Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

SCHOOL OF ELECTRONIC ENGINEERING
DUBLIN CITY UNIVERSITY (DCU)

On-demand Crowdsourced Federated Learning over Edge Devices

Author
Mehreen Tahir, B.S.

Supervisors
Dr. Ali Intizar

A Dissertation submitted in fulfillment of the requirements for the award of Doctor of Philosophy (Ph.D.)

July 2025

Declaration

Dedication

To my mother, my rock, my strength, and the quiet force behind everything I've achieved. I know, more than anyone, the sacrifices you've made so your children could dream freely and stand tall in the world. From the quiet strength you carried through every hardship, to the countless times you put our needs above your own, I have watched, I have learned, and I carry those lessons with me always. Every word of this thesis, every page, is rooted in the love, resilience, and courage you showed me. I hope this makes you proud in the way you've always made me proud to be your daughter.

To my siblings, Ambreen, Furqan, Farhan, and Taskeen, thank you for always having my back, even from afar. Your constant support, prayers, and belief in me have meant more than I can express. Your encouragement gave me the strength on the days I needed it most, and I'm deeply grateful for that. And to my two little nieces, whose laughter and giggles brightened even the hardest days. You may be too young to understand it now, but your joy gave me the kind of comfort no words ever could.

To my friends, Aditya, Sidra, Fatima, and Hamza, I couldn't have survived this journey without you. Thank you for the game nights, the spontaneous hangouts, endless venting sessions, and most of all, for listening when I needed to vent, cry, or just sit in silence. Aditya, thank you for being there in ways big and small, seen and unseen. From all the coffees and endless chai sessions to the silly debates, shared frustrations, and laughter, you were always a constant. You helped lighten the hardest days, and your friendship carried me through more than you know. I'm endlessly grateful for your support and for standing by me without fail, through tight deadlines, caffeine-fueled rants, and everything

in between. And of course, special thanks for those late-night grammar checks during submissions. Even though I suspect Grammarly did most of the heavy lifting, you were the emotional support human I didn't know I needed, and your last-minute reviews still saved the day more often than not. Your support meant more than words can say. Sidra, you always believed in me. Every time I doubted myself, you reminded me how far we had come and how close we were to the finish line. Your faith lifted me more than you know. From all the loud chats in the lab to just random rant calls, I appreciated it all, even when I didn't say it in so many words. Fatima, thank you for bringing joy and warmth into our lives. From the very first meal I had in Ireland to all the homely gatherings you called us over for, you created pockets of home for us and reminded me what comfort and friendship feel like. Hamza, thank you for being there right from the very beginning. From Zoom introduction to picking me up from the airport and literally sorting everything, then and in many moments that followed, you've always been there for us. You've always been someone I knew I could rely on, calm and steady, like an older brother. Over time, you and Fatima became more than friends; you became family.

To my lab mates and colleagues at DCU, Amina, Ali, Halima and Mayug, thank you for the shared struggles and the small moments of camaraderie that made this long journey feel just a bit lighter. Whether it was a passing comment in the hallway or a cup of coffee or the lunches and dinners we shared, your presence and support helped shape this work in more ways than one.

Finally, to everyone who stood by me, cheered me on, listened to the rants, and kept me going, whether through a kind word, a shared coffee, or a quiet moment of understanding, thank you. This journey has been long and testing, but your support made it meaningful. This thesis carries a piece of each one of you, and I will always be grateful.

Acknowledgements

First and foremost, I am deeply grateful to Allah (SWT) for giving me the strength, patience, and resilience to complete this journey. Through every moment of doubt, every sleepless night, and every challenge along the way, it was His guidance and mercy that gave me the courage to carry on. I am truly humbled and thankful for the countless blessings that made this accomplishment possible.

I would like to express my sincere and heartfelt gratitude to my supervisor, Dr. Ali Intizar, for his continuous support and guidance throughout this research journey. Your critiques challenged me to refine my thinking, revisit my assumptions, and continuously improve the quality of my work. While the journey was not always easy, I recognize the role your direction played in shaping the final outcome of this thesis. I am thankful for the opportunity to learn and grow under your supervision.

I would also like to extend a special thanks to Dr. Feras Awaysheh and Dr. Sadi Alawadi, whose mentorship and encouragement made an immense difference throughout this journey. Your guidance as co-authors and thoughtful feedback helped me navigate both the challenges and milestones of this work, and I truly value everything I learned from working with you.

My heartfelt appreciation also goes to Prof. Ana Fernández Vilas (External Examiner), Prof. Derek Molloy (Internal Examiner), and Dr. Conor McArdle (Chairperson) for reviewing my thesis and for their valuable time, insights, and constructive feedback during the viva. Your thoughtful questions and suggestions helped me improve its quality.

I am deeply grateful to the ML-Labs and the SFI Centre for Research Training in Machine Learning for providing the funding and infrastructure that made this work possible. The resources, academic support, and training opportunities offered through the program gave me the space to grow as a researcher and pursue my ideas with freedom and focus.

To my lab mates and fellow researchers, thank you for the insightful conversations, peer support, and all the moments of shared learning along the way. Whether during group meetings or informal discussions, your input helped shape my understanding and strengthened the work.

Finally, I would like to thank all the participants and collaborators involved in the research for your time and contribution. This work would not have been possible without your involvement.

Publications

1. **Mehreen Tahir**, and Muhammad Intizar Ali. ”On the Performance of Federated Learning Algorithms for IoT.” IoT, vol. 3, no. 2, 2022, pp. 273–284. https://doi.org/10.3390/iot3020016.

2. **Mehreen Tahir**, and Muhammad Intizar Ali. ”Enabling On-Demand Crowdsourced Federated Learning Over IoT.” in Eighth International Conference on Fog and Mobile Edge Computing (FMEC). IEEE, 2023, pp. 128-134. https://doi.org/10.1109/FMEC59375.2023.10306078.

3. **Mehreen Tahir**, and Muhammad Intizar Ali. ”Client Selection in Federated Learning: Challenges, Strategies, and Contextual Considerations.” In: Rehman, M.H.u., Gaber, M.M. (eds) Federated Learning Systems. Studies in Computational Intelligence, vol 832. Springer, Cham. https://doi.org/10.1007/978-3-031-78841-3_3

4. **Mehreen Tahir**, and Muhammad Intizar Ali. ”Multi-Criterion Client Selection for Efficient Federated Learning.” Proceedings of the AAAI Symposium Series, vol. 3, no. 1, 2024, pp. 318–322. https://doi.org/10.1609/aaaiss.v3i1.31227.

5. **Mehreen Tahir**, Tanjila Mawla, Muhammad Intizar Ali, Sadi Alawadi, Maanak Gupta, and Feras Awaysheh. ”SecureFedPROM: A Zero-Trust Federated Learning Approach with Multi-Criteria Client Selection.” in IEEE Journal on Selected Areas in Communications, vol. 43, no. 6, pp. 2025-2041, June 2025, doi: 10.1109/JSAC.2025.3560008.

6. **Mehreen Tahir**, Feras Awaysheh, Sadi Alawadi, and Muhammad Intizar Ali. ”Bayesian Federated Learning with Stochastic Variational Inference.” 2nd International Conference on Federated Learning Technologies and Applications (FLTA), Valencia, Spain, 2024, pp. 290-297, doi: 10.1109/FLTA63145.2024.10840014.

Contents

List of Figures

List of Tables

On-demand Crowdsourced Federated Learning over Edge Devices

Mehreen Tahir

Abstract

Federated learning (FL) is attributed to training a machine learning (ML) model over a number of distributed devices while keeping all their training data localized. Under these settings, the edge devices perform computations on their local data before sending the required updates to the central server to improve the global model. This approach has shown great potential since hundreds of devices can potentially contribute to learning a single task without sharing their local data. Despite its success in many domains, current FL systems face significant challenges in scaling client participation and handling data heterogeneity, which impede the training of high-performing models.

To address these limitations, this thesis argues that there's a need for a dynamic learning platform where edge devices could volunteer to collaboratively learn a task through FL. It further proposes FedOnDemand, an on-demand crowdsourced FL framework that dynamically incorporates edge devices based on demand and availability. The research aims to optimize client participation and resource allocation in FL systems to ensure an efficient and scalable learning process.

We present a novel client selection mechanism designed to optimize the contribution of clients based on their computational resources and data quality. By implementing a multi-criterion client selection protocol, the system dynamically selects clients based on their suitability for a given FL task. To secure this process, we incorporate attribute-based access control measures, ensuring that client selection is both effective and secure. This approach not only enhances the quality of the contributions but also safeguards the

integrity of the FL process.

To manage data heterogeneity and improve model robustness, we model FL as a Bayesian process. Clients employ Stochastic Variational Inference (SVI) to approximate local posterior distributions, while the server utilizes Bayesian learning techniques to aggregate these updates, effectively managing uncertainty. Furthermore, we explore fairness-aware incentive mechanisms based on data valuation, ensuring clients are rewarded proportionally to their contributions. These mechanisms are designed to foster active and robust participation across diverse network environments. Empirical evaluations using benchmark datasets demonstrate significant improvements in convergence speed, model accuracy, and system scalability compared to traditional FL approaches.

This research contributes to the field by providing a framework that enhances the operational efficiency of FL models and ensures greater participant engagement and system integrity. The implications of this study are far-reaching, potentially influencing future designs of ML systems that require decentralized data inputs across highly dynamic and privacy-sensitive environments.

Chapter 1

Introduction

1.1 Research Motivation

In recent years, the rapid proliferation of the Internet of Things (IoT) and edge devices has led to an unprecedented surge in data generated at the network's edge. According to recent data from Statista, the number of connected IoT devices is projected to exceed 27.1 billion by 2028[1]. During the same period, global data volumes are forecast to rise to approximately 394 zettabytes (ZB)[2], a significant portion of which will be generated by IoT and edge devices. These IoT devices are an integral part of smart ecosystems such as smart homes, wearable health monitors, industrial sensors, etc. These devices continuously generate diverse, context-rich data streams, which can be exploited to train sophisticated machine learning (ML) models. For instance, in the healthcare domain, wearable devices such as Fitbit and Apple Watch continuously monitor patient vitals, detecting early signs of conditions like atrial fibrillation or hypertension. Machine learning models trained on such data can improve early diagnosis and personalized healthcare interventions, allowing hospitals and telemedicine platforms to detect anomalies more effectively and optimize treatment plans [1]. Similarly, in smart cities, sensors embedded in roadways and public transportation systems continuously collect data on traffic con-

[1] https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/
[2] https://www.statista.com/statistics/871513/worldwide-data-created/

gestion, pedestrian movements, and energy usage. ML models trained on this data can enable dynamic traffic light adjustments, predictive maintenance of public infrastructure, and real-time energy grid optimization, improving urban efficiency [2].

However, Traditional ML approaches rely on centralized data collection and processing, where data from various sources is aggregated in data centers or cloud servers for the purpose of training models. While effective in controlled environments, this approach faces significant challenges in the context of edge environments. The sheer volume of data generated by edge devices makes centralized collection impractical due to bandwidth limitations, high communication costs, and increased latency [3]. Moreover, transmitting sensitive data over networks raises serious privacy and security concerns. Stringent data privacy regulations like the General Data Protection Regulation (GDPR) and the California Consumer Privacy Act (CCPA) impose strict requirements on data handling and sharing, limiting the feasibility of centralized data collection.

To mitigate the aforementioned issues, edge computing has become increasingly popular, enabling data processing directly at the device level rather than relying on centralized cloud infrastructure [4]. Fog computing emerged as an extension of edge computing, incorporating intermediary nodes such as gateways and local servers to further distribute computation and storage closer to the data source. Initially, these paradigms were introduced to execute simple queries over low-powered, distributed devices [5, 6]. More recently, research in this field has focused on training ML models centrally and then deploying the trained models on local devices to provide personalization, and mobile user modeling [7]. However, with the advancements in storage and computational capabilities of edge devices, we can utilize their local resources to train ML models using distributed data sets. This led to the inception of Federated Learning (FL), a decentralized ML paradigm that enables the edge devices to collaboratively train ML models. Unlike traditional ML approaches, FL shifts the learning process closer to the data source. In this paradigm, a central server distributes an untrained or partially trained ML model, known as the global model, to participating edge devices. Each participating device,

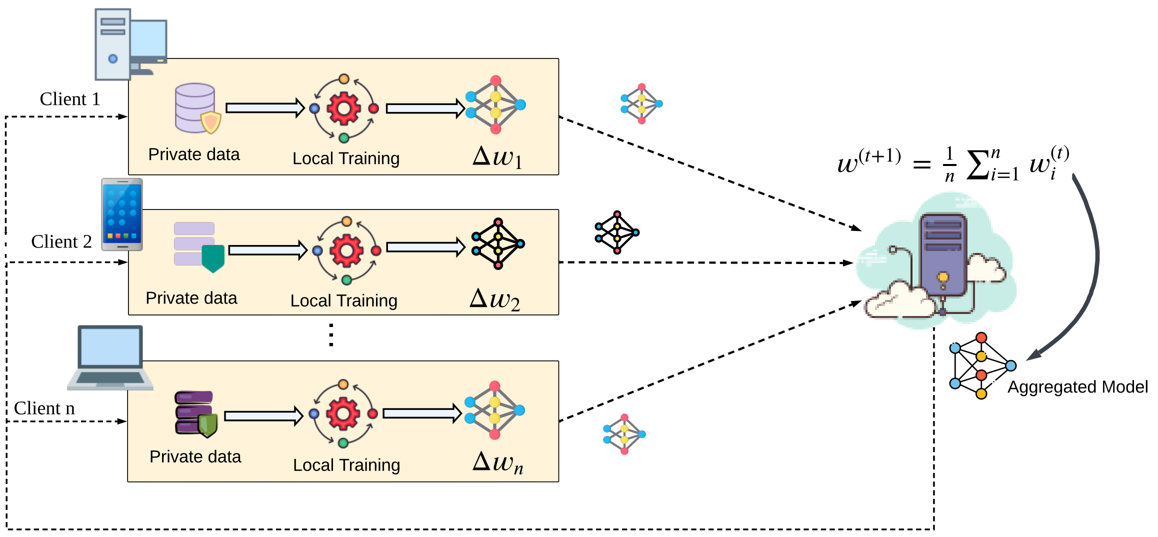


Figure 1.1: Federated Learning Architecture

often referred to as a client, performs localized training using its local data, updating the model parameters through optimization techniques like stochastic gradient descent (SGD). Once the local training phase is complete, clients transmit these updated model parameters back to the central server. The server aggregates the updates from all the clients using techniques such as model averaging to update the global model parameters and sends the updated model back to the clients. This process is repeated over multiple communication rounds until the global model converges to a stationary point or predefined communication rounds are complete [8]. A classic example of this is the Google keyboard (Gboard). When the keyboard shows a possible suggestion, the phone locally stores the information about the current context and whether the suggestion was clicked. This on-device information is then used to improve the next iteration of Gboard suggestions [9]. Figure 1.1 provides a visual representation of classic FL architecture and its training process.

The standard FL problem involves learning a globally shared statistical model from data residing on tens to potentially thousands of remote devices. This model is trained under the constraint that the data generated on the edge device is kept and processed locally, with only periodic updates communicated to the central server. By keeping data local to the device, FL minimizes the risk of data breaches and complies with privacy regulations, making it especially suitable for applications in healthcare [10, 11], finance [12], recommendation systems [13], and other sensitive domains. It also reduces communication overhead, as only model parameters or gradients are transmitted over the network instead of sending large volumes of raw data.

Despite the success and its benefits, FL continues to face a range of challenges that limit its broader applicability. These challenges include non-independent and identically distributed data (non-IID) across clients, communication efficiency, security, and, more importantly, auditing and scaling both the data and clients to optimize the performance of the federated network. Since the edge devices only have limited data to train the learning model, a key open research challenge is to motivate a large and diverse pool of clients, collectively providing sufficient data samples to ensure cooperation for training high-quality models. However, existing FL approaches largely operate under controlled assumptions, such as homogeneous data distributions and fixed participation patterns. These assumptions do not align with the realities of edge networks, where devices and data exhibit significant heterogeneity and dynamic availability. For example, in intelligent transportation systems, autonomous vehicles and roadside sensors continuously collect traffic and environmental data. However, these clients are highly mobile and may only contribute intermittently, making traditional FL approaches impractical due to their reliance on predefined client participation. In such dynamic environments, the need for an adaptable and scalable FL framework becomes evident—one that can adjust to varying participation levels, support heterogeneous data sources, and incentivize long-term engagement.

Additionally, the existing implementations of FL primarily focus on learning a single task with a single objective. This narrow scope significantly limits the potential of FL, as it restricts the range of applications and the adaptability of the models to dynamic, real-world scenarios. These implementations also assume that participating devices inherently possess subsets of the required training data, which oversimplifies the complexities of practical deployments. Such constraints limit the FL from evolving into a truly versatile framework capable of supporting diverse applications across a wide range of edge environments.

