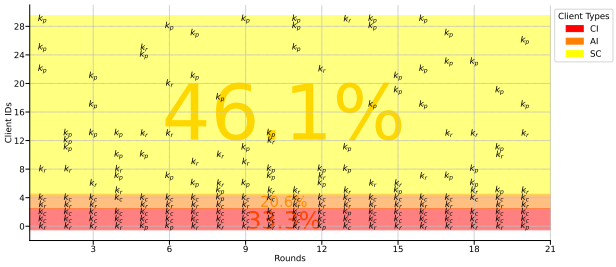
(b) Simulation of the sampling strategy of FedDiverse on the $\text{Spawrious}_{\text{GCI}}$ dataset. With uniform random sampling, clients belonging to each specific type would have been sampled with the following proportions: CI=66.6%, AI=16.7%, SC=16.7%.



(c) Simulation of the sampling strategy of FedDiverse on the $\text{Spawrious}_{\text{GAI}}$ dataset. With uniform random sampling, clients belonging to each specific type would have been sampled with the following proportions: CI=12.0%, AI=60.0%, SC=8.0%.



(d) Simulation of the sampling strategy of FedDiverse on the Spawrious_{4} dataset. With uniform random sampling, clients belonging to each specific type would have been sampled with the following proportions: CI=16.0%, AI=16.0%, SC=68.0%.



(e) Simulation of the sampling strategy of FedDiverse on the $\text{Waterbirds}_{\text{dist}}$ dataset. With uniform random sampling, clients belonging to each specific type would have been sampled with the following proportions: CI=10.0%, AI=6.7%, SC=83.3%.

Figure 17

5.8 Conclusion

In this work, we have introduced a novel framework for characterizing statistical data heterogeneity in FL, we have presented seven datasets to evaluate the performance of FL methods in the presence of different types of data heterogeneity, and we have proposed FedDiverse, a novel and efficient client selection method that selects clients with diverse types of statistical data heterogeneity. In extensive experiments, we demonstrate FedDiverse's competitive performance on all datasets while requiring low communication and computation overhead.

In the next chapter, we explore a use-case for FedDiverse: we investigate the capabilities of FedDiverse's diversity-driven client selection to improve group algorithmic fairness in federated learning.

Chapter 6

Group fairness: a use case of FedDiverse

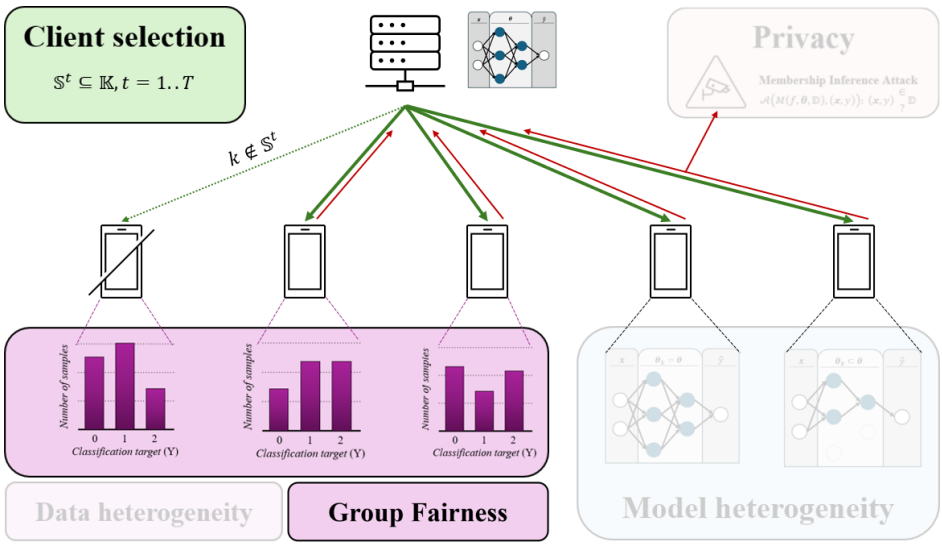


Figure 18. Content presented in this chapter in the scope of the thesis.

6.1 Introduction

In chapter 3, we illustrate how client selection can be a powerful tool to achieve a variety of objectives in federated learning. Building on the foundation from chapter 5, where we introduced FedDiverse, a novel client selection algorithm that leverages client diversity in non-IID distributed datasets to train robust models, we describe in this chapter ongoing work that explores a use case of FedDiverse: group algorithmic fairness in FL by mitigating the bias in distributed computer vision datasets.

Algorithmic fairness [DHP+12] has been extensively studied in machine learning for over a decade. Its importance has grown as machine learning methods are increasingly applied in high-stakes scenarios, such as automated decision-making in healthcare or finance [BHN23]. However, achieving fairness in machine learning is a difficult task given the biases present in the data, the design choices, and the use of the models.

As described in chapter 1, federated learning (FL) has been proposed as a privacy-preserving solution for distributed machine learning, where clients (*e.g.* hospitals) have access to sensitive data and are willing to cooperate to train a global machine learning model together without sharing the private data. Therefore, a central server is designed to collect model updates instead of the sensitive information to construct a robust global model through aggregation [MMR+17]. However, building a fair FL system involves additional challenges compared to traditional algorithmic fairness, given the restrictions of private data and efficient communication, and the interests of multiple stakeholders in a distributed system [KMA+21].

As described in chapter 5, FedDiverse is a novel client selection algorithm for FL proposed in this thesis and designed to mitigate data heterogeneity in biased federated datasets. It leverages the diversity of the client data [NFN+25], hence its name. Clients estimate a data heterogeneity triplet based on the class imbalance, attribute imbalance, and spurious correlation statistics of their data. The server selects diverse clients based on these scalar triplets submitted by the clients. We have also proposed a method to determine the interaction matrix of the client data when the spurious attributes are unknown.

In this chapter, we investigate whether FedDiverse can be applied to improve group algorithmic fairness in scenarios where there is a bias in client data that leads to unfairness in the trained models. The diversity in the distributions of the data in the clients allows for training federated models with client selection to promote the participation of clients with data from globally underrepresented protected groups.

To understand the background of fairness in federated learning, we introduce key concepts in section 2.6. The structure of the rest of the chapter is as follows: and in section 6.3 we present the most relevant existing research on fairness in federated learning. Section 6.4 discusses the application of FedDiverse for group algorithmic fairness and sections 6.5 and 6.6 summarizes our results. Finally, in section 6.7 we outline future research directions.

6.2 Fairness in FL

6.2.1 FL fairness notions

Given that federated learning is a multi-agent system where the interests of the different actors do not always align, fairness can be addressed from different perspectives. [SYL23] and [HYS+24] categorize the notions of fairness in federated learning as fairness in performance and collaboration. On the one hand, *performance fairness* includes the notion of *group fairness* that aims to ensure the fairness of protected groups represented in multiple clients, and *performance distribution fairness* that aims to achieve equal performance between clients. On the other hand, *collaboration fairness* or cooperation fairness includes notions in which the goal is to reward clients according to their contribution to the federation. Table 20 summarizes the notions of fairness in federated learning, following [SYL23].

6.2.2 Algorithmic fairness in federated learning

In this research, we focus on group algorithmic fairness in federated learning, *i.e.*, how fairly a trained model performs across protected groups, which may be defined by attributes such as gender, race, religion or age. In *cross-silo* federated learning, where each client is typically an organization, *e.g.*a bank or a hospital, with private datasets, fairness concerns often relate

Table 20. Fairness notions introduced in federated learning

Fairness notion	**Description**
Fair performance	
Good-Intent Fairness	Minimize maximum loss of *protected group*
Group Fairness	Improve algorithmic fairness metrics for *protected group*
Accuracy Parity	Reduce performance differences between *clients*
Fair collaboration	
Selection Fairness	Ensuring fair selection of underrepresented *clients*
Contribution Fairness	A *client*'s reward should be proportional to it's contribution
Regret Distribution Fairness and Expectation Fairness	The *clients* are motivated fairly through *time* until the training terminates

to how well the model performs across subgroups within these datasets. In contrast, in *cross-device* federated learning, each client typically corresponds to an individual (*e.g.* a personal smartphone), such that fairness efforts focus on ensuring equitable performance at the individual or client level. A recent systematic literature review highlights the growing interest in addressing group fairness in FL research [SACA24].

6.2.3 Group fairness metrics

Measuring group algorithmic fairness has long been a subject of debate in the scientific community. A widely accepted guideline is that the choice of fairness metric should depend on the nature of the real-world task at hand [VR18]. However, when it comes to toy datasets commonly used in federated learning research, there is often no consensus on which fairness metric is most appropriate, even for the same task.

Table 21 provides an overview of the most commonly used group fairness metrics in FL for image classification tasks, which is the focus of this thesis. Based on our literature review, the most commonly used group algorithmic fairness metrics for federated learning are *equal opportunity* and *demographic parity*.

These metrics capture fairness from different: while *demographic parity* focuses on equal prediction outcomes across groups, regardless of true labels, *equal opportunity* requires equal true positive rates, aligning fairness with accuracy for the positive class. Ultimately, the choice of metric should reflect the societal priorities and risk implications derived from the specific application domain.

6.3 Related work

[SACA24] categorizes group fairness methods in FL based on *where* the fairness intervention takes place. In the first category, fairness is enforced *locally*, *i.e.*, each client applies fairness techniques based on the statistics of its own dataset. However, in non-IID settings, the data distribution observed by a clikent can differ significantly from the global data distribution seen by the server. For example, a client k might identify a certain group as underrepresented within its own data and apply a data re-weighting strategy to reduce bias. Yet, this same group could be well-represented –or even be the majority– across the entire federation. This mismatch can lead to unintended unfair results in the global model. In the second

Table 21. Commonly used fairness metrics in Federated Learning for binary label (Y) prediction ($\hat{Y}$) and sensitive attribute (A) groups

Metric	Definition	Methods
Equal Opportunity (EOp)	$\Pr(\hat{Y}=1\|Y=1,A=0)=\Pr(\hat{Y}=1\|Y=1,A=1)$	[GGvDS21; KPDG22; YJ22; WPLK23; MLZL24; DBT+24; DS24]
Equalized Odds (EOd)	$\Pr(\hat{Y}=1\|Y=y,A=0)=\Pr(\hat{Y}=1\|Y=y,A=1),\forall y\in\{0,1\}$	[KPDG22]
Accuracy Parity (AP)	$\Pr(\hat{Y}=Y\|A=0)=\Pr(\hat{Y}=Y\|A=1)$	[GGvDS21; KPDG22; HWS24]
Demographic Parity (DemPar)	$\Pr(\hat{Y}=1\|A=0)=\Pr(\hat{Y}=1\|A=1)$	[YJ22; HWS24; MLZL24; DS24; WPLK23; DBT+24; CZZZ25]
Predictive Equality (PE)	$\Pr(\hat{Y}=1\|Y=0,A=0)=\Pr(\hat{Y}=1\|Y=0,A=1)$	[YJ22]
Worst group accuracy (WGAcc)	$\min\Pr(\hat{Y}=y\|Y=y,A=a),\forall y,a\in\{0,1\}$	[PMB+22; HWS24]
Well-Calibration (WCal)	$\Pr(Y=1\|\hat{Y}=\hat{y},A=0)=\Pr(Y=1\|\hat{Y}=\hat{y},A=1),\forall\hat{y}\in 0,1$	[WPLK23]
Discrimination Index (DI)	$\left\|\Pr(\hat{Y}=1\|A=0)-\Pr(\hat{Y}=1\|A=1)\right\|$	[DBT+24]

category, fairness is enforced *globally* by the server. An example of this approach can consist of *client selection* to increase the participation of clients with data from underrepresented groups [ZKW20; SFAA23]. Finally, in the *hybrid* category, the server and the clients work together to achieve fairness, which typically requires the adaptation of the training algorithm on both the clients and the server.

Recent group fairness methods in federated learning for image classification include [GGvDS21], where the authors present an approach that adapts the differential multipliers method to FL to solve the constrained optimization problem of minimizing the loss with a fairness constraint. In [KPDG22], the server weights the clients based on their model fairness on a server-side validation set $\mathbb{D}_v$ (*e.g.* assigns the weights that maximize $\frac{Acc}{\ell_{fair}}$ in $\mathbb{D}_v$). [YJ22] trains an encoder-decoder network to generate adversarial samples on the clients to mitigate bias.

Additional methods include [PMB+22], where clients compute the empirical risk score of the previous global model on their private dataset and send it back to the server together with their model update. The server assigns weights to the clients based on this additional information. [HWS24] follows the same idea combined with a fairness regularizer in the client-side optimizer. [WPLK23] applies a fairness regularizer on the server side, where the bias of the local datasets is close to their shared gradient updates. [MLZL24] implements a constrained optimization method on the client level while grouping clients on the server to use equal weights for the groups in the aggregation. The approach proposed by [DBT+24] uses an evolution algorithm to apply fair aggregation on the server side. [DS24] applies a fairness regularizer with a learnable λ weight updated along with the model weights in a federated manner. [CZZZ25] trains a generator on the server using the client models that is shared with the clients to generate synthetic data for bias mitigation.

Alternatively, [SGAA24] uses clustered federated learning to train multiple global models, each trained by similar clients with local bias mitigation.

Several authors have proposed assigning different weights to the clients to achieve group fairness. The first article that compared local data re-weighting with client weighting to achieve group fairness by manipulating the participation of the clients from the server side was presented by Abay et al. in 2020 [AZB+20]. Some of the previously described methods find inspiration in this work and also use client weighting to mitigate bias in the clients [KPDG22; PMB+22; HWS24; MLZL24].

Alternatively, client selection can be applied to improve group fairness. The differences between client weighting and client selection are explained in the taxonomy of presented in chapter 3. In [SFAA23], the server uses a validation set $\mathbb{D}_V$ to infer the importance of the client models for fairness and hence select the best candidates. [ZKW20] designs a multi-agent reinforcement learning system to optimize client selection for fairness in FL.

6.3.1 Algorithmic Fairness Datasets in Federated Learning

Similarly to group fairness metrics, there is no consensus on benchmark datasets used to evaluate group fairness in federated learning. In this section, we summarize the most commonly used datasets, focusing on image datasets –which is the focus of this thesis– with realistic protected groups (*e.g.*datasets involving human data).

CelebA

The CelebFaces Attributes Dataset (CelebA) is an image dataset containing more than 200K faces from 10K individuals annotated with 40 binary attributes [LLWT15]. The dataset is used in federated learning because the images can be distributed based on the identities [CDW+19] to simulate *cross-device* FL, or with multiple individuals used in one client for *cross-silo* setting [CZZZ25]. While most of the FL literature has used *gender* as the protected attribute, the reported experiments vary depending on the target of the classification task, including *attractiveness* [KPDG22; YJ22; DS24], *smiling* [WPLK23; MLZL24; DBT+24], and *wavy hair* [CZZZ25].

UTKFace

The UTKFace Large Scale Face Dataset [ZSQ17] consists of 20k face images labeled by *age*, *gender* and *ethnicity*. FL fairness research mostly uses *ethnicity* as the protected attribute and *gender* [KPDG22; CZZZ25] or binarized *age* [MLZL24; DS24] as the classification target.

FairFace

The FairFace dataset consists of 100k images balanced on ethnicity [KJ21]. It contains *ethnicity*, *gender* and *age* groups. [CZZZ25] uses *ethnicity* as the protective attribute and *gender* as the classification target.

Additional image datasets

Other image datasets with orchestrated sensitive groups include FEMNIST, a dataset containing 800k handwritten characters by 3500 users commonly used in FL literature [CDW+19]. One fairness research study from the literature has reported using the color of the pen as the protected group, which is a somewhat arbitrary decision: the dataset consists of black pen

and blue pen characters on a white background, and white chalk on a blackboard [GGvDS21]. Datasets like FashionMNIST [XRV17] and CIFAR-10 [Kri09] are also used with each label considered both as the target label and protected attribute [PMB+22]. Additionally, a new image dataset adapted by the fairness community but not yet used by the federated learning researchers is the FACET [GRR+23] dataset, which includes 32k annotated images of 50k people.

Tabular data

Tabular datasets are commonly used to evaluate group fairness in centralized and federated learning research. These experiments typically involve learning logistic regression [DXWT21; AZB+20; CPL+21; ZKL+21] or small multilayer perceptron [PDG21; GGvDS21] models.

In this context, the most commonly used fairness datasets are: the **Adult** [KB96] dataset, where the target classification is $> \$50K$ salary and the sensitive group can be *gender* or *race* [AZB+20; ZKW20; DXWT21]; and the **COMPAS** [ALMK19] dataset, consisting of 7000 samples with the probability of "re-offend" as the target variable and *gender* or/and *race* as the protected attributes, is also frequently used in the literature [AZB+20; ZKW20].

The **ACS Employment** dataset, consisting of 1 million samples from the American Community Survey via Folktables, filtered for *employment* classification [DHMS21] and with *race* or *gender* as protected attributes is of particular interest for FL research because it is possible to naturally distribute the data among states as clients in the federation [HWS24; YPN24].

6.4 Group fairness with FedDiverse

We address fairness in federated learning as a use case of FedDiverse, presented in chapter 5. Intuitively, if the lack of fairness originates from a representation bias in the training dataset across clients, a FL method that is mitigates the effects of data heterogeneity in the clients could be applied to achieve algorithmic fairness.

Given $\mathcal{X}$ input space, $\mathcal{Y} = \{0, 1\}$ binary label space and $\mathcal{A}$ protected attribute space and Ω parameter space, let $f : \mathcal{X} \to \mathcal{Y}$ be a predictor function parameterized by $\boldsymbol{\theta} \in \Omega$. For $\mathbb{D} = \{\boldsymbol{x}_i, y_i, a_i\}$ dataset, where each $(\boldsymbol{x}_i, y_i, a_i) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{A}$, the goal is to determine the $\boldsymbol{\theta}$ parameter that minimizes the loss function:

$$\min_{\boldsymbol{\theta} \in \Omega} \mathcal{L}(\boldsymbol{\theta}) = \frac{1}{|\mathbb{D}|} \sum_{(\boldsymbol{x}, y, a) \in \mathbb{D}} \ell_{BCE}(y, f(\boldsymbol{x}, \boldsymbol{\theta})),$$

where ℓ_{BCE} is the binary cross-entropy loss, while maintaining good fairness measured by the three most commonly used group fairness metrics in federated learning (see section 2.6):

$$\textit{Equal opportunity}\text{: } \Pr(\hat{Y} = 1 | Y = 1, A = 0) = \Pr(\hat{Y} = 1 | Y = 1, A = a), \forall a \in \mathcal{A}, \tag{35}$$

$$\textit{Demographic parity}\text{: } \Pr(\hat{Y} = 1 | A = 0) = \Pr(\hat{Y} = 1 | A = a), \forall a \in \mathcal{A} \tag{36}$$

$$\text{and } \textit{Worst group accuracy}\text{: } \min \Pr(\hat{Y} = y | Y = y, A = a), \forall y \in \mathcal{Y}, a \in \mathcal{A}, \tag{37}$$

We consider $\mathbb{D} = \bigcup_{k \in \mathbb{K}} \mathbb{D}_i$ the union of the private datasets $\mathbb{D}_k$ of $k \in \mathbb{K}$ clients. We define the interaction matrix $\boldsymbol{R} \in \mathbb{R}^{|\mathcal{Y}| \times |\mathcal{A}|}$ such that each element of the matrix $r_{k,l} \in \boldsymbol{R}$

contains the number of samples of the respective population in the dataset $r_{l,m} = |\mathbb{G}|$, where $\mathbb{G} = \{(\boldsymbol{x}, y, a) \in \mathbb{D}, y = l, a = m\}$. We denote $\boldsymbol{R}_k$ the interaction matrix of the dataset $\mathbb{D}_k$ of client k.

FEDDIVERSE [NFN+25] consists of 3 steps:

(1) First, an interaction matrix predictor infers $\tilde{\boldsymbol{R}}_k$ if a is unknown for $(\boldsymbol{x}, y, a) \in \mathbb{D}_k$.

(2) Next, the clients generate a data heterogeneity triplet Δ_k from the interaction matrix $\boldsymbol{R}_k$ to preserve privacy.

(3) Finally, the server $\mathcal{S}$ selects clients that are distant in the *data heterogeneity space* ($\Delta \in [0, 1]^3$) to maximize diversity and complementarity in the data distributions.

Given a minority group (y, a), if $\exists k \in \mathbb{K} : \frac{r^k_{y,a}}{\sum_{(i,j)\in(|\mathcal{Y}|\times\mathcal{A})} r^k_{i,j}} > \frac{r_{y,a}}{\sum_{(i,j)\in(|\mathcal{Y}|\times\mathcal{A})} r_{i,j}}$, increasing the participation of that client k can be seen as a resampling method for fairness optimization [KC12].

6.5 Experiments

6.5.1 Datasets

As previously described, we evaluate the performance of FEDDIVERSE and related methods on three commonly used image datasets, namely:

CelebA: To created a federated version of this dataset, we select images of 20 individuals of the CelebA dataset for every client. To simulate bias in the training data, we adjust the number of individuals per $(target, attribute)$ group, where individuals are placed in the group represented by the majority of their image instances. We use the binary variable *wavy hair* as the target label and *gender* as the binary protected attribute (male/female). We adopt a similar data distribution to that of the $\text{Spawrious}_{\text{GSC}}$ dataset presented in table 13, where instead of 90 and 10 samples, we use 9 and 1 individuals, respectively [NFN+25]. This results in a dataset with (wavy hair, male) as the underrepresented group and varying numbers of samples depending on the selected individuals.

UTKFace: Unlike CelebA, the UTKFace dataset does not have natural federated splits. Hence, we apply the data split described in section 5.5 for the Spawrious dataset simulating spurious correlations ($\text{Spawrious}_{\text{GSC}}$). We use *race* in a binarized form (white – non-white) as the protected attribute and *gender* (male – female) as the binary target variable. Note that 47% of the samples in the UTKFace dataset are labeled as white, we obtain a fairly balanced pool for the data split to select from. In the FL simulated scenario, the minority groups are (white, female) and (black, male).

FairFace: Similarly to UTKFace, this dataset does not have a natural federated split, thus we apply the same data split and use the same target and sensitive attribute labels as for UTKFace.

6.5.2 Experimental setup

We simulate a federated learning scenario with 24 clients using both the Flower [BTM+20] and PyTorch [PGM+19] frameworks. Our code is available at `https://github.com/Erosinho13/SpuriousFL`.

Table 22. Comparing FedDiverse with FedAvg random client selection for algorithmic fairness in federated learning

Dataset	Method	Acc(%)	WGAcc (%)	ΔEOp (↓)	ΔDemPar (↓)
$\text{CelebA}_{\text{GSC}}$	FedDiverse	83.95 ± 0.06	**55.12** ± 2.29	**0.13** ± 0.02	**0.37** ± 0.02
	Random	**84.41** ± 0.14	50.72 ± 1.41	0.14 ± 0.03	0.39 ± 0.03
$\text{UTKFace}_{\text{GSC}}$	FedDiverse	87.60 ± 0.45	79.44 ± 1.61	0.10 ± 0.03	**0.10** ± 0.01
	Random	**87.99** ± 0.75	**81.01** ± 2.95	**0.09** ± 0.01	0.11 ± 0.01
$\text{FairFace}_{\text{GSC}}$	FedDiverse	79.00 ± 0.67	70.08 ± 3.23	**0.13** ± 0.01	**0.11** ± 0.01
	Random	**79.82** ± 0.58	**72.22** ± 2.69	0.14 ± 0.01	0.12 ± 0.01

FL architecture The server and the clients train a MobileNet v2 [How17] model, where batch normalization layers are replaced with group normalization layers and initial weights are pre-trained on Imagenet. We apply the *categorical crossentropy* loss function with 0.001 learning rate and a batch size of 28. We train the federation for $T = 200$ rounds with equally weighted clients. In experiments without client selection, all clients (24) participate in the federation in every round. In the cases where client selection is performed, the server selects 9 clients to participate in the federation in each round. We report *average accuracy*, *worst group accuracy*, *demographic parity*, and *equal opportunity* averaged over 3 runs.