To address these limitations, this thesis envisions a more dynamic and generalized FL framework that transcends the constraints of existing approaches. The goal is to design a platform where devices can volunteer to participate and collaboratively learn multiple tasks and objectives in a flexible and crowdsourced manner. This vision relies on creating mechanisms that enable seamless integration of devices with varying capabilities, while dynamically adapting to the heterogeneity of data, resources, and network conditions. By fostering an open and inclusive ecosystem, this platform would support on-demand analytics over a large number of edge devices, enabling FL to unlock its full potential. Achieving this outcome requires a paradigm shift towards frameworks that embrace the inherent complexities of real-world edge networks while promoting voluntary, equitable, and secure collaboration across diverse participants.

1.2 Research Questions

Based on the challenges, vision, and opportunities identified above, we have formulated the following key research questions to guide this thesis:

RQ1: How can we design a dynamic and open FL ecosystem that supports voluntary client participation while improving scalability and model performance? Specifically, what mechanisms can enable on-demand participation, while adapting to client availability and ensuring their effective contribution to the global model?

RQ2: How can the FL training process be optimized to effectively handle client and data heterogeneity to enhance efficiency and accuracy? Specifically:

RQ2.1: What optimization techniques can be used to dynamically manage client participation and effective resource utilization, ensuring training efficiency and model accuracy in diverse environments?

RQ2.2: What techniques can be used to manage non-IID data in FL to ensure robust model convergence and accuracy, particularly in the presence of uncertainty arising from diverse data distributions?

RQ2.3: How can the challenges of heterogeneous client updates be addressed to maintain consistent global model performance while minimizing variability and uncertainty?

RQ3: How can we ensure the security and trustworthiness of an open FL ecosystem, protecting the global model and client data from adversarial threats, unauthorized access, and vulnerabilities inherent to decentralized systems?

RQ4: How can fairness-aware incentive mechanisms be designed to motivate active and sustained client participation, ensuring equitable compensation while promoting long-term engagement and scalability?

1.3 Contribution of the Thesis

This dissertation aims to enhance the scalability, efficiency, security, and fairness of FL in dynamic environments, particularly over edge devices. To achieve this, we introduce a crowdsourced FL framework that enables voluntary client participation while addressing key challenges related to client heterogeneity, security risks, and incentive mechanisms. The proposed approach integrates multiple advancements in FL, including dynamic client selection, optimization techniques for training on heterogeneous data, security mechanisms for decentralized learning, and incentive models for sustained participation. The primary contributions of this dissertation are summarized as follows:

1. **Crowdsourced Federated Learning with Dynamic and Voluntary Client Participation**

   To improve FL scalability and flexibility, we propose a crowdsourced framework to enable dynamic and voluntary client participation in the FL training process.

This approach allows clients to join and leave the FL process on demand, effectively managing the variability in client availability and willingness to participate. By facilitating on-demand client participation, the ecosystem leverages the computing power and data contributions of a diverse set of clients, enhancing the scalability and performance of the global model. The open participation model addresses the challenge of engaging a wide range of devices in the FL process, ensuring a scalable and adaptable system for real-world FL deployments.

2. **Optimizing FL Training in Heterogeneous Environments**

   To address the challenges posed by data and system heterogeneity, we developed strategies to optimize the FL training process, improving model performance and reducing overall training time. This includes:

   - **Effective Utilization of Client Resources and Adaptive Client Management through Multi-Criterion Client Selection**

     We designed a sophisticated multi-criterion client selection mechanism that evaluates and selects clients based on various criteria such as computational resources, data quality, network conditions, and device availability. By dynamically adapting to the current state of the network and the specific requirements of the FL task, this approach ensures that the FL process capitalizes on the most relevant data and computational power available. This enhances training efficiency and model accuracy by selecting clients whose participation will most benefit the global model, thereby optimizing the trade-offs between performance and training time.

   - **Bayesian Federated Learning**

     To handle the uncertainty and variability inherent in non-IID data across clients, we integrated Bayesian methods into the FL framework. Specifically, we employed Stochastic Variational Inference (SVI) on the client side and Hierarchical Bayesian Modeling on the server side. This Bayesian approach allows

FL to model the distributional diversity of client data more effectively, ensuring robust model convergence and accuracy despite heterogeneous data distributions. By accounting for uncertainty and incorporating prior knowledge, the Bayesian methods enhance the learning process, making it more resilient to the challenges posed by data heterogeneity.

3. **Implementation of Zero-Trust Security Architecture for Federated Learning**

   Recognizing the critical importance of security in FL, we implemented robust security mechanisms to protect the global model and client data from unauthorized access and adversarial threats, especially in crowdsourced environments. We integrated zero-trust security principles into the FL framework, developing a secure Attribute-Based Access Control (ABAC) mechanism. This system allows for fine-grained control over client participation, ensuring that only authorized and trustworthy devices can contribute to the model training process. By verifying the attributes of each client before granting access, the ABAC mechanism prevents unauthorized participation and reduces the risk of malicious activities. This security layer is essential for protecting the integrity of the global model against potential adversarial attacks, such as data poisoning, and ensures that the learning process remains secure and reliable.

4. **A Fairness-Aware Incentive Mechanism for Crowdsourced Federated Learning**

   To encourage active and sustained participation from clients, we designed a fairness-aware incentive mechanism grounded in data valuation. The mechanism calculates the marginal contribution of each client's data to the overall model performance, ensuring that rewards are distributed proportionally based on the value of their contributions. By fairly compensating clients based on the quality and quantity of their data, the incentive mechanism addresses concerns over resource consumption and privacy. This promotes equitable participation and maintains client trust in the FL network, which is crucial for the long-term success and scalability of the FL ecosystem.

5. **Empirical Validation**

   We provide empirical evidence through extensive simulations and experiments using benchmark datasets and real-world data to validate the proposed methods. The results demonstrated significant improvements in model accuracy, convergence speed, and overall system robustness compared to traditional FL approaches. This validation confirms the effectiveness of integrating crowdsourcing mechanisms, adaptive client management, Bayesian learning, robust security measures, and fair incentivization within the FL framework, showcasing the practical applicability of the proposed solutions and their potential impact on real-world FL deployments.

Together, these contributions constitute a practical and extensible framework for enabling scalable FL in dynamic, real-world settings. By addressing challenges related to participation, heterogeneity, trust, and sustainability, the proposed approach advances the broader goal of making FL more inclusive, secure, and adaptable.

1.4 Organization

The rest of this dissertation is organized as follows: Chapter 2 provides the technical background, discussing the foundational concepts of distributed learning, edge computing, and the principles of FL. It also highlights the key challenges FL faces, including heterogeneity, security, scalability, and incentive mechanisms. Chapter 3 presents an extensive literature review, examining existing solutions and identifying gaps in key areas, including heterogeneity in FL, client selection strategies, Bayesian learning approaches, security mechanisms, incentivization, and the use of crowdsourcing in distributed systems.

The subsequent chapters detail the primary contributions of this work. Chapter 4 introduces the concept of Crowdsourced Federated Learning, focusing on enabling dynamic and voluntary client participation to create an open ecosystem. Chapter 5 focuses on optimizing the FL training process in heterogeneous environments. It introduces a multi-criterion client selection mechanism, details its design and implementation, and discusses its impact on model performance and training efficiency. Chapter 6 explores Bayesian Federated Learning, employing Stochastic Variational Inference (SVI) to address the challenges of non-IID data distributions and improve global model robustness and accuracy.

Chapter 7 discusses the implementation of robust security mechanisms within the FL framework. It explores the integration of zero-trust security principles and the development of a Secure Attribute-Based Access Control (ABAC) mechanism and Public Key Infrastructure (PKI) certificates to protect the global model and client data from unauthorized access and adversarial threats. Chapter 8 introduces a fairness-aware incentive mechanism using the Shapley Value to calculate the contribution of each client's data, ensuring equitable rewards and sustained participation. Finally, Chapter 9 concludes the dissertation by summarizing the key findings, reflecting on the limitations of the research, and offering directions for future work.

Chapter 2

Background

The exponential growth of data and the increasing demand for real-time processing have propelled advancements in machine learning (ML) technologies. Traditional centralized ML approaches, which rely on aggregating large volumes of data in a single location, face numerous challenges, including scalability limitations, data privacy concerns, and substantial communication overheads. To address these issues, distributed computing frameworks have emerged, enabling the parallel processing of data across multiple nodes and devices. This chapter explores the evolution of these paradigms, highlighting their principles, benefits, and limitations, and examines how FL integrates their strengths to enable privacy-preserving and decentralized model training. The chapter concludes by discussing the key challenges in FL, setting the stage for the research and contributions presented in subsequent chapters.

2.1 Distributed Learning

Distributed learning involves distributing the training process of ML models across multiple machines to manage extensive datasets and overcome the computational limitations inherent in single-machine environments. This approach enhances both the scalability and efficiency of model training by leveraging the combined computational power and stor-

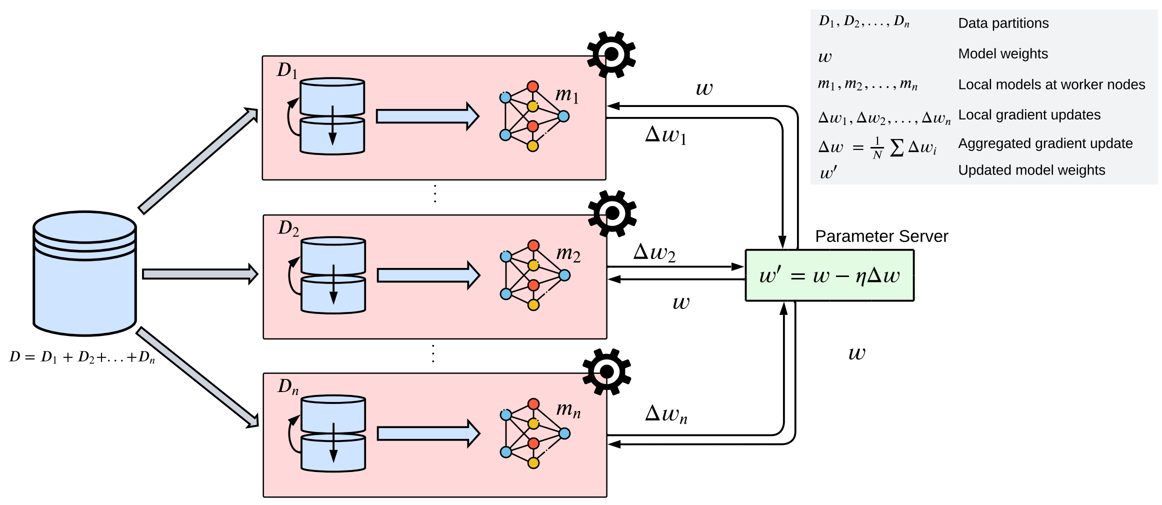

$D_1, D_2, \ldots, D_n$ Data partitions
w Model weights
$m_1, m_2, \ldots, m_n$ Local models at worker nodes
$\Delta w_1, \Delta w_2, \ldots, \Delta w_n$ Local gradient updates
$\Delta w = \frac{1}{N}\sum \Delta w_i$ Aggregated gradient update
w' Updated model weights
D_1
m_1
D_2
m_2
D_n
m_n
w
Δw_1
Δw_2
Δw_n
Parameter Server
$w' = w - \eta \Delta w$
$D = D_1 + D_2 + \ldots + D_n$

Figure 2.1: An example of Data Parallelism in Distributed Learning

age capacity of a distributed network [14, 15]. By partitioning the workload, distributed machine learning facilitates the handling of large-scale data that would otherwise be impractical to process on a single machine, thereby accelerating the training process and improving model performance [16].

One of the primary methodologies in distributed learning is distributed data parallelism, where a large dataset is partitioned into smaller subsets that are distributed across different computational nodes. It involves duplicating the model across multiple GPUs or CPU cores, and each processes a subset of the data simultaneously [17]. Periodically, the results of the models are combined through a parameter server, resulting in a global model [18]. This approach significantly reduces computation time with negligible or no model performance degradation and enables the handling of datasets that exceed the memory capacity of individual machines. Figure 2.1 provides a visual representation of data parallelism in distributed learning.

The effectiveness of distributed data parallelism has been demonstrated in various large-scale machine learning tasks. For instance, Google's DistBelief system utilized distributed data parallelism to train deep neural networks on massive datasets, achieving substantial improvements in training speed and model accuracy [19]. Similarly, Facebook's implementation of synchronous distributed training enabled the training of ResNet-50 on the ImageNet dataset in record time by leveraging numerous GPUs across multiple servers [20].

Despite its advantages, distributed learning introduces several challenges such as communication overhead, synchronization efficiency, and above all, data privacy and security. Moreover, a critical assumption in many distributed learning frameworks is that the data distribution across different nodes is independent and identically distributed (IID) or that the system designer can manipulate and adjust the data distribution to achieve IID conditions, thereby facilitating faster convergence [21, 22]. However, in real-world scenarios, data is often non-independent and identically distributed (non-IID), posing significant challenges to model training and convergence rates. Addressing these assumptions is essential for developing more robust and flexible distributed learning systems that can operate effectively in heterogeneous environments.

2.2 Edge Computing

Edge computing extends distributed learning by bringing computation and data storage closer to the data sources, typically at the edge of the network, on devices like smartphones, sensors, and IoT devices [3, 23]. This approach addresses several key challenges identified in distributed learning, particularly those related to communication overhead and latency. By processing data locally, edge computing significantly reduces the need to transmit large volumes of raw data to centralized servers, thereby conserving bandwidth and enabling real-time analytics [24]. This is especially beneficial for latency-sensitive applications, such as autonomous vehicles [25, 26], industrial automation [27], and healthcare diagnostics [28]. This decentralization also ensures that each device can independently process and analyze the data it generates, creating a more distributed approach to resource utilization. Figure 2.2 provides a visual representation of an edge computing architecture.

Unlike distributed learning systems that require a robust communication infrastruc-

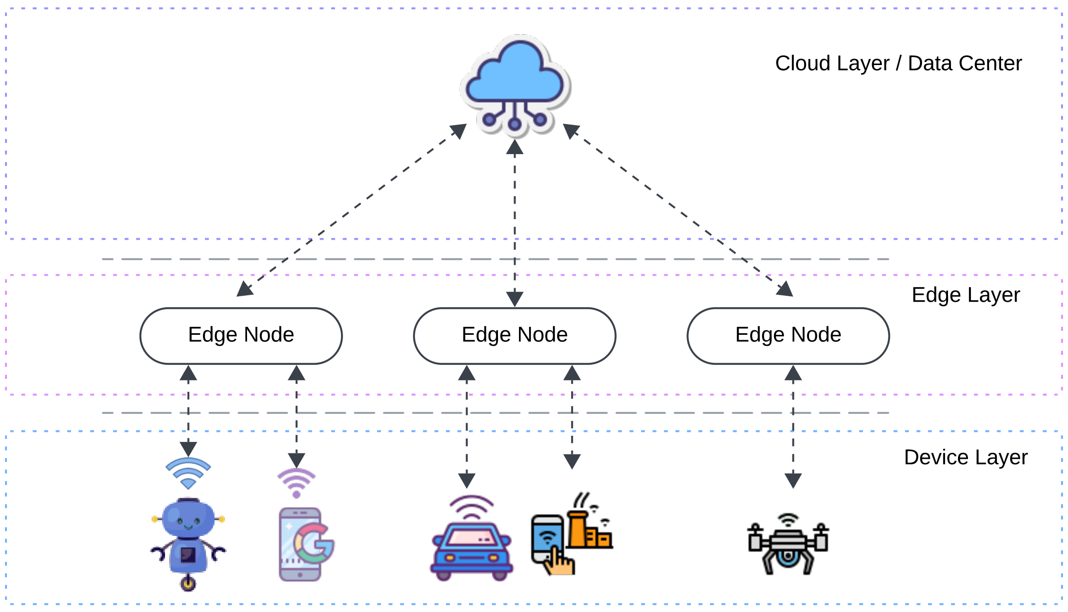


Figure 2.2: An example Edge Computing Architecture

ture to synchronize nodes or aggregate global insights, edge computing primarily focuses on localized operations. Each edge device processes its data independently, often tailored to specific contextual requirements, without necessitating regular communication or synchronization with other devices in the system [29]. This approach makes edge computing particularly suitable for environments with limited or unreliable network connectivity, where maintaining consistent communication between devices is impractical.

While distributed learning frameworks often assume a homogeneous environment with uniform computational resources and data distributions, edge computing is inherently designed to operate in heterogeneous environments. Edge devices vary significantly in terms of hardware capabilities, energy availability, and operational conditions. This diversity is not merely accommodated but forms a fundamental characteristic of edge computing, requiring algorithms and models that can adapt dynamically to varying resource constraints [30, 31]. This flexibility allows edge computing to thrive in settings where traditional distributed systems would struggle to function effectively. Edge computing also emphasizes task-specific optimization, wherein models and algorithms are often designed or pruned to fit the specific requirements and constraints of individual devices. This differs from broader distributed learning strategies, which typically aim to achieve global optimization and uniformity across all participating nodes [32, 33]. By tailoring computations to localized needs, edge computing maximizes efficiency while minimizing unnecessary overhead.

While edge computing is a powerful paradigm for enabling localized and resource-aware computation, it faces challenges in coordinating learning as the scale and complexity of distributed applications grow. The device-centric design of edge computing and the lack of a cohesive framework for collaborative learning across devices keep the learning isolated, leading to fragmented models that fail to generalize across the system. Additionally, the diverse and dynamic environments of edge devices make it difficult to coordinate their contributions effectively [34]. Ensuring that devices with varying capabilities, data distributions, and connectivity can collectively improve learning outcomes requires a framework that balances autonomy with collaboration, which edge computing alone cannot achieve. These limitations set the stage for FL, which introduces mechanisms to enable coordinated, decentralized learning while preserving the autonomy of edge devices.

2.3 Federated Learning

Federated Learning (FL) builds upon the edge computing paradigm by introducing a structured approach to decentralized learning, where multiple edge devices collaboratively train a shared model without exchanging raw data. It provides a framework that allows distributed devices to contribute to a global model while maintaining local autonomy. This section provides the necessary background on FL's core principles, its iterative training process, and the fundamental methods used to aggregate model updates across participating clients.

The goal of FL is to learn a global model w that fits all client's $C = \{c_1, c_2, \ldots, c_n\}$ local data, where c_i represents the i^{th} client in the system. In a typical FL setup, the central server initializes the global model w_0 at communication round $t = 0$. For each subsequent round $t \geq 1$, the server selects a subset of clients $S_t \subseteq C$ from the set of all available clients to participate in that round. The global model w_t is then sent to each selected client $c_i \in S_t$. The clients proceed to train the received global model w_t using its local dataset D_i by performing E epochs of Stochastic Gradient Descent (SGD). The goal of local training is to minimize the client's loss function, defined as:

$$F_i(w_i) = \min_{w_i} \left(\frac{1}{|D_i|} \sum_{j \in D_i} f_j(w_i) \right) \tag{2.1}$$

Here, w_i represents the local model parameters for client c_i, and $f_j(w_i)$ is the loss for data sample j within the dataset D_i. The objective is to determine the optimal model parameters w_i that minimize this local loss function $F_i(w_i)$. To achieve this, the local model is iteratively refined over E epochs, following the update rule:

$$w_i^{r+1} = w_i^r - \eta \nabla F_i(w_i^r) \tag{2.2}$$

where $r = 0, 1, \ldots, E-1$ represents the local epochs, η is the local learning rate, and $\nabla F_i(w_i^r)$ is the gradient of the local loss function with respect to w_i^r.
After local training, each client obtains an updated local model w_i^{t+1}. The client then sends this updated model back to the central server. Upon receiving the updated models $\{w_i^{t+1}\}_{c_i \in S_t}$ from the clients, the central server aggregates them to update the global model w_{t+1}. A common aggregation method is Federated Averaging (FedAvg), which computes a weighted average of the received updates based on the number of data samples at each client [8]:

$$w_{t+1} = \frac{1}{N_t} \sum_{c_i \in S_t} |D_i| w_i^{t+1} \tag{2.3}$$

where $N_t = \sum_{c_i \in S_t} |D_i|$ is the total number of data points across all selected clients in round t. This weighting ensures that clients with more data have a proportionally greater influence on the updated global model.

The server then broadcasts the updated global model w_{t+1} to a (potentially different) subset of clients S_{t+1} in the next communication round $t + 1$. This iterative process continues until convergence criteria are met, such as a satisfactory level of model accuracy or a maximum number of communication rounds.

By minimizing the aggregate loss function across all clients, FL seeks to find a global model that generalizes well despite the data being distributed and potentially heterogeneous. The use of FedAvg as the aggregation method is particularly effective under the assumption of data being IID across clients. However, in practical scenarios where data is often non-IID, additional techniques and considerations are required to ensure convergence and robust model performance. It is also important to note that the learning rate η, the number of local epochs E, and the client selection strategy significantly impact the convergence behavior and performance of the FL process. Balancing these parameters is essential for efficient training, especially in environments with resource constraints and varying data distributions.

While the decentralized architecture and privacy-preserving mechanisms of FL make it a powerful tool for enabling collaborative ML, it continues to face various challenges both internally in its design and in its external environment. These challenges include managing communication overhead, ensuring security and trust within the federated system, and addressing the scalability of clients, data, and models. Additionally, incentivizing a larger pool of clients to participate voluntarily, despite limited local data and computational resources, remains a key open research problem.

2.4 Challenges in Federated Learning

This section explores the challenges in FL in detail.

2.4.1 Scalability

Scalability is a primary challenge in FL, particularly as the number of participating clients increases. To train sophisticated models capable of generalizing well, FL systems require contributions from a large and diverse set of clients [35]. In many edge learning scenarios, individual clients possess limited amounts of data, which may not be representative or diverse enough to train an effective model independently. For example, in an FL setup for smart city applications, individual sensors or devices may only collect data related to their immediate surroundings, such as local traffic flow, air quality, or energy usage. Aggregating data from numerous sensors distributed across the city is essential to capture comprehensive patterns and variability necessary for training robust models that can optimize urban traffic management, environmental monitoring, or energy distribution [36]. Without sufficient data from a wide array of sources, the models may fail to accurately represent city-wide conditions, limiting their effectiveness in making informed decisions for smart city initiatives.

However, as the number of participating clients increases, issues such as load balancing, efficient client selection, and resource allocation become increasingly complex. Managing and coordinating training across a vast and potentially dynamic set of clients can also strain the central server [37], leading to bottlenecks and degraded performance if not managed effectively. The exchange of model updates between the server and clients results in significant communication costs and latency. Even though FL reduces the need to transmit raw data, the cumulative communication required for model updates in large-scale networks can overwhelm network resources, especially in bandwidth-constrained environments [38]. Additionally, the variability in client availability and participation rates further complicates the scalability of FL systems, necessitating robust mechanisms to ensure consistent and efficient model training across diverse and fluctuating client populations [39].

2.4.2 Communication Efficiency

FL usually requires iterative and frequent data exchanges between clients and the central server. The local model updates from clients, which include gradients or weight parameters, must be transmitted across the network after each local training iteration. Modern deep learning models often contain millions of parameters, resulting in substantial data transmission requirements each time an update is sent or received [40]. This continual communication can lead to significant bandwidth consumption and increased latency, particularly in environments with limited or unreliable network connectivity [41, 42, 43], as the time consumed in transmitting large model updates can significantly slow down the overall training process, delaying the convergence of the global model [44].

Moreover, the heterogeneous nature of client connections complicates the communication efficiency challenge further. Clients exhibit diverse network conditions, such as varying bandwidth capacities and fluctuating connectivity, resulting in asynchronous and inconsistent communication patterns [45]. These disparities can lead to delays in model aggregation, as the central server may need to wait for updates from slower or intermittently connected clients, thereby extending the training time. Techniques such as model compression [46, 47], gradient compression [48], quantization [49], federated dropout [50] and sparse updates [51] have been proposed to mitigate these issues by reducing the size of the transmitted updates [52, 53]. However, achieving a balance between reducing communication overhead and maintaining model accuracy remains a challenge, as excessive compression can degrade the quality of the aggregated model.

2.4.3 Heterogeneous Data and Non-IID Distributions

Distributed optimization problems are often modeled under the assumption that data is IID, which means that each data point is drawn from the same distribution independently of others. However, the data residing on each client device in FL often originates from diverse sources and reflects varying usage patterns, environments, and user behaviors. For instance, in a mobile keyboard prediction application, different users may have unique typing habits, vocabularies, and contextual usage, resulting in highly personalized and heterogeneous datasets on their respective devices [9]. Similarly, in healthcare applications, patient data can vary widely based on demographic factors, medical histories, and treatment protocols, leading to non-uniform data distributions across different hospitals or clinics [54]. This inherent diversity leads to non-IID data distribution across clients, which adds complexity to problem modeling, solution formulation, analysis, and optimization, as the global model must effectively learn from heterogeneous data sources without being biased towards any particular client's data distribution.

One of the primary implications of non-IID data in FL is the potential for slower convergence and reduced model accuracy. The non-IID nature of data can cause the local models trained on individual clients to diverge significantly. Traditional aggregation mechanisms in FL, such as Federated Averaging (FedAvg), which aggregate model updates by averaging gradients or weights from participating clients, may struggle to aggregate these updates to form a generalized global model. If certain clients have disproportionately influential data due to the size or nature of their datasets, the resulting model may become biased towards those clients, and the global model struggles to generalize well across all clients [55]. This inconsistency complicates the optimization process and leads to a slower convergence or even divergence of the global model in some cases. As a result, ensuring consistent and reliable model performance in the face of heterogeneous data remains a challenge for federated networks.

2.4.4 Resource Constraints and Client Heterogeneity

FL operates in highly decentralized environments where participating clients exhibit significant heterogeneity not only in terms of data but also computational capabilities, memory capacity, and energy resources. This dual aspect of heterogeneity presents significant obstacles to effective and efficient FL deployment. Clients range from powerful servers with abundant processing power and storage to resource-constrained devices such as smartphones, IoT sensors, and embedded systems that have limited computational resources and battery life. This variability in hardware capabilities means that while high-performance devices can perform extensive local computations and participate actively in multiple training rounds, the devices with constrained resources may struggle to perform intensive local computations required for training complex ML models, leading to delays in their participation, or even dropout from the training process altogether [56]. This inconsistency can result in uneven contributions to the model, where the updates from more capable clients disproportionately influence the global model, potentially biasing its performance and limiting its generalizability across diverse data distributions [57, 58]. Moreover, memory limitations on edge devices restrict the size and complexity of the models that can be trained locally. Larger models may not fit into the available memory, often necessitating the use of either lightweight and compressed models or forcing devices to offload parts of the computation to external resources, which can introduce additional latency and security concerns.

The client heterogeneity further extends to the variability in energy constraints and network connectivity. Intensive local computations required for training can lead to rapid battery depletion in battery-operated devices, reducing the operational lifespan of these devices and limiting their availability for ongoing participation in the FL process [59, 60]. This energy drain not only affects the individual device's usability but also impacts the overall training timeline, as devices may need to conserve energy by limiting their participation or by shutting down intermittently. Additionally, varying network speeds across clients can create disparities in the time it takes for updates to be transmitted and received, leading to delayed or missed updates and affecting the convergence of the global model [61]. Combined, these factors make it challenging to maintain an efficient training workflow under heterogeneous clients.

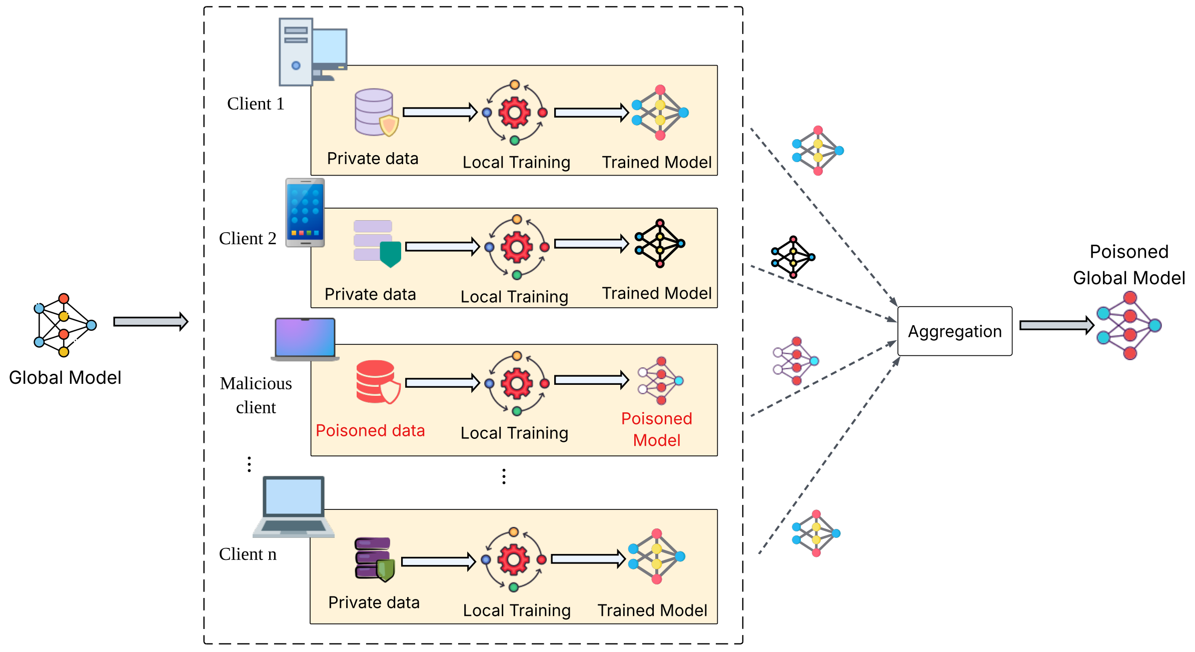


Figure 2.3: An example of Data and Model Poisoning in FL

2.4.5 Data Privacy and Security

FL fundamentally aims to enhance data privacy by keeping the client data localized, thereby mitigating the risks associated with centralized data storage and transmission. However, the decentralized and distributed nature of FL introduces unique security challenges, particularly concerning the trustworthiness of participating clients. In traditional centralized machine learning, data and computations are managed within secure, controlled environments. However, in FL, the server must rely on numerous clients to perform local computations and share model updates without direct oversight, raising concerns about the integrity and reliability of the training process.

One of the primary security issues in FL is the risk of malicious or unreliable clients participating in the training, as shown in Figure 2.3. furthermore, since clients operate independently and may join or leave the network dynamically, there is a potential for adversaries to infiltrate the system by masquerading as legitimate clients. Malicious clients can impair the training process in several ways:

- **Unauthorized Access:** Without proper mechanisms to verify and authenticate clients, unauthorized entities could gain access to the FL process. Even though raw data is not shared, the gradients or weights transmitted during the training process can inadvertently reveal sensitive information about the local datasets [38]. Adversaries with access to these updates can employ inference attacks to extract private data or infer characteristics about individual data points [62]. This poses significant privacy risks, especially in scenarios where clients hold highly sensitive or proprietary information.

- **Data Poisoning:** Clients with compromised or malicious intent might use contaminated or fabricated data for local training. This can lead to the global model learning incorrect patterns, affecting its reliability and trustworthiness [63, 64].
- **Model Poisoning:** Adversaries may intentionally alter their local model updates to degrade the performance of the global model. By submitting incorrect or corrupted updates, they can introduce biases or reduce the model's accuracy [65, 66].

Additionally, the absence of robust client authentication and authorization mechanisms in many FL systems further aggravates these security concerns. Unlike centralized systems, where access control can be enforced strictly, FL's distributed nature makes it challenging to monitor and regulate client activities effectively. The server often has limited information about the clients, making it difficult to assess their trustworthiness or enforce participation policies.

2.4.6 Performance Trade-offs

FL systems often involve inherent trade-offs between model performance [67], computational efficiency [68, 69], communication costs[70, 71] and privacy [72]. Achieving high model performance often requires including a large number of clients, running more communication rounds, or aggregating richer updates from diverse data distributions. However, these improvements come at the cost of increased communication overhead, computation time, and energy consumption. In contrast, limiting the number of participating clients or reducing the frequency and size of updates can speed up training and lower resource usage, but may compromise model accuracy or lead to biased learning if the selected clients are not representative.

FL systems must also balance privacy and security requirements with both performance and efficiency. Techniques such as differential privacy, secure aggregation, and fine-grained access control can enhance data protection and system integrity but introduce overhead in terms of computation and communication. For example, secure aggregation protocols often require multiple cryptographic operations and additional message exchanges, which can slow down training and increase latency. Similarly, adding noise to updates for privacy may degrade model performance if not carefully tuned. These privacy-preserving mechanisms must be carefully integrated to avoid undermining either model quality or training efficiency.

In practical FL deployments, striking the right balance across these dimensions is crucial. Overemphasizing any single aspect—such as maximizing performance without considering privacy or enforcing strict security at the cost of scalability—can limit the real-world applicability of the system. Designing protocols that adaptively manage these trade-offs based on application needs, resource availability, and data sensitivity remains an open and critical area of research in FL.