Baseline We compare the proposed methods with a **uniform random** selection, where clients are randomly selected according to a uniform distribution.

6.6 Results

6.6.1 FedDiverse for group fairness

Experiments with the original FedDiverse privacy settings are summarized in table 22 and show promising results to achieve fairness in image classification tasks. For the CelebA dataset, FedDiverse outperforms the baseline in *worst group accuracy* (4.4% increase), *equalized opportunity* (0.01 reduction) and *demographic parity* (0.02 reduction) while maintaining competitive accuracy (0.46% drop compared to random). For UTKFace, FedDiverse outperforms the baseline if *demographic parity* (0.01 reduction) and for FairFace it outperforms in *demographic parity* (0.01 reduction) and *equalized opportunity* (0.01 reduction).

6.6.2 Relaxed privacy requirements for improved group fairness

In section 5.6, we presented three variations of the FedDiverse algorithm with varying levels of privacy implications depending on the metadata that is shared as part of the federation.

The most privacy-preserving option would be not to share any metadata. In this case, it is not possible to apply diversity-driven client selection. Hence, we use random client selection with FedAvg.

In the second level, the sensitive attributes are excluded from the training, aiming to achieve what is known in the literature as fairness under unawareness [CKM+19]. In FedDiverse,

the majority and minority groups are inferred based on the performance of a biased model: examples with poor performance are hypothesized as members of the minority group instead of using the minority group given by the protected attributes.

In the next privacy level, the protected attributes are known to the clients but they are not shared with the server. In FedDiverse, this entails sharing the data heterogeneity triplets (DHT) with the server.

In the least private level, the clients share their interaction matrix with the server which uses this information to re-weight the clients, adopting a similar methodology as that of the ReWeight method [LBC+20] in centralized learning. In this case, we use the FedRK method described in section 5.6, which assigns a selection probability to the clients based on the weights derived from the clients' interaction matrix.

Table 23 summarizes the results of the experiments applying these four levels of attribute privacy on the CelebA, UTKFace and FairFace datasets. As seen in the table, the best performing method –in terms of group fairness– is FedRK. On the CelebA dataset, FedRK achieves an increase of 8.15 percentage points in *worst group accuracy*, a decrease of 0.49 percentage points in *accuracy*, and an improvement of 0.03 and 0.05 in *equalized opportunity* and *demographic parity*, respectively, when compared to the baseline. On the UTKFace dataset, FedRK achieves an increase of 3.81 percentage points in *worst group accuracy*, of 0.29 percentage points in *accuracy*, and an improvement of 0.05 and 0.06 in *equalized opportunity* and *demographic parity*, respectively, when compared to the baseline. On the FairFace dataset, we observe an increase of 3.55 percentage points in *worst group accuracy*, of 0.67 percentage points in *accuracy*, and an improvement of 0.07 in both *equalized opportunity* and *demographic parity*, with respect to the baseline. Both variations of FedDiverse achieve very competitive levels of accuracy and worst group accuracy while improving the group fairness metrics when compared to the baseline.

6.7 Conclusion and future work

Our experiments illustrate the value of applying diversity-driven client selection algorithms for group algorithmic fairness in federated learning. We also demonstrate the interplay between the level of privacy in the shared diversity descriptors and the algorithmic fairness of the trained model.

One limitation of FedDiverse is that it expects a binary attribute space, internally classifying samples as belonging to a majority or minority group. Although most fairness datasets include binary groups, there are also datasets that consist of multiple sensitive attributes. Therefore, an future research should explore the possibility of predicting binary tensors for the interaction matrix in FedDiverse. One such direction consists of leveraging clustering algorithms to infer the interaction matrix.

Interestingly, FedDiverse does not rely on knowledge of the protected attribute for bias mitigation. Future research could investigate whether this design internalizes the discrimination between protected groups or the model classifies hard examples orthogonally to the sensitive attribute.

Table 23. Performance of the client selection method given different levels of privacy requirements on the CelebA, UTKFace and FairFace datasets

(a) CelebA

Privacy setup	Acc(%)	WGAcc (%)	ΔEOp (↓)	ΔDemPar (↓)
Random (No shared stats)	**84.41** ± 0.14	50.72 ± 1.41	0.14 ± 0.03	0.39 ± 0.03
FedDiverse (Predicted DHT)	83.95 ± 0.06	55.12 ± 2.29	0.13 ± 0.02	0.37 ± 0.02
FedDiverse (Known DHT)	83.83 ± 0.14	58.54 ± 0.65	0.14 ± 0.01	0.37 ± 0.02
FedRK (Known $\boldsymbol{R}^k$)	83.92 ± 0.25	**59.57** ± 5.59	**0.11** ± 0.03	**0.34** ± 0.02

(b) UTKFace

Privacy setup	Acc(%)	WGAcc (%)	ΔEOp (↓)	ΔDemPar (↓)
Random (No shared stats)	87.99 ± 0.75	81.01 ± 2.95	0.09 ± 0.01	0.11 ± 0.01
FedDiverse (Predicted DHT)	87.60 ± 0.45	79.44 ± 1.61	0.10 ± 0.03	0.10 ± 0.01
FedDiverse (Known DHT)	**88.60** ± 0.60	82.62 ± 1.64	0.08 ± 0.02	0.08 ± 0.01
FedRK (Known $\boldsymbol{R}^k$)	88.28 ± 0.10	**84.86** ± 2.92	**0.04** ± 0.01	**0.05** ± 0.00

(c) FairFace

Privacy setup	Acc(%)	WGAcc (%)	ΔEOp (↓)	ΔDemPar (↓)
Random (No shared stats)	79.82 ± 0.58	72.22 ± 2.69	0.14 ± 0.01	0.12 ± 0.01
FedDiverse (Predicted DHT)	79.00 ± 0.67	70.08 ± 3.23	0.13 ± 0.01	0.11 ± 0.01
FedDiverse (Known DHT)	79.99 ± 0.28	70.49 ± 2.17	0.12 ± 0.01	0.10 ± 0.00
FedRK (Known $\boldsymbol{R}^k$)	**80.49** ± 0.49	**75.77** ± 2.72	**0.07** ± 0.01	**0.05** ± 0.01

Chapter 7

Conclusion

In the previous chapters, we presented our work done in the field of federated learning, specifically focusing on the impact of data, model and participation heterogeneity on performance, privacy and fairness in federated learning. In this chapter we summarize the key contributions, discuss the limitations of our work and propose future research directions.

Summary of key contributions: Federated Learning (FL) is an emerging field with impact on a variety of domains. The diverse list of challenges and solutions proposed in FL makes comparing and evaluating novel FL methods a complex endeavor. To simplify this task, in chapter 3 we proposed a novel taxonomy of client selection in FL that enables the categorization and comparison of different FL methods. A key yet unrealistic assumption in FL is homogeneity in the clients and their data. In this thesis, we relax such an assumption and contribute to the field of FL with several novel contributions that allow for heterogeneous FL settings.

We start with model heterogeneity in chapter 4 where we investigate FL architectures where the clients learn models of different complexities depending on their computation capabilities. We present a novel taxonomy of model integration methods and perform an in-depth study of the privacy implications of such strategies: in the case of IID in the clients, we find that the model integration strategy can have a significant impact on privacy. However, non-IID setups are challenging for state-of-the-art methods and require further investigation.

We address such a challenge in chapter 5, where we propose FedDiverse, a novel client selection algorithm that leverages the data heterogeneity in the clients – namely attribute imbalance, class imbalance, and spurious correlations – to mitigate the difficulties that arise from having non-IID data.

Finally, in chapter 6, we explore a socially impactful application of FedDiverse by investigating how its client selection strategy can be leveraged to enhance group fairness in FL. Specifically, we examine scenarios where data is heterogeneously distributed across demographic or socially-defined groups – such as race, gender, or geographic region – and show how FedDiverse can mitigate disparate model performance across these groups. Through targeted client participation, our method promotes equitable representation in the learning process, thereby contributing to more inclusive and fair FL models.

In sum, this thesis provides key contributions in the field of federated learning with client or data heterogeneity to yield robust, accurate, and fair models that respect the clients' privacy. In the next paragraph, we address the limitations of our work.

Limitations and challenges: Federated Learning is an engineering-heavy research field. Like all research in FL, ours is also bounded by design choices, namely: (1) We limit our experiments in chapter 4 to attacks taking place in the last client model update; (2) We measure privacy with membership inference attacks; (3) In chapter 5, we limit experiments to small, but well-controlled data distributions and datasets; (4) Furthermore, we design our client selection and model-agnostic methods to be modular and easy to combine with existing frameworks. However, this comes with a cost of competitiveness compared to complex methods designed for the sole purpose of dealing with the task at hand.

Future work should be designed to address these limitations. Additionally, in the next paragraph, we describe interesting research ideas that can serve as the basis for the continuance of the research presented in this thesis.

Future research directions: A direction for future research concerns the work described in chapters 5 and 6. Current research on non-IID data in FL focuses on building general, robust methods. In this chapter, we adopted a different approach and designed experiments with control of the statistical heterogeneity in the datasets. We believe that this research can be extended to develop metrics that measure the expected non-IIDness in the distributed data *before* the training, allowing us to design better task-specific FL methods. Additionally, future research should include a more in-depth analysis of the interplay between different objectives in FL, such as privacy in chapter 4 and fairness in chapter 6.

Appendix A

Appendix for chapter 3

A.1 Highlighted client selection algorithms

In this section, we provide a summary of the client selection algorithms included in table 4 and discussed in this paper. These methods have been selected because of their relevance in the field and/or because they represent a specific category of our taxonomy.

Baseline selection strategies

The two most commonly used baseline client selection strategies are EqRep and EqW, presented below.

$\mathbf{EqRep_{baseline}}$

Equal representation of clients ($\text{EqRep}_{\text{baseline}}$): with a total of $|\mathbb{K}|$ clients and $|\mathbb{S}^t|$ selected clients in round t of the training, each client has a $\Pr(k \in \mathbb{S}^t) = \frac{|\mathbb{S}^t|}{|\mathbb{K}|}$ probability to participate in the round. This approach is often referred to as *random* client selection. Its motivation is to have all clients participating equally. The original FedAvg method use $\text{EqRep}_{\text{baseline}}$ to reduce the number of participating clients.

$\mathbf{EqW_{baseline}}$

Equal weights of data samples ($\text{EqW}_{\text{baseline}}$): the probability of participation for client k is proportional to the fraction of data that the client has access to $\Pr(k \in \mathbb{S}^t) = \frac{|\mathbb{D}_k|}{|\mathbb{D}|}$. Thus, to compute the probability the server needs to know the clients' dataset size, $|\mathbb{D}_k|,\ \forall k \in \mathbb{K}$. The motivation is to achieve equal representation for all data samples in the training.

Methods motivated by improving training efficiency

$\mathbf{AFL_G}$

In Active Federated Learning (AFL_G)[10] the client k has a valuation in training round t, v_k^t, based on its reported loss [GMB+19]. If a client participates in training round t, its valuation is updated, otherwise it remains the same as in the previous training round $t-1$, $v_k^t = v_k^{t-1}, k \notin \mathbb{S}^t$. All the clients start with a negative infinite valuation, $v_k^0 = -\infty, \forall k \in \mathbb{K}$. The value interpretation is a probability function based on this value with additional clients

[10] AFL refers to Agnostic Federated Learning [MSS19], Active Federated Learning [GMB+19] and Anarchic Federated Learning [YZKL22]. We separate them using the initials of the first authors, AFL_M, AFL_G and AFL_Y respectively.

selected for exploration. The motivation is to achieve the same performance as the baseline client selection algorithms but with fewer training epochs.

pow-d

In Power-of-Choice Strategy (POW-D), first a candidate set ($\mathbb{A} \subset \mathbb{K}$) is selected based on the fraction of their data, $|\mathbb{D}_i|$ (as in $\text{EQW}_{\text{baseline}}$). Then the clients in $\mathbb{A}$ compute their local losses and report them back to the server. The clients with the highest losses are selected $\mathbb{S}^t \subset \mathbb{A}$ [CWJ22]. This method improves both the convergence rate and the accuracy when compared to random selection ($\text{EqRep}_{\text{baseline}}$).

S-FedAvg

In the Shapley-value based federated averaging (S-FEDAVG) method proposed by Nagalapatti and Narayanam [NN21], the server uses a verification dataset ($\mathbb{D}_V$) to determine the clients' contributions using Shapley values. Then, it selects the clients with a probability proportional to their contribution value. This method reaches higher accuracy than FEDAVG.

***k*-FED**

In k-FED, a local k-mean clustering is proposed to generate a compressed data description [DLS21]. Then, the cluster centroids in each client, $\mathbb{G}_k, k \in \mathbb{K}$, are sent to the server. The server does not include in the training clients with similar cluster centroids. This approach is evaluated with a combination of POW-D and the results show that it can boost the training convergence even further.

LAG

The Lazily Aggregated Gradient (LAG) [CGSY18] method stores previous updates of the clients and predicts their future performance using a Lipschitz-smoothness (L_f) estimator. In this approach, the updates are computed only on valuable clients as decided by the client or the server. Early stopping is possible if none of the clients sends updates. The motivation for the development of this method is to reduce the communication between clients and the server.

FL-CIR

Federated learning with class imbalance reduction (FL-CIR) [YWZ+21] proposes a multi-armed bandit method to select a client set with minimum class imbalance. Each client represents an arm, and the selected clients are a super-arm (set of arms). The reward of the super-arm is computed based on the results of the current training round. The motivation is to reduce the impact of non-IIDness in the data and hence improve the overall model's accuracy.

FAVOR

Optimizing Federated Learning on Non-IID Data with Reinforcement Learning (FAVOR) [WKNL20] formulates the client selection problem as a RL task, where the server acts as an agent, the state $s^t = \boldsymbol{\theta}^t, \boldsymbol{\theta}_1^t, ..., \boldsymbol{\theta}_N^t$ summarizes the current parameters in each client, the action consists of selecting a client for the next training round and the reward is the server-side accuracy. The server has a local validation set ($\mathbb{D}_V$) to evaluate the clients' models. The method's effectiveness is demonstrated with experiments where the target accuracy is reached with fewer communication rounds than previous methods.

Other global constrains

In addition to improving the training efficiency, several research works have explored other global constrains to guide the client selection process.

FedCS

Nishio and Yonetani [NY19] proposed one of the first client selection methods called Federated Learning with Client Selection (FedCS). The clients report their resource requests to the server in the form of an estimated compute time $\mathcal{T}$. The global constrain is defined as a maximum compute time per client. Thus, the server selects the clients that fulfill such a constrain.

AFL$_\mathbf{M}$

Agnostic Federated Learning (AFL$_\text{M}$), Mohri, Sivek, and Suresh [MSS19] aims to minimize the maximum loss of the clients' performance (good-intent fairness). The server updates a λ distribution function in every training round based on these local losses, and λ is used to reweigh the samples or to select clients that match a target data distribution.

q-FFL

q-Fair Federated Learning (q-FFL) aims to achieve a uniform accuracy across all clients Li *et al.* [LSBS20]. First, the server select clients according to a uniform probability. Then, the server reweighs the client parameters based on the Lipschitz constant (L_f) of the local loss functions.

Oort

In Oort, Lai *et al.* [LZMC21] utility scores are used to rank the clients and the server selects the top-K performers. The utility function depends on the local loss and the training time and it is given by: $Util(k) = |\mathbb{B}_k|\sqrt{\frac{1}{\mathbb{B}_k}\sum_{i\in\mathbb{B}_k}\ell(f(\boldsymbol{x}_i), y_i)^2} \times \frac{\mathcal{T}}{\mathcal{T}_k}^{\mathbb{1}(\mathcal{T}<\mathcal{T}_k)\times\alpha}$, where $\mathbb{B}_k \subset \mathbb{D}_k$ is a subset of samples in client k; $\mathcal{T}$ and $\mathcal{T}_k$ are the global and local needed training time, respectively; $\mathbb{1}(x)$ is 1 if x true, 0 otherwise, and α is a hyperparameter. Oort also includes an exploration component to increase the data diversity.

FLAME

FLAME: Federated Learning Across Multi-device Environments (FLAME$_\text{C}$)[11] builds energy profiles for each client and clients have a maximum allowed budget of energy consumption. The clients compute their utility scores using the training loss ($\mathcal{L}$), the energy consumption ($\mathcal{E}$) and the required computational time($\mathcal{T}$). The server selects the clients with the largest overall utility. In their experiments, even though clients have limited energy and therefore, limited participation [CMK22].

DDaBA

Dynamic Defense Against Byzantine Attacks (DDaBA) [RMLH22] tries to identifiy adversarial clients based on the local update's accuracy on the server-side validation set $\mathbb{D}_V$. It flags potentially harmful clients when the accuracy of their local update changes dramatically between rounds. Experiments show that this approach is effective against byzantine attacks.

Client inclusion policies

Regarding client inclusion policies, we highlight two methods proposed in the literature.

[11] There are at least 3 FL papers with the FLAME acronym. We include the initial of the first author as a subscript to differentiate between them.

LT-FL

Loss Tolerant Federated Learning (LT-FL) [ZFH21] classifies the clients into two categories, depending on whether they have enough resources to retrieve lost communication packages or not. The clients in the first group are expected to send an updated model to the server in all cases. However, the weights of the clients from the second group may be set to zero if their packages get lost.

FD

Federated Dropout (FDropout) [CKMT18] use model compression (via dropped neurons) to reduce the size of the clients. The dropped neurons are not randomized but computed in such a way to have the same smaller matrix dimension in each client. Moreover, the server is able to map them back together to their original model size. This technique increases the server-client communication efficiency while keeping the original prediction accuracy. Later work [DDT21; LWW+22] use the same idea to allow for heterogeneous model sizes in clients with different capabilities.

Incentive mechanisms

Finally, we include a summary of the three highlighted methods that propose different incentive mechanisms.

CI-MR

In Contribution Index Multi Round Reconstruction (CI-MR), all clients perform their local training and send the results back to the server which weights them depending on the size of the local datasets in each client, similarly to $\text{EqW}_{\text{baseline}}$. At the end of the training, the server uses Shapley values to determine the data quality of the clients to define the payment of their incentive [STW19].

FMore

FMore: an Incentive Scheme of Multi-dimensional Auction (FMore) [ZZWC20] is an incentive mechanism where the server shares a scoring function (Value Generation function) with the clients, the clients compute their utility scores based on their local resources and offer a bid with a payment proposal. Based on this information, the server selects the clients for that round of training.

CBIM

Contract-Based Incentive Mechanism (CBIM) [KXN+19a] uses contract theory to select the right cluster of clients based on the clients' utility scores. It also keeps track of the reputation of the clients via a blockchain mechanism.

Appendix B

Appendix for chapter 4

B.1 Detailed experiment results

Results on the two CIFAR datasets are summarized in the main paper in included in detail in table 24 and table 25.

Table 26 includes the detailed experiment results for the FEMNIST dataset.

Table 24. Detailed results on the CIFAR-10 dataset. Experiments averaged over 3 runs. Best in each category highlighted with bold. Methods are grouped by number of *large* clients and ordered based on the frequency a client receives the same parameters.