2.4.7 Architectural Challenges

While significant progress has been made in addressing the inherent challenges of FL, the current FL architectures still face significant challenges limiting their scalability, interoperability, and overall effectiveness. As the number of participating clients increases, managing communication overhead, ensuring efficient synchronization, and maintaining model consistency becomes increasingly complex [37]. However, traditional centralized architectures may not scale efficiently, leading to bottlenecks and degraded performance in large-scale deployments. Additionally, current architectures often lack the adaptability required to handle clients with varying levels of participation and fluctuating resource availability [73]. This rigidity can result in poor model generalization and inefficient resource utilization, as the system may not effectively leverage the full spectrum of client capabilities and data diversity [74].

To overcome these architectural challenges, there is a growing recognition of the need for an **open and interoperable ecosystem** in FL that fosters collaboration and standardization. An open ecosystem can facilitate scalability by promoting interoperable protocols and frameworks that allow seamless integration of diverse clients [39]. By encouraging open standards, FL systems can better accommodate heterogeneous devices and support large-scale deployments. Such an ecosystem would also encourage the participation of a broader range of clients, increasing the diversity and volume of data available for training. This inclusivity can help mitigate data scarcity and imbalance issues, improving model robustness and generalization. By lowering barriers to entry and providing incentives for participation, an open ecosystem can harness the collective data and computational resources of a vast network of clients. Furthermore, such an ecosystem could also enhance flexibility by enabling the development of modular and extensible architectures where developers and researchers contribute and share components that address specific challenges. This collaborative approach accelerates innovation and allows FL architectures to evolve in response to emerging needs and technologies.

2.4.8 Incentivization

Encouraging active participation from clients in FL is a significant challenge, particularly when their resources are limited and the benefits are not immediately apparent. Clients may be reluctant to participate in the training process due to concerns over resource consumption, privacy risks, and a lack of direct incentives. Local training requires computational power, memory, and energy, which can be burdensome for devices with constrained resources or battery life. Despite FL's aim to preserve data privacy by keeping data localized, clients may still fear potential privacy breaches through inference attacks or unintended information leakage. Moreover, the absence of clear and tangible benefits can deter clients from contributing their data and computational resources to FL. The voluntary nature of client participation means that clients must perceive sufficient value or incentive to allocate their limited resources towards contributing to the global model. Without appropriate incentives, FL systems risk experiencing low participation rates, which can lead to insufficient data diversity, reduced model accuracy, and biased learning outcomes [75].

To address these issues, developing robust incentive mechanisms is crucial. Implementing fair and transparent reward systems ensures that clients are adequately compensated for their contributions, thereby encouraging active and sustained participation [76]. Additionally, incorporating privacy-preserving techniques and trust-building measures can help alleviate privacy concerns, making clients more willing to engage in FL and contribute high-quality data and resources.

Chapter 3

Literature Review & Empirical Analysis

3.1 Introduction

FL faces significant challenges, including data and system heterogeneity, communication inefficiency, and security vulnerabilities. These challenges have also been acknowledged and thoroughly discussed in literature [35, 37, 38, 77, 78, 79, 80]. Over the years, numerous optimizations have been proposed to address these limitations, ranging from novel aggregation mechanisms to advanced client selection strategies, robust security frameworks, and incentive-driven participation models. These advancements aim to improve the performance, efficiency, scalability, and reliability of FL systems, especially under realistic heterogeneous conditions.

This chapter provides a critical examination of the state-of-the-art techniques developed to tackle the key challenges in the field. It reviews existing approaches for managing heterogeneity, optimizing client selection, improving aggregation methods, and tackling security and incentivization issues. Additionally, it includes experimental analysis to empirically assess the effectiveness of these techniques. The progress achieved and the remaining gaps are identified, offering insights that serve as the foundation for the contributions detailed in the subsequent chapters.

3.2 Addressing Heterogeneity in Federated Learning

FedAvg [8] has served as the foundational algorithm for FL, offering an efficient way to aggregate local model updates. However, it does not necessarily account for heterogeneity in its design. As a result, FedAvg often struggles under heterogeneous conditions, leading to slower convergence, biased global models, and inefficient resource utilization [38]. To mitigate these issues, extensive research has proposed various strategies tailored to address both types of heterogeneity in FL, data heterogeneity, and system heterogeneity. The following sections explore these challenges in detail, reviewing state-of-the-art methods designed to enhance FL performance under such conditions.

3.2.1 Data Heterogeneity

Data heterogeneity, often characterized by non-IID data across clients, poses significant challenges to FL since it can lead to model divergence, reduced generalization capabilities, and slower convergence rates. Several studies, such as those by [57, 81, 82], have investigated the impact of data heterogeneity on FL performance and demonstrate that non-IID data could cause significant degradation in model accuracy and slow down convergence. The impact of data heterogeneity becomes more pronounced in scenarios with a small client population or when clients possess limited data samples. In such cases, the divergence in local updates can significantly skew the global model, resulting in poor generalization.

To mitigate the effects of data heterogeneity, various aggregation algorithms and optimization techniques have been proposed. One common approach is to modify the FedAvg algorithm to account for statistical heterogeneity. Methods like FedProx [83] introduce a proximal term in the local objective to restrict the divergence between local and global models, thereby improving stability and convergence in heterogeneous environments. The authors of [84] employ control variates to mitigate client drift resulting from non-IID data, ensuring more accurate and consistent updates during the aggregation process. While a few other approaches propose to tackle heterogeneity using sample weights [85], regularization techniques [86] and virtual homogeneity learning [87].

Another effective strategy involves personalized Federated Learning, where models are tailored to individual clients while still benefiting from shared global knowledge. Meta-learning-based techniques, such as those proposed in [88, 89, 90], enable the global model to adapt quickly to each client's specific data distribution by learning a good initialization that can be fine-tuned locally. Multi-task learning frameworks also allow for the simultaneous training of a global model and multiple personalized models, thereby accommodating the unique characteristics of each client's data [91].

Clustering-based methods have also been employed to group clients with similar data distributions, allowing for more specialized aggregation within each cluster [92, 93]. For instance, FedCluster dynamically identifies and clusters clients based on their data characteristics, facilitating the training of cluster-specific models that better capture the underlying data patterns [94]. These approaches not only aim to enhance model accuracy but also the scalability of FL systems by reducing the complexity of managing highly heterogeneous data. Besides clustering-based approaches, a few other studies have also explored Transfer Learning and domain adaptation techniques to address data heterogeneity [95]. Furthermore, advanced optimization algorithms also play a significant role in addressing data heterogeneity. Adaptive Federated Optimization techniques, such as FedAdam and FedYogi, incorporate adaptive learning rates and momentum terms to better handle the variability in client updates [96]. These optimizers enhance the convergence properties of FL by dynamically adjusting the optimization process based on the observed update patterns.

However, despite these advancements, our field experiments over the MNIST dataset [97] indicate that many proposed solutions may not fully address the challenges posed by non-IID data [98]. We evaluate the performance of algorithms like FedAvg, FedProx, FedPD, and SCAFFOLD under varying degrees of data heterogeneity and find that while all algorithms performed well under IID conditions, they experienced significant accuracy drops when faced with non-IID data distributions. Specifically, our experiments show that FedAvg's accuracy decreased by 7.09% under non-IID settings, reflecting its lack of mechanisms to handle heterogeneity effectively. More notably, algorithms designed to mitigate heterogeneity, such as FedPD [99] and SCAFFOLD [84], exhibit even larger accuracy drops of 30.24% and 19.22%, respectively, while QFedAvg behaves similarly to FedAvg, showing no improvement despite its weighted aggregation mechanism. This is likely due to the random selection of clients in each round, which prevents QFedAvg from consistently prioritizing high-loss clients, effectively neutralizing its intended advantage. Furthermore, some algorithms demonstrate sensitivity to local hyperparameters like batch size and aggregation weighting. For instance, SCAFFOLD may require smaller batch sizes to perform adequately, but this can lead to increased training times and may not be feasible in resource-constrained environments, while QFedAvg's effectiveness depends on the choice of the scaling factor q, which controls how much higher-loss clients are prioritized in the aggregation. These unexpected results suggest that existing algorithms may not be as robust to statistical heterogeneity as claimed. Specifically, there is a gap in developing algorithms that can dynamically adapt to varying degrees of data heterogeneity without relying heavily on hyperparameter tuning, which can be impractical in real-world scenarios. Moreover, the lack of mechanisms to consistently prioritize clients with significant contributions to model improvement highlights a need for more sophisticated client selection strategies. We summarize the results of our experiments in Figure 3.1.

These observations highlight the need for more effective strategies to handle data heterogeneity in FL. Potential approaches could include developing aggregation methods that are robust to distributional differences and designing client selection mechanisms that consider the statistical properties of client data. Moreover, incorporating personalized models that account for individual client distributions, while contributing to a shared global model, may offer a promising direction [100].

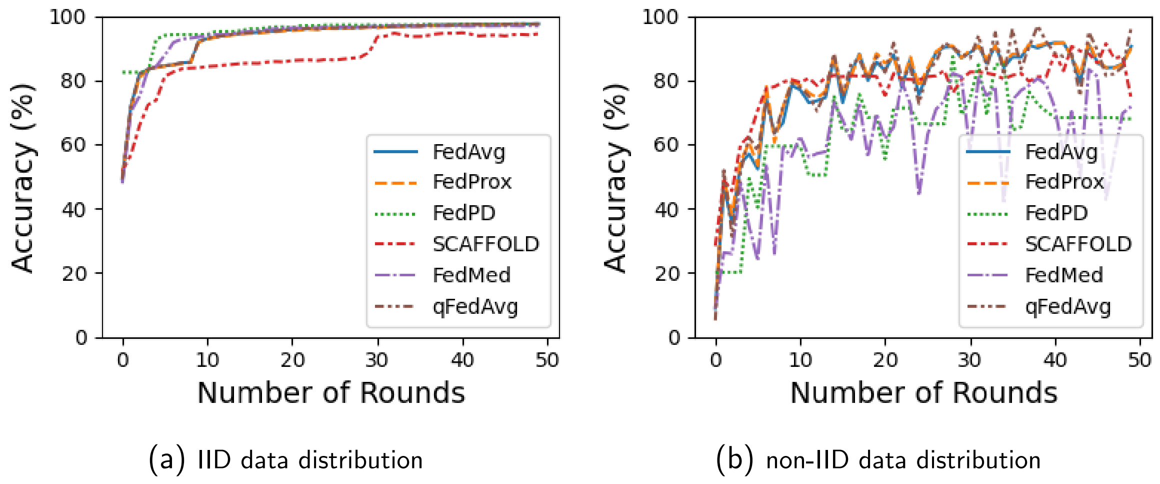


Figure 3.1: Average accuracy of FL algorithms for MNIST data set

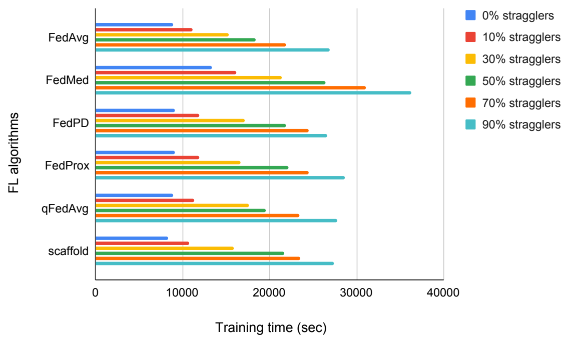
3.2.2 System Heterogeneity

In addition to data heterogeneity, FL systems face significant challenges due to system heterogeneity, which refers to the variations in computational capabilities, memory, energy resources, and network connectivity among participating clients. These diversities introduce significant delays in the FL process and can lead to biased updates if certain devices consistently contribute less due to resource limitations [35, 101, 102].

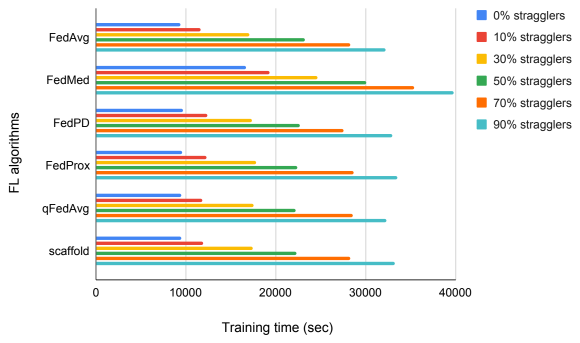
Similar to data heterogeneity, a few studies, such as those by [103, 104, 105] have made attempts to characterise the impact of device and system heterogeneity and report significant performance degradation and longer convergence times. One of the primary challenges reported by these studies is the presence of stragglers — clients that, due to limited computational power or poor network connectivity, take significantly longer to complete local training tasks compared to others. This delay occurs because FL servers typically wait to receive updates from all participating clients in a given round before proceeding to aggregate the results and update the global model. This synchronous aggregation approach means that faster clients, which complete their local training quickly, remain idle while waiting for stragglers to finish. Consequently, the overall training process slows down, leading to increased latency and delayed convergence of the global model

[106].

Inspired by these studies, we evaluate the performance of various FL algorithms under varying levels of device heterogeneity. Specifically, we conducted extensive experiments to assess how stragglers affect both training time and model accuracy. Each client was allotted a fixed number of local epochs to complete the training process. Clients who failed to do so within the specified time frame were considered stragglers. Furthermore, we set a threshold for the number of stragglers to simulate varying levels of system heterogeneity and conducted the experiments over both IID and non-IID data distributions. Our results, as shown in Figure 3.2, demonstrate that system heterogeneity prolongs the training time, regardless of whether the data distribution is IID or non-IID.



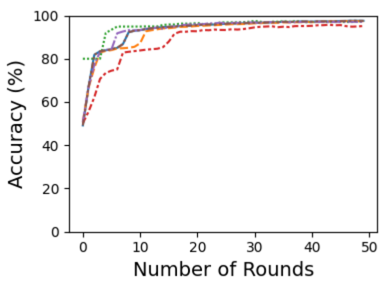
(a) IID data distribution



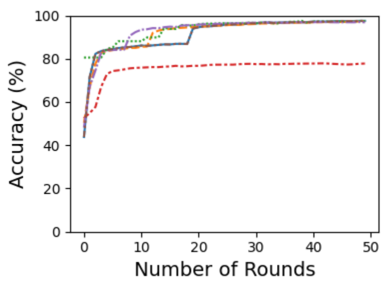
(b) non-IID data distribution

Figure 3.2: Training time of FL algorithms over MNIST data for varying levels of stragglers

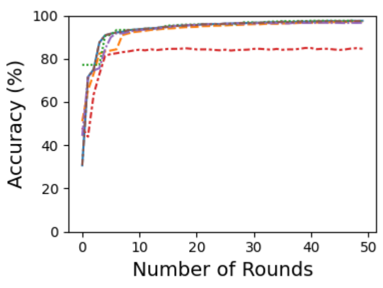
The results also show that the impact of heterogeneity varies depending on the scenario and sensitivity of the algorithm. A small threshold of stragglers is less likely to affect the model performance if the network has an IID data distribution. However, this impact becomes more significant in scenarios where both statistical and system heterogeneity come into the picture. For instance, we did not observe any accuracy drop for a threshold of 10% stragglers over an IID data distribution, as can be seen in Figure 3.3. On the other hand, Figure 3.4 shows the same 10% threshold reports an accuracy drop of 16.04% over non-IID data for FedMed.