No. of *large*	Freq.	Method name	↑ Server Acc %	↑ Client Acc %	↓ AUC % tMIA	↓ AUC % LiRA	↓ AUC % Yeom	↓ AUC % Avg	↓ Attack Adv % tMIA	↓ Attack Adv % LiRA	↓ Attack Adv % Yeom	↓ Attack Adv % Avg	↓ TPR% at 0.1% FPR tMIA	↓ TPR% at 0.1% FPR LiRA	↓ TPR% at 0.1% FPR Yeom	↓ TPR% at 0.1% FPR Avg
10	1	FedAvg100k	**78.48** ± 0.23	**78.47** ± 0.18	53.62 ± 0.15	51.77 ± 0.77	52.62 ± 0.55	52.67 ± 0.42	3.94 ± 0.40	2.49 ± 0.17	3.03 ± 0.65	3.16 ± 0.41	0.16 ± 0.05	**0.08** ± 0.01	0.14 ± 0.04	0.13 ± 0.03
0	1	FedAvg30k	69.04 ± 0.24	68.82 ± 0.21	**52.17** ± 0.35	**50.83** ± 0.15	**51.81** ± 0.27	**51.60** ± 0.14	**2.11** ± 1.01	**0.69** ± 0.54	**1.66** ± 0.05	**1.49** ± 0.18	**0.09** ± 0.02	0.09 ± 0.00	**0.09** ± 0.02	**0.09** ± 0.01
2	1	OFM (HeteroFL)	77.79 ± 1.53	69.17 ± 0.17	53.30 ± 0.31	51.13 ± 0.16	52.47 ± 0.27	52.30 ± 0.16	3.22 ± 1.14	1.46 ± 0.33	3.04 ± 0.76	2.57 ± 0.24	0.10 ± 0.02	**0.06** ± 0.01	0.12 ± 0.04	0.10 ± 0.02
2	1	OFR	78.22 ± 1.53	**69.95** ± 1.18	53.30 ± 0.55	50.94 ± 0.56	52.28 ± 0.44	52.17 ± 0.43	2.60 ± 1.09	0.68 ± 1.55	1.95 ± 0.57	1.74 ± 0.34	0.14 ± 0.04	0.08 ± 0.02	0.11 ± 0.05	0.11 ± 0.03
2	1/4	OSM	**78.70** ± 1.28	58.52 ± 1.91	51.73 ± 1.19	50.48 ± 0.45	51.40 ± 1.03	51.20 ± 0.84	2.16 ± 0.41	1.19 ± 0.67	2.38 ± 0.21	1.91 ± 0.29	0.10 ± 0.01	0.08 ± 0.03	**0.07** ± 0.05	**0.08** ± 0.02
2	1/4	GFM	77.89 ± 1.09	68.13 ± 1.02	52.54 ± 0.27	50.84 ± 0.48	51.98 ± 0.39	51.79 ± 0.33	3.01 ± 0.71	0.67 ± 0.87	2.04 ± 0.33	1.91 ± 0.06	0.09 ± 0.05	0.10 ± 0.03	0.12 ± 0.03	0.10 ± 0.03
2	1/4	GFR	78.19 ± 1.32	67.44 ± 1.26	52.17 ± 0.15	50.67 ± 0.78	51.73 ± 0.31	51.53 ± 0.20	1.89 ± 0.66	0.59 ± 0.01	1.53 ± 0.34	1.34 ± 0.10	0.10 ± 0.03	0.08 ± 0.01	0.14 ± 0.06	0.11 ± 0.03
2	1/10	UFR	77.87 ± 1.03	63.64 ± 1.44	52.03 ± 0.47	50.58 ± 0.29	51.78 ± 0.42	51.46 ± 0.29	2.29 ± 0.18	0.39 ± 0.78	2.23 ± 0.27	1.64 ± 0.29	0.10 ± 0.02	0.08 ± 0.03	0.11 ± 0.01	0.10 ± 0.01
2	$1/10^4$	OSR	77.29 ± 0.72	32.30 ± 0.65	**51.06** ± 0.52	50.32 ± 0.43	**50.97** ± 0.51	**50.78** ± 0.35	**1.05** ± 0.62	0.23 ± 0.51	**1.10** ± 0.55	**0.79** ± 0.56	0.11 ± 0.03	0.10 ± 0.04	0.10 ± 0.04	0.11 ± 0.01
2	$1/10^4$	GSR	75.61 ± 1.34	37.87 ± 3.30	51.63 ± 0.16	**50.09** ± 0.14	51.19 ± 0.25	50.97 ± 0.12	1.87 ± 0.32	**0.03** ± 0.30	1.79 ± 0.62	1.23 ± 0.41	0.09 ± 0.02	0.07 ± 0.03	**0.07** ± 0.02	**0.08** ± 0.02
2	$1/10^4$	USR (FDropout)	76.26 ± 0.95	40.61 ± 1.55	51.52 ± 0.89	50.27 ± 0.19	51.19 ± 0.89	51.00 ± 0.54	2.48 ± 0.05	0.16 ± 0.09	1.23 ± 0.03	1.29 ± 0.02	**0.08** ± 0.03	0.11 ± 0.02	0.10 ± 0.05	0.10 ± 0.02
5	1	OFM (HeteroFL)	78.69 ± 0.59	71.00 ± 0.92	53.55 ± 0.37	51.38 ± 0.43	52.77 ± 0.25	52.56 ± 0.32	3.66 ± 0.62	1.87 ± 0.37	2.66 ± 0.08	2.73 ± 0.36	0.13 ± 0.02	0.10 ± 0.04	0.12 ± 0.04	0.12 ± 0.02
5	1	OFR	78.84 ± 0.65	**72.02** ± 0.37	53.76 ± 0.39	50.93 ± 0.19	52.68 ± 0.17	52.46 ± 0.07	4.24 ± 0.30	0.70 ± 0.51	3.60 ± 0.67	2.85 ± 0.05	**0.10** ± 0.03	0.07 ± 0.00	**0.07** ± 0.03	**0.08** ± 0.02
5	1/4	OSM	**79.00** ± 0.33	66.22 ± 1.96	52.72 ± 0.60	51.10 ± 0.04	51.94 ± 0.64	51.92 ± 0.41	3.38 ± 0.08	1.59 ± 0.48	**1.47** ± 0.18	2.15 ± 0.07	**0.10** ± 0.05	0.08 ± 0.01	0.13 ± 0.06	0.10 ± 0.04
5	1/4	GFM	78.89 ± 0.45	68.23 ± 0.64	53.04 ± 0.06	50.89 ± 0.22	52.19 ± 0.40	52.04 ± 0.18	4.11 ± 0.38	1.11 ± 0.22	2.66 ± 0.19	2.62 ± 0.12	0.12 ± 0.01	0.10 ± 0.05	0.09 ± 0.01	0.10 ± 0.02
5	1/4	GFR	78.95 ± 0.48	68.92 ± 0.30	52.58 ± 0.43	**50.76** ± 0.15	51.92 ± 0.35	51.75 ± 0.18	**2.55** ± 0.06	1.14 ± 0.00	1.87 ± 0.53	**1.85** ± 0.16	0.12 ± 0.03	**0.06** ± 0.02	0.10 ± 0.03	0.09 ± 0.01
5	1/10	UFR	78.83 ± 0.56	64.68 ± 0.70	52.56 ± 0.44	51.04 ± 0.43	51.87 ± 0.46	51.83 ± 0.18	2.65 ± 0.76	1.01 ± 0.02	2.05 ± 0.83	1.90 ± 0.54	0.12 ± 0.02	0.10 ± 0.04	0.08 ± 0.02	0.10 ± 0.01
5	$1/10^4$	OSR	78.86 ± 0.48	51.41 ± 3.51	52.64 ± 0.22	51.23 ± 0.24	51.89 ± 0.53	51.92 ± 0.17	3.41 ± 0.73	1.55 ± 0.23	2.68 ± 0.62	2.55 ± 0.37	0.12 ± 0.02	0.10 ± 0.00	0.08 ± 0.01	0.10 ± 0.01
5	$1/10^4$	GSR	78.50 ± 0.95	52.09 ± 1.57	52.49 ± 0.10	50.89 ± 0.38	51.82 ± 0.30	51.73 ± 0.03	3.12 ± 0.42	**0.76** ± 0.50	1.92 ± 0.83	1.93 ± 0.25	0.12 ± 0.03	0.08 ± 0.03	0.09 ± 0.04	0.10 ± 0.03
5	$1/10^4$	USR (FDropout)	78.44 ± 0.38	52.31 ± 1.11	**52.29** ± 0.34	51.11 ± 0.49	**51.67** ± 0.31	**51.69** ± 0.22	2.69 ± 0.58	1.00 ± 0.01	2.39 ± 0.53	2.03 ± 0.37	0.13 ± 0.04	0.09 ± 0.02	0.11 ± 0.04	0.11 ± 0.01
8	1	OFM (HeteroFL)	79.07 ± 0.42	**73.96** ± 0.48	54.19 ± 0.42	51.88 ± 0.15	53.10 ± 0.36	53.06 ± 0.29	3.63 ± 0.28	2.03 ± 0.07	3.55 ± 0.83	3.07 ± 0.39	0.13 ± 0.04	0.09 ± 0.01	0.11 ± 0.04	0.11 ± 0.02
8	1	OFR	78.69 ± 0.20	73.37 ± 0.39	53.91 ± 0.26	51.76 ± 0.13	53.07 ± 0.10	52.92 ± 0.11	4.18 ± 0.20	2.12 ± 0.07	2.86 ± 1.39	3.05 ± 0.51	0.14 ± 0.02	0.10 ± 0.02	0.12 ± 0.05	0.12 ± 0.02
8	1/4	OSM	78.85 ± 0.60	70.91 ± 1.25	53.40 ± 0.33	51.96 ± 0.43	52.63 ± 0.42	52.66 ± 0.15	**3.06** ± 0.57	2.15 ± 0.61	2.74 ± 0.50	**2.65** ± 0.23	0.11 ± 0.02	0.09 ± 0.03	**0.07** ± 0.03	**0.09** ± 0.01
8	1/4	GFM	78.82 ± 0.06	71.02 ± 0.38	53.93 ± 0.22	**51.50** ± 0.28	52.84 ± 0.02	52.76 ± 0.16	4.01 ± 0.74	**1.76** ± 0.55	3.38 ± 0.87	3.05 ± 0.72	0.11 ± 0.03	0.09 ± 0.03	0.12 ± 0.04	0.11 ± 0.03
8	1/4	GFR	78.64 ± 0.71	72.26 ± 1.06	53.63 ± 0.47	52.07 ± 0.30	52.53 ± 0.39	52.74 ± 0.27	3.21 ± 0.11	2.06 ± 0.55	2.95 ± 0.53	2.74 ± 0.03	0.16 ± 0.06	**0.08** ± 0.02	0.16 ± 0.04	0.13 ± 0.03
8	1/10	UFR	**79.10** ± 0.33	68.97 ± 1.19	53.94 ± 0.72	51.79 ± 0.43	53.09 ± 0.43	52.94 ± 0.44	3.92 ± 1.58	2.05 ± 1.13	3.07 ± 0.47	3.01 ± 1.06	0.12 ± 0.01	0.09 ± 0.03	0.12 ± 0.02	0.11 ± 0.01
8	$1/10^4$	OSR	78.82 ± 0.72	67.32 ± 0.08	53.27 ± 0.35	52.19 ± 0.16	52.52 ± 0.37	52.66 ± 0.27	3.33 ± 0.67	2.33 ± 0.81	2.93 ± 0.98	2.86 ± 0.37	**0.09** ± 0.02	0.10 ± 0.05	0.11 ± 0.02	0.10 ± 0.01
8	$1/10^4$	GSR	78.38 ± 0.25	67.51 ± 0.32	**53.08** ± 0.25	51.68 ± 0.37	**52.45** ± 0.29	**52.40** ± 0.27	3.74 ± 0.80	1.94 ± 0.64	**2.46** ± 0.12	2.71 ± 0.09	0.14 ± 0.04	**0.08** ± 0.00	0.12 ± 0.03	0.12 ± 0.01
8	$1/10^4$	USR (FDropout)	78.87 ± 0.30	67.74 ± 0.38	53.55 ± 0.57	51.95 ± 0.25	52.72 ± 0.33	52.74 ± 0.36	3.65 ± 1.62	2.18 ± 0.05	2.58 ± 0.37	2.80 ± 0.68	0.13 ± 0.01	0.11 ± 0.02	0.13 ± 0.03	0.13 ± 0.01

Table 25. Detailed results on the CIFAR-100 dataset. Experiments averaged over 3 runs. Best in each category highlighted with bold. Methods are grouped by number of *large* clients and ordered based on the frequency a client receives the same parameters.

No. of *large*	Freq.	Method name	↑ Server Acc %	↑ Client Acc %	↓ AUC % tMIA	↓ AUC % LiRA	↓ AUC % Yeom	↓ AUC % Avg	↓ Attack Adv % tMIA	↓ Attack Adv % LiRA	↓ Attack Adv % Yeom	↓ Attack Adv % Avg	↓ TPR% at 0.1% FPR tMIA	↓ TPR% at 0.1% FPR LiRA	↓ TPR% at 0.1% FPR Yeom	↓ TPR% at 0.1% FPR Avg
10	1	FedAvg100k	**45.44** ± 0.75	**44.98** ± 0.52	55.20 ± 0.06	51.82 ± 0.69	54.65 ± 0.12	53.89 ± 0.19	4.64 ± 0.69	1.60 ± 0.80	3.10 ± 0.33	3.11 ± 0.52	0.13 ± 0.02	0.10 ± 0.02	0.06 ± 0.01	0.10 ± 0.01
0	1	FedAvg30k	34.01 ± 0.43	33.88 ± 0.67	**52.78** ± 0.12	**50.91** ± 0.30	**52.68** ± 0.13	**52.12** ± 0.03	**2.89** ± 1.19	**0.35** ± 1.15	**1.10** ± 2.17	**1.45** ± 0.89	**0.09** ± 0.03	**0.09** ± 0.04	**0.05** ± 0.01	**0.08** ± 0.02
2	1	OFM (HeteroFL)	44.14 ± 2.59	32.46 ± 0.99	54.71 ± 0.19	51.15 ± 0.38	54.31 ± 0.18	53.39 ± 0.23	4.07 ± 1.51	0.95 ± 0.67	3.04 ± 3.11	2.69 ± 1.24	0.17 ± 0.03	0.10 ± 0.05	0.07 ± 0.06	0.11 ± 0.03
2	1	OFR	44.36 ± 1.86	**33.02** ± 0.72	54.62 ± 0.26	51.33 ± 0.36	54.28 ± 0.17	53.41 ± 0.19	6.11 ± 2.23	1.50 ± 0.49	2.43 ± 1.58	3.35 ± 0.63	0.10 ± 0.01	**0.07** ± 0.01	**0.06** ± 0.03	**0.08** ± 0.01
2	1/4	OSM	**45.13** ± 2.25	22.54 ± 2.16	53.06 ± 0.36	51.05 ± 0.36	52.97 ± 0.42	52.36 ± 0.19	3.94 ± 2.46	0.92 ± 1.81	1.70 ± 1.79	2.19 ± 1.60	0.10 ± 0.04	0.09 ± 0.02	0.09 ± 0.05	0.09 ± 0.04
2	1/4	GFM	43.05 ± 1.76	31.70 ± 1.37	53.11 ± 0.62	50.96 ± 0.49	52.95 ± 0.38	52.34 ± 0.31	4.24 ± 1.58	0.50 ± 0.85	**0.88** ± 2.32	1.88 ± 0.14	0.10 ± 0.03	0.08 ± 0.02	0.11 ± 0.04	0.10 ± 0.03
2	1/4	GFR	43.17 ± 2.74	30.44 ± 0.50	53.08 ± 0.39	51.10 ± 0.22	53.08 ± 0.39	52.42 ± 0.24	4.47 ± 0.61	1.27 ± 0.47	2.13 ± 1.55	2.62 ± 0.47	0.12 ± 0.03	0.09 ± 0.04	0.09 ± 0.01	0.10 ± 0.02
2	1/10	UFR	43.16 ± 1.83	26.58 ± 1.38	52.66 ± 0.56	50.79 ± 0.13	52.73 ± 0.48	52.06 ± 0.30	2.68 ± 1.70	**-0.23** ± 1.61	1.29 ± 1.29	**1.25** ± 0.48	0.12 ± 0.04	0.11 ± 0.02	0.07 ± 0.02	0.10 ± 0.01
2	$1/10^4$	OSR	43.05 ± 1.53	10.85 ± 0.28	52.28 ± 0.22	50.87 ± 0.26	52.11 ± 0.12	51.75 ± 0.04	2.86 ± 1.54	0.75 ± 1.96	0.98 ± 3.26	1.53 ± 0.28	0.11 ± 0.06	0.08 ± 0.04	0.14 ± 0.10	0.11 ± 0.04
2	$1/10^4$	GSR	42.32 ± 1.99	12.51 ± 0.94	**52.12** ± 0.70	50.68 ± 0.15	**51.89** ± 0.61	**51.56** ± 0.40	**2.24** ± 2.40	0.99 ± 0.71	1.08 ± 2.60	1.43 ± 0.46	0.10 ± 0.03	**0.07** ± 0.03	0.10 ± 0.01	0.09 ± 0.01
2	$1/10^4$	USR (FDropout)	41.31 ± 1.67	12.85 ± 0.62	52.36 ± 0.62	**50.54** ± 0.10	52.03 ± 0.35	51.65 ± 0.33	3.41 ± 2.31	1.51 ± 0.31	1.15 ± 0.78	2.02 ± 0.91	**0.09** ± 0.02	**0.07** ± 0.02	0.10 ± 0.02	0.09 ± 0.01
5	1	OFM (HeteroFL)	44.98 ± 1.22	35.89 ± 0.21	55.23 ± 0.31	51.58 ± 0.09	54.75 ± 0.12	53.85 ± 0.14	4.87 ± 2.11	1.89 ± 0.87	3.75 ± 1.41	3.50 ± 0.06	0.14 ± 0.03	0.09 ± 0.04	0.05 ± 0.01	0.10 ± 0.03
5	1	OFR	45.39 ± 1.33	**36.08** ± 0.36	54.82 ± 0.15	51.34 ± 0.26	54.50 ± 0.16	53.55 ± 0.06	5.11 ± 1.48	1.06 ± 0.61	**1.31** ± 2.56	2.49 ± 0.83	0.11 ± 0.05	**0.08** ± 0.02	0.09 ± 0.02	**0.09** ± 0.02
5	1/4	OSM	**45.70** ± 1.24	29.81 ± 0.66	53.97 ± 0.46	51.14 ± 0.66	53.69 ± 0.25	52.93 ± 0.36	4.85 ± 0.43	**-0.36** ± 0.84	2.49 ± 1.66	2.33 ± 0.38	0.13 ± 0.02	0.11 ± 0.02	**0.06** ± 0.03	0.10 ± 0.01
5	1/4	GFM	44.57 ± 0.22	32.49 ± 0.66	53.92 ± 0.02	51.38 ± 0.31	53.59 ± 0.05	52.96 ± 0.11	4.82 ± 1.32	1.49 ± 0.61	2.50 ± 1.53	2.94 ± 0.83	**0.10** ± 0.01	0.07 ± 0.01	0.11 ± 0.03	0.10 ± 0.01
5	1/4	GFR	45.22 ± 1.00	32.81 ± 0.37	54.04 ± 0.08	51.61 ± 0.37	53.77 ± 0.15	53.14 ± 0.18	4.04 ± 0.89	0.87 ± 0.62	1.62 ± 1.54	**2.18** ± 0.85	**0.10** ± 0.01	0.12 ± 0.04	0.07 ± 0.02	0.10 ± 0.00
5	1/10	UFR	44.98 ± 0.59	29.36 ± 1.24	53.78 ± 0.27	51.19 ± 0.09	53.57 ± 0.29	52.85 ± 0.16	5.52 ± 1.74	0.94 ± 1.29	1.88 ± 0.79	2.78 ± 1.10	0.12 ± 0.03	0.10 ± 0.02	0.08 ± 0.00	0.10 ± 0.01
5	$1/10^4$	OSR	44.92 ± 1.22	24.27 ± 0.54	53.76 ± 0.73	50.88 ± 0.32	53.19 ± 0.47	52.61 ± 0.37	5.02 ± 0.81	1.38 ± 0.81	2.80 ± 0.79	3.07 ± 0.70	0.12 ± 0.03	0.09 ± 0.06	0.11 ± 0.02	0.11 ± 0.03
5	$1/10^4$	GSR	44.84 ± 0.89	24.20 ± 0.40	**53.56** ± 0.34	**50.81** ± 0.24	53.24 ± 0.12	**52.54** ± 0.22	**3.98** ± 0.46	1.41 ± 0.44	2.09 ± 1.72	2.49 ± 0.49	**0.10** ± 0.03	0.10 ± 0.02	0.07 ± 0.01	**0.09** ± 0.02
5	$1/10^4$	USR (FDropout)	45.52 ± 0.60	24.41 ± 0.10	53.73 ± 0.08	51.16 ± 0.18	**53.06** ± 0.09	52.65 ± 0.12	4.04 ± 1.25	1.13 ± 1.92	1.96 ± 1.90	2.38 ± 0.84	**0.10** ± 0.01	0.12 ± 0.00	0.07 ± 0.04	0.10 ± 0.01
8	1	OFM (HeteroFL)	44.85 ± 1.32	39.02 ± 1.43	54.95 ± 0.26	51.59 ± 0.34	54.61 ± 0.23	53.72 ± 0.21	6.01 ± 1.22	2.40 ± 0.67	2.17 ± 0.27	3.53 ± 0.64	**0.09** ± 0.04	0.10 ± 0.01	**0.07** ± 0.04	**0.09** ± 0.03
8	1	OFR	45.71 ± 0.99	**40.06** ± 0.67	55.18 ± 0.35	51.63 ± 0.10	54.88 ± 0.07	53.90 ± 0.07	5.09 ± 1.22	1.37 ± 1.14	3.17 ± 1.31	3.21 ± 1.03	0.11 ± 0.03	0.15 ± 0.03	0.10 ± 0.01	0.12 ± 0.00
8	1/4	OSM	45.97 ± 0.62	37.98 ± 0.28	54.33 ± 0.29	51.59 ± 0.43	54.05 ± 0.22	53.32 ± 0.30	4.09 ± 0.49	1.17 ± 1.53	1.22 ± 2.29	2.16 ± 0.98	0.13 ± 0.03	0.09 ± 0.00	**0.07** ± 0.01	0.10 ± 0.01
8	1/4	GFM	45.55 ± 0.76	37.98 ± 1.28	54.49 ± 0.27	51.56 ± 0.22	54.01 ± 0.14	53.35 ± 0.08	6.14 ± 1.10	1.91 ± 1.19	**1.02** ± 1.62	3.02 ± 0.29	0.12 ± 0.01	0.09 ± 0.03	**0.07** ± 0.02	**0.09** ± 0.01
8	1/4	GFR	46.30 ± 0.56	38.44 ± 0.66	54.66 ± 0.26	51.55 ± 0.48	54.17 ± 0.14	53.46 ± 0.09	5.68 ± 1.43	1.72 ± 0.84	1.71 ± 1.91	3.04 ± 0.42	0.13 ± 0.05	0.09 ± 0.03	**0.07** ± 0.01	0.10 ± 0.02
8	1/10	UFR	**46.54** ± 0.69	38.03 ± 0.39	54.50 ± 0.13	51.74 ± 0.47	54.06 ± 0.20	53.43 ± 0.06	5.46 ± 0.31	1.22 ± 0.42	2.98 ± 1.45	3.22 ± 0.51	0.11 ± 0.04	**0.08** ± 0.02	0.09 ± 0.02	**0.09** ± 0.01
8	$1/10^4$	OSR	45.30 ± 0.47	37.03 ± 0.25	**54.00** ± 0.20	51.68 ± 0.25	53.68 ± 0.05	53.12 ± 0.16	4.56 ± 1.31	1.97 ± 0.47	2.73 ± 0.73	3.09 ± 0.82	0.19 ± 0.00	0.10 ± 0.03	0.12 ± 0.07	0.13 ± 0.01
8	$1/10^4$	GSR	45.65 ± 0.42	36.81 ± 0.59	54.03 ± 0.21	51.74 ± 0.45	53.68 ± 0.34	53.15 ± 0.33	**3.72** ± 0.66	**0.56** ± 0.95	1.27 ± 1.01	**1.85** ± 0.61	0.14 ± 0.02	0.09 ± 0.04	**0.07** ± 0.02	0.10 ± 0.02
8	$1/10^4$	USR (FDropout)	45.55 ± 0.44	37.14 ± 0.28	53.77 ± 0.35	**51.48** ± 0.36	**53.47** ± 0.54	**52.91** ± 0.41	4.45 ± 0.23	1.26 ± 0.50	2.19 ± 2.23	2.63 ± 0.66	0.12 ± 0.03	0.09 ± 0.01	0.13 ± 0.05	0.12 ± 0.01

Table 26. Detailed results on the FEMNIST dataset. Experiments averaged over 3 runs. Best in each category highlighted with bold. Methods are grouped by number of *large* clients and ordered based on the frequency a client receives the same parameters.