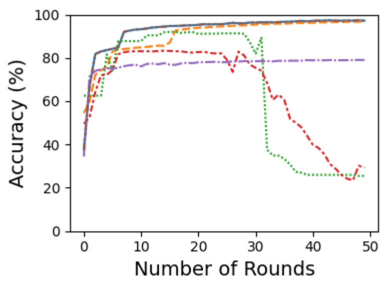
(a) 10% stragglers



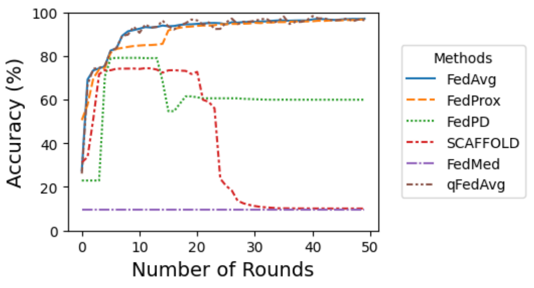
(b) 30% stragglers



(c) 50% stragglers



(d) 70% stragglers

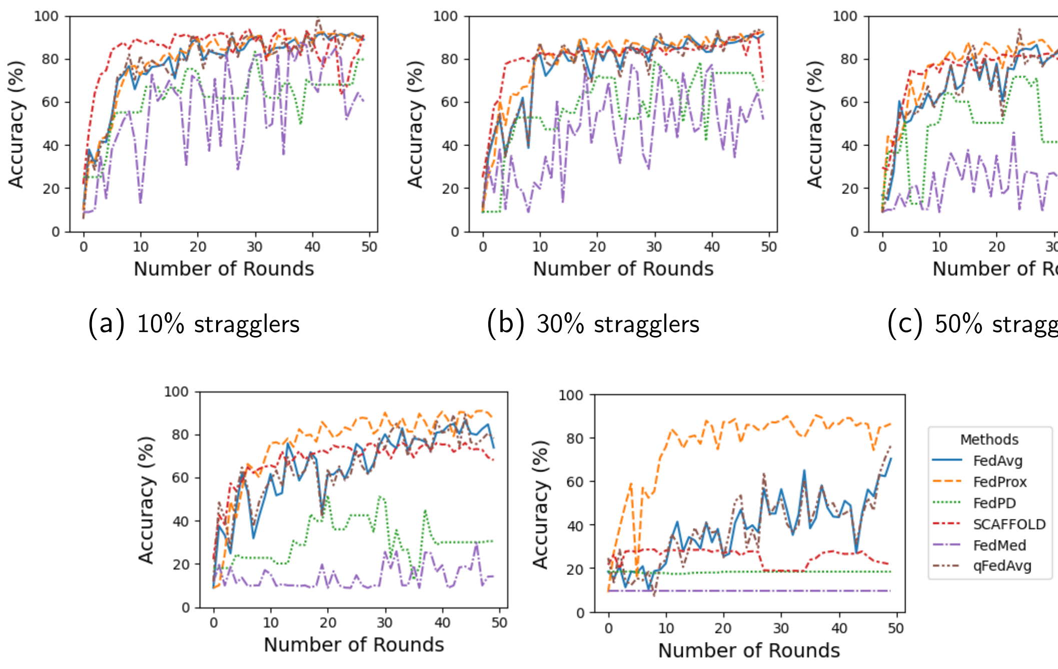


(e) 90% stragglers

Figure 3.3: Average accuracy comparison of different aggregation protocols over IID data for the MNIST dataset under varying levels of straggler devices

We extended our experiments to include the highly heterogeneous environment where data is distributed in a non-IID manner and up to 90% of the participating devices are stragglers. Under these settings, FedAvg reported an accuracy drop of 22.54% and a training time up to 3.42× longer. This behavior can be expected because FedAvg does not account for system or statistical heterogeneity in its design. We also observed an accuracy drop of 20.03% and 4.84% in the performance of qFedAvg and FedProx, respectively, while FedMed, FedPD, and SCAFFOLD led to model divergence.

These findings indicate that while some FL algorithms show resilience to moderate levels of heterogeneity, many are still highly sensitive to system constraints. Algorithms that perform well under IID data often struggle in real-world non-IID scenarios where system heterogeneity is prevalent. For example, FedProx showed a smaller accuracy drop than FedAvg due to its design, which takes heterogeneity into account. However, under extreme heterogeneity, even these solutions struggle to maintain model performance.



(a) 10% stragglers (b) 30% stragglers (c) 50% stragglers

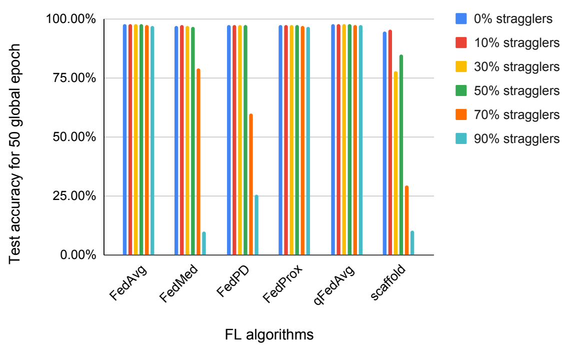
(d) 70% stragglers (e) 90% stragglers

Figure 3.4: Average accuracy comparison of different aggregation protocols over non-IID data for the MNIST dataset under varying levels of straggler devices

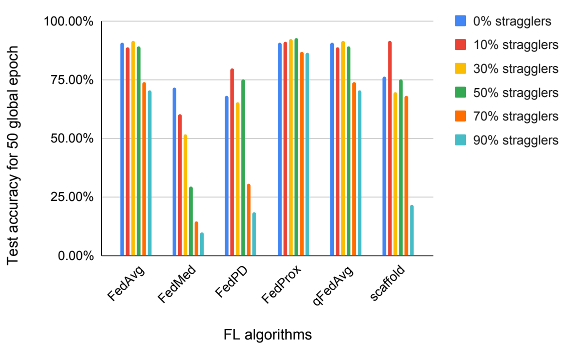
Figure 3.5 summarises these results over varying levels of stragglers. Here, the devices are chosen at random for each training iteration, which skews the distribution more and results in model divergence.

Our experiments illustrate the significant impact of both system and statistical heterogeneity on FL performance and support the findings reported by earlier studies. Since then, researchers have proposed a number of techniques to improve the training efficiency of FL under heterogeneous conditions [107, 108, 109, 110, 111], most notable of which is asynchronous federated learning (AFL). Unlike synchronous methods, AFL allows clients to send their model updates independently without waiting for all clients to finish their local computations [112, 113, 114]. This reduces waiting times and improves overall training efficiency, though it introduces complexities in ensuring the consistency and stability of the global model [115].

Client selection strategies have also been developed to address system heterogeneity.



(a) IID Data Distribution



(b) non-IID Data Distribution

Figure 3.5: Average accuracy of FL algorithms over the MNIST dataset for varying levels of stragglers

The idea is to select a subset of clients that can reliably participate in each training round. Techniques such as resource-aware client selection prioritize clients based on their current computational load, battery levels, and network conditions [56, 116]. By selecting clients that are more likely to complete their training tasks efficiently, these strategies help maintain a steady flow of updates and prevent the training process from being stalled by less capable or unreliable devices.

Beyond individual client selection, a few researchers have also explored structural approaches such as Hierarchical federated learning (HFL) to mitigate heterogeneity. HFL is organizing clients into clusters and leveraging intermediate aggregation points like edge servers to manage subsets of clients and offload some of the aggregation workload from the central server. This hierarchical restructuring reduces communication bottlenecks [117, 118] and enables more localized coordination [119]. Moreover, this approach allows clients with similar characteristics to collaborate more effectively, improving the efficiency of local updates before they are aggregated at the global level [120].

Adaptive aggregation algorithms complement these structural strategies by dynamically adjusting the aggregation process based on the varying capabilities and contributions of clients. Techniques like FedProx [83] and FedDyn [121] ensure that updates from clients are appropriately weighted, accounting for differences in local training duration or steps.

Furthermore, resource allocation algorithms ensure that clients with limited resources are not overburdened, thereby preventing premature dropout and ensuring active participation throughout the training process [122, 123, 124]. These algorithms dynamically allocate computational tasks and communication resources based on the real-time availability and capacity of each client, promoting a balanced and efficient FL ecosystem. A few techniques also allow adaptive partial training, enabling more clients to participate in the training process [125]. However, despite significant progress, existing techniques often fall short in fully mitigating the complexities introduced by system heterogeneity. Challenges such as ensuring consistent model performance, maintaining fairness across clients, and optimizing resource utilization under dynamic conditions persist, highlighting the need for ongoing research and more sophisticated solutions.

3.3 Client Selection in Federated Learning

FL relies on decentralized training across edge devices, making the performance and efficiency of the system closely tied to the characteristics of the participating clients. Due to resource constraints and scalability requirements, only a subset of clients is typically selected to participate in each training round. This selective participation is not only practical but also critical for managing computational and communication costs, improving convergence efficiency, and maintaining the quality of the global model. However, client selection strategies were largely rudimentary in the early stages of FL development, with random selection being the foundational approach [8]. The underlying principle of random selection is to randomly select clients to participate in the training process to ensure unbiased participation across all clients. While it guarantees an equitable opportunity for diverse clients to contribute to the learning model, it exhibits limitations in effectively managing the heterogeneity of client devices in practical FL applications and can lead to convergence issues, as discussed in Section 3.2. Despite its simplicity, this approach provided insights into the dynamics of client participation and highlighted the need for

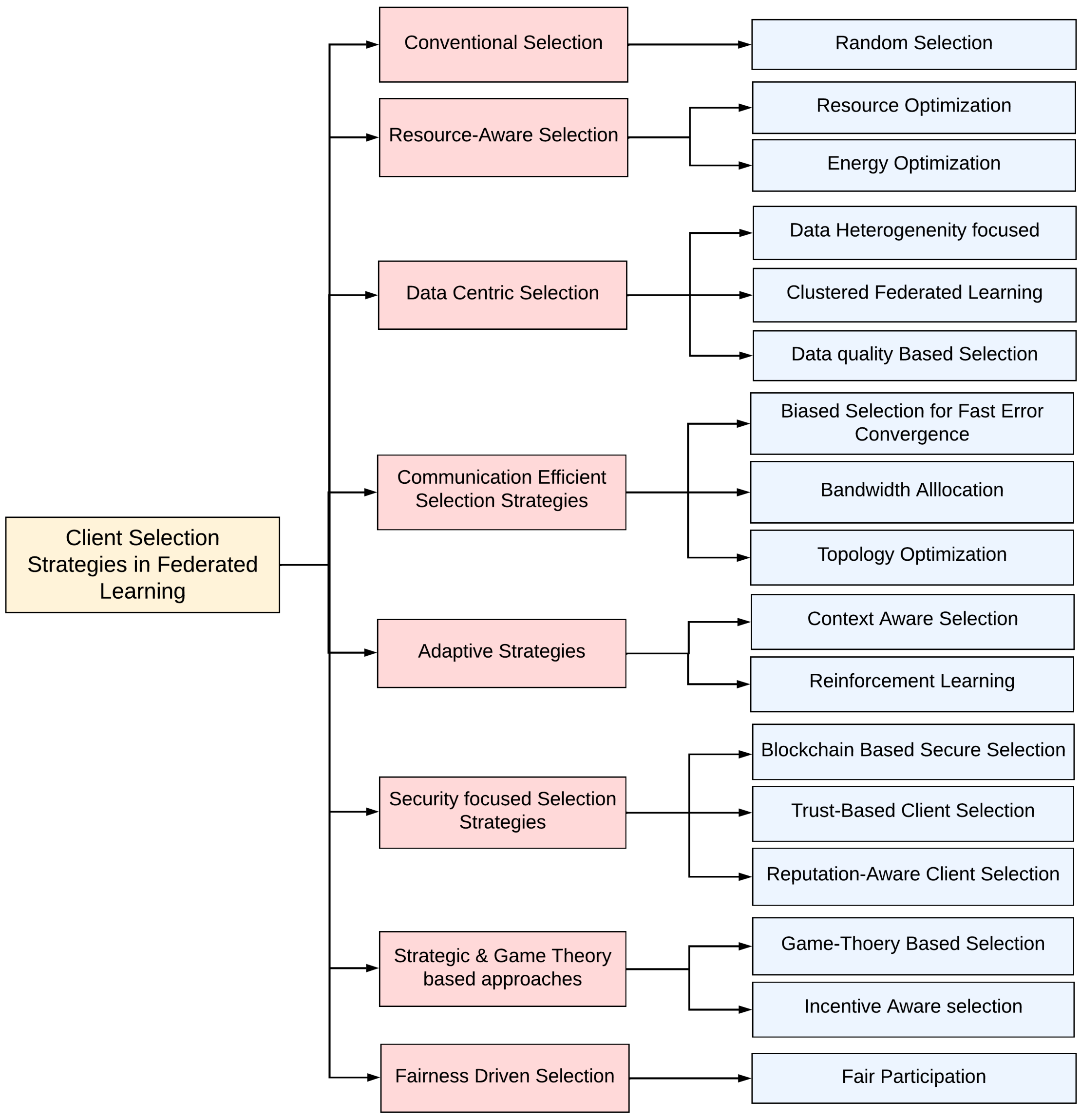

Client Selection Strategies in Federated Learning
Conventional Selection
Random Selection
Resource-Aware Selection
Resource Optimization
Energy Optimization
Data Centric Selection
Data Heterogenenity focused
Clustered Federated Learning
Data quality Based Selection
Communication Efficient Selection Strategies
Biased Selection for Fast Error Convergence
Bandwidth Alllocation
Topology Optimization
Adaptive Strategies
Context Aware Selection
Reinforcement Learning
Security focused Selection Strategies
Blockchain Based Secure Selection
Trust-Based Client Selection
Reputation-Aware Client Selection
Strategic & Game Theory based approaches
Game-Thoery Based Selection
Incentive Aware selection
Fairness Driven Selection
Fair Participation

Figure 3.6: Taxonomy of Client Selection Strategies in Federated Learning

strategies that consider client diversity and operational factors. Since then, several studies have made significant advancements to optimize FL performance in various aspects [126, 127] such as minimizing computational and communication costs and dealing with heterogeneity issues [128, 129]. These strategies can be broadly categorized into several dimensions, as illustrated in Figure 3.6

The following subsections provide a comprehensive overview of these strategies, highlighting their key characteristics, advantages, and limitations.

Data-Centric and Heterogeneity Management Strategies

Client selection strategies in this category focus on optimizing client selection based on the data characteristics and distribution. They specifically target challenges arising from non-IID data, data skewness, and variations in data quality among clients, thereby aiming to improve both the efficiency and performance of the learning model. For instance, Zhang et al. [130] proposed to select the clients with the least degree of non-IID data. Adaptive algorithms, such as those introduced by Chen et al. [131], propose to select clients based on their local statistical heterogeneity and previous training performance, while others consider both the system and data heterogeneity to dynamically adjust the number of selected clients [132]. On the other hand, the authors in [133] proposed a Determinantal Point Process (DPP) based method, which diversifies the participants' datasets in each round of training to accelerate FL training. More sophisticated approaches employ Deep Reinforcement Learning (RL) to optimize learning over Non-IID data. For instance, the authors in [134] model the client selection as a Markov decision process. They integrate a top-p sampling strategy into the Double Deep Q-Network (double-DQN) algorithm and introduce a novel client sampling algorithm.

Clustered FL was proposed as a strategy to efficiently mitigate the adversities of non-IID data. Fraboni [135] demonstrated that clustered sampling-based client selection leads to better client representativity and reduces variance in aggregation weights. Some strategies in this domain, such as those proposed by Albaseer et al. [136] and Arisdakessian et al. [137], leverage the statistical as well as device heterogeneity for clustering-based client selection, while others, further improved clustered FL with active client selection, using active learning to speed up the process and enhance model accuracy [138].

Concurrently, the focus on data quality has become increasingly prominent, with strategies aiming to select clients with high-quality, rich, and diverse datasets to improve model performance. Techniques like the irrelevance sampling method and Federated Data Quality Profiling (FDQP), as proposed by [139] and [140] respectively, consider both the quality and quantity of data for client selection. Authors in [141] propose an RL-based framework that considers multiple factors, including data size, data quality, and learning budget, to automatically learn client selection policies based on the observed client status. Some other studies [142, 143, 144] in this area have proposed techniques that consider the client contribution to adjust the probability of client selection. However, client selection in FL is not solely dependent on the client's data. Other factors, such as the client's hardware configurations, communication resources, and other environmental factors, also play a significant role.

Resource Aware Selection Strategies

Resource-aware selection strategies in FL focus on optimizing client participation based on their available resources, ensuring the efficient operation of the FL system under varying resource and energy constraints. For instance, the authors in [56] suggest managing the clients based on their available resources and allowing as many client updates as possible. Yu et al. [145] proposed a similar approach to maximizing client updates but emphasized minimizing energy consumption through fixed resource allocation. Similarly, several other studies suggest client selection strategies based on the learning quality of the clients, within a limited resource budget [146, 147]. More sophisticated approaches in this regard propose a deep RL-based approach with a focus on minimizing energy consumption [148] and enhancing communication and computational efficiency [149, 150]. Several studies, including those by He et al. [151] and Luo et al. [152], focus on reducing both time and computational costs.

More recent works in this regard emphasize addressing computational challenges, adaptability, and efficiency. Chen et al. [153] suggest dynamically adjusting client selection based on clients' trained models and their available computational resources while Niu et al. [154] propose a framework that expedites the learning process by selecting clients with a significant impact on the global model, enabling faster convergence and incorporating an early stopping mechanism to conserve resources. A few other studies, such as [116, 155], further emphasize resource conservation while maintaining model accuracy.

[155] in particular, tackles the challenges posed by waiting times and straggler devices by dynamically adjusting training based on clients' resources.

Communication Efficient and Network-Aware Selection Strategies

To improve the efficiency and practicality of the FL learning process, many studies have focused on client selection to minimize the communication costs while maintaining, or even enhancing, the learning efficacy. Studies such as those by Cho et al. [156] suggest that biased client selection towards those with higher local loss results in faster error convergence. They propose a bandit-based communication-efficient client selection strategy to achieve faster convergence with lower communication overhead. Ribero et al. [157] and Marnissi et al. [158] support the same idea and propose to select a subset of clients with significant weight updates and with the highest norms of gradient values, respectively. Recent works in this regard also suggest using Shapley Value to identify and select clients with the highest marginal contribution to model accuracy to complete model training within the fixed budget of communication rounds.