# large *clients*	Freq.	Method name	↑ Server Acc %	↑ Client Acc %	↓ AUC %				↓ Attack Adv %				↓ TPR% at 0.1% FPR			
					tMIA	LiRA	Yeom	Avg	tMIA	LiRA	Yeom	Avg	tMIA	LiRA	Yeom	Avg
10	1	FedAvg100k	**87.20** ± 0.46	**87.18** ± 0.42	56.41 ± 0.43	53.17 ± 0.25	55.98 ± 0.10	55.18 ± 0.10	7.67 ± 0.23	5.20 ± 0.38	7.26 ± 0.42	6.71 ± 0.25	0.27 ± 0.02	0.25 ± 0.06	0.15 ± 0.03	0.22 ± 0.01
0	1	FedAvg30k	84.10 ± 0.97	84.20 ± 1.04	**55.45** ± 0.51	**51.20** ± 0.16	**55.41** ± 0.58	**54.02** ± 0.38	**6.86** ± 0.71	**2.26** ± 0.26	**7.01** ± 0.61	**5.38** ± 0.38	**0.15** ± 0.02	**0.14** ± 0.02	**0.16** ± 0.02	**0.15** ± 0.01
2	1	OFM (HeteroFL)	83.19 ± 0.83	82.59 ± 0.26	57.24 ± 0.73	52.62 ± 0.24	56.74 ± 0.37	55.53 ± 0.29	9.21 ± 0.43	4.02 ± 0.51	8.50 ± 0.54	7.24 ± 0.40	0.24 ± 0.04	0.16 ± 0.05	0.13 ± 0.03	0.18 ± 0.01
2	1	OFR	82.95 ± 0.09	**82.92** ± 1.04	56.93 ± 0.63	52.51 ± 0.34	56.49 ± 0.60	55.31 ± 0.35	8.68 ± 0.92	3.09 ± 1.07	8.17 ± 1.00	6.65 ± 0.83	0.28 ± 0.06	0.11 ± 0.03	0.11 ± 0.01	0.17 ± 0.02
2	1/4	OSM	84.39 ± 0.60	73.18 ± 0.81	56.47 ± 0.46	51.55 ± 0.39	56.33 ± 0.48	54.78 ± 0.18	8.60 ± 0.57	2.35 ± 0.64	8.39 ± 0.72	6.44 ± 0.26	0.22 ± 0.03	0.15 ± 0.06	**0.08** ± 0.03	0.15 ± 0.01
2	1/4	GFM	85.23 ± 0.21	81.80 ± 0.19	56.27 ± 0.58	51.72 ± 0.17	56.18 ± 0.56	54.72 ± 0.30	8.29 ± 0.99	2.69 ± 0.18	7.66 ± 1.11	6.21 ± 0.45	0.22 ± 0.02	0.16 ± 0.02	0.13 ± 0.02	0.17 ± 0.01
2	1/4	GFR	83.76 ± 0.71	81.32 ± 0.27	55.95 ± 0.84	51.78 ± 0.18	56.10 ± 0.62	54.61 ± 0.41	7.56 ± 0.82	2.39 ± 0.88	7.21 ± 1.41	5.72 ± 0.46	0.19 ± 0.04	0.20 ± 0.13	0.15 ± 0.05	0.18 ± 0.07
2	1/10	UFR	84.42 ± 0.39	79.12 ± 0.01	55.53 ± 0.27	51.15 ± 0.37	55.54 ± 0.34	54.08 ± 0.12	7.13 ± 0.80	1.95 ± 0.97	6.44 ± 0.38	5.17 ± 0.23	**0.16** ± 0.03	0.13 ± 0.05	0.13 ± 0.03	0.14 ± 0.00
2	$1/10^4$	OSR	83.26 ± 1.22	31.17 ± 1.30	54.93 ± 0.59	50.49 ± 0.46	55.14 ± 0.77	53.52 ± 0.43	6.26 ± 0.88	1.06 ± 0.89	6.02 ± 0.86	4.45 ± 0.53	0.17 ± 0.04	0.12 ± 0.03	0.09 ± 0.04	0.13 ± 0.02
2	$1/10^4$	GSR	84.78 ± 1.01	41.59 ± 1.78	54.80 ± 0.25	50.28 ± 0.57	**54.95** ± 0.34	53.35 ± 0.07	**5.89** ± 0.11	0.73 ± 1.22	5.67 ± 1.01	**4.10** ± 0.72	**0.16** ± 0.01	**0.07** ± 0.01	0.10 ± 0.02	**0.11** ± 0.01
2	$1/10^4$	USR (FDropout)	**85.43** ± 0.54	44.05 ± 0.93	**54.78** ± 0.54	**50.18** ± 0.37	**54.95** ± 0.47	**53.31** ± 0.22	6.20 ± 1.02	**0.69** ± 0.69	**5.76** ± 0.89	4.22 ± 0.14	0.17 ± 0.05	0.10 ± 0.03	0.10 ± 0.02	0.12 ± 0.02
5	1	OFM (HeteroFL)	86.18 ± 0.25	**83.21** ± 0.30	57.19 ± 0.00	52.86 ± 0.20	57.23 ± 0.14	55.76 ± 0.02	8.88 ± 0.41	4.46 ± 0.86	8.70 ± 0.11	7.35 ± 0.46	0.28 ± 0.01	0.19 ± 0.04	**0.05** ± 0.00	0.17 ± 0.02
5	1	OFR	85.49 ± 0.22	83.10 ± 0.12	57.60 ± 0.33	52.99 ± 0.24	57.15 ± 0.25	55.91 ± 0.07	9.28 ± 0.70	5.24 ± 0.33	8.71 ± 1.50	7.74 ± 0.57	0.27 ± 0.03	0.20 ± 0.06	0.06 ± 0.01	0.18 ± 0.02
5	1	OSM	**86.74** ± 0.47	76.85 ± 1.05	56.20 ± 0.08	52.16 ± 0.34	56.22 ± 0.55	54.86 ± 0.29	7.42 ± 0.49	2.98 ± 0.38	7.57 ± 1.02	5.99 ± 0.54	**0.20** ± 0.06	0.16 ± 0.02	0.06 ± 0.01	**0.14** ± 0.02
5	1/4	GFM	86.33 ± 0.19	79.94 ± 0.36	56.55 ± 0.42	52.03 ± 0.22	56.37 ± 0.19	54.98 ± 0.14	9.23 ± 0.02	2.86 ± 0.75	7.21 ± 0.16	6.44 ± 0.30	0.25 ± 0.05	0.13 ± 0.01	**0.05** ± 0.03	**0.14** ± 0.01
5	1/4	GFR	86.27 ± 0.07	81.04 ± 0.29	56.86 ± 0.17	52.33 ± 0.44	56.64 ± 0.50	55.28 ± 0.34	8.52 ± 1.24	3.65 ± 0.59	8.08 ± 0.87	6.75 ± 0.62	0.26 ± 0.05	0.15 ± 0.04	0.08 ± 0.01	0.16 ± 0.03
5	1/10	UFR	86.27 ± 0.54	76.24 ± 0.49	56.24 ± 0.23	51.99 ± 0.28	56.18 ± 0.40	54.80 ± 0.18	8.09 ± 0.23	3.11 ± 0.20	8.06 ± 0.72	6.42 ± 0.31	0.30 ± 0.09	0.12 ± 0.03	0.07 ± 0.03	0.16 ± 0.03
5	$1/10^4$	OSR	86.54 ± 0.22	49.75 ± 3.31	**55.35** ± 0.18	**51.31** ± 0.45	**55.28** ± 0.51	**53.98** ± 0.35	6.91 ± 0.57	**1.68** ± 0.24	**6.33** ± 0.53	**4.97** ± 0.36	0.23 ± 0.06	**0.11** ± 0.04	0.07 ± 0.03	**0.14** ± 0.03
5	$1/10^4$	GSR	86.61 ± 0.07	53.28 ± 2.41	55.62 ± 0.14	51.64 ± 0.38	55.84 ± 0.22	54.37 ± 0.03	6.95 ± 1.57	2.36 ± 1.11	6.71 ± 0.85	5.34 ± 1.13	0.26 ± 0.02	0.14 ± 0.01	0.07 ± 0.05	0.16 ± 0.03
5	$1/10^4$	USR (FDropout)	86.58 ± 0.53	55.04 ± 1.73	55.52 ± 0.41	51.38 ± 0.37	55.59 ± 0.49	54.16 ± 0.29	**6.44** ± 0.88	2.10 ± 0.49	6.75 ± 0.84	5.10 ± 0.50	0.25 ± 0.12	0.17 ± 0.04	**0.05** ± 0.02	0.16 ± 0.02
8	1	OFM (HeteroFL)	87.04 ± 0.21	**83.93** ± 0.33	57.97 ± 0.32	53.41 ± 0.75	57.73 ± 0.15	56.37 ± 0.20	8.96 ± 0.77	4.22 ± 1.97	8.62 ± 0.12	7.27 ± 0.44	**0.27** ± 0.03	0.22 ± 0.04	**0.05** ± 0.06	0.18 ± 0.04
8	1	OFR	86.47 ± 0.99	83.44 ± 1.12	57.44 ± 0.05	53.23 ± 0.10	57.47 ± 0.01	56.05 ± 0.01	9.32 ± 0.71	4.79 ± 1.34	8.40 ± 0.09	7.51 ± 0.18	0.32 ± 0.02	0.25 ± 0.13	0.07 ± 0.06	0.21 ± 0.07
8	1/4	OSM	87.33 ± 0.14	79.80 ± 1.58	56.77 ± 0.53	52.66 ± 0.26	56.87 ± 0.01	55.43 ± 0.09	9.18 ± 0.69	3.84 ± 0.77	7.58 ± 0.33	6.87 ± 0.08	0.28 ± 0.04	**0.13** ± 0.05	0.12 ± 0.11	0.18 ± 0.03
8	1/4	GFM	87.38 ± 0.04	80.89 ± 0.80	**56.29** ± 0.27	52.72 ± 0.19	56.63 ± 0.26	55.21 ± 0.11	8.37 ± 0.17	4.38 ± 0.56	7.50 ± 1.56	6.75 ± 0.39	**0.27** ± 0.01	0.20 ± 0.01	0.13 ± 0.05	0.20 ± 0.02
8	1/4	GFR	87.12 ± 0.38	80.74 ± 0.68	56.86 ± 0.39	53.06 ± 0.49	56.79 ± 0.12	55.57 ± 0.01	8.98 ± 0.83	4.73 ± 0.87	8.52 ± 0.34	7.41 ± 0.45	0.23 ± 0.02	0.20 ± 0.04	0.07 ± 0.05	**0.17** ± 0.04
8	1/10	UFR	87.05 ± 1.28	76.97 ± 0.44	56.86 ± 0.70	52.91 ± 0.57	56.82 ± 0.20	55.53 ± 0.11	9.03 ± 0.69	4.32 ± 1.53	8.86 ± 0.53	7.40 ± 0.10	0.31 ± 0.06	0.20 ± 0.04	0.10 ± 0.03	0.20 ± 0.02
8	$1/10^4$	OSR	87.27 ± 0.36	72.63 ± 0.09	56.33 ± 0.20	52.62 ± 0.53	**56.45** ± 0.35	**55.13** ± 0.36	8.39 ± 1.25	3.86 ± 1.05	7.20 ± 1.37	6.48 ± 1.22	0.30 ± 0.03	0.20 ± 0.03	0.16 ± 0.11	0.22 ± 0.06
8	$1/10^4$	GSR	**87.64** ± 0.05	73.19 ± 0.40	56.46 ± 0.32	**52.61** ± 0.17	56.47 ± 0.14	55.18 ± 0.10	8.50 ± 0.78	**3.71** ± 0.42	**7.16** ± 0.51	**6.45** ± 0.29	0.32 ± 0.02	0.17 ± 0.04	0.10 ± 0.02	0.20 ± 0.03
8	$1/10^4$	USR (FDropout)	86.99 ± 0.05	72.76 ± 0.59	56.40 ± 0.13	52.63 ± 0.36	56.50 ± 0.37	55.18 ± 0.05	**8.07** ± 0.05	4.63 ± 1.16	7.70 ± 0.80	6.80 ± 0.67	0.30 ± 0.00	0.15 ± 0.01	0.12 ± 0.05	0.19 ± 0.01

Appendix C

Appendix for chapter 5

C.1 FedDiverse pseudo-code

Algorithm 6 contains the pseudo-code of the FedDiverse algorithm. Specifically:

- Algorithm 1 contains the pseudo-code for FedDiverse's pre-training phase.
- Algorithm 2 shows how clients in FedDiverse train the biased model.
- Algorithm 3 corresponds to how clients in FedDiverse train the attribute classifier.
- Algorithm 4 illustrates how FedDiverse computes the data heterogeneity triplets.
- Algorithm 5 contains FedDiverse's sampling strategy according to the data heterogeneity triplets.
- Algorithm 6 depicts the overall procedure.

Algorithm 1: - Pre-training(κ, T_0, τ_0)

Let: Model f, global parameters $\boldsymbol{\theta}^0$
for each round $t \in [T_0]$ **do**
  Randomly sample $\mathbb{S}^t \subseteq \mathbb{K}$ such that $|\mathbb{S}^t| = \kappa$
  $\mathcal{S}$ sends global parameters $\boldsymbol{\theta}^t$ to all the clients in $\mathbb{S}^t$
  for each client $k \in \mathbb{S}^t$, *in parallel* **do**
    $\boldsymbol{\theta}_k = \boldsymbol{\theta}^t$
    $\boldsymbol{\theta}_k^{t+1} = \text{ERM}(f(\boldsymbol{\theta}_k); \ell_{\text{CE}}; \mathbb{D}_k; \tau_0)$
    Send $\boldsymbol{\theta}_k^{t+1}$ to $\mathcal{S}$
  end for
  $\boldsymbol{\theta}^{t+1} = \sum_{k \in \mathbb{S}^t} \frac{n_k}{n} \boldsymbol{\theta}_k^{t+1}$
end for
Return $\boldsymbol{\theta}^{T_0}$

Algorithm 2: - Biased-model-training($\boldsymbol{\theta}^{T_0}$, τ_{bias}, q)

Let: $\ell_{\text{GCE}}(\bar{f}_k(\boldsymbol{x};\boldsymbol{\theta}), y) = \frac{1-\bar{f}_k(\boldsymbol{x};\boldsymbol{\theta})^q}{q}$
Initialize parameters $\boldsymbol{\theta}_k$ of the biased model $\bar{f}_k$ with pre-trained parameters $\boldsymbol{\theta}^{T_0}$
$\boldsymbol{\theta}_k^{\text{bias}} = \text{ERM}(\bar{f}_k(\boldsymbol{\theta}_k); \ell_{\text{GCE}}; \mathbb{D}_k; \tau_{\text{bias}})$
$\tilde{\mathbb{G}}_k$ = set of inputs $\boldsymbol{x}$ such that the samples $(\boldsymbol{x}, y) \in \mathbb{D}_k$ are correctly predicted by $\bar{f}_k$, *i.e.* $\bar{f}_k(\boldsymbol{x}; \boldsymbol{\theta}_k^{\text{bias}}) = y$
$\tilde{\mathbb{g}}_k$ = set of inputs $\boldsymbol{x}$ such that the samples $(\boldsymbol{x}, y) \in \mathbb{D}_k$ are incorrectly predicted by $\bar{f}_k$, *i.e.* $\bar{f}_k(\boldsymbol{x}; \boldsymbol{\theta}_k^{\text{bias}}) \neq y$
Return $\boldsymbol{\theta}_k^{\text{bias}}$, $\tilde{\mathbb{G}}_k$, $\tilde{\mathbb{g}}_k$

Algorithm 3: - Attribute-classifier-training($\tilde{\mathbb{G}}^k$, $\tilde{\mathbb{g}}^k$, $\boldsymbol{\theta}_k^{\text{bias}}$, τ_{attr})

Let: $\hat{f} = \hat{\psi} \circ \varphi$; $\hat{\psi}$ is the attribute classifier, φ is the feature extractor
Compute the pivot class $\hat{y} = \arg\min_{y \in \mathcal{Y}} \left| \left| \tilde{\mathbb{G}}_y^k \right| - \left| \tilde{\mathbb{g}}_y^k \right| \right|$
Construct the dataset $\hat{\mathbb{D}}_k$ of pairs $(\boldsymbol{x}, \tilde{a})$, for all $\boldsymbol{x}$ such that $(\boldsymbol{x}, \hat{y}) \in \mathbb{D}_k$. $\tilde{a} = 0$ if $\boldsymbol{x} \in \mathbb{G}_{\hat{y}}^k$, $\tilde{a} = 1$ otherwise
Initialize parameters $\boldsymbol{\theta}_k = \boldsymbol{\theta}_k^{\text{bias}}$
Fix the parameters of the feature extractor φ
$\boldsymbol{\theta}_k^{\text{attr}} = \text{ERM}(\hat{f}(\boldsymbol{\theta}_k); \ell_{\text{CE}}; \mathbb{D}_k; \tau_{\text{attr}})$
Return $\boldsymbol{\theta}_k^{\text{attr}}$, $\hat{y}$

Algorithm 4: - DHT-computation($\hat{y}$, $\tilde{\mathbb{G}}_{\hat{y}}^k$, $\tilde{\mathbb{g}}_{\hat{y}}^k$)

Let: $\mathbb{D}_k^y \subset \mathbb{D}_k$ is the set of images in $\mathbb{D}_k$ with label y
$\tilde{\boldsymbol{R}} = 0_{|\mathcal{Y}| \times 2}$
for each $y \in \mathcal{Y}$: **do**
  if $y = \hat{y}$ **then**
    $\tilde{\boldsymbol{R}}_{y0} = \left| \tilde{\mathbb{G}}_{\hat{y}}^k \right|$, $\tilde{\boldsymbol{R}}_{y1} = \left| \tilde{\mathbb{g}}_{\hat{y}}^k \right|$
  else
    for each $(x, y) \in \mathbb{D}_k^y$ **do**
      $\tilde{a} = \hat{f}(\boldsymbol{x}; \boldsymbol{\theta}_k^{\text{attr}})$
      $\tilde{\boldsymbol{R}}_{y\tilde{a}} = \tilde{\boldsymbol{R}}_{y\tilde{a}} + 1$
    end for
  end if
end for
$\tilde{\Delta}^k = [\Delta_{\text{CI}}(\tilde{\boldsymbol{R}}^k), \Delta_{\text{AI}}(\tilde{\boldsymbol{R}}^k), \Delta_{\text{SC}}(\tilde{\boldsymbol{R}}^k)]^\top$
Return $\tilde{\Delta}^k$

Algorithm 5: - Triplet-sampling($\tilde{\Delta}$, $\mathbb{K}_{\text{left}}$, i)

1) Probabilistic selection

Compute the probability vector $p = \frac{\tilde{\Delta}_i}{\left\|\tilde{\Delta}_i\right\|_1}$

Sample $k_p \in \mathbb{K}_{\text{left}}$ according to p

Remove k_p from $\mathbb{K}_{\text{left}}$: $\mathbb{K}_{\text{left}} = \mathbb{K}_{\text{left}} \setminus \{k_p\}$

2) Complementary selection

Compute the normalized matrix $\underline{\tilde{\Delta}}$ such that $\underline{\tilde{\Delta}}^k = \frac{\tilde{\Delta}^k}{\left\|\tilde{\Delta}^k\right\|_1}$, $\forall k \in \mathbb{K}$

Find the complementary client $k_c = \arg\min_{k \in \mathbb{K}_{\text{left}}} \left\langle \underline{\tilde{\Delta}}^{k_p}, \underline{\tilde{\Delta}}^k \right\rangle$

Remove k_c from $\mathbb{K}_{\text{left}}$: $\mathbb{K}_{\text{left}} = \mathbb{K}_{\text{left}} \setminus \{k_c\}$

3) Orthogonal selection

Find the remaining orthogonal client $k_r = \arg\max_{k \in \mathbb{K} \setminus \{k_p, k_c\}} \left\langle \underline{\tilde{\Delta}}^{k_p} \times \underline{\tilde{\Delta}}^{k_c}, \underline{\tilde{\Delta}}^k \right\rangle$

Remove k_r from $\mathbb{K}_{\text{left}}$: $\mathbb{K}_{\text{left}} = \mathbb{K}_{\text{left}} \setminus \{k_r\}$

Return k_p, k_c, k_r, $\mathbb{K}_{\text{left}}$

Algorithm 6: - FedDiverse algorithm. Here, we assume that $\kappa \mod 3 \equiv 0$ for readability. If $\kappa \mod 3 \not\equiv 0$, the last time algorithm 5 is executed in one round, it will return fewer clients accordingly.

Input: Number of clients sampled per round κ, number of pre-training rounds T_0, number of training rounds T, number of steps of local training τ_0, number of steps of local biased model training τ_{bias}, number of steps of local attribute classifier training τ_{attr}, ℓ_{GCE} hyper-parameter $q \in (0, 1]$, FL optimization algorithm Opt, FL aggregator Agg

$\tilde{\Delta} = 0_{3\times K}$

$\boldsymbol{\theta}^{T_0}$ = Pre-training(κ, T_0, τ_0)

$\mathcal{S}$ sends $\boldsymbol{\theta}^{T_0}$ to all the clients $k \in \mathbb{K}$

for each client $k \in \mathbb{K}$ *in parallel* **do**

  $\boldsymbol{\theta}_k^{\text{bias}}$, $\tilde{\mathbb{G}}^k$, $\tilde{\mathrm{g}}^k$ = Biased-model-training($\boldsymbol{\theta}^{T_0}$, τ_{bias}, q)

  $\boldsymbol{\theta}_k^{\text{attr}}$, $\hat{y}$ = Attribute-classifier-training($\tilde{\mathbb{G}}^k$, $\tilde{\mathrm{g}}^k$, $\boldsymbol{\theta}_k^{\text{bias}}$, τ_{attr})

  $\tilde{\Delta}^k$ = DHT-computation($\hat{y}$, $\tilde{\mathbb{G}}_{\hat{y}}^k$, $\tilde{\mathrm{g}}_{\hat{y}}^k$)

end for

for each round $t \in [T]$ **do**

  $\mathbb{K}_{\text{left}} = \mathbb{K}$

  $\mathbb{S}^t = \emptyset$

  while $|\mathbb{S}^t| < \kappa$ **do**

    k_p, k_c, k_r, $\mathbb{K}_{\text{left}}$ = Triplet-sampling($\tilde{\Delta}$, $\mathbb{K}_{\text{left}}$, $t \mod 3$)

    $\mathbb{S}^t = \mathbb{S}^t \cup \{k_p, k_c, k_r\}$

  end while

  $\mathcal{S}$ sends global parameters $\boldsymbol{\theta}^t$ (and, eventually, additional information) to all the clients in $\mathbb{S}^t$

  for each client $k \in \mathbb{S}^t$ *in parallel* **do**

    $\boldsymbol{\theta}_k^{t+1}$, ... = Opt($\boldsymbol{\theta}^t$, ...) # Specific parameters and returned values depend on the chosen Opt algorithm

    Send $\boldsymbol{\theta}_k^{t+1}$ and eventual other information to $\mathcal{S}$ # Specific message depends on the chosen Opt algorithm

  end for

  $\boldsymbol{\theta}^{t+1}$, ... =Agg($\{\boldsymbol{\theta}_k^{t+1}\}_{k\in\mathbb{S}^t}$, ...) # Specific parameters and returned values depend on the chosen Agg algorithm

end for

Apéndice D

Resumen en castellano

D.1 Introducción

El Aprendizaje Federado (Federated Learning, FL) es un enfoque de aprendizaje automático que busca abordar las preocupaciones relacionadas con la privacidad y la seguridad presentes en el aprendizaje automático centralizado. Consiste en una arquitectura colaborativa y distribuida, en la cual un servidor se comunica con múltiples clientes (por ejemplo, dispositivos móviles), de modo que estos mantienen localmente sus datos potencialmente sensibles y privados, y solo comparten con el servidor los pesos del modelo entrenado y cierta información de metadatos. La tarea principal del servidor es agregar los parámetros del modelo recibidos de los clientes para aprender un modelo global mejorado, que luego es redistribuido entre los clientes. El servidor central puede establecer distintas condiciones de finalización del proceso de entrenamiento y realizar la agregación del modelo utilizando diversas estrategias y optimizadores. Este enfoque para llevar a cabo aprendizaje distribuido ha sido descrito como un ejemplo de "privacidad desde el diseño" (privacy by design), y por tanto representa una solución prometedora en aplicaciones sensibles a la privacidad, como la atención médica y el sector bancario.

Desde su propuesta en 2017 por McMahan et al. [MMR+17], el área del aprendizaje federado ha crecido rápidamente. Una revisión exhaustiva realizada en 2021 describe los avances y problemas abiertos en FL, y recopila más de 500 trabajos relacionados [KMA+21]. En ella se analizan los distintos tipos de aprendizaje federado, se exponen los desafíos más relevantes, y se resumen las direcciones que la comunidad investigadora ha seguido para abordarlos. Esta tesis adopta la misma notación y definiciones presentadas en Kairouz et al. [KMA+21].

Tipos de FL En cuanto al alcance de un sistema de aprendizaje federado, consideramos dos categorías principales:

1. FL horizontal vs. vertical Dependiendo de la naturaleza de la distribución de los datos, los sistemas de FL se dividen en aprendizaje federado horizontal y vertical. El aprendizaje federado vertical implica que los clientes tienen acceso a diferentes atributos sobre los mismos individuos, relacionados mediante un identificador único. La motivación para colaborar en este contexto es construir modelos más precisos al incorporar información complementaria sobre los individuos, mientras se preserva su privacidad y se evita enviar los datos al servidor central. En contraste, el aprendizaje federado horizontal se refiere a una federación en la que las características son las mismas en todos los clientes. En este caso, la motivación para

colaborar es disponer de una mayor cantidad de datos al agregarlos de forma distribuida, manteniéndolos privados en los dispositivos de los clientes. El servidor central aprende un modelo con mejor desempeño que los modelos locales de cualquiera de los clientes individuales y lo envía de vuelta, permitiendo que todos se beneficien de la federación. En esta tesis, nos enfocamos en técnicas de aprendizaje federado horizontal.

2. FL entre silos vs. entre dispositivos Considerando la naturaleza de los clientes, existen dos tipos diferenciables de arquitecturas FL: entre silos y entre dispositivos [KMA+21]. En el aprendizaje federado entre silos, se espera que los clientes sean confiables, estén disponibles, tengan estado y sean direccionables. En cambio, en el aprendizaje federado entre dispositivos, los clientes son actores individuales, diversos y desconectados, que pueden no participar en la federación por diversas razones, como pérdida de conectividad o alto consumo energético.

Desafíos en FL El campo del aprendizaje federado abarca numerosos temas de investigación relevantes y se utiliza cada vez más en aplicaciones sensibles a la privacidad [LLGL24]. A continuación, resumimos los desafíos más importantes en FL y algunas estrategias comúnmente empleadas para abordarlos:

1. Eficiencia, cuyo objetivo es entrenar modelos más precisos de manera más rápida, con una comunicación reducida y liviana con los clientes, ya que esta representa un cuello de botella importante en las aplicaciones de FL [KMY+16; CSP+21; DLX+24; ZZL+24]. Este desafío también incluye la necesidad de construir sistemas FL robustos frente a la heterogeneidad de los datos, fenómeno comúnmente conocido como datos no-IID (no independientes ni idénticamente distribuidos) [MRA+12]. Un ejemplo realista donde estos problemas pueden manifestarse es una empresa que entrena modelos FL en los dispositivos móviles de usuarios distribuidos globalmente. Los datos pueden depender de las preferencias demográficas de los usuarios, mientras que la comunicación en línea (posiblemente en redes con tarifas) y las distancias geográficas pueden motivar el uso de comunicaciones comprimidas y robustas. Las estrategias para abordar este desafío incluyen la selección de clientes [CWJ22; NLQO22], la reducción parcial de la complejidad del modelo para clientes con recursos limitados [DDT21; NLQO25], y la personalización [LHBS21].