A few other studies in this domain propose a joint client selection and bandwidth allocation strategy to reduce the convergence time and model transmission latency [159, 160]. On the other hand, the authors in [161] dynamically adjust client selection and gradient compression in response to client capabilities to reduce communication loads and address the straggler effect. Another approach to accelerating FL is through topology optimization. For instance, the Topology-Optimized Federated Edge Learning (TOFEL) scheme has been proposed to tackle the issue of excessive learning time due to the existence of straggler devices in Federated Edge Learning [162]. Several other studies also suggest topology optimization to accelerate FL and highlight that optimizing FL models, frameworks, and algorithms with a focus on their network topologies can lead to improved performance [163].

Security-Focused Selection Strategies

In FL, the participating clients impact the robustness, security, and overall performance of the learning process. Various approaches, including blockchain [164], trust-based [165, 166], and reputation-based client selection strategies [167] have been proposed to address these concerns.

The integration of blockchain technology into client selection strategies offers increased security through its immutable and auditable characteristics. Nguyen et al. [164] leverage this idea to enforce secure selection over random clients, while Qammar et al. [168] propose forward bidding to select optimized clients. These strategies help mitigate the biased and malicious influence of participants.

Many studies use trust and reputation as criteria for selecting clients. For instance, Rjoub et al. [165] and Wehbi et al. [166] both propose trust-based client selection strategies. Rjoub's approach uses trust-based deep RL to optimize client selection, while Wehbi's approach is based on mutual trust and matching game theory to select trustworthy clients. Concurrently, Wang et al. [167] and Tan et al. [169] propose to evaluate and select clients based on their reputation profiles.

Fairness-Driven Selection Strategies

Client Selection strategies in this category emphasize the importance of fair participation, equitable resource distribution, and bias mitigation. Strategies such as the one proposed by Huang et al. [170] propose a client selection method with a fairness guarantee. Shi et al. [171] proposed a similar approach where they dynamically adjust FL clients' selection probabilities by jointly considering their reputations, times of participation in FL tasks, and contributions to the resulting model performance. The authors in [172], however, proposed a different approach where they use a biased client selection strategy but dynamically adjust the weight assigned to each client to improve the resulting fairness. However, focusing on fairness may increase the final global accuracy but might result in lower training efficiency due to the performance-fairness trade-off.

Adaptive Strategies

FL environments are inherently dynamic and necessitate adaptive client selection strategies that can effectively respond and adjust to the changes in the environment. This is particularly relevant in scenarios where the data distribution, model structure, and participating clients of the overall framework are not fixed but change over time [173]. A key strategy in this context is proposed by Qu et al. [174] that uses the Contextual Combinatorial Multi-Armed Bandit framework to optimize client selection by analyzing the computing and transmission context of client-edge Server pairs, significantly improving training efficiency. Moreover, RL-based approaches have also been implemented to address the challenges of dynamic environments [148, 175]. For instance, a multi-agent hybrid deep RL-based algorithm, as discussed in [176], optimizes client selection and payment actions in scenarios with dynamic client participation, resource constraints, and limited budgets. Similarly, Zhang et al. [177] also discuss an efficient client selection approach in FL using multi-agent RL. However, the sophistication of these methods often demands higher computational resources and advanced algorithmic tuning to ensure effectiveness.

Strategic and Game Theory-based Approaches

Client selection strategies in this category employ economic and algorithmic principles to enhance system efficiency and incentivize participants. For instance, Nagalapatti and Narayanam address the problem of selecting relevant clients by introducing a Shapley value-based method. Wehbi et al. [178] use game theory and matching algorithms to design preference functions for client IoT devices and federated servers, allowing them to rank each other. This approach focuses on maximizing both client revenues and model accuracy and takes into account the preferences of both parties in their selection, leading to more effective client selection. Yin et al. [179] propose a Stackelberg game-based client selection framework that considers system heterogeneity, client heterogeneity, and client fairness to reduce latency. Incentive mechanisms, on the other hand, are designed to motivate client participation through various forms of rewards [180], which can be monetary or otherwise. These mechanisms range from auction frameworks, where clients bid for participation [181], to collaborative models that offer benefits like data or computing resource exchanges [182]. The goal is to align client participation with the objectives of the FL system, ensuring robust data contribution and efficient model training.

While these client selection strategies build a strong foundation to optimize the various aspects of FL, many of these approaches tend to simplify the problem, focusing solely on optimizing a single objective, such as computational efficiency. This singular focus can lead to an imbalance, where gains in one area might result in shortcomings in another. For instance, strategies that excel in reducing computational load might not adequately address the challenges posed by non-IID data distribution, potentially affecting the overall model performance. Moreover, if the algorithm continually selects the fastest devices, it might result in unfair selection, omitting certain data portions and reducing data diversity, which could adversely affect model training performance. This observation suggests that a single-dimensional approach might be insufficient for the complex ecosystem of FL. Instead, there is a clear need for multi-criterion client selection. Such an approach would not just focus on one aspect of client selection but would simultaneously consider a range of factors, including data quality, energy consumption, and computational constraints. By doing so, it can provide a more holistic solution that aligns with the diverse requirements and challenges of FL systems.

FL systems are complex, involving diverse data sources, varying computational and communication capabilities, and a range of application needs, necessitating a more comprehensive approach. Moreover, the effectiveness of these strategies is highly context-dependent. What works well in one setting, like networks of mobile devices, might not be as effective in another, such as in industrial applications involving the Internet of Things (IoT). This variability signifies the importance of flexible and adaptable algorithms that are not only efficient but can also be tailored to meet the unique demands of various FL environments.

3.4 Bayesian Federated Learning

The traditional FL framework has made significant progress in enabling decentralized model training over potentially a large number of clients. Within this framework, aggregation mechanisms, such as FedAvg, serve as the backbone of the collaborative training process, orchestrating how individual client updates are combined to form a global model. Acknowledging the limitations of FedAvg, subsequent aggregation strategies have been developed to address specific challenges inherent in FL, such as heterogeneity [96, 121, 183], communication efficiency [184], robustness to adversarial attacks [185, 186], and ensuring model convergence and accuracy [187]. While these strategies lay a robust groundwork and have been effective in improving model performance and efficiency to a certain extent, they remain limited by their deterministic nature, which restricts their ability to model and adapt to the complex, probabilistic relationships inherent in diverse datasets. These limitations have pushed researchers to seek more sophisticated methodologies to enhance the FL framework that can better handle uncertainty and variability.

Bayesian Federated Learning (BFL) emerges as a comprehensive extension of the traditional FL framework, incorporating Bayesian principles to provide a probabilistic approach to model aggregation and parameter estimation. Unlike deterministic aggregation mechanisms, BFL treats model parameters as random variables with associated probability distributions [188]. This probabilistic treatment allows BFL to quantify uncertainty in model predictions and parameter estimates, enabling a more adaptive and robust learning process. By maintaining and updating posterior distributions of model parameters, BFL can effectively capture the variability across different clients' data, thereby improving the robustness and generalizability of federated models [189]. Additionally, the Bayesian framework facilitates the incorporation of prior knowledge, which can be particularly beneficial in scenarios with limited or highly heterogeneous data.

Several key studies have advanced the integration of Bayesian learning principles within the FL framework, demonstrating the potential of BFL to overcome the limitations of traditional methods. Early studies, such as those by Yurochkin et al. [190], introduced Bayesian nonparametric methods that effectively synthesize global models from local neural network weights with minimal communication, showcasing significant utility in image classification tasks. While the authors in [191] provide a comprehensive survey of BFL methodologies, highlighting their advantages in addressing heterogeneity and data constraints compared to classical FL approaches.

Recent advancements in BFL have focused on addressing challenges related to personalization, communication efficiency, and model robustness. For example, Zhang et al. [192] developed the pFedBayes framework, which employs Bayesian variational inference to optimize personalized models that account for client-specific data distributions. This was further extended by Chen et al., who introduced frameworks that enable shared and individual uncertainty representations, fostering robust collaborations across heterogeneous clients [193]. To address the communication efficiency challenge, Lee et al. (2020) introduced the scalable-BFL (SBFL) algorithm, which reduces communication overhead through optimized aggregation of quantized gradients [194]. Similarly, Barbieri et al. proposed a compressed BFL method that reduces communication overhead without compromising model calibration and accuracy [195].

BFL has also made significant contributions to improving the security and reliability of federated systems. For instance, Kumari et al. [196] suggest that probabilistic updates can be leveraged to identify and exclude malicious contributions, thus providing defense against backdoor attacks. Likewise, the hierarchical Bayesian frameworks have also been shown to ensure secure and private model training while offering strong theoretical guarantees on convergence and generalization [197]. Additionally, some other advancements in the field, such as the online Laplace approximation by Liu et al. [198] and adaptive dropout strategies by Xue et al. [199], have enhanced the operational efficiency and robustness of BFL frameworks in heterogeneous and non-IID data environments.

Despite these advancements, several gaps remain in the current body of research on BFL. Scalability and computational efficiency continue to pose significant challenges, as

Bayesian methods often involve complex probabilistic computations that can be computationally intensive and difficult to scale to large numbers of clients or high-dimensional models. Additionally, the increased communication overhead resulting from the transmission of probabilistic model parameters, such as distributions or variational approximations, remains an area requiring further research efforts to match the efficiency of deterministic methods. Furthermore, there is a lack of standardized benchmarks and evaluation protocols for Bayesian FL methods, making it difficult to systematically compare and assess the effectiveness of different approaches. Lastly, the theoretical foundations of BFL require further development to better understand the convergence properties and generalization capabilities of Bayesian FL algorithms, particularly in highly heterogeneous settings.

3.5 Enhancing Security in Federated Learning Systems

While FL inherently promotes data privacy by keeping raw data localized on client devices, it is still vulnerable to various security threats and privacy attacks that can compromise both the model and the data, as explained in Section 2.4.5. Recent advancements in the literature have thus focused extensively on mitigating these security threats through a combination of robust aggregation techniques, access control protocols, and client authentication mechanisms. For example, robust aggregation techniques like Krum [200, 201] and Trimmed Mean [202, 203, 204] have been proposed to identify and exclude anomalous updates from malicious clients, thereby preserving the integrity of the global model. Additionally, Byzantine-resilient algorithms have been introduced to tolerate a certain proportion of adversarial clients without degrading the overall model performance [205, 206, 207].

Beyond data and model poisoning attacks, the adversaries can also carry out inference attacks where they attempt to extract sensitive information from the model updates

[208, 209, 210]. Additionally, communication between clients and the central server can be intercepted or tampered with, leading to breaches in confidentiality and integrity [37]. Defense against these attacks requires the integration of cryptographic techniques such as homomorphic encryption [211] and secure multi-party computation (SMPC) [212] for the confidentiality of model updates during the aggregation process. Homomorphic encryption allows the server to perform computations on encrypted data without decrypting it, thereby preventing potential information leakage [213, 214, 215]. The authors in [216] have also explored combining homomorphic encryption with digital signatures to enhance model integrity by preventing unauthorized modifications and ensuring traceability. Similarly, SMPC protocols enable multiple parties to jointly compute functions over their inputs while keeping those inputs private, further enhancing the security of the FL process [217, 218]. These cryptographic approaches ensure that sensitive information remains protected even if the server is compromised. However, these approaches come at the cost of computational overhead and significantly suffer from inherent trade-offs, such as the privacy-utility tradeoff.

Another critical aspect of enhancing FL security is to ensure that only legitimate clients participate in the FL process. Authentication clients before they are granted access to the FL system can help prevent unauthorized access and potential attacks [219]. For instance, Li et al. propose a certificateless authentication-based trustworthy federated learning framework to mutually authenticate clients and servers and prevent model poisoning attacks [220]. Yuan et al. propose a pseudonym-based authentication mechanism [221] while several other studies have suggested blockchain-based decentralized authentication for lightweight and anonymous communication [222, 223]. These approaches not only verify the identities of participating clients but also ensure that the transmitted updates remain secure from eavesdropping and tampering. However, most current solutions focus on specific attack vectors, leaving other potential vulnerabilities unaddressed, which can easily be exploited by adaptive adversaries who modify their strategies to bypass existing defenses.

Despite these advancements, the existing security models still struggle to scale effectively, particularly as the number of participating clients increases. Many robust aggregation and authentication techniques become computationally intensive and inefficient in large-scale deployments. Additionally, there is an absence of comprehensive access control mechanisms that can dynamically manage client permissions and roles within the FL framework. Furthermore, adaptive and sophisticated adversaries, who can modify their attack strategies based on the defense mechanisms in place, pose ongoing challenges, necessitating the development of more resilient and adaptive security protocols.

To address these challenges, zero-trust security mechanisms are gaining traction as a means to enhance the robustness and reliability of FL systems. These security models do not assume any implicit trust in any user, device or network entity [224]. Instead, every interaction must be authenticated, authorized, and validated before access is granted [225]. In a zero-trust FL system, every request to access the FL platform experiences thorough authorization checks to mitigate the risks associated with compromised devices and unauthorized access [226, 227]. Advanced access control techniques, such as Blockchain [228, 229] or Attribute-Based Access Control (ABAC) [230, 231], are required to implement zero-trust environments. Notably, ABAC enables fine-grained access control based on client attributes, such as role, location, or device characteristics, making it suitable for the heterogeneous nature of FL clients [232].

3.6 Crowdsourcing in Machine Learning and Distributed Systems

Crowdsourcing refers to the practice of obtaining data, services, or contributions from a large group of participants. This approach leverages the collective intelligence and diversity of participants, making it a scalable and cost-effective solution for tasks that are otherwise time-consuming or resource-intensive [233]. Originating from the notion of outsourcing tasks to a distributed workforce, crowdsourcing enables the distribution of various tasks, such as data annotation, problem-solving, and resource provisioning, across a vast and often heterogeneous pool of contributors [234]. In the context of machine learning and distributed systems, crowdsourcing is particularly valuable for collecting and labeling large-scale datasets. It aids in model evaluation, addresses challenges such as data imbalance, and optimizes system performance, thereby contributing to the development of robust and adaptive technologies.

3.6.1 Crowdsourcing in Machine Learning

Crowdsourcing has become an essential tool for acquiring and annotating comprehensive datasets that enhance the generalizability and performance of ML models. Platforms such as Amazon Mechanical Turk (MTurk) enable researchers and developers to collect large-scale labeled data for various tasks, including image classification, natural language processing, and recommendation systems [235]. For instance, Callison [236] demonstrates the effectiveness of MTurk in generating high-quality labeled data for natural language processing tasks, highlighting the platform's potential in accelerating ML model development. A few other studies, such as those by Kittur et al. [237], Jennifer [238], and Abhigna et al. [239] explored the dynamics of crowd-powered systems, emphasizing how diverse contributor inputs can lead to more robust and unbiased models.

The use of crowdsourcing in ML extends beyond data labeling to address challenges like imbalanced or sparse datasets. For instance, Ratner et al. introduced Snorkel, a system that leverages crowdsourced weak supervision to train models without extensive labeled data, thereby reducing the reliance on manual annotation [240]. Shi et al. proposed a Distribution Aware Self-training approach that enhances annotation distribution, significantly improving the performance of models trained on skewed datasets [241]. Xu et al. [242] demonstrated how integrating blockchain into crowdsourcing workflows can improve transparency and traceability, enhancing the overall reliability of crowdsourced data.

The scalability and flexibility of crowdsourcing have also enabled its adoption in specialized ML domains. Frameworks like WeCrowd [243] leverage social networks to increase user engagement and participation in crowdsourcing tasks. This integration of social and technological factors highlights the potential of crowdsourcing to address complex challenges in ML, particularly those requiring diverse and representative datasets.

3.6.2 Crowdsourcing in Distributed Systems

In distributed systems, crowdsourcing enhances the efficiency and scalability of computational tasks by distributing workloads across multiple participants. Distributed systems inherently involve the coordination of multiple interconnected components. Crowdsourcing provides a mechanism to leverage the collective resources and expertise of distributed participants to address complex system-level challenges. This helps mitigates the limitations of traditional systems, such as bottlenecks and single points of failure, by leveraging the collective processing power of a distributed network. Crowdsourced distributed computing platforms, like BOINC (Berkeley Open Infrastructure for Network Computing), exemplify how distributed resources can be utilized to perform large-scale computations for scientific research, data analysis, and simulation tasks [244]. Crowdware combines GPU-based public resource computing with energy-aware incentive mechanisms to optimize resource allocation while minimizing energy consumption [245]. Hosseini et al. introduced Crowdcloud, a crowdsourced cloud infrastructure system that pools computational resources from the public [246]. Other frameworks such as *SETI@home*[1], *Folding@home*[2] and *Milkyway@home*[3] also demonstrate the power of crowdsourcing in distributed systems. *Folding@home*, for example, stands at the computing power of approximately 16.9 petaFLOPS as of 11th November 2024, leveraging the computational resources and processing power from volunteers worldwide. Such computational capac-

[1]http://setiathome.ssl.berkeley.edu/
[2]https://en.wikipedia.org/wiki/Folding@home
[3]http://milkyway.cs.rpi.edu/milkyway/

ity has been achieved using almost 1,133 AMD GPUs, 7,256 NVidia GPUs, and 22,905 CPUs[4]. Similarly, *SETI@home* has historically utilized the unused computing cycles of volunteer devices to process signals from space, showcasing the ability of crowdsourcing to aggregate vast distributed resources for a shared goal. These platforms highlight the potential of crowdsourced distributed computing to tackle computational challenges at scale.