2. Protección frente a actores maliciosos. A diferencia del aprendizaje automático centralizado, donde el entrenamiento se realiza en un sistema central bajo control administrativo, en FL los clientes pueden comportarse de forma maliciosa y la comunicación con el servidor puede quedar expuesta a oyentes curiosos. Se distinguen los ataques a la privacidad, en los que observadores pasivos intentan extraer información sensible de las comunicaciones [TLC+20; NLQO25], y los ataques adversariales, en los que actores manipulan el entrenamiento federado para alcanzar sus propios fines [BCMC19; YDK+24]. Las defensas posibles incluyen restricciones en la comunicación, como el cómputo multipartito seguro [LZS+24], el uso de privacidad diferencial [TLC+20], optimizadores robustos [YDK+24], e incentivos para fomentar la participación de clientes confiables [STW19].

3. Equidad, que abarca tanto el desempeño justo —generalizado desde los conceptos tradicionales de equidad algorítmica a un entorno distribuido— como la colaboración justa, que aborda la heterogeneidad en la contribución y motivación de los clientes a través de esquemas de recompensa [SYL23; HYS+24; SACA24]. Las estrategias incluyen personalización [LHBS21], optimización con restricciones, evaluación de equidad en el servidor [KPDG22], e incentivos [NLQO22].

4. Desafíos del sistema, que incluyen problemas derivados de comunicaciones no confiables,

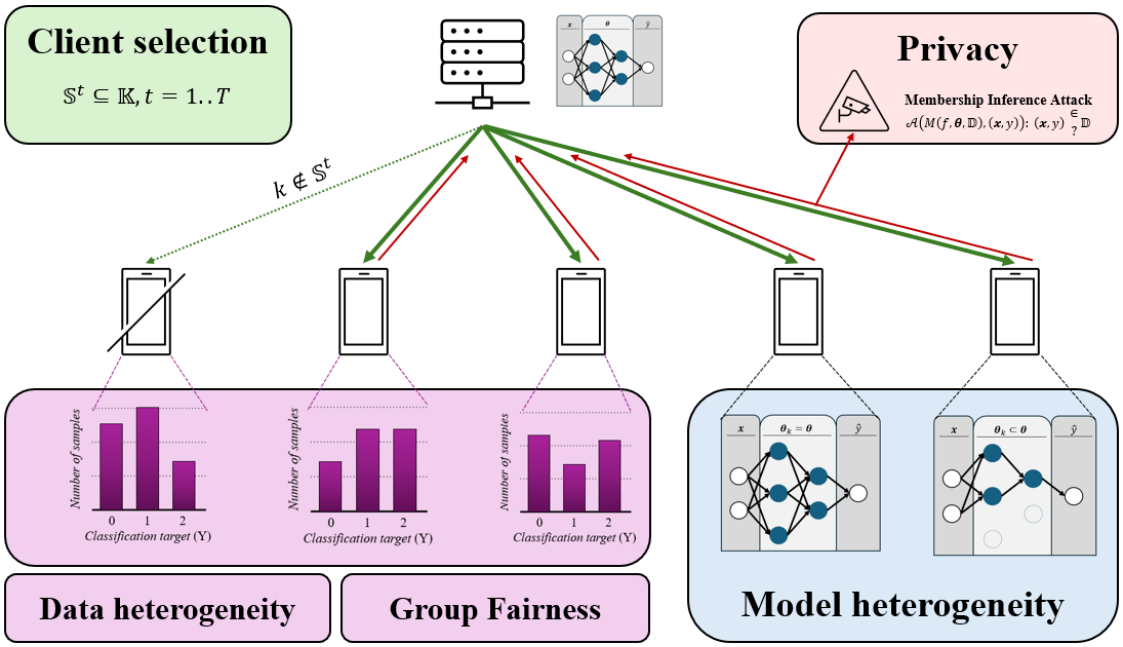


Figura 19. Resumen de los temas de investigación de esta tesis: en una arquitectura de aprendizaje federado, el servidor central puede utilizar la «selección de clientes» para abordar la «heterogeneidad de datos», reducir las «correlaciones espurias» no deseadas o mejorar la «equidad algorítmica». Además, el servidor permite a los clientes emplear la heterogeneidad de modelos: los clientes pueden optar por entrenar modelos de menor complejidad para mejorar su privacidad frente a ataques de inferencia de pertenencia.

dispositivos que se desconectan, o con capacidades computacionales heterogéneas [BEG+19]. Las soluciones propuestas incluyen agrupar clientes con capacidades similares [ZZWC20], reducir la carga computacional para clientes con recursos limitados [DDT21; NLQO25], y usar esquemas de agregación robustos frente a fallos [MÖB22].

La tabla 1 resume estos desafíos y estrategias. En esta tesis, destacamos los conceptos abordados con el objetivo de relajar las suposiciones fundamentales del aprendizaje federado horizontal respecto a la heterogeneidad. Investigamos los efectos de la heterogeneidad de los datos y diseñamos métodos que aprovechan tanto la heterogeneidad de modelos como las diferencias en la participación de los clientes.

D.2 Antecedentes

En este capítulo presentamos la notación matemática utilizada en esta tesis, en particular en lo que se refiere al *aprendizaje federado* (nos centramos en el aprendizaje federado horizontal en todos los capítulos), la *selección de clientes* y los *ataques de inferencia de pertenencia*. En los capítulos derivados de artículos de investigación, hemos actualizado las ecuaciones para mantener una notación coherente en toda la tesis [NLQO22; NLQO25; NFN+25]. Seguimos las prácticas comunes en FL, así como la notación empleada en el libro Deep Learning de Ian Goodfellow, Yoshua Bengio y Aaron Courville [GBC16]. La tabla 2 resume la notación utilizada a lo largo del documento.

Además, la figura 19 ilustra los retos y estrategias abordados en esta tesis.

D.2.1 Aprendizaje federado

El aprendizaje federado es una colaboración entre clientes, donde cada cliente $k \in \mathbb{K}$, con $|\mathbb{K}| = K$, tiene acceso a un conjunto de datos $\mathbb{D}_k$ considerado privado. Los clientes colaboran bajo la coordinación de un servidor central $\mathcal{S}$ para entrenar un modelo global $f : \mathcal{X} \longrightarrow \mathcal{Y}$ con parámetros $\boldsymbol{\theta} \in \Omega$.

En el FL vertical, el espacio de entrada global es una combinación de las entradas de los distintos clientes: $\bigcup_{k\in\mathbb{K}} \mathcal{X}k = \mathcal{X}$, con algún identificador compartido entre clientes, $\mathcal{X}id \subseteq \bigcap_{k\in\mathbb{K}} \mathcal{X}_k$. En el FL horizontal, los clientes comparten el mismo espacio de entrada, es decir, $\mathcal{X}_k = \mathcal{X}$. En este trabajo, nos enfocamos en el FL horizontal.

En este contexto horizontal, donde los clientes comparten el mismo espacio de entrada y salida, consideramos $\mathbb{D} = \bigcup_{i=1}^{K} \mathbb{D}_i$ como el conjunto de datos total y $|\mathbb{D}|$ como el número total de muestras distribuidas entre los clientes. Cada par $(\boldsymbol{x}, y) \in \mathbb{D}_k$ representa una muestra del conjunto de entrenamiento del cliente k, y ℓ denota la función de pérdida. El objetivo del servidor central es minimizar la función de coste global $\mathcal{L}(\boldsymbol{\theta})$, dada por la ecuación (38).

$$\min_{\boldsymbol{\theta}\in\Omega} \mathcal{L}(\boldsymbol{\theta}) = \frac{1}{|\mathbb{D}|} \sum_{k\in\mathbb{K}} \sum_{(\boldsymbol{x},y)\in\mathbb{D}_k} \ell(y, f(\boldsymbol{x}, \boldsymbol{\theta})) \tag{38}$$

Sin embargo, el servidor no tiene acceso directo a los datos privados de los clientes. Por ello, distribuye los parámetros del modelo global $\boldsymbol{\theta}$ a todos los clientes, quienes los utilizan como punto de partida para calcular sus parámetros locales $\boldsymbol{\theta}_k$, minimizando su función de coste local $\mathcal{L}_k(\boldsymbol{\theta}_k, \mathbb{D}_k)$ como se muestra en la ecuación (39).

$$\min_{\boldsymbol{\theta}_k\in\Omega} \mathcal{L}k(\boldsymbol{\theta}_k, \mathbb{D}_k) = \frac{1}{|\mathbb{D}_k|} \sum (\boldsymbol{x}, y) \in \mathbb{D}_k \ell(y, f(\boldsymbol{x}, \boldsymbol{\theta}_k)) \tag{39}$$

Una vez que el servidor ha recopilado los parámetros locales $\boldsymbol{\theta}_k$, los agrega para obtener la siguiente iteración del parámetro global $\boldsymbol{\theta}$, con el objetivo de seguir minimizando la función de coste global $\mathcal{L}(\boldsymbol{\theta})$. Cada iteración de actualización del modelo global se denomina *ronda de entrenamiento*, y los parámetros en la ronda t se denotan como $\boldsymbol{\theta}^t$. Una estrategia de agregación simple pondera los parámetros locales de cada cliente proporcionalmente al tamaño de su conjunto de datos, como se indica en la ecuación (40). El entrenamiento federado continúa hasta que se alcanza la convergencia de $\mathcal{L}$.

$$\boldsymbol{\theta}^{t+1} = \sum_{k\in\mathbb{K}} \frac{|\mathbb{D}_k|}{|\mathbb{D}|} \boldsymbol{\theta}_k^t \tag{40}$$

La comunicación con cada cliente tras cada actualización local puede suponer una sobrecarga significativa, especialmente cuando el número de clientes es elevado. Para mitigar este problema, McMahan et al. [MMR+17] introdujeron el algoritmo FedAvg, en el cual los clientes se comunican con el servidor solo después de e épocas locales. Desde el punto de vista del servidor, los parámetros globales se actualizan en rondas de entrenamiento $t = 1, ..., T$. Desde el punto de vista de los clientes, cada ronda de entrenamiento t consiste en $\tau = 1, ..., e$ épocas locales. Por lo tanto, los clientes realizan un total de eT épocas de entrenamiento.

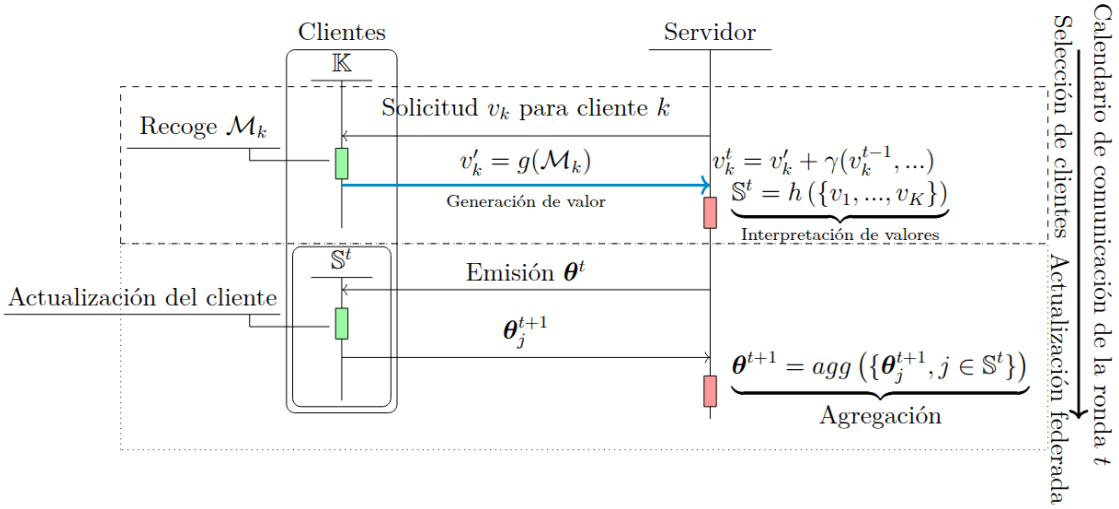


Figura 20. Diagrama de una ronda general de entrenamiento de aprendizaje federado con selección de clientes. El servidor se comunica con el conjunto de clientes $\mathbb{K}$ para seleccionar el subconjunto de clientes participantes $\mathbb{S}^t$. Sólo este subconjunto se entrena con sus datos locales en la ronda t e informa de sus parámetros al servidor. No todos los pasos son necesarios para todos los algoritmos.

D.2.2 Selección de clientes

En los capítulos 3 y 5, discutimos la *selección de clientes*, una técnica de FL en la que, en la ronda t, en lugar de comunicarse con todos los clientes en $\mathbb{K}$, el servidor elige un subconjunto de clientes ($\mathbb{S}^t$),

$$\mathbb{S}^t \subseteq \mathbb{K} \tag{41}$$

para que realicen el entrenamiento local y actualicen el modelo global en función de su respuesta.

FedAvg McMahan et al. [MMR+17] utiliza una selección aleatoria $\mathbb{S}^t$ de K' clientes participantes en cada ronda de entrenamiento t, donde $\mathbb{S}^t \subseteq \mathbb{K}$ y $|\mathbb{S}^t| = K'$, de modo que un cliente k se selecciona con probabilidad $p(k \in \mathbb{S}^t) = \frac{K'}{K}$. Los pasos de actualización y agregación de clientes se definen como $\boldsymbol{\theta}_j^{t+1} = \texttt{ClientUpdate}(\boldsymbol{\theta}^t, \mathbb{D}_j)$ y $\boldsymbol{\theta}^{t+1} = \texttt{agg}(\boldsymbol{\theta}_j^{t+1},\ j \in \mathbb{S}^t)$, respectivamente.

Después del trabajo seminal de FedAvg, investigaciones posteriores han propuesto diversos métodos de selección de clientes, la mayoría de los cuales siguen el flujo ilustrado en la figura 2. En primer lugar, el servidor solicita a cada cliente un valor v_k, que el cliente calcula basándose en su información local $\mathcal{M}_k$. Una vez que el servidor ha recibido los valores v_k de todos los clientes, selecciona un subconjunto $\mathbb{S}^t$ de ellos según la función $h(v_1, \ldots, v_K)$. Los métodos de selección *stateful* también utilizan información previa sobre los clientes, calculando v_k a partir de la retroalimentación v'_k y una función γ que toma en cuenta valores anteriores del cliente. Después, el servidor difunde sus parámetros globales $\boldsymbol{\theta}^t$ sólo a los clientes seleccionados $\mathbb{S}^t$, quienes actualizan sus modelos locales y devuelven sus parámetros actualizados $\boldsymbol{\theta}_k^{t+1}$. Con esta información, el servidor calcula el nuevo parámetro global $\boldsymbol{\theta}^{t+1}$.

Tabla 27. Dos grandes grupos de métodos FL con heterogeneidad del tamaño del modelo. **(a)** Métodos FL con selección dinámica del tamaño del modelo de los clientes. Se supone que todos los clientes tienen un modelo de la misma complejidad que el modelo del servidor. Estos métodos no son aplicables a entornos en los que los clientes tienen restricciones de datos y cálculo, como es nuestro caso. Por tanto, quedan fuera del ámbito de este documento. **b)** FL métodos en los que los clientes tienen un tamaño de modelo fijo, que puede ser menor que el tamaño del modelo del servidor. En este caso, los clientes aplican diferentes estrategias para seleccionar las porciones del modelo del servidor que utilizarán en su entrenamiento. La fuente azul corresponde a los nuevos métodos propuestos en esta tesis.

(a) Métodos FL con selección dinámica del tamaño del modelo de cliente.

		Métodos dinámicos de selección del tamaño	
		Random	**Gradiente**
Update	Cada ronda		FLADO [LGZX23]
	Cada batch	FJORD [HLA+21]	

(b) Métodos FL en los que los clientes tienen un tamaño de modelo fijo.

		Estrategia de selección	
		Remuestreado (S)	**Fijado (F)**
Cobertura	Un grupo (O)	OSM, OSR	HETEROFL [DDT21], OFR
	Varios grupos(G)	GSR	GFM, GFR
	Único (U)	FDROPOUT [CKMT18]	UFR

D.2.3 Heterogeneidad de los modelos

El método descrito anteriormente sigue el enfoque estándar de FL horizontal: los clientes y el servidor comparten la arquitectura del modelo, mientras mantienen los datos privados localmente. En el capítulo 4, investigamos un caso más generalizado, la *heterogeneidad de modelos*, en el que algunos clientes emplean un modelo con una complejidad reducida pero con una arquitectura similar. A nivel abstracto, lo ilustramos como:

$$f_k = f \text{ y } \boldsymbol{\theta}_k \subset \boldsymbol{\theta}, \tag{42}$$

lo que indica que la arquitectura del modelo del cliente k comparte la misma estructura que el modelo global, aunque sus parámetros constituyen un subconjunto de los del servidor.

Suponemos una arquitectura FL heterogénea aplicada a una tarea de visión por computador, en la que tanto el modelo del servidor como el de los clientes son redes neuronales convolucionales (CNN), con el mismo número de capas pero con distinto número de canales por capa.

Sean $\boldsymbol{\theta}$ los parámetros del modelo CNN del servidor, compuesto por L capas representadas mediante matrices de pesos $\boldsymbol{A}^{N\times M,l} \in \boldsymbol{\theta}$ en cada capa l. En este escenario, la reducción del modelo $\boldsymbol{\theta}_k \subset \boldsymbol{\theta}$ en el cliente k se logra limitando el tamaño de cada capa l en la CNN del cliente, según el siguiente principio: una capa en el servidor representada por la matriz de pesos $\boldsymbol{A}^{N\times M,l} \in \boldsymbol{\theta}$ se reduce a un tamaño $N_k \times M_k$, donde $N_k < N$ y $M_k < M$, de modo

que cada celda $a_k^{(i_k,j_k),l}$ en la matriz reducida $\boldsymbol{A}_k^{N_k\times M_k,l}$ corresponde a una celda $a^{(i,j),l}$ en la matriz del servidor $\boldsymbol{A}^{N\times M,l}$:

$$\forall i_k, j_k, l : a_k^{(i_k,j_k),l} \in \boldsymbol{A}_k^l, \exists i, j : a^{(i,j),l} \in \boldsymbol{A}, a_k^{(i_k,j_k),l} = a^{(i,j),l} \tag{43}$$

En este contexto, la literatura propone dos grandes grupos de métodos para integrar los modelos de los clientes en el del servidor, como se muestra en la tabla 27. El primer grupo, ilustrado en la tabla 27(a), incluye algoritmos que seleccionan *dinámicamente* el tamaño de los modelos de los clientes, aunque todos los clientes pueden manejar modelos del mismo tamaño que el del servidor.

Estos métodos definen una función $Ms(\cdot)$ que determina las dimensiones $N_k^l \times M_k^l$ del modelo del cliente k, es decir, qué subconjunto de la matriz de pesos $\boldsymbol{A}_k^l$ se emplea en la capa l del modelo $\boldsymbol{\theta}_k$. Entre los enfoques más representativos de esta categoría se encuentran FLADO[LGZX23] y FJORD[HLA+21]. Cabe señalar que estos métodos no son adecuados para entornos con restricciones de datos y capacidad de cómputo por parte de los clientes, como es el caso que abordamos. Por lo tanto, quedan fuera del alcance de esta tesis.

El segundo grupo, representado en la tabla 27(b) e ilustrado en la figura 8, abarca métodos en los que los clientes tienen *modelos de tamaño fijo*, generalmente más pequeños que el del servidor. Dado que trabajamos con CNNs, nos referimos a esta familia como métodos de selección de canales. Los pesos de cada capa l en una CNN 2D están definidos por un tensor de dimensiones $(N, M, H, W)^l$, donde M y N representan los canales de entrada y salida de la capa, y H y W son la altura y anchura del kernel, respectivamente.

En este contexto, $A^{N\times M,l}$ denota la matriz de pesos de cada capa lineal l en el modelo del servidor, donde M y N corresponden a las dimensiones de entrada y salida [ZTII88], y $a^{(i,j),l}$ representa los pesos del kernel en la posición (i, j). Aquí, la función $Ch(\cdot) : \boldsymbol{A}^{N\times M,l} \rightarrow \boldsymbol{A}_k^{N_k\times M_k,l}$ define el mapeo entre las celdas de la matriz de pesos del servidor $\boldsymbol{A}^l$ y la matriz $\boldsymbol{A}_k^l$ más pequeña del cliente k en cada capa l. Sin pérdida de generalidad, se asume que los canales están ordenados.

D.2.4 Heterogeneidad de los datos

En el capítulo 5, nos centramos en la heterogeneidad de los datos en el aprendizaje automático, específicamente en las *correlaciones espurias*, el *desequilibrio de clases* y el *desequilibrio de atributos*.

Sea $f : \mathcal{X} \rightarrow \mathcal{Y}$ una función de predicción parametrizada por $\boldsymbol{\theta} \in \Omega$, donde $\mathcal{X}$ es el espacio de características, $\mathcal{Y}$ es el espacio de salida y Ω es el espacio de parámetros. Suponemos que el espacio de características se compone de dos subespacios: $\mathcal{X} \subseteq \mathcal{X}y \times \mathcal{X}a$, donde $\mathcal{X}y$ y $\mathcal{X}a$ representan los espacios de características *intrínsecas a la tarea* y *de atributo*, respectivamente. La etiqueta de clase $y \in \mathcal{Y}$ de una muestra $\boldsymbol{x} \coloneqq (\boldsymbol{x}_y, \boldsymbol{x}_a)$ está determinada por la característica discriminativa $\boldsymbol{x}_y$, mientras que la etiqueta de atributo $a \in \mathcal{A}$ está determinada por la característica de atributo $\boldsymbol{x}_a$, donde $\mathcal{A}$ es el espacio de atributos. El conjunto de datos de entrenamiento $\mathbb{D}$ está compuesto por n pares de muestras característica-objetivo, $\mathbb{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, donde cada muestra se extrae de forma independiente e idénticamente distribuida según la distribución de entrenamiento $\Pr\text{tr}$.

La heterogeneidad estadística de los datos surge cuando hay un cambio en las subpoblaciones, es decir, cuando la representación de estas difiere entre las distribuciones de entrenamiento $\Pr_{\text{tr}}$ y de prueba $\Pr_{\text{te}}$. Aquí, las subpoblaciones se definen por las combinaciones

de etiquetas objetivo y atributos, es decir, por el espacio $\mathcal{Y} \times \mathcal{A}$. Consideramos tres tipos de heterogeneidad estadística en los datos:

Desequilibrio de clases (CI – Class imbalance)

La distribución de las etiquetas objetivo y difiere entre los conjuntos de entrenamiento y de prueba, de modo que ciertas clases son mucho más frecuentes en el conjunto de entrenamiento, *i.e.*: $\Pr_{\text{tr}}(Y = y) \gg \Pr_{\text{tr}}(Y = y')$ para algunos $y, y' \in \mathcal{Y}$ con $y \neq y'$. Este desequilibrio puede producir un clasificador sesgado que tenga un mal desempeño en muestras pertenecientes a la clase minoritaria.

Desequilibrio de atributos (AI – Attribute imbalance)

La probabilidad de aparición de un determinado atributo $a' \in \mathcal{A}$ en el conjunto de entrenamiento es mucho menor que la de otros atributos a, y esta disparidad en la prevalencia no se mantiene en la distribución de datos de prueba, *i.e.*: $\Pr_{\text{tr}}(A = a) \gg \Pr_{\text{tr}}(A = a')$. Este desequilibrio puede generar un clasificador sesgado hacia el atributo mayoritario a.

Correlación espuria (SC – Spurious correlation)

Existe una dependencia estadística entre la clase Y y el atributo A en la distribución de entrenamiento que no se mantiene en la distribución de prueba, *i.e.*: $\Pr_{\text{tr}}(Y = y \mid A = a) \gg \Pr_{\text{tr}}(Y = y) \gg \Pr_{\text{tr}}(Y = y \mid A = a')$, para algún $y \in \mathcal{Y}$ y $a, a' \in \mathcal{A}$. Esta dependencia espuria puede hacer que un clasificador funcione bien en muestras donde la relación espuria se mantiene (*e.g.*, $(Y = y, A = a)$), pero tenga un rendimiento inferior donde dicha relación no se presenta (*e.g.*, $(Y = y, A = a')$).

D.2.5 Privacidad

En el capítulo 4, analizamos la privacidad de los modelos de aprendizaje federado (FL). Existen dos enfoques principales para evaluarla: el uso de aproximaciones matemáticas para medir la privacidad diferencial [Dwo06], o la evaluación de la privacidad frente a ataques específicos. Los ataques pueden clasificarse según sus objetivos, como replicar el comportamiento del modelo objetivo o reconstruir información a partir de los datos de entrenamiento [LDS+21]. En este trabajo, empleamos ataques de inferencia de pertenencia (*membership inference attacks*, MIA), en los que el objetivo de un atacante curioso es determinar si un par de entrada-salida formaba parte de los datos de entrenamiento del modelo observado:

$$\mathscr{A}(M(f, \boldsymbol{\theta}, \mathbb{D}), (\boldsymbol{x}, y)) : (\boldsymbol{x}, y) \underset{?}{\in} \mathbb{D}. \tag{44}$$

Nos centramos en ataques de caja negra, es decir, aquellos en los que el atacante no tiene acceso directo a los parámetros del modelo $\boldsymbol{\theta}_g$ ni a la arquitectura f, pero puede consultar el modelo con instancias de datos y observar su predicción $\hat{y}$. El objetivo del atacante es construir un modelo $\mathscr{A}$ que prediga si una instancia de datos $(\boldsymbol{x}, y)$ pertenecía al conjunto de entrenamiento $\mathbb{D}_k$ del modelo $M(f, \boldsymbol{\theta}_k, \mathbb{D}_k)$ correspondiente al cliente $c \in \mathbb{K}$. Además, consideramos ataques *pasivos*, en los que el atacante se limita a observar el comportamiento del sistema sin interferir en él.

Formalmente, el modelo de un atacante perfecto $\mathscr{A}$ se define como:

$$\mathscr{A}(f, \boldsymbol{\theta}_k, (\boldsymbol{x}, y)) = \begin{cases} 1, & \text{si } (\boldsymbol{x}, y) \in \mathbb{D}_k, \\ 0, & \text{en otro caso.} \end{cases} \tag{45}$$

Estudiamos el rendimiento de tres variantes de MIA pasivos y de caja negra, que se resumen a continuación y se describen en mayor detalle en sección 4.5:

- YEOM **attack**, donde el atacante elige un umbral global ν y considera como miembros del conjunto de entrenamiento, en cada cliente, aquellas instancias con una pérdida inferior a ν[YGFJ18].
- LIRA **attack**, donde el atacante dispone de un conjunto de datos auxiliar $\mathbb{D}_a$ y entrena modelos sombra $M_{sw}(f, \mathbb{D}_{sw})$ sobre subconjuntos aleatorios $\mathbb{D}_{sw} \subset \mathbb{D}_a$. Una instancia se predice como miembro si la puntuación de confianza del modelo objetivo se asemeja a la distribución de confianza de las instancias miembro en los modelos sombra[CCN+22].
- TMIA **attack**, que utiliza destilación de conocimiento para recopilar trayectorias de pérdida con el fin de identificar qué instancias pertenecen al conjunto de entrenamiento [LZBZ22]. Este ataque se basa en que las trayectorias de pérdida tras cada época de entrenamiento discriminan mejor entre instancias miembro y no miembro que la pérdida final del modelo.

D.2.6 Equidad

La equidad algorítmica aborda las implicaciones éticas y sociales de los sistemas automatizados de toma de decisiones. Dado que los sistemas de aprendizaje automático se emplean cada vez más en ámbitos sensibles como la sanidad, las finanzas y la aplicación de la ley, garantizar su imparcialidad resulta esencial para evitar la perpetuación de prejuicios sociales.

Mientras que el concepto de *equidad individual* define la equidad como el trato similar a individuos similares, el concepto de *equidad de grupo* se basa en la definición de grupos poblacionales y exige un trato igualitario entre ellos [BHN23]. Los grupos pueden definirse en función de atributos protegidos como la etnia, el sexo o la edad. Estas dos nociones de equidad suelen entrar en conflicto, lo que refleja las compensaciones y tensiones inherentes a la cuantificación de la equidad.

Para medir la equidad de grupo en el aprendizaje automático, es fundamental considerar el entorno social en el que se aplica un modelo entrenado. En este contexto, la literatura propone diversas métricas de equidad [VR18]. En sección 6.2.3, revisamos varios algoritmos de aprendizaje federado justos para identificar las métricas de equidad más comunes en este campo. Según nuestra revisión, estas métricas son: la *igualdad de oportunidades*,

$$\Pr(\hat{Y} = 1 | Y = 1, A = 0) = \Pr(\hat{Y} = 1 | Y = 1, A = a), \forall a \in \mathcal{A}$$

y la *paridad demográfica*,

$$\Pr(\hat{Y} = 1 | A = 0) = \Pr(\hat{Y} = 1 | A = a), \forall a \in \mathcal{A}.$$

Utilizamos estas métricas para evaluar la equidad de los algoritmos de aprendizaje federado en el capítulo 6.

D.3 Estructura de la tesis

En el capítulo 1, introducimos los conceptos básicos del aprendizaje federado, un enfoque de aprendizaje automático centrado en la privacidad de los datos, mediante el diseño de un sistema distribuido en el que los nodos locales, también conocidos como clientes, mantienen sus datos privados y solo comparten los parámetros del modelo con un servidor que orquesta el entrenamiento. Analizamos los principales tipos de aprendizaje federado, como el horizontal y el vertical, así como los entornos multisilo y multidispositivo. Resumimos los principales retos del aprendizaje federado y los métodos comúnmente utilizados por la comunidad para abordarlos.

En el capítulo 2, presentamos la notación y los antecedentes matemáticos, así como los temas tratados a lo largo de esta tesis. Explicamos el concepto de aprendizaje federado e introducimos la selección de clientes, un método para reducir los costes computacionales y de comunicación en FL. Resumimos los fundamentos de la heterogeneidad de modelos y datos en sistemas distribuidos. Asimismo, introducimos conceptos del aprendizaje automático tradicional, como la privacidad de los datos y la equidad algorítmica de grupo.

En el capítulo 3, nos centramos en la selección de clientes. La necesidad de manejar un gran número de clientes fue la motivación original para aplicar este enfoque en FL. Los primeros métodos de FL [MMR+17] utilizaban una selección aleatoria uniforme para limitar el número de clientes con los que el servidor se comunica en cada ronda. Trabajos posteriores demostraron que estos métodos pueden mantener el rendimiento del modelo, mejorar la tasa de convergencia del entrenamiento [NY19], reducir el número de rondas necesarias [GMB+19] o mejorar la equidad en datos desbalanceados [LSBS20]. Proponemos una taxonomía de selección de clientes que categoriza los métodos según el objetivo, las capacidades de los clientes y los requisitos de comunicación. Un ejemplo es la heterogeneidad de modelos, propuesta originalmente para permitir la participación de clientes con menor capacidad computacional entrenando redes neuronales más pequeñas [DDT21]. Las principales contribuciones de este capítulo se presentan en [NLQO22].

En el capítulo 4, diseñamos un conjunto de métodos basados en la heterogeneidad de modelos para abordar el tema de la privacidad. El FL permite entrenar modelos sin necesidad de transferir datos sin procesar desde los dispositivos al servidor central. Se considera una solución que preserva la privacidad desde su diseño, ya que los datos nunca abandonan los clientes y solo se comparten los parámetros del modelo, lo cual es crucial en escenarios prácticos como la salud [XGS+21; LWC+22], las finanzas [LTJZ20; BP20] o las interfaces inteligentes de teléfonos inteligentes [APA+22]. Sin embargo, trabajos recientes han demostrado que se puede inferir información sensible a partir de los parámetros del modelo compartidos [FJR15; CLE+19]. Una de las razones es que los modelos complejos tienden a sobreajustar los datos [YGFJ18]. Por ello, investigamos si la heterogeneidad de modelos puede ayudar a mitigar la pérdida de privacidad, permitiendo que los clientes con menos datos entrenen modelos más pequeños. Este capítulo se basa en [NLQO25].

En el capítulo 5, abordamos el desafío de la heterogeneidad de datos en los clientes. En escenarios reales, los datos de los clientes están moldeados por factores locales como comportamientos de usuario [TYCY22], entornos específicos de recolección [FMO20; YAE+18], y sesgos culturales o socioeconómicos [BCM+18], lo que resulta en heterogeneidad estadística de los datos, donde estos son no-IID y desbalanceados. Esto dificulta la generalización del modelo del servidor y reduce su rendimiento [LHY+20; CCC22]. Proponemos `FedDiverse`,

un novedoso método de selección de clientes que aprovecha la heterogeneidad para reducir las correlaciones espurias aprendidas por el modelo. La idea es que, si muchos clientes tienen datos similares, seleccionar aquellos con datos diferentes puede ayudar a reducir esas correlaciones. Este capítulo se basa en [NFN+25].

Finalmente, en el capítulo 6, exploramos las capacidades de FedDiverse para lograr equidad grupal en FL [SYL23]. Adaptamos la selección de clientes de FedDiverse para mejorar la equidad entre clientes afectados por sesgos en los datos de entrenamiento. El objetivo es seleccionar clientes diversos para aumentar la participación de aquellos con datos valiosos pero subrepresentados.

D.4 Conclusión

D.4.1 Resumen de las principales contribuciones

El aprendizaje federado es un campo emergente con repercusiones en diversos ámbitos. La variada lista de retos y soluciones propuestas en FL hace que la comparación y evaluación de nuevos métodos sea una tarea compleja. Para simplificar esta tarea, en el capítulo 3 hemos propuesto una nueva taxonomía de selección de clientes en FL que permite la categorización y comparación de diferentes métodos. Un supuesto clave pero poco realista en FL es la homogeneidad de los clientes y sus datos. En esta tesis, relajamos este supuesto y contribuimos al campo del aprendizaje federado con varias aportaciones novedosas que permiten entornos heterogéneos.

Comenzamos con la heterogeneidad de modelos en el capítulo 4, donde investigamos arquitecturas FL en las que los clientes aprenden modelos de diferentes complejidades según sus capacidades de cómputo. Presentamos una taxonomía novedosa de métodos de integración de modelos y realizamos un estudio en profundidad de las implicaciones para la privacidad de dichas estrategias. En el caso IID entre los clientes, encontramos que la estrategia de integración de modelos puede tener un impacto significativo en la privacidad. Sin embargo, las configuraciones no IID suponen un reto para los métodos más avanzados y requieren más investigación.

Abordamos este reto en el capítulo 5, donde proponemos FedDiverse, un novedoso algoritmo de selección de clientes que aprovecha la heterogeneidad de datos —a saber, el desequilibrio de atributos, el desequilibrio de clases y las correlaciones espurias— para mitigar las dificultades que surgen al disponer de datos no IID.

Por último, en el capítulo 6, exploramos una aplicación socialmente impactante de FedDiverse, investigando cómo su estrategia de selección de clientes puede aprovecharse para mejorar la equidad grupal en FL. En concreto, examinamos situaciones en las que los datos se distribuyen de forma heterogénea entre grupos demográficos o socialmente definidos, como la etnia, el sexo o la región geográfica, y mostramos cómo FedDiverse puede mitigar el rendimiento dispar del modelo en estos grupos. Mediante la participación selectiva de los clientes, nuestro método promueve la representación equitativa en el proceso de aprendizaje, contribuyendo así a modelos de FL más inclusivos y justos.

En resumen, esta tesis aporta contribuciones clave en el campo del aprendizaje federado con heterogeneidad de clientes o datos para obtener modelos robustos, precisos y justos que respeten la privacidad de los participantes.

D.4.2 Limitaciones y retos

El aprendizaje federado es un campo de investigación que requiere mucha ingeniería. Como toda investigación en FL, la nuestra también está limitada por decisiones de diseño, incluyendo: (1) Limitamos nuestros experimentos en el capítulo 4 a ataques que ocurren en la última actualización del modelo del cliente; (2) Medimos la privacidad mediante ataques de inferencia de membresía; (3) En el capítulo 5, limitamos los experimentos a distribuciones de datos y conjuntos de datos pequeños pero bien controlados; y (4) Diseñamos nuestros métodos de selección de clientes y agnósticos al modelo para que sean modulares y fáciles de combinar con marcos existentes. Sin embargo, esto conlleva un coste en competitividad frente a métodos más complejos diseñados específicamente para la tarea.

D.4.3 Futuras líneas de investigación

Una línea de investigación futura se relaciona con el trabajo descrito en los capítulos 5 y 6. La investigación actual sobre datos no IID en FL se centra en la construcción de métodos generales y robustos. En este trabajo adoptamos un enfoque diferente, diseñando experimentos con control de la heterogeneidad estadística en los conjuntos de datos. Consideramos que esta investigación puede ampliarse para desarrollar métricas que midan la no-IID esperada en los datos distribuidos *antes* del entrenamiento, lo que permitiría diseñar mejores métodos FL específicos para cada tarea. Además, investigaciones futuras futura deberían incluir un análisis más profundo de la interacción entre los diferentes objetivos en FL, como la privacidad (en el capítulo 4) y la equidad (en el capítulo 6).

Bibliography

[ABGL19] M. Arjovsky, L. Bottou, I. Gulrajani, and D. Lopez-Paz, "Invariant risk minimization", *arXiv preprint arXiv:1907.02893*, 2019 (cit. on pp. 59, 66).

[ACG+16] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy", in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 308–318 (cit. on p. 23).

[ALMK19] J. Angwin, J. Larson, S. Mattu, and L. Kirchner, "Machine bias: There's software used across the country to predict future criminals. and it's biased against blacks.—-2016", *Access mode: https://www. propublica. org/article/machine-bias-risk-assessments-in-criminal-sentencing (online*, 2019 (cit. on p. 82).

[ANA+24] M. Asif, S. Naz, F. Ali, A. Salam, F. Amin, F. Ullah, and A. Alabrah, "Advanced zero-shot learning (azsl) framework for secure model generalization in federated learning", *IEEE Access*, 2024 (cit. on p. 53).

[APA+22] K. Arikumar, S. B. Prathiba, M. Alazab, T. R. Gadekallu, S. Pandya, J. M. Khan, and R. S. Moorthy, "Fl-pmi: Federated learning-based person movement identification through wearable devices in smart healthcare systems", *Sensors*, 22nd vol., 4th no., p. 1377, 2022 (cit. on pp. 4, 110).

[AZB+20] A. Abay, Y. Zhou, N. Baracaldo, S. Rajamoni, E. Chuba, and H. Ludwig, "Mitigating bias in federated learning", *arXiv preprint arXiv:2012.02447*, 2020 (cit. on pp. 81, 82).

[AZM+21] D. A. E. Acar, Y. Zhao, R. Matas, M. Mattina, P. Whatmough, and V. Saligrama, "Federated learning based on dynamic regularization", in *International Conference on Learning Representations*, 2021. [Online]. Available: `https://openreview.net/forum?id=B7v4QMR6Z9w` (cit. on pp. 58, 59).

[BCM+18] T. S. Brisimi, R. Chen, T. Mela, A. Olshevsky, I. C. Paschalidis, and W. Shi, "Federated learning of predictive models from federated electronic health records", *International journal of medical informatics*, 112th vol., pp. 59–67, 2018 (cit. on pp. 4, 58, 110).

[BCMC19] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Analyzing federated learning through an adversarial lens", in *International conference on machine learning*, PMLR, 2019, pp. 634–643 (cit. on pp. 2, 102).

[BEG+19] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečnỳ, S. Mazzocchi, B. McMahan, *et al.*, "Towards federated learning at scale: System design", *Proceedings of machine learning and systems*, 1st vol., pp. 374–388, 2019 (cit. on pp. 2, 103).

[BEGS17] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent", *Advances in Neural Information Processing Systems*, 30th vol., 2017 (cit. on pp. 17, 22, 60).

[BFA21] C. Briggs, Z. Fan, and P. Andras, "Federated learning for short-term residential energy demand forecasting", *arXiv preprint arXiv:2105.13325*, 2021 (cit. on pp. 29, 30).

[BHN23] S. Barocas, M. Hardt, and A. Narayanan, *Fairness and machine learning: Limitations and opportunities*. MIT press, 2023 (cit. on pp. 11, 77, 109).

[BIK+17] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning", in *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1175–1191 (cit. on p. 23).

[Bla98] C. L. Blake, "Repository of machine learning database", *http://www. ics. uci. edu/mlearn/MLRepository. html*, 1998 (cit. on p. 29).

[BP20] D. Byrd and A. Polychroniadou, "Differentially private secure multi-party computation for federated learning in financial applications", in *Proceedings of the First ACM International Conference on AI in Finance*, 2020, pp. 1–9 (cit. on pp. 4, 110).

[BRG+21] D. Bernau, J. Robl, P. W. Grassal, S. Schneider, and F. Kerschbaum, "Comparing local and central differential privacy using membership inference attacks", in *Data and Applications Security and Privacy XXXV: 35th Annual IFIP WG 11.3 Conference, DBSec 2021, Calgary, Canada, July 19–20, 2021, Proceedings*, Springer, 2021, pp. 22–42 (cit. on p. 37).

[BTM+20] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, J. Fernandez-Marques, Y. Gao, L. Sani, H. L. Kwing, T. Parcollet, P. P. d. Gusmão, and N. D. Lane, "Flower: A friendly federated learning research framework", *arXiv preprint arXiv:2007.14390*, 2020 (cit. on pp. 51, 67, 83).

[CATV17] G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik, "Emnist: Extending mnist to handwritten letters", in *2017 international joint conference on neural networks (IJCNN)*, IEEE, 2017, pp. 2921–2926 (cit. on p. 29).

[CCC22] D. Caldarola, B. Caputo, and M. Ciccone, "Improving generalization in federated learning by seeking flat minima", in *European Conference on Computer Vision*, Springer, 2022, pp. 654–672 (cit. on pp. 4, 58, 110).

[CCGR22] G. Cheng, Z. Charles, Z. Garrett, and K. Rush, "Does federated dropout actually work?", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 3387–3395 (cit. on p. 49).

[CCN+22] N. Carlini, S. Chien, M. Nasr, S. Song, A. Terzis, and F. Tramer, "Membership inference attacks from first principles", in *2022 IEEE Symposium on Security and Privacy (SP)*, IEEE, 2022, pp. 1897–1914 (cit. on pp. 11, 38, 43, 109).

[CDW+19] S. Caldas, S. M. K. Duddu, P. Wu, T. Li, J. Konečnỳ, H. B. McMahan, V. Smith, and A. Talwalkar, "Leaf: A benchmark for federated settings", in *Workshop on Federated Learning for Data Privacy and Confidentiality*, 2019 (cit. on pp. 29, 30, 53, 81).

[CGSY18] T. Chen, G. Giannakis, T. Sun, and W. Yin, "Lag: Lazily aggregated gradient for communication-efficient distributed learning", *Advances in Neural Information Processing Systems*, 31st vol., 2018 (cit. on pp. 14, 17, 23, 24, 26, 28, 60, 90).

[CHR20] W. Chen, S. Horvath, and P. Richtarik, "Optimal client sampling for federated learning", *arXiv preprint arXiv:2010.13723*, 2020 (cit. on pp. 17, 28).

[CKM+19] J. Chen, N. Kallus, X. Mao, G. Svacha, and M. Udell, "Fairness under unawareness: Assessing disparity when protected class is unobserved", in *Proceedings of the conference on fairness, accountability, and transparency*, 2019, pp. 339–348 (cit. on p. 84).

[CKMT18] S. Caldas, J. Konečny, H. B. McMahan, and A. Talwalkar, *Expanding the reach of federated learning by reducing client resource requirements*, 2018. DOI: 10.48550/ARXIV.1812.07210. [Online]. Available: https://arxiv.org/abs/1812.07210 (cit. on pp. 9, 18, 33, 34, 37, 40, 49, 60, 92, 106).

[CLE+19] N. Carlini, C. Liu, Ú. Erlingsson, J. Kos, and D. Song, "The secret sharer: Evaluating and testing unintended memorization in neural networks.", in *USENIX Security Symposium*, vol. 267, 2019 (cit. on pp. 4, 34, 37, 110).

[CMK22] H. Cho, A. Mathur, and F. Kawsar, *Flame: Federated learning across multi-device environments*, 2022. DOI: 10.48550/ARXIV.2202.08922. [Online]. Available: https://arxiv.org/abs/2202.08922 (cit. on pp. 17, 24, 26, 28, 91).

[CPL+21] S. Cui, W. Pan, J. Liang, C. Zhang, and F. Wang, "Addressing algorithmic disparity and performance inconsistency in federated learning", *Advances in Neural Information Processing Systems*, 34th vol., pp. 26 091–26 102, 2021 (cit. on p. 82).

[CSP+21] M. Chen, N. Shlezinger, H. V. Poor, Y. C. Eldar, and S. Cui, "Communication-efficient federated learning", *Proceedings of the National Academy of Sciences*, 118th vol., 17th no., e2024789118, 2021 (cit. on pp. 2, 102).

[CV25] H. Chen and H. Vikalo, "Heterogeneity-guided client sampling: Towards fast and efficient non-iid federated learning", *Advances in Neural Information Processing Systems*, 37th vol., pp. 65 525–65 561, 2025 (cit. on p. 60).

[CWJ22] Y. J. Cho, J. Wang, and G. Joshi, "Towards understanding biased client selection in federated learning", in *International Conference on Artificial Intelligence and Statistics*, PMLR, 2022, pp. 10 351–10 375 (cit. on pp. 2, 17, 23, 24, 26, 59, 60, 69, 90, 102).

[CWKM20] Y. Chen, C. Wei, A. Kumar, and T. Ma, "Self-training avoids using spurious features under domain shift", in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20 ser., Vancouver, BC, Canada: Curran Associates Inc., 2020, ISBN: 9781713829546 (cit. on p. 58).

[CYZ19] C. Clark, M. Yatskar, and L. Zettlemoyer, "Don't take the easy way out: Ensemble based methods for avoiding known dataset biases", in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, K. Inui, J. Jiang, V. Ng, and X. Wan, Eds., Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 4069–4082. DOI: 10.18653/v1/D19-1418. [Online]. Available: https://aclanthology.org/D19-1418 (cit. on p. 60).

[CZZZ25] H. Chen, T. Zhu, W. Zhou, and W. Zhao, "Afed: Algorithmic fair federated learning", *IEEE Transactions on Neural Networks and Learning Systems*, 2025 (cit. on pp. 80, 81).

[DBT+24] Y. Djebrouni, N. Benarba, O. Touat, P. De Rosa, S. Bouchenak, A. Bonifati, P. Felber, V. Marangozova, and V. Schiavoni, "Bias mitigation in federated learning for edge computing", *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 7th vol., 4th no., pp. 1–35, 2024 (cit. on pp. 80, 81).

[DDT21] E. Diao, J. Ding, and V. Tarokh, "Hetero{fl}: Computation and communication efficient federated learning for heterogeneous clients", in *International Conference on Learning Representations*, 2021 (cit. on pp. 2, 3, 9, 19, 33, 34, 36, 37, 40, 48, 49, 51, 92, 102, 103, 106, 110).

[DHMS21] F. Ding, M. Hardt, J. Miller, and L. Schmidt, "Retiring adult: New datasets for fair machine learning", *Advances in neural information processing systems*, 34th vol., pp. 6478–6490, 2021 (cit. on p. 82).