While crowdsourcing has significantly advanced machine learning and distributed systems, its application to leveraging both data and computational resources presents unique opportunities and challenges. By integrating crowdsourcing principles with distributed systems, there is immense potential to create frameworks that harness the collective computational power and diverse data resources of participants. This integration can enable scalable, privacy-preserving, and cost-effective solutions for collaborative model training, particularly when combined with the decentralized nature of edge devices. Such frameworks promise to enhance real-time data processing, reduce communication overhead, and foster resilient, adaptive systems for distributed learning.

However, realizing these opportunities requires addressing critical challenges, particularly in assessing the quality of contributions and incentivizing participation. Ensuring the reliability of computational and data resources contributed by diverse participants is essential for maintaining system robustness. Additionally, sophisticated mechanisms are needed to fairly reward contributions, balancing cost-efficiency with participant engagement.

3.7 Incentive Mechanisms

Incentive mechanisms have long been studied in the literature to encourage active volunteer participation, ensure quality data, and enhance the robustness of environments reliant on volunteer contributions. These mechanisms are essential for motivating participants to

[4]https://stats.foldingathome.org/os

contribute their resources, whether it be data, computational power, or expertise, thereby ensuring the sustainability and effectiveness of collaborative systems such as crowdsourcing and, more recently, FL.

In crowdsourcing, incentive mechanisms were introduced to attract and retain contributors who perform tasks such as data annotation, transcription, and problem-solving. Early studies, such as the one by Kittur et al. [247], highlighted the importance of monetary rewards in platforms like Amazon Mechanical Turk to compensate workers for their efforts. However, monetary incentives alone were found to be insufficient for ensuring high-quality outputs. Subsequent research introduced non-monetary incentives, such as computation credits, reputation systems, tokenized incentives, and gamification elements, which recognize and reward contributors based on their performance and reliability [248, 249]. These approaches not only motivated participants to maintain high standards but also fostered a sense of community and achievement among workers [250].

As crowdsourcing matured, more sophisticated incentive mechanisms were developed to address specific challenges such as task complexity and worker reliability. A few studies, such as [251, 252], explored performance-based incentives to adjust rewards based on the accuracy and reliability of submitted work, promoting high-quality data annotation and reducing the incidence of errors or malicious contributions. Auction-based mechanisms have also been explored to dynamically price tasks based on demand and supply, ensuring efficient task allocation and fair compensation for contributors [253, 254, 255]. Xu et al. took a different approach and proposed an online reverse auction model to encourage participation while ensuring fair compensation in mobile crowdsourcing systems [256]. While Pang and Liu explored the use of linear variable compensation schemes to weed out low-quality participants and incentivize higher-quality contributions [257].

In FL, incentive mechanisms have been explored to promote the submission of high-quality data, ensuring that the aggregated updates are both reliable and informative [258]. However, FL presents unique challenges for incentive design, requiring that incentive mechanisms must account for the computational costs, data contribution, and potential privacy risks involved. Early adaptations of incentive mechanisms from crowdsourcing and distributed learning emphasize the need for mechanisms that ensure participants are adequately rewarded for their contributions without compromising the integrity of the federated model [38]. Consequently, various approaches have been proposed to design effective incentive mechanisms in FL. Game theory-based frameworks are commonly employed to model the interactions between the central server and the clients. In these frameworks, the server aims to maximize the utility of the global model, while clients seek to maximize their own rewards [259, 260]. For instance, Kang et al. [250] introduced a contract theory-based approach where the server designs contracts that specify the rewards for different levels of client participation and data quality. However, these mechanisms often face limitations such as scalability issues and the complexity of designing fair and optimal contracts in dynamic environments [261].

Another innovative approach involves the use of Shapley values from cooperative game theory to fairly allocate rewards based on each client's marginal contribution to the global model [262, 263]. Shapley values provide a principled method for quantifying the contribution of each client by considering all possible subsets of clients, ensuring that rewards are distributed fairly and proportionately. This method not only incentivizes participation but also encourages clients to provide high-quality and diverse data, as their rewards are directly tied to the actual improvement they bring to the model [264]. However, calculating Shapley values is computationally expensive, especially as the number of clients increases, making it impractical for large-scale FL systems. While the existing works lay a solid foundation for incentivizing clients, challenges remain in designing scalable and efficient incentive mechanisms that can balance fairness, computational feasibility while ensuring sustainable client participation.

Chapter 4

FedOnDemand: A Crowdsourced Federated Learning framework

4.1 Introduction

Motivated by the requirement of an open ecosystem for FL, this chapter introduces the concept of Crowdsourced Federated Learning (CFL) which integrates the decentralized training capabilities of FL with the collaborative and participatory nature of crowdsourcing. This integration establishes an open ecosystem where a diverse and extensive network of devices can voluntarily contribute to the training of machine learning models. While FL enables the development of global models by aggregating updates from individual clients without sharing raw data, crowdsourcing introduces mechanisms for voluntary participation, incentivization, and dynamic engagement from a broad range of contributors. By utilizing the collective resources and varied data sources available in a crowdsourced environment, this integration effectively overcomes major limitations of traditional FL, including scalability challenges and the necessity for continuous client participation.

By facilitating on-demand participation, CFL can accommodate the dynamic availability and varying capabilities of devices, thereby increasing the system's flexibility and robustness. Additionally, fair and transparent incentive mechanisms encourage high-quality contributions and sustained engagement, which are essential for achieving optimal model accuracy and scalability.

In this chapter, we propose FedOnDemand, a CFL framework designed to support dynamic participation, incentivize meaningful contributions, and ensure efficient collaboration in distributed environments. We outline its core components, including device registration, task discovery, model training, and incentive mechanisms, while also addressing practical challenges such as security, data quality, and client heterogeneity. This chapter serves as a foundation for the in-depth exploration of client selection, secure access control, and fairness-aware reward allocation presented in later chapters.

4.2 Objectives

The goal of this chapter is to develop a Crowdsourced Federated Learning framework that supports scalable, secure, and equitable participation across a wide range of devices. By merging crowdsourcing principles with FL, the proposed FedOnDemand framework aims to address the critical challenges of participant heterogeneity, intermittent availability, and the need for effective incentive strategies.

Specifically, the objectives include:

- Supporting voluntary and dynamic participation from diverse edge devices.
- Optimizing collaborative model training in the presence of device and data heterogeneity.
- Establishing robust mechanisms for secure client registration and participation.
- Promoting fairness and engagement through contribution-based reward distribution.

Together, these objectives guide the design of FedOnDemand to improve the practicality and effectiveness of FL in decentralized and real-world environments.

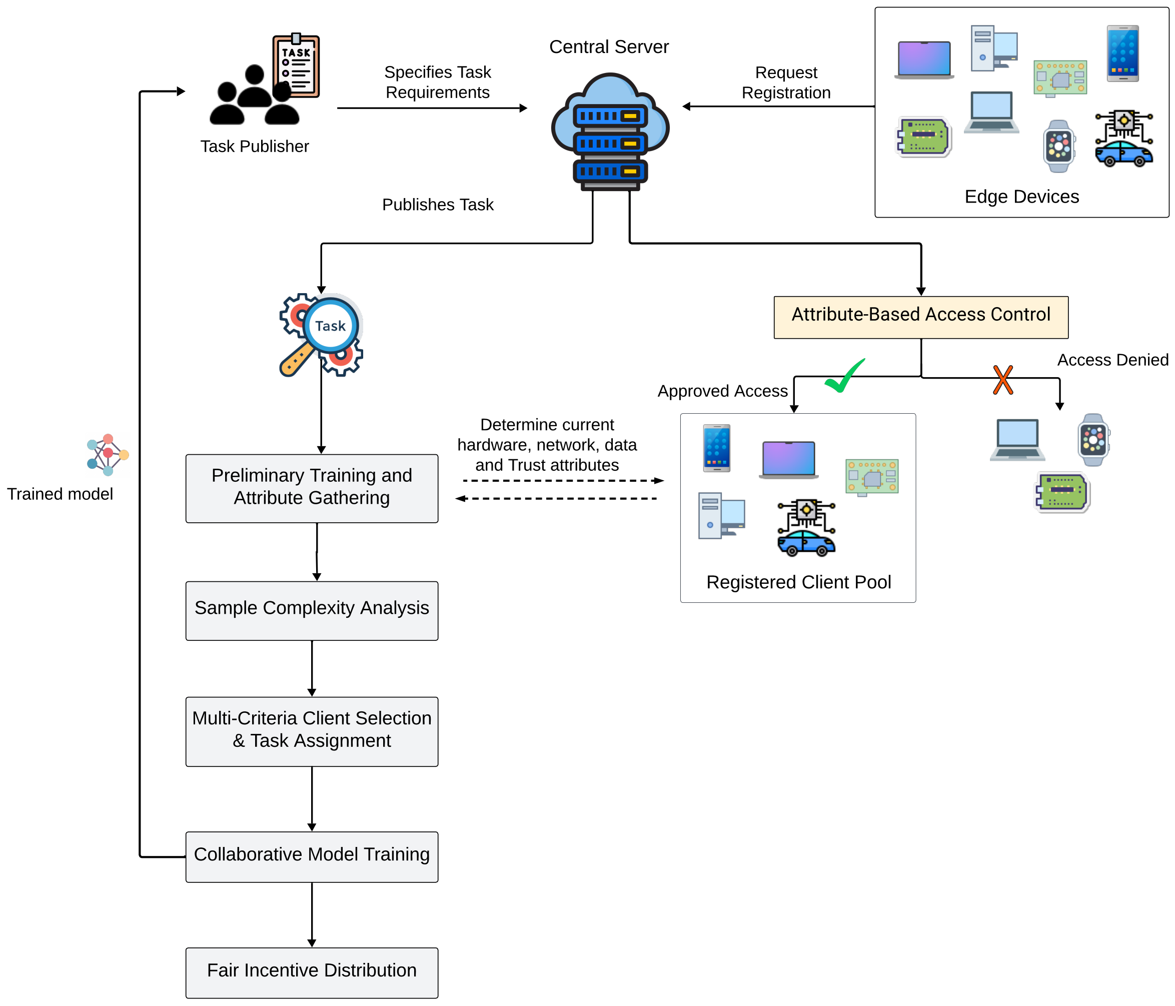

Task Publisher
Specifies Task Requirements
Central Server
Request Registration
Edge Devices
Publishes Task
Task
Attribute-Based Access Control
Access Denied
Approved Access
Determine current hardware, network, data and Trust attributes
Trained model
Preliminary Training and Attribute Gathering
Registered Client Pool
Sample Complexity Analysis
Multi-Criteria Client Selection & Task Assignment
Collaborative Model Training
Fair Incentive Distribution

Figure 4.1: An overview of FedOnDemand: Crowdsourced Federated Learning over Edge Devices

4.3 FedOnDemand: Enabling Crowdsourced Federated Learning

This section presents the design and workflow of FedOnDemand, focusing on its core components—device registration, task publishing, client selection, training, and incentivization. The section outlines how these elements interact to support dynamic, secure, and efficient collaboration at scale.

Figure 4.1 illustrates the proposed FedOnDemand framework. At its core, the system is designed around user registration, where participants voluntarily list their edge devices for training purposes. The central server is responsible for client registration, task distribution, client coordination, global model management, and maintaining system scalability. The server is optimized to support a large number of devices and frequent model updates with minimal overhead. Participants register their devices by specifying relevant attributes, such as computational capabilities and communication protocols. The registration process incorporates a robust security mechanism, including Attribute-Based Access Control (ABAC), which evaluates each requesting device against a predefined set of computation and communication criteria (using security policies), ensuring that only devices meeting the essential thresholds are added to the system. Once registered, these devices join the system's pool of available training devices.

When the platform receives a request for training an ML model, it is translated into the requirements of FL using edge devices. The central server plays a vital role in selecting suitable clients to participate in the training process for a given task. This decision takes into account multiple factors, including the device's capabilities, the relevance of its data, and the current resource availability. Selected clients then undertake the training task using their local data. Each client independently trains the model while preserving the privacy of their data. The updated model parameters are then sent back to the central server. During this exchange, the server also collects performance statistics from the client devices, assessing their individual contributions to the overall model.

To ensure the integrity and accuracy of the global model, the central server selectively accepts the model updates from the clients based on each client's performance and alignment with the request. By evaluating the received updates and aggregating them appropriately, the server improves the global model's quality. Once the learning process concludes, the participating devices are recognized and compensated for their contributions, reinforcing their importance in the on-demand FL paradigm.

The following sections present a comprehensive explanation of each component of FedOnDemand.

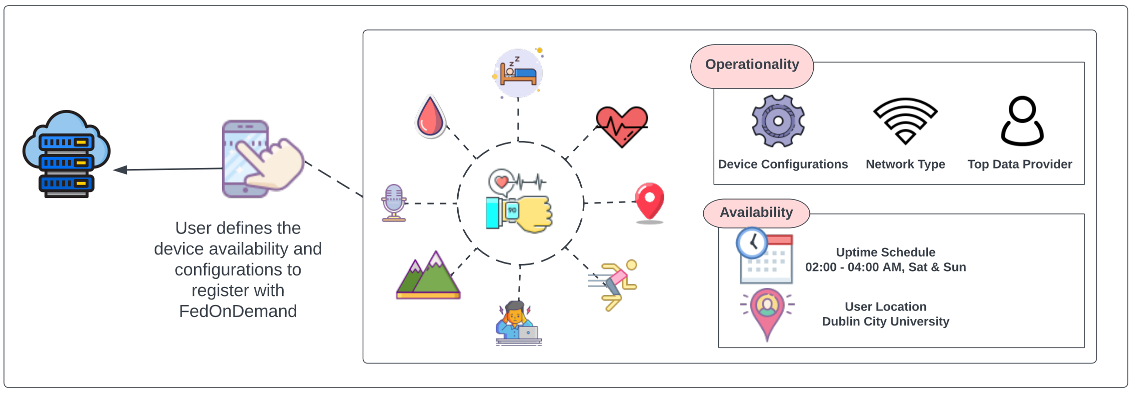


Figure 4.2: An example of crowdsourcing smartwatch data, through a mobile device, for FL with FedOnDemand

4.3.1 Device Registration and Profiling

As shown in Figure 4.2, consider a scenario where a device owner intends to share a specific type of data generated by one of their devices. With the Device Registration and Profiling component in FedOnDemand, device owners can register their devices and provide essential parameters to participate in the training process. This registration not only allows their devices to be part of the collaborative model training but also augments the platform's diversity of resources.

When registering a device on the platform, owners typically provide the following parameters:

- Device Configuration: Specifies the device's hardware details, including CPU, memory, and storage capacity. This information helps assess the computational capabilities of the device and determine the device's suitability for participation in various FL tasks.

- Data Type: Denotes the type of data the device generates, whether it's sensor readings, images, text, or audio. This allows the platform to match devices with relevant tasks and ensure that the data contributed aligns with the learning objectives. To ensure semantic consistency among these data types, the platform employs a standardized ontology-based mapping system, cross-verifying the data's semantic integrity with the specified type.

- Availability: Indicates when the device can be tapped for FL tasks. This information aids in scheduling and resource allocation, ensuring that devices are efficiently utilized during the training process. FedOnDemand assumes that devices remain available during their specified availability window. However, if a device becomes unreachable or drops out mid-training, it is simply excluded from that round. The system is designed to tolerate such dropouts without affecting overall convergence.
- Communication Capabilities: Details the supported communication protocols, such as Wi-Fi, Bluetooth, or cellular networks. This information helps determine the feasibility of device participation based on the task-specific communication requirements.

The device registration process is tightly controlled by an Attribute-Based Access Control (ABAC) mechanism. ABAC evaluates the requesting devices against pre-defined computation, communication, data compliance, and security benchmarks to authorize the registration. These policies are enforced using security policies. Only devices that meet these thresholds are authorized to be registered with the platform. This design supports fine-grained access control and allows for dynamic enforcement of policies as device attributes evolve. Figure 4.3 shows a basic ABAC security model.