[DHP+12] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel, "Fairness through awareness", in *Proceedings of the 3rd innovations in theoretical computer science conference*, 2012, pp. 214–226 (cit. on p. 77).

[DJW13] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Local privacy and statistical minimax rates", in *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, IEEE, 2013, pp. 429–438 (cit. on p. 23).

[DLR+21a] Y. Deng, F. Lyu, J. Ren, Y.-C. Chen, P. Yang, Y. Zhou, and Y. Zhang, "Fair: Quality-aware federated learning with precise user incentive and model aggregation", in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*, IEEE, 2021, pp. 1–10 (cit. on p. 25).

[DLR+21b] Y. Deng, F. Lyu, J. Ren, H. Wu, Y. Zhou, Y. Zhang, and X. Shen, "Auction: Automated and quality-aware client selection framework for efficient federated learning", *IEEE Transactions on Parallel and Distributed Systems*, 33rd vol., 8th no., pp. 1996–2009, 2021 (cit. on p. 26).

[DLS21] D. K. Dennis, T. Li, and V. Smith, "Heterogeneity for the win: One-shot federated clustering", in *International Conference on Machine Learning*, PMLR, 2021, pp. 2611–2620. [Online]. Available: https://proceedings.mlr.press/v139/dennis21a.html (cit. on pp. 14, 17, 23, 27, 60, 90).

[DLX+24] Y. Deng, F. Lyu, T. Xia, Y. Zhou, Y. Zhang, J. Ren, and Y. Yang, "A communication-efficient hierarchical federated learning framework via shaping data distribution at edge", *IEEE/ACM Transactions on Networking*, 32nd vol., 3rd no., pp. 2600–2615, 2024 (cit. on pp. 2, 102).

[DS24] G. W. M. Dunda and S. Song, "Fairness-aware federated minimax optimization with convergence guarantee", in *2024 IEEE Conference on Artificial Intelligence (CAI)*, IEEE, 2024, pp. 563–568 (cit. on pp. 80, 81).

[Dwo06] C. Dwork, "Differential privacy", in *International colloquium on automata, languages, and programming*, Springer, 2006, pp. 1–12 (cit. on pp. 10, 108).

[DXWT21] W. Du, D. Xu, X. Wu, and H. Tong, "Fairness-aware agnostic federated learning", in *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*, SIAM, 2021, pp. 181–189 (cit. on p. 82).

[DZC+23] J. Dong, D. Zhang, Y. Cong, W. Cong, H. Ding, and D. Dai, "Federated incremental semantic segmentation", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 3934–3943 (cit. on p. 74).

[FCC23] E. Fanì, M. Ciccone, and B. Caputo, "Feddrive v2: An analysis of the impact of label skewness in federated semantic segmentation for autonomous driving", in *5th Italian Conference on Robotics and Intelligent Machines (I-RIM)*, 2023 (cit. on p. 74).

[FDM+08] R. Farzan, J. M. DiMicco, D. R. Millen, C. Dugan, W. Geyer, and E. A. Brownholtz, "Results from deploying a participation incentive mechanism within the enterprise", in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08 ser., Florence, Italy: Association for Computing Machinery, 2008, pp. 563–572, ISBN: 9781605580111. DOI: 10.1145/1357054.1357145. [Online]. Available: https://doi.org/10.1145/1357054.1357145 (cit. on p. 20).

[FJR15] M. Fredrikson, S. Jha, and T. 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 (cit. on pp. 4, 34, 110).

[FMO20] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach", *Advances in neural information processing systems*, 33rd vol., pp. 3557–3568, 2020 (cit. on pp. 4, 58, 110).

[GBC16] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org (cit. on pp. 5, 103).

[GBH09] A. Go, R. Bhayani, and L. Huang, "Twitter sentiment classification using distant supervision", *CS224N project report, Stanford*, 1st vol., 12th no., p. 2009, 2009 (cit. on p. 29).

[GBX22] Y. Gu, Y. Bai, and S. Xu, "Cs-mia: Membership inference attack based on prediction confidence series in federated learning", *Journal of Information Security and Applications*, 67th vol., p. 103 201, 2022 (cit. on p. 37).

[GGvDS21] B. R. Gálvez, F. Granqvist, R. van Dalen, and M. Seigel, "Enforcing fairness in private federated learning via the modified method of differential multipliers", in *NeurIPS 2021 Workshop Privacy in Machine Learning*, 2021 (cit. on pp. 80, 82).

[GJM+20] R. Geirhos, J.-H. Jacobsen, C. Michaelis, R. Zemel, W. Brendel, M. Bethge, and F. A. Wichmann, "Shortcut learning in deep neural networks", *Nature Machine Intelligence*, 2nd vol., 11th no., pp. 665–673, 2020 (cit. on pp. 58, 59).

[GKN17] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective", *arXiv preprint arXiv:1712.07557*, 2017 (cit. on pp. 18, 20, 28).

[GMB+19] J. Goetz, K. Malik, D. Bui, S. Moon, H. Liu, and A. Kumar, "Active federated learning", *arXiv preprint arXiv:1909.12641*, 2019. [Online]. Available: `https://arxiv.org/abs/1909.12641` (cit. on pp. 3, 14, 23, 26, 27, 89, 110).

[GRR+23] L. Gustafson, C. Rolland, N. Ravi, Q. Duval, A. Adcock, C.-Y. Fu, M. Hall, and C. Ross, "Facet: Fairness in computer vision evaluation benchmark", in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 20 370–20 382 (cit. on p. 82).

[GTL23] Y. Guo, X. Tang, and T. Lin, "Fedrc: Tackling diverse distribution shifts challenge in federated learning by robust clustering", *arXiv preprint arXiv:2301.12379*, 2023 (cit. on pp. 60, 65).

[HG20] R. Hu and Y. Gong, "Trading data for learning: Incentive mechanism for on-device federated learning", in *GLOBECOM 2020-2020 IEEE Global Communications Conference*, IEEE, 2020, pp. 1–6 (cit. on pp. 19, 25).

[HLA+21] S. Horvath, S. Laskaridis, M. Almeida, I. Leontiadis, S. Venieris, and N. Lane, "Fjord: Fair and accurate federated learning under heterogeneous targets with ordered dropout", *Advances in Neural Information Processing Systems*, 34th vol., pp. 12 876–12 889, 2021 (cit. on pp. 8, 9, 34, 37, 40, 48, 106, 107).

[How17] A. G. Howard, "Mobilenets: Efficient convolutional neural networks for mobile vision applications", *arXiv preprint arXiv:1704.04861*, 2017 (cit. on pp. 67, 84).

[HQB19] T.-M. H. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification", *arXiv preprint arXiv:1909.06335*, 2019 (cit. on pp. 54, 59, 67, 71, 72).

[HS92] R. W. Hahn and R. N. Stavins, "Economic incentives for environmental protection: Integrating theory and practice", *The American economic review*, 82nd vol., 2nd no., pp. 464–468, 1992 (cit. on p. 20).

[HSF+23] H. Huang, W. Shi, Y. Feng, C. Niu, G. Cheng, J. Huang, and Z. Liu, "Active client selection for clustered federated learning", *IEEE Transactions on Neural Networks and Learning Systems*, 2023 (cit. on pp. 60, 65).

[HSS+22] H. Hu, Z. Salcic, L. Sun, G. Dobbie, P. S. Yu, and X. Zhang, "Membership inference attacks on machine learning: A survey", *ACM Computing Surveys (CSUR)*, 54th vol., s11th no., pp. 1–37, 2022 (cit. on p. 36).

[HWS24] S. Hu, Z. S. Wu, and V. Smith, "Fair federated learning via bounded group loss", in *2024 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, IEEE, 2024, pp. 140–160 (cit. on pp. 80–82).

[HYS+24] W. Huang, M. Ye, Z. Shi, G. Wan, H. Li, B. Du, and Q. Yang, "Federated learning for generalization, robustness, fairness: A survey and benchmark", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024 (cit. on pp. 2, 78, 102).

[HZRS16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778 (cit. on p. 72).

[JWK+21] B. Jayaraman, L. Wang, K. Knipmeyer, Q. Gu, and D. Evans, "Revisiting membership inference under realistic assumptions", *Proceedings on Privacy Enhancing Technologies*, 2021st vol., 2nd no., 2021 (cit. on p. 37).

[JWN+20] Y. Jiao, P. Wang, D. Niyato, B. Lin, and D. I. Kim, "Toward an automated auction framework for wireless federated learning services market", *IEEE Transactions on Mobile Computing*, 20th vol., 10th no., pp. 3034–3048, 2020 (cit. on pp. 25, 26).

[KB96] R. Kohavi and B. Becker, "Adult data set", *UCI machine learning repository*, 5th vol., p. 2093, 1996 (cit. on p. 82).

[KC12] F. Kamiran and T. Calders, "Data preprocessing techniques for classification without discrimination", *Knowledge and information systems*, 33rd vol., 1st no., pp. 1–33, 2012 (cit. on p. 83).

[KD21] Y. Kaya and T. Dumitras, "When does data augmentation help with membership inference attacks?", in *Proceedings of the 38th International Conference on Machine Learning*, M. Meila and T. Zhang, Eds., Proceedings of Machine Learning Research ser., vol. 139, PMLR, Jul. 2021, pp. 5345–5355 (cit. on pp. 37, 48).

[KHL19] D. Kaushik, E. Hovy, and Z. C. Lipton, "Learning the difference that makes a difference with counterfactually-augmented data", *arXiv preprint arXiv:1909.12434*, 2019 (cit. on p. 60).

[KJ21] K. Karkkainen and J. Joo, "Fairface: Face attribute dataset for balanced race, gender, and age for bias measurement and mitigation", in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021, pp. 1548–1558 (cit. on p. 81).

[KJK+20] S. P. Karimireddy, M. Jaggi, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh, "Mime: Mimicking centralized stochastic algorithms in federated learning", *arXiv preprint arXiv:2008.03606*, 2020 (cit. on p. 59).

[KKK+19] B. Kim, H. Kim, K. Kim, S. Kim, and J. Kim, "Learning not to learn: Training deep neural networks with biased data", in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 9012–9020 (cit. on p. 60).

[KKM+20] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning", in *International conference on machine learning*, PMLR, 2020, pp. 5132–5143 (cit. on pp. 58, 59).

[KMA+21] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, R. G. L. D'Oliveira, H. Eichner, S. E. Rouayheb, D. Evans, J. Gardner, Z. Garrett, A. Gascón, B. Ghazi, P. B. Gibbons, M. Gruteser, Z. Harchaoui, C. He, L. He, Z. Huo, B. Hutchinson, J. Hsu, M. Jaggi, T. Javidi, G. Joshi, M. Khodak, J. Konecný, A. Korolova, F. Koushanfar, S. Koyejo, T. Lepoint, Y. Liu, P. Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, H. Qi, D. Ramage, R. Raskar, M. Raykova, D. Song, W. Song, S. U. Stich, Z. Sun, A. T. Suresh, F. Tramèr, P. Vepakomma, J. Wang, L. Xiong, Z. Xu, Q. Yang, F. X. Yu, H. Yu, and S. Zhao, "Advances and open problems in federated learning", *Foundations and Trends® in Machine Learning*, 14th vol., –21st no., pp. 1–210, 2021, ISSN: 1935-8237. DOI: 10.1561/2200000083. [Online]. Available: http://dx.doi.org/10.1561/2200000083 (cit. on pp. 1, 2, 21, 58, 59, 78, 101, 102).

[KMY+16] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency", *arXiv preprint arXiv:1610.05492*, 2016 (cit. on pp. 2, 102).

[KND+00] C. Kremen, J. O. Niles, M. Dalton, G. C. Daily, P. R. Ehrlich, J. P. Fay, D. Grewal, and R. P. Guillery, "Economic incentives for rain forest conservation across scales", *Science*, 288th vol., 5472nd no., pp. 1828–1832, 2000 (cit. on p. 20).

[KPDG22] S. Kanaparthy, M. Padala, S. Damle, and S. Gujar, "Fair federated learning for heterogeneous data", in *Proceedings of the 5th Joint International Conference on Data Science & Management of Data (9th ACM IKDD CODS and 27th COMAD)*, 2022, pp. 298–299 (cit. on pp. 2, 80, 81, 102).

[KRA+20] A. Kuznetsova, H. Rom, N. Alldrin, J. Uijlings, I. Krasin, J. Pont-Tuset, S. Kamali, S. Popov, M. Malloci, A. Kolesnikov, *et al.*, "The open images dataset v4", *International Journal of Computer Vision*, 128th vol., 7th no., pp. 1956–1981, 2020 (cit. on p. 24).

[Kri09] A. Krizhevsky, "Learning multiple layers of features from tiny images", *Master's thesis, University of Tront*, 2009 (cit. on pp. 29, 48, 82).

[KXN+19a] J. Kang, Z. Xiong, D. Niyato, S. Xie, and J. Zhang, "Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory", *IEEE Internet of Things Journal*, 6th vol., 6th no., pp. 10 700–10 714, 2019 (cit. on pp. 19, 20, 22, 23, 26, 28, 92).

[KXN+19b] J. Kang, Z. Xiong, D. Niyato, H. Yu, Y.-C. Liang, and D. I. Kim, "Incentive design for efficient federated learning in mobile networks: A contract theory approach", in *2019 IEEE VTS Asia Pacific Wireless Communications Symposium (APWCS)*, IEEE, 2019, pp. 1–5 (cit. on pp. 19, 25, 26).

[LAS+20] Y. Liu, Z. Ai, S. Sun, S. Zhang, Z. Liu, and H. Yu, "Fedcoin: A peer-to-peer payment system for federated learning", in *Federated Learning*, Springer, 2020, pp. 125–138 (cit. on pp. 19, 25).

[LBC+20] P. Lahoti, A. Beutel, J. Chen, K. Lee, F. Prost, N. Thain, X. Wang, and E. Chi, "Fairness without demographics through adversarially reweighted learning", *Advances in neural information processing systems*, 33rd vol., pp. 728–740, 2020 (cit. on p. 85).

[LDKS23] A. Lynch, G. J.-S. Dovonon, J. Kaddour, and R. Silva, *Spawrious: A benchmark for fine control of spurious correlation biases*, 2023. arXiv: 2303.05470 [cs.CV] (cit. on p. 66).

[LDS+21] B. Liu, M. Ding, S. Shaham, W. Rahayu, F. Farokhi, and Z. Lin, "When machine learning meets privacy: A survey and outlook", *ACM Computing Surveys (CSUR)*, 54th vol., 2nd no., pp. 1–36, 2021 (cit. on pp. 10, 108).

[LeC98] Y. LeCun, "The mnist database of handwritten digits", *http://yann. lecun. com/exdb/mnist/*, 1998 (cit. on p. 29).

[LGZX23] D. Liao, X. Gao, Y. Zhao, and C.-Z. Xu, "Adaptive channel sparsity for federated learning under system heterogeneity", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 20 432–20 441 (cit. on pp. 8, 9, 34, 37, 49, 106, 107).

[LHBS21] T. Li, S. Hu, A. Beirami, and V. Smith, "Ditto: Fair and robust federated learning through personalization", in *International conference on machine learning*, PMLR, 2021, pp. 6357–6368 (cit. on pp. 2, 102).

[LHC+21] E. Z. Liu, B. Haghgoo, A. S. Chen, A. Raghunathan, P. W. Koh, S. Sagawa, P. Liang, and C. Finn, "Just train twice: Improving group robustness without training group information", in *International Conference on Machine Learning*, PMLR, 2021, pp. 6781–6792 (cit. on pp. 58, 60).

[LHS21] Q. Li, B. He, and D. Song, "Model-contrastive federated learning", in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 10 713–10 722 (cit. on pp. 54, 59).

[LHY+20] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data", *International Conference on Learning Representations*, 2020 (cit. on pp. 4, 58, 110).

[LLGL24] B. Liu, N. Lv, Y. Guo, and Y. Li, "Recent advances on federated learning: A systematic survey", *Neurocomputing*, p. 128 019, 2024 (cit. on pp. 2, 102).

[LLWT15] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild", in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 3730–3738 (cit. on p. 81).

[LNW99] A. Lindbeck, S. Nyberg, and J. W. Weibull, "Social norms and economic incentives in the welfare state", *The Quarterly Journal of Economics*, 114th vol., 1st no., pp. 1–35, 1999 (cit. on p. 20).

[LSBS20] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning", in *International Conference on Learning Representations*, 2020. [Online]. Available: `https://openreview.net/forum?id=ByexElSYDr` (cit. on pp. 3, 14, 18, 23, 26, 91, 110).

[LSL+21] A. Li, J. Sun, P. Li, Y. Pu, H. Li, and Y. Chen, "Hermes: An efficient federated learning framework for heterogeneous mobile clients", in *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*, 2021, pp. 420–437 (cit. on p. 37).

[LSZ+20] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks", *Proceedings of Machine learning and systems*, 2nd vol., pp. 429–450, 2020 (cit. on pp. 54, 58, 59, 71, 72).

[LTJZ20] G. Long, Y. Tan, J. Jiang, and C. Zhang, "Federated learning for open banking", in *Federated Learning: Privacy and Incentive*, Springer, 2020, pp. 240–254 (cit. on pp. 4, 110).

[LW19] D. Li and J. Wang, "Fedmd: Heterogenous federated learning via model distillation", in *NeurIPS Workshop on Federated Learning for Data Privacy and Confidentiality*, 2019 (cit. on p. 37).

[LWC+22] Z. Li, L. Wang, G. Chen, Z. Zhang, M. Shafiq, and Z. Gu, "E2egi: End-to-end gradient inversion in federated learning", *IEEE Journal of Biomedical and Health Informatics*, 2022 (cit. on pp. 4, 34, 38, 110).

[LWW+22] R. Liu, F. Wu, C. Wu, Y. Wang, L. Lyu, H. Chen, and X. Xie, "No one left behind: Inclusive federated learning over heterogeneous devices", in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022, pp. 3398–3406 (cit. on pp. 19, 37, 92).

[LXM+20] W. Y. B. Lim, Z. Xiong, C. Miao, D. Niyato, Q. Yang, C. Leung, and H. V. Poor, "Hierarchical incentive mechanism design for federated machine learning in mobile networks", *IEEE Internet of Things Journal*, 7th vol., 10th no., pp. 9575–9588, 2020 (cit. on pp. 19, 25).

[LZBZ22] Y. Liu, Z. Zhao, M. Backes, and Y. Zhang, "Membership inference attacks by exploiting loss trajectory", in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, 2022, pp. 2085–2098 (cit. on pp. 11, 38, 45, 109).

[LZMC21] F. Lai, X. Zhu, H. V. Madhyastha, and M. Chowdhury, "Oort: Efficient federated learning via guided participant selection", in *15th USENIX Symposium on Operating Systems Design and Implementation (OSDI 21)*, USENIX Association, Jul. 2021, pp. 19–35, ISBN: 978-1-939133-22-9. [Online]. Available: `https://www.usenix.org/conference/osdi21/presentation/lai` (cit. on pp. 24, 28, 91).

[LZS+24] F. Liu, Z. Zheng, Y. Shi, Y. Tong, and Y. Zhang, "A survey on federated learning: A perspective from multi-party computation", *Frontiers of Computer Science*, 18th vol., 1st no., p. 181 336, 2024 (cit. on pp. 2, 102).

[MBB24] A. Mora, A. Bujari, and P. Bellavista, "Enhancing generalization in federated learning with heterogeneous data: A comparative literature review", *Future Generation Computer Systems*, 2024 (cit. on pp. 58, 59).

[MLZL24] S. I. A. Meerza, L. Liu, J. Zhang, and J. Liu, "Glocalfair: Jointly improving global and local group fairness in federated learning", *arXiv preprint arXiv:2401.03562*, 2024 (cit. on pp. 80, 81).

[MMR+17] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data", in *Artificial intelligence and statistics*, PMLR, 2017, pp. 1273–1282 (cit. on pp. 1, 3, 7, 13, 30, 36, 48, 71, 78, 101, 104, 105, 110).

[MO12] C. Marantes and D. Openshaw, "Low carbon london a large scale low carbon smart grid project", *Journal of International Council on Electrical Engineering*, 2nd vol., 2nd no., pp. 231–236, 2012 (cit. on pp. 29, 30).

[MÖB22] M. Mansouri, M. Önen, and W. Ben Jaballah, "Learning from failures: Secure and fault-tolerant aggregation for federated learning", in *Proceedings of the 38th Annual Computer Security Applications Conference*, 2022, pp. 146–158 (cit. on pp. 2, 103).

[MRA+12] J. G. Moreno-Torres, T. Raeder, R. Alaiz-Rodríguez, N. V. Chawla, and F. Herrera, "A unifying view on dataset shift in classification", *Pattern recognition*, 45th vol., 1st no., pp. 521–530, 2012 (cit. on pp. 2, 102).

[MSC+23] X. Mu, Y. Shen, K. Cheng, X. Geng, J. Fu, T. Zhang, and Z. Zhang, "Fedproc: Prototypical contrastive federated learning on non-iid data", *Future Generation Computer Systems*, 143rd vol., pp. 93–104, 2023 (cit. on p. 54).

[MSS19] M. Mohri, G. Sivek, and A. T. Suresh, "Agnostic federated learning", in *International Conference on Machine Learning*, PMLR, 2019, pp. 4615–4625 (cit. on pp. 14, 18, 23, 26, 89, 91).

[NAN20] V. Nagarajan, A. Andreassen, and B. Neyshabur, "Understanding the failure modes of out-of-distribution generalization", *arXiv preprint arXiv:2010.15775*, 2020 (cit. on p. 59).

[NCA+20] J. Nam, H. Cha, S. Ahn, J. Lee, and J. Shin, "Learning from failure: De-biasing classifier from biased classifier", *Advances in Neural Information Processing Systems*, 33rd vol., pp. 20 673–20 684, 2020 (cit. on pp. 60, 63, 64).

[Nes13] Y. Nesterov, "Gradient methods for minimizing composite functions", *Mathematical programming*, 140th vol., 1st no., pp. 125–161, 2013 (cit. on p. 71).

[NFN+25] G. D. Németh, E. Fanì, Y. J. Ng, B. Caputo, M. Á. Lozano, N. Oliver, and N. Quadrianto, "Diversity-driven learning: Tackling spurious correlations and data heterogeneity in federated models", *arXiv preprint arXiv:2504.11216*, 2025 (cit. on pp. ix, xi, xiii, 4, 5, 57, 78, 83, 103, 111).

[NHD+23] H. N. C. Neto, J. Hribar, I. Dusparic, D. M. F. Mattos, and N. C. Fernandes, "A survey on securing federated learning: Analysis of applications, attacks, challenges, and trends", *IEEE Access*, 11th vol., pp. 41 928–41 953, 2023 (cit. on p. 37).

[Nic84] A. L. Nichols, *Targeting economic incentives for environmental protection*. Massachusetts Institute of Technology Press, Cambridge, MA, 1984 (cit. on p. 20).

[NLQO22] G. D. Németh, M. A. Lozano, N. Quadrianto, and N. M. Oliver, "A snapshot of the frontiers of client selection in federated learning", *Transactions on Machine Learning Research*, 2022, ISSN: 2835-8856. [Online]. Available: `https://openreview.net/forum?id=vwOKBldzFu` (cit. on pp. ix, xi, xiii, 2, 3, 5, 13, 60, 102, 103, 110).

[NLQO25] G. D. Németh, M. Á. Lozano, N. Quadrianto, and N. Oliver, "Privacy and accuracy implications of model complexity and integration in heterogeneous federated learning", *IEEE Access*, 13th vol., pp. 40 258–40 274, 2025. DOI: `10.1109/ACCESS.2025.3546478` (cit. on pp. ix, xi, xiii, 2, 4, 5, 33, 102, 103, 110).

[NN21] L. Nagalapatti and R. Narayanam, "Game of gradients: Mitigating irrelevant clients in federated learning", in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, 2021, pp. 9046–9054 (cit. on pp. 23, 25, 26, 90).

[NY19] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge", in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, 2019, pp. 1–7. DOI: `10.1109/ICC.2019.8761315` (cit. on pp. 3, 14, 17, 23, 26, 28, 91, 110).

[OEK+13] J. F. Olson, M. Eaton, S. A. Kells, V. Morin, and C. Wang, "Cold tolerance of bed bugs and practical recommendations for control", *Journal of economic entomology*, 106th vol., 6th no., pp. 2433–2441, 2013 (cit. on p. xv).

[OV23] A. Oprea and A. Vassilev, "Adversarial machine learning: A taxonomy and terminology of attacks and mitigations", National Institute of Standards and Technology US Department of Commerce, Tech. Rep., 2023 (cit. on p. 37).

[PDG21] M. Padala, S. Damle, and S. Gujar, "Federated learning meets fairness and differential privacy", in *International Conference on Neural Information Processing*, Springer, 2021, pp. 692–699 (cit. on pp. 18, 82).

[PGC+17] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch", in *NIPS 2017 Workshop Autodiff Submission*, 2017 (cit. on p. 51).

[PGM+19] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library", in *Advances in Neural Information Processing Systems 32*, Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf` (cit. on pp. 67, 83).

[PLY23] P. Pene, W. Liao, and W. Yu, "Incentive design for heterogeneous client selection: A robust federated learning approach", *IEEE Internet of Things Journal*, 11th vol., 4th no., pp. 5939–5950, 2023 (cit. on pp. 60, 65).

[PMB+22] A. Papadaki, N. Martinez, M. Bertran, G. Sapiro, and M. Rodrigues, "Minimax demographic group fairness in federated learning", in *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*, 2022, pp. 142–159 (cit. on pp. 80–82).

[PTB+19] S. R. Pandey, N. H. Tran, M. Bennis, Y. K. Tun, Z. Han, and C. S. Hong, "Incentivize to build: A crowdsourcing framework for federated learning", in *2019 IEEE Global Communications Conference (GLOBECOM)*, IEEE, 2019, pp. 1–6 (cit. on pp. 19, 25).

[Raw04] J. Rawls, "A theory of justice", in *Ethics*, Routledge, 2004, pp. 229–234 (cit. on p. 18).

[RBL+22] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models", in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10 684–10 695 (cit. on p. 66).

[RCZ+20] S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečnỳ, S. Kumar, and H. B. McMahan, "Adaptive federated optimization", *arXiv preprint arXiv:2003.00295*, 2020 (cit. on pp. 30, 59).

[RMLH22] N. Rodríguez-Barroso, E. Martínez-Cámara, M. V. Luzón, and F. Herrera, "Dynamic defense against byzantine poisoning attacks in federated learning", *Future Generation Computer Systems*, 133rd vol., pp. 1–9, 2022 (cit. on pp. 17, 22, 60, 91).

[RMR+21] J.-F. Rajotte, S. Mukherjee, C. Robinson, A. Ortiz, C. West, J. M. L. Ferres, and R. T. Ng, "Reducing bias and increasing utility by federated generative modeling of medical images using a centralized adversary", in *Proceedings of the Conference on Information Technology for Social Good*, 2021, pp. 79–84 (cit. on p. 24).

[SACA24] T. Salazar, H. AraÃējo, A. Cano, and P. H. Abreu, "A survey on group fairness in federated learning: Challenges, taxonomy of solutions and directions for future research", *arXiv preprint arXiv:2410.03855*, 2024 (cit. on pp. 2, 79, 102).

[SCT+15] J. R. Schofield, R. Carmichael, S. Tindemans, M. Bilton, M. Woolf, G. Strbac, *et al.*, "Low carbon london project: Data from the dynamic time-of-use electricity pricing trial, 2013", *uK Data Service, SN*, 7857th vol., 2015th no., pp. 7857–1, 2015 (cit. on pp. 29, 30).

[SE19] Y. Sarikaya and O. Ercetin, "Motivating workers in federated learning: A stackelberg game perspective", *IEEE Networking Letters*, 2nd vol., 1st no., pp. 23–27, 2019 (cit. on pp. 19, 20, 25).

[SFAA23] T. Salazar, M. Fernandes, H. Araújo, and P. H. Abreu, "Fair-fate: Fair federated learning with momentum", in *International Conference on Computational Science*, Springer, 2023, pp. 524–538 (cit. on pp. 80, 81).

[SFT+23] D. Shenaj, E. Fanì, M. Toldo, D. Caldarola, A. Tavera, U. Michieli, M. Ciccone, P. Zanuttigh, and B. Caputo, "Learning across domains and devices: Style-driven source-free domain adaptation in clustered federated learning", in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 2023, pp. 444–454 (cit. on p. 74).

[SGAA24] T. Salazar, J. Gama, H. Araújo, and P. H. Abreu, "Unveiling group-specific distributed concept drift: A fairness imperative in federated learning", *arXiv preprint arXiv:2402.07586*, 2024 (cit. on p. 80).

[Sha51] L. S. Shapley, *Notes on the N-person Game–I: Characteristic-point Solutions of the Four-person Game*. Rand Corporation, 1951 (cit. on p. 25).

[Sha71] L. S. Shapley, "Cores of convex games", *International journal of game theory*, 1st vol., 1st no., pp. 11–26, 1971 (cit. on p. 20).

[SHK+14] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting", *The journal of machine learning research*, 15th vol., 1st no., pp. 1929–1958, 2014 (cit. on p. 40).

[SHK+20] M. Shin, C. Hwang, J. Kim, J. Park, M. Bennis, and S.-L. Kim, "Xor mixup: Privacy-preserving data augmentation for one-shot federated learning", *arXiv preprint arXiv:2006.05148*, 2020 (cit. on p. 37).

[SKHL20] S. Sagawa, P. W. Koh, T. B. Hashimoto, and P. Liang, "Distributionally Robust Neural Networks for Group Shifts: On the Importance of Regularization for Worst-Case Generalization", in *International Conference on Learning Representations*, 2020. arXiv: `1911.08731` (cit. on pp. 59, 67).

[SO21] M. Savi and F. Olivadese, "Short-term energy consumption forecasting at the edge: A federated learning approach", *IEEE Access*, 9th vol., pp. 95 949–95 969, 2021 (cit. on pp. 29, 30).

[SSG+23] D. Song, G. Shen, D. Gao, L. Yang, X. Zhou, S. Pan, W. Lou, and F. Zhou, "Fast heterogeneous federated learning with hybrid client selection", in *Uncertainty in Artificial Intelligence*, PMLR, 2023, pp. 2006–2015 (cit. on pp. 60, 69).

[SSM19] L. Song, R. Shokri, and P. Mittal, "Membership inference attacks against adversarially robust deep learning models", in *2019 IEEE Security and Privacy Workshops (SPW)*, IEEE, 2019, pp. 50–56 (cit. on p. 37).

[STW19] T. Song, Y. Tong, and S. Wei, "Profit allocation for federated learning", in *2019 IEEE International Conference on Big Data (Big Data)*, IEEE, 2019, pp. 2577–2586 (cit. on pp. 2, 19, 20, 23, 25, 26, 61, 92, 102).

[SYL23] Y. Shi, H. Yu, and C. Leung, "Towards fairness-aware federated learning", *IEEE Transactions on Neural Networks and Learning Systems*, 2023 (cit. on pp. 2, 4, 18, 78, 102, 111).

[TCK+21] F. Träuble, E. Creager, N. Kilbertus, F. Locatello, A. Dittadi, A. Goyal, B. Schölkopf, and S. Bauer, "On disentangled representations learned from correlated data", in *Proceedings of the 38th International Conference on Machine Learning*, M. Meila and T. Zhang, Eds., Proceedings of Machine Learning Research ser., vol. 139, PMLR, Jul. 2021, pp. 10 401–10 412 (cit. on p. 58).

[TLC+20] S. Truex, L. Liu, K.-H. Chow, M. E. Gursoy, and W. Wei, "Ldp-fed: Federated learning with local differential privacy", in *Proceedings of the third ACM international workshop on edge systems, analytics and networking*, 2020, pp. 61–66 (cit. on pp. 2, 102).

[TYCY22] A. Z. Tan, H. Yu, L. Cui, and Q. Yang, "Towards personalized federated learning", *IEEE transactions on neural networks and learning systems*, 34th vol., 12th no., pp. 9587–9603, 2022 (cit. on pp. 4, 58, 110).

[VMFS21] C. Vigurs, C. Maidment, M. Fell, and D. Shipworth, "Customer privacy concerns as a barrier to sharing data about energy use in smart local energy systems: A rapid realist review", *Energies*, 14th vol., 5th no., p. 1285, 2021 (cit. on p. 30).

[VR18] S. Verma and J. Rubin, "Fairness definitions explained", in *Proceedings of the international workshop on software fairness*, 2018, pp. 1–7 (cit. on pp. 11, 79, 109).

[WBW+11] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The caltech-ucsd birds-200-2011 dataset", 2011 (cit. on p. 66).

[WCX+21] J. Wang, Z. Charles, Z. Xu, G. Joshi, H. B. McMahan, B. A. y Arcas, M. Al-Shedivat, G. Andrew, S. Avestimehr, K. Daly, D. Data, S. N. Diggavi, H. Eichner, A. Gadhikar, Z. Garrett, A. M. Girgis, F. Hanzely, A. Hard, C. He, S. Horvath, Z. Huo, A. Ingerman, M. Jaggi, T. Javidi, P. Kairouz, S. Kale, S. P. Karimireddy, J. Konecný, S. Koyejo, T. Li, L. Liu, M. Mohri, H. Qi, S. J. Reddi, P. Richtárik, K. Singhal, V. Smith, M. Soltanolkotabi, W. Song, A. T. Suresh, S. U. Stich, A. Talwalkar, H. Wang, B. E. Woodworth, S. Wu, F. X. Yu, H. Yuan, M. Zaheer, M. Zhang, T. Zhang, C. Zheng, C. Zhu, and W. Zhu, "A field guide to federated optimization", *CoRR*, abs/2107.06917 vol., 2021. [Online]. Available: `https://arxiv.org/abs/2107.06917` (cit. on pp. 22, 23).

[WDZ19] G. Wang, C. X. Dang, and Z. Zhou, "Measure contribution of participants in federated learning", in *2019 IEEE International Conference on Big Data (Big Data)*, IEEE, 2019, pp. 2597–2604 (cit. on pp. 19, 25).

[WFK+24] Y. Wang, H. Fu, R. Kanagavelu, Q. Wei, Y. Liu, and R. S. M. Goh, "An aggregation-free federated learning for tackling data heterogeneity", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 26 233–26 242 (cit. on p. 59).

[Wil14] E. Wilson, "Commercial and residential hourly load profiles for all tmy3 locations in the united states", DOE Open Energy Data Initiative (OEDI); National Renewable Energy Lab.(NREL …, Tech. Rep., 2014 (cit. on p. 30).

[WK23] A. Wu and Y.-W. Kwon, "Enhancing recommendation capabilities using multi-head attention-based federated knowledge distillation", *IEEE Access*, 11th vol., pp. 45 850–45 861, 2023 (cit. on p. 37).

[WKNL20] H. Wang, Z. Kaplan, D. Niu, and B. Li, "Optimizing federated learning on non-iid data with reinforcement learning", in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, IEEE, 2020, pp. 1698–1707 (cit. on pp. 14, 23, 25, 26, 90).

[WLL+20] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "Tackling the objective inconsistency problem in heterogeneous federated optimization", *Advances in neural information processing systems*, 33rd vol., pp. 7611–7623, 2020 (cit. on pp. 61, 69).

[WPLK23] G. Wang, A. Payani, M. Lee, and R. Kompella, "Mitigating group bias in federated learning: Beyond local fairness", *arXiv preprint arXiv:2305.09931*, 2023 (cit. on pp. 80, 81).

[WSK+22] J. Wolfrath, N. Sreekumar, D. Kumar, Y. Wang, and A. Chandra, "Haccs: Heterogeneity-aware clustered client selection for accelerated federated learning", in *2022 IEEE international parallel and distributed processing symposium (IPDPS)*, IEEE, 2022, pp. 985–995 (cit. on p. 60).

[WW22] H. Wu and P. Wang, "Node selection toward faster convergence for federated learning on non-iid data", *IEEE Transactions on Network Science and Engineering*, 9th vol., 5th no., pp. 3099–3111, 2022 (cit. on pp. 60, 69).

[WXX+24] C. Wang, H. Xia, S. Xu, H. Chi, R. Zhang, and C. Hu, "Fedbnr: Mitigating federated learning non-iid problem by breaking the skewed task and reconstructing representation", *Future Generation Computer Systems*, 153rd vol., pp. 1–11, 2024 (cit. on p. 54).

[WZNK24] X. Wang, H. Zhao, K. Nahrstedt, and S. Koyejo, "Personalized federated learning with spurious features: An adversarial approach", *Transactions on Machine Learning Research*, 2024 (cit. on pp. 60, 74).

[WZQ+19] J. Wang, H. Zhong, J. Qin, W. Tang, R. Rajagopal, Q. Xia, and C. Kang, "Incentive mechanism for sharing distributed energy resources", *Journal of Modern Power Systems and Clean Energy*, 7th vol., 4th no., pp. 837–850, 2019. DOI: 10.1007/s40565-019-0518-5 (cit. on p. 20).

[WZY+19] T. Wang, J. Zhao, M. Yatskar, K.-W. Chang, and V. Ordonez, "Balanced datasets are not enough: Estimating and mitigating gender bias in deep image representations", in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 5310–5319 (cit. on p. 60).

[XGS+21] J. Xu, B. S. Glicksberg, C. Su, P. Walker, J. Bian, and F. Wang, "Federated learning for healthcare informatics", *Journal of Healthcare Informatics Research*, 5th vol., pp. 1–19, 2021 (cit. on pp. 4, 110).

[XRV17] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms", *arXiv preprint arXiv:1708.07747*, 2017 (cit. on pp. 29, 82).

[XYG+20] B. Xin, W. Yang, Y. Geng, S. Chen, S. Wang, and L. Huang, "Private fl-gan: Differential privacy synthetic data generation based on federated learning", in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2020, pp. 2927–2931 (cit. on p. 24).

[XZLD22] F. Xin, J. Zhang, J. Luo, and F. Dong, "Federated learning client selection mechanism under system and data heterogeneity", in *2022 IEEE 25th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, IEEE, 2022, pp. 1239–1244 (cit. on p. 60).

[YAE+18] T. Yang, G. Andrew, H. Eichner, H. Sun, W. Li, N. Kong, D. Ramage, and F. Beaufays, "Applied federated learning: Improving google keyboard query suggestions", *arXiv preprint arXiv:1812.02903*, 2018 (cit. on pp. 4, 58, 110).

[YBR20] L. Yang, C. Beliard, and D. Rossi, "Heterogeneous data-aware federated learning", *IJCAI 2020 Federated Learning Workshop*, 2020 (cit. on p. 53).

[YDK+24] A. Yazdinejad, A. Dehghantanha, H. Karimipour, G. Srivastava, and R. M. Parizi, "A robust privacy-preserving federated learning model against model poisoning attacks", *IEEE Transactions on Information Forensics and Security*, 2024 (cit. on pp. 2, 102).

[YFD+23] M. Ye, X. Fang, B. Du, P. C. Yuen, and D. Tao, "Heterogeneous federated learning: State-of-the-art and research challenges", *ACM Computing Surveys*, 56th vol., 3rd no., pp. 1–44, 2023 (cit. on p. 34).

[YGFJ18] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha, "Privacy risk in machine learning: Analyzing the connection to overfitting", in *2018 IEEE 31st computer security foundations symposium (CSF)*, IEEE, 2018, pp. 268–282 (cit. on pp. 4, 11, 33, 34, 36, 38, 42, 48, 53, 109, 110).

[YGQ+22] C.-H. Yao, B. Gong, H. Qi, Y. Cui, Y. Zhu, and M.-H. Yang, "Federated multi-target domain adaptation", in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 1424–1433 (cit. on p. 74).

[YHSC17] G. Yang, S. He, Z. Shi, and J. Chen, "Promoting cooperation by the social incentive mechanism in mobile crowdsensing", *IEEE Communications Magazine*, 55th vol., 3rd no., pp. 86–92, 2017 (cit. on p. 20).

[YJ22] Y. Yang and B. Jiang, "Towards group fairness via semi-centralized adversarial training in federated learning", in *2022 23rd IEEE International Conference on Mobile Data Management (MDM)*, IEEE, 2022, pp. 482–487 (cit. on pp. 80, 81).

[YLCT20] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A survey of autonomous driving: Common practices and emerging technologies", *IEEE access*, 8th vol., pp. 58 443–58 469, 2020 (cit. on p. 58).

[YNX+23] R. Ye, Z. Ni, C. Xu, J. Wang, S. Chen, and Y. C. Eldar, "Fedfm: Anchor-based feature matching for data heterogeneity in federated learning", *IEEE Transactions on Signal Processing*, 71st vol., pp. 4224–4239, 2023 (cit. on p. 59).

[YPN24] Y. Yang, A. Payani, and P. Naghizadeh, "Enhancing group fairness in federated learning through personalization", *arXiv preprint arXiv:2407.19331*, 2024 (cit. on p. 82).

[YWL+22] H. Yao, Y. Wang, S. Li, L. Zhang, W. Liang, J. Zou, and C. Finn, "Improving Out-of-Distribution Robustness via Selective Augmentation", in *Proceedings of the 39th International Conference on Machine Learning*, Proceedings of Machine Learning Research ser., vol. 162, PMLR, Jul. 2022, pp. 25 407–25 437 (cit. on p. 59).

[YWZ+21] M. Yang, X. Wang, H. Zhu, H. Wang, and H. Qian, "Federated learning with class imbalance reduction", in *2021 29th European Signal Processing Conference (EUSIPCO)*, IEEE, 2021, pp. 2174–2178 (cit. on pp. 23, 24, 90).

[YZC+24] W. Ye, G. Zheng, X. Cao, Y. Ma, and A. Zhang, "Spurious correlations in machine learning: A survey", *arXiv preprint arXiv:2402.12715*, 2024 (cit. on pp. 58, 59, 66).

[YZJE23] C. Yun-Hin, J. Zhihan, D. Jing, and N. C.-H. Edith, "Fedin: Federated intermediate layers learning for model heterogeneity", *arXiv preprint arXiv:2304.00759*, 2023 (cit. on p. 37).

[YZKG23] Y. Yang, H. Zhang, D. Katabi, and M. Ghassemi, "Change is hard: A closer look at subpopulation shift", in *International Conference on Machine Learning*, PMLR, 2023, pp. 39 584–39 622 (cit. on pp. 58, 61, 63, 72).

[YZKL22] H. Yang, X. Zhang, P. Khanduri, and J. Liu, "Anarchic federated learning", in *International Conference on Machine Learning*, PMLR, 2022 (cit. on p. 89).

[ZFH21] P. Zhou, P. Fang, and P. Hui, "Loss tolerant federated learning", *arXiv preprint arXiv:2105.03591*, 2021 (cit. on pp. 14, 18, 23, 27, 60, 92).

[ZKL+21] F. Zhang, K. Kuang, Y. Liu, L. Chen, C. Wu, F. Wu, J. Lu, Y. Shao, and J. Xiao, "Unified group fairness on federated learning", *arXiv preprint arXiv:2111.04986*, 2021 (cit. on p. 82).

[ZKW20] D. Y. Zhang, Z. Kou, and D. Wang, "Fairfl: A fair federated learning approach to reducing demographic bias in privacy-sensitive classification models", in *2020 IEEE International Conference on Big Data (Big Data)*, 2020, pp. 1051–1060. DOI: 10.1109/BigData50022.2020.9378043 (cit. on pp. 80–82).

[ZLF+24] S. Zhao, T. Liao, L. Fu, C. Chen, J. Bian, and Z. Zheng, "Data-free knowledge distillation via generator-free data generation for non-iid federated learning", *Neural Networks*, 179th vol., p. 106 627, 2024 (cit. on p. 54).

[ZLQ+20] Y. Zhan, P. Li, Z. Qu, D. Zeng, and S. Guo, "A learning-based incentive mechanism for federated learning", *IEEE Internet of Things Journal*, 7th vol., 7th no., pp. 6360–6368, 2020 (cit. on p. 26).

[ZLT+23] J. Zhang, A. Li, M. Tang, J. Sun, X. Chen, F. Zhang, C. Chen, Y. Chen, and H. Li, "Fed-cbs: A heterogeneity-aware client sampling mechanism for federated learning via class-imbalance reduction", in *International Conference on Machine Learning*, PMLR, 2023, pp. 41 354–41 381 (cit. on pp. 60, 65).

[ZMMN21] C. Zhou, X. Ma, P. Michel, and G. Neubig, "Examining and combating spurious features under distribution shift", in *International Conference on Machine Learning*, PMLR, 2021, pp. 12 857–12 867 (cit. on pp. 53, 54).

[ZS18] Z. Zhang and M. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels", *Advances in neural information processing systems*, 31st vol., 2018 (cit. on pp. 63, 64).

[ZSQ17] Z. Zhang, Y. Song, and H. Qi, "Age progression/regression by conditional adversarial autoencoder", in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2017 (cit. on p. 81).

[ZTII88] W. Zhang, J. Tanida, K. Itoh, and Y. Ichioka, "Shift-invariant pattern recognition neural network and its optical architecture", in *Proceedings of annual conference of the Japan Society of Applied Physics*, Montreal, CA, vol. 564, 1988 (cit. on pp. 9, 107).

[ZTL+21] A. Ziller, A. Trask, A. Lopardo, B. Szymkow, B. Wagner, E. Bluemke, J.-M. Nounahon, J. Passerat-Palmbach, K. Prakash, N. Rose, *et al.*, "Pysyft: A library for easy federated learning", in *Federated Learning Systems*, Springer, 2021, pp. 111–139 (cit. on p. 28).

[ZWL+24] J. Zhang, J. Wang, Y. Li, F. Xin, F. Dong, J. Luo, and Z. Wu, "Addressing heterogeneity in federated learning with client selection via submodular optimization", *ACM Transactions on Sensor Networks*, 20th vol., 2nd no., pp. 1–32, 2024 (cit. on p. 59).

[ZXS+23] C. Zhang, Z. Xiaoman, E. Sotthiwat, Y. Xu, P. Liu, L. Zhen, and Y. Liu, "Generative gradient inversion via over-parameterized networks in federated learning", in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 5126–5135 (cit. on p. 34).

[ZZH+21] Y. Zhan, J. Zhang, Z. Hong, L. Wu, P. Li, and S. Guo, "A survey of incentive mechanism design for federated learning", *IEEE Transactions on Emerging Topics in Computing*, 2021. [Online]. Available: `https://ieeexplore.ieee.org/abstract/document/9369019` (cit. on pp. 20, 21).

[ZZJ+20] Y. Zhao, J. Zhao, L. Jiang, R. Tan, D. Niyato, Z. Li, L. Lyu, and Y. Liu, "Privacy-preserving blockchain-based federated learning for iot devices", *IEEE Internet of Things Journal*, 8th vol., 3rd no., pp. 1817–1829, 2020 (cit. on pp. 19, 20, 22).

[ZZL+24] W. Zhang, T. Zhou, Q. Lu, Y. Yuan, A. Tolba, and W. Said, "Fedsl: A communication-efficient federated learning with split layer aggregation", *IEEE Internet of Things Journal*, 11th vol., 9th no., pp. 15 587–15 601, 2024 (cit. on pp. 2, 102).

[ZZW+21] R. Zeng, C. Zeng, X. Wang, B. Li, and X. Chu, "A comprehensive survey of incentive mechanism for federated learning", *arXiv preprint arXiv:2106.15406*, 2021 (cit. on pp. 20, 21).

[ZZWC20] R. Zeng, S. Zhang, J. Wang, and X. Chu, "Fmore: An incentive scheme of multi-dimensional auction for federated learning in mec", in *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*, IEEE, 2020, pp. 278–288 (cit. on pp. 2, 14, 19, 20, 23, 25, 28, 61, 92, 103).

EiDUA Escuela Internacional de Doctorado
Escola Internacional de Doctorat
International Doctoral School

eidua.ua.es