In FedOnDemand, ABAC not only controls the registration process but also governs the entire lifecycle of a client in the system, from onboarding to model contribution. A detailed treatment of the ABAC framework and how it enforces zero-trust principles in a federated setting is provided in Chapter 7.

Once registered, the devices become part of the available pool for FL tasks. The system dynamically selects devices based on task requirements, device capabilities, and real-time availability, ensuring that the most suitable devices are leveraged for efficient model training.

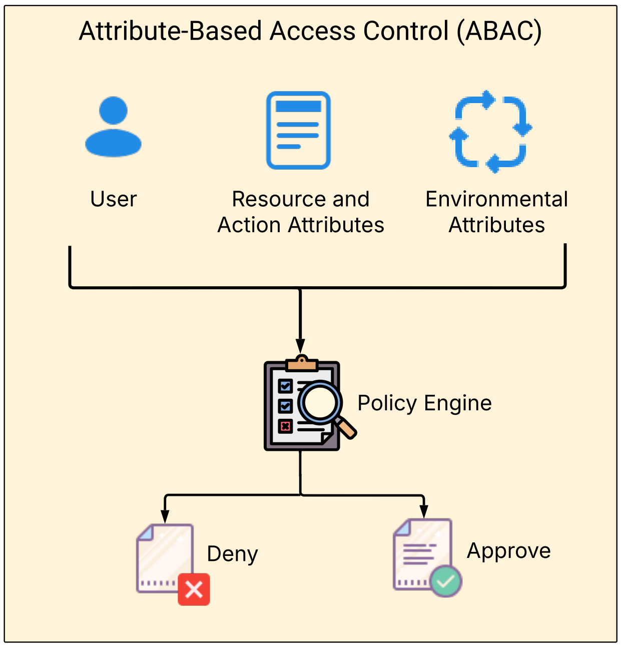

Attribute-Based Access Control (ABAC)
User
Resource and Action Attributes
Environmental Attributes
Policy Engine
Deny
Approve

Figure 4.3: Attribute-Based Access Control Security Model

4.3.2 Task Publishing and Requirements Specification

In the FedOnDemand platform, task publishing operates as a dynamic marketplace where requesters submit tasks for training specific machine learning models, along with their associated requirements and test set T. By specifying detailed parameters and constraints, requesters can define a well-structured and customized FL process that aligns with their objectives. This mechanism ensures that tasks are matched with suitable devices, optimizing the performance and efficiency of the training process.

To initiate the FL process, requesters provide the particular needs and constraints of their tasks. These parameters, though they may vary across problem domains, generally include desired data features, necessary solution functions, the selected learning model, and specific hyperparameters. Furthermore, requesters allocate a budget B and establish corresponding incentives for the task, encouraging device owners to contribute their resources and expertise.

For example, consider a requester with the goal of training an image classification model for an e-commerce platform using a large and diverse image dataset. In this scenario, they would specify preferences such as the convolutional neural network architecture (e.g., ResNet, VGG), image preprocessing techniques (e.g., normalization, data augmentation), and specific hyperparameters (e.g., learning rate, batch size). Depending on the complexity and scale of the task, they would also establish the budget and corresponding compensation.

Upon receiving all necessary details, the FedOnDemand platform acts as a central hub for task publishing and discovery. It facilitates the dissemination of task details to potential workers, enabling them to browse and identify tasks that align with their capabilities and interests.

4.3.3 Preliminary Training & Attribute Gathering

Once clients opt into a task, the server distributes the initial global model w_0 and invites clients to perform preliminary training on a subset of their local data. During this process, the platform gathers critical system-level metrics $H = \{h_1, h_2, ...\}$ (where h_i represents available CPU, memory, battery level, etc.), network-level attributes $N = \{n_1, n_2, ...\}$ (like bandwidth and latency), and a data quality measure Q (data size and local loss value gauged against the publisher's test set) as well as current trust score T. For each client C_i, the collected attributes form a vector $V_i = [g_j(c_i) \mid j \in \{H, N, Q, T\}]$. This data is crucial for the subsequent steps of the system, including estimating the sample complexity and selecting the most suitable clients for the training task.

4.3.4 Sample Complexity Estimation

In the FedOnDemand platform, we estimate the *sample complexity* to achieve the desired level of model performance and accuracy. Sample complexity refers to the minimum number of data points necessary to ensure the global model achieves a target generalization error ϵ. Given the distributed and non-IID nature of data across clients, we employ a PAC-Bayesian framework to derive bounds for this estimation.

The PAC-Bayesian framework is commonly used to provide probabilistic guarantees on the generalization error by combining prior knowledge and empirical observations. In this context, we derive a PAC-Bayesian bound, refined using Bernstein's inequality, to account for the variance in client data and the divergence between client-specific posterior and prior distributions. This approach ensures that our bound incorporates both the uncertainty in the data and the model's complexity.

At a high level, the sample complexity $N = \sum_{k=1}^{K} n_k$ is derived to ensure that the generalization error ϵ remains within a desired threshold with high probability $1 - \delta$. The detailed mathematical derivation of this estimation, along with assumptions and practical considerations, can be found in **Appendix A.1**. In summary, the key result of this derivation is as follows:

$$N \geq \frac{8\epsilon^2}{C^2} \left(\sum_{k=1}^{K} p(k) \left(D_{\mathrm{KL}}(Q_k \| P_k) + \sigma_k^2 \right) + \frac{\log(1/\delta)}{2n_k} \right)$$

Where:

- ϵ is the target generalization error,
- $D_{\mathrm{KL}}(Q_k \| P_k)$ represents the KL divergence between the posterior and prior distributions for client k,
- σ_k^2 denotes the variance in the loss function for client k,
- δ is the confidence level, and
- C bounds the loss function.

By estimating N and given the local data size each client holds, the platform can dynamically determine the number of clients required for the training process, adjusting the selection process accordingly.

4.3.5 Multi-Criteria Client Selection and Task Assignment

In our proposed system, we assume that the edge devices voluntarily participate in the FL training, expecting rewards. Therefore, the central server must assess the suitability of a participant for a given learning task to ensure an efficient learning process and high-quality solution. To address this intricate decision-making task, we reconceptualize client selection as a ranking problem and employ a widely adopted multi-criteria decision-making (MCDM) technique called PROMETHEE. PROMETHEE is a family of methods designed to rank and select alternatives based on multiple, often conflicting criteria [265]. Within this family:

- **PROMETHEE I** provides a partial ranking through pairwise comparisons.
- **PROMETHEE II**, which is employed in our framework, offers a complete ranking of the alternatives. This is achieved using the concept of a ”net flow”, which consolidates various criteria into a single score for each alternative, enabling a clear ranking from best to worst.

Unlike methods that rely on hierarchical comparisons or strict scoring schemes, PROMETHEE II uses preference functions to compare each pair of clients across all criteria to compute the net flow score. This allows us to flexibly encode task-specific priorities and generate a clear ranking of participating devices. Furthermore, it supports both qualitative and quantitative attributes, allowing fine-tuned preference modeling via customizable functions, and avoids the full pairwise comparison matrix required by methods like AHP. This makes it suitable for real-time decision-making in FL systems that must adapt to heterogeneous and dynamic environments.

In the context of FedOnDemand, the PROMETHEE II-based client selection method, namely FedPROM, systematically evaluates clients based on availability, hardware and network specifications, reputation scores, data quality, and budget constraints. By aggregating these factors, the method identifies the top-performing clients for each FL task, ensuring efficient performance while adhering to budgetary limits. The specifics of this client selection process are detailed in Chapter 5.

However, PROMETHEE has some limitations, particularly around computational scalability with a large number of clients. The method is also sensitive to the choice of preference functions, requires normalization, and may become difficult to manage as the number of criteria increases. To mitigate these issues, we implement a sorting-based variant of PROMETHEE that avoids exhaustive pairwise comparisons and streamlines the net flow computation. This improves speed and scalability without compromising selection quality. We detail the process of multi-criteria client selection using sorting-based PROMETHEE in Section 5.7.

We also considered other MCDM techniques such as AHP, TOPSIS, and ELECTRE. While each has its merits, AHP struggles with scalability due to its reliance on full pairwise comparisons. TOPSIS is computationally efficient but assumes linear trade-offs between criteria, which can oversimplify client heterogeneity. ELECTRE supports more complex decision logic but is less interpretable and more expensive to compute. PROMETHEE, particularly in its optimized form, offers a middle ground—providing flexibility, scalability, and clear rankings, making it well-suited for the client selection needs of FedOnDemand.

Through this comprehensive selection approach, FedOnDemand improves the overall performance and accuracy of the FL process by:

- Selecting clients with sufficient resources and stable network connectivity.
- Ensuring only trustworthy clients with relevant, high-quality data participate, thus improving security and reliability.
- Prioritizing clients that best align with task requirements, minimizing communication delays, and operational costs.

After selecting the top-ranked clients, FedOnDemand distributes the initial model to the selected devices for local data training.

4.3.6 Collaborative Model Training

Clients in FedOnDemand collaboratively update the global model parameters using their local datasets and send these updates to the central server. The process follows a standard flow, where local learning and model aggregation occur iteratively across multiple communication rounds, with the goal of improving the global model over time. We model FL as a Bayesian process to handle the uncertainty associated with varying data distributions. The training process follows the steps outlined below:

- **Model Initialization:** The central server initializes the global model and its parameters, distributing it to the selected clients for training. The initial model may undergo pre-training to establish a starting point. In order to reduce communication overhead, the model size is optimized to ensure efficient transfers between the server and client devices.

- **Local Learning:** Each client performs local training using its private dataset. At this stage, Stochastic Variational Inference (SVI) is employed to approximate the posterior distribution over the client-specific parameters. Clients minimize their local loss function based on this distribution, generating updated parameters that reflect both their local data and the prior global model. These updated parameters are then sent back to the server.

- **Aggregation:** Upon receiving updates from participating clients, the central server performs Hierarchical Bayesian aggregation. To account for the variability in data quality across crowdsourced devices, updates are variance-weighted—clients with more consistent local training (lower variance) contribute more to the global model—ensuring that noisy updates have a reduced influence on the final parameters. This process incorporates the client-specific updates while accounting for uncertainty and variability across the clients' data distributions. The aggregated model reflects the diverse contributions of the clients, ensuring that the global model evolves in a robust and reliable manner.

In a crowdsourced environment, the challenge of efficiently managing communication and resource constraints across a wide variety of devices is significant. By utilizing Bayesian learning principles, the system can effectively handle the non-IID nature of client data, ensuring that updates are aggregated in a way that balances contribution quality and performance. Furthermore, the updates in FedOnDemand are expected to be dynamic, where each round may involve different clients based on availability, and the influence of each client's update changes depending on its variance. This allows the global model to continuously adapt in response to real-time participation and varying data characteristics.

This process of local training and global aggregation is repeated over several communication rounds or until the global model reaches a desired level of accuracy. We detail the process of Bayesian learning for FL in Chapter 6.

4.3.7 Fair Incentive Mechanism

In a crowdsourced environment, the client devices volunteer to participate in the training process with the expectation of receiving rewards. Traditional incentive mechanisms often fail to account for the varying quality of contributions from individual participants and adopt a one-size-fits-all pricing system. This approach risks undervaluing high-quality data contributions and may inadvertently attract malicious actors.

To address these limitations, FedOnDemand utilizes a Shapley value-based incentive mechanism, which ensures a fair and rational allocation of rewards based on the value of each participant's data contribution [266, 267]. Figure 4.4 shows a high-level overview of the adopted incentive mechanism.

The Shapley value treats each data contributor as a player in a cooperative game, where the utility of any subset of contributors is determined by the collective value they add to the trained model. This profit-sharing mechanism fairly distributes the total gains

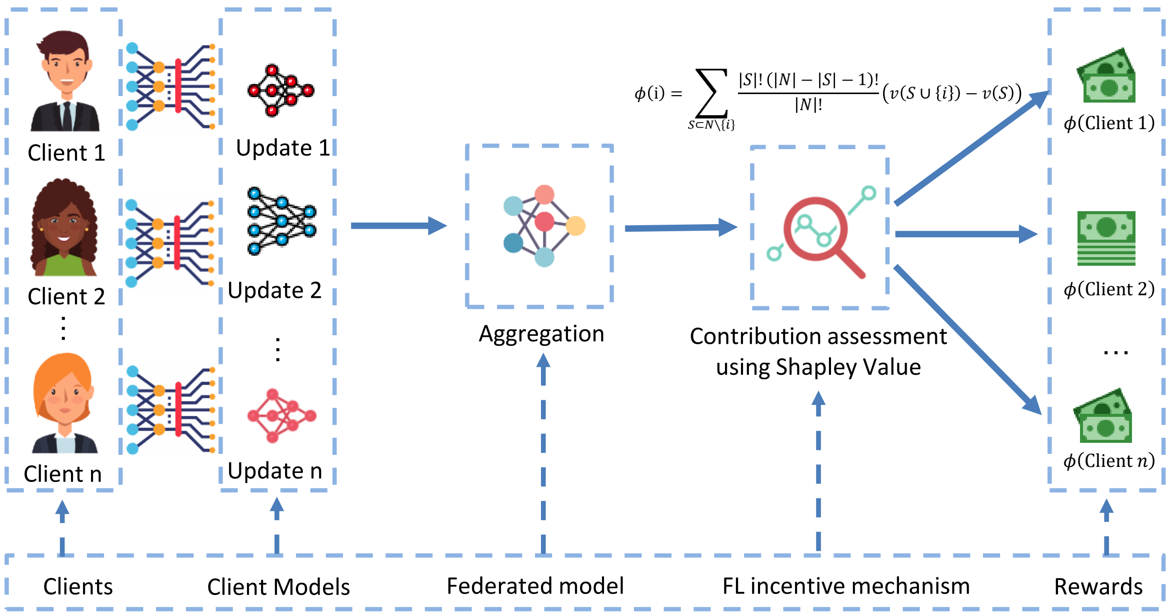


Figure 4.4: Illustration of a fairness-aware incentive mechanism in Federated Learning. Individual client updates are assessed using Shapley values to determine equitable reward distribution based on each client's contribution.

among all participants and is characterized by its fairness, rationality, and decentralized nature.

The Shapley value for each participant is calculated based on their marginal contribution to the coalition. For any participant i, the Shapley value, denoted as $\phi(i)$, is defined as the average of the marginal contributions across all possible permutations of participants. Mathematically, it can be expressed as:

$$\phi(i) = \sum_{S\subset N\setminus i} \frac{|S|!(N-|S|-1)!}{|N|!}(v(S\cup i)-v(S)) \tag{4.1}$$

Where N represents the set of all participants in the system, $v(S)$ denotes the contribution of the model collaboratively trained by the subset S, and $(v(S\cup i)-v(S))$ represents the marginal contribution of participant i to the subset S.

By employing the Shapley value-based incentive mechanism, FedOnDemand promotes fair distribution of rewards among the clients as it recognizes and rewards the individual contributions of participants based on the unique value they bring to the collaborative training process. Consequently, it incentivizes active participation, encourages the sharing of high-quality data, and enhances the overall performance and effectiveness of the crowdsourced FL ecosystem.

4.4 Summary

This chapter introduced FedOnDemand, a crowdsourced federated learning framework designed to support dynamic client participation, task-driven training, and fair contribution-based incentives. It outlined the system's architecture and core components, including device registration with ABAC-based access control, task publishing, adaptive client selection using PROMETHEE, Bayesian model training, and Shapley value-based reward allocation.

Each component is designed to address a specific operational challenge in deploying federated learning across diverse and intermittently available devices. Together, they enable a scalable and secure learning environment without assuming consistent participation or homogeneous resources.

The remainder of this dissertation builds on this foundation by examining and formalizing the key submodules of FedOnDemand. The next chapter begins with the design and optimization of the client selection mechanism to ensure efficient and context-aware participation across large client pools.

Chapter 5

Multi-Criteria Client Selecting for Efficient Federated Learning

5.1 Introduction

The practical deployment of FL in real-world scenarios is often limited by the inherent heterogeneity of both data and system resources. Clients in a federated network often possess vastly different computational capabilities, network conditions, and data distributions. This heterogeneity can lead to inefficient training processes, prolonged convergence times, and degraded model performance. In such scenarios, the challenge is to optimize the FL training process in the presence of system and data heterogeneity, and the participating clients effectively determine the success of the learning task.

Unlike traditional centralized machine learning, FL involves a large number of distributed clients, each equipped with its own local data and computational resources. These clients perform training on local data and periodically send model updates to a central server. Given the variability in client devices and data quality, selecting the appropriate clients to participate in each round of training is crucial to ensuring both the efficiency and effectiveness of the global model.

Client selection, therefore, must balance multiple factors, including the availability of computational resources, the quality and quantity of local data, and the network conditions of each client. Furthermore, in many practical FL systems, the number of clients far exceeds the number that can participate in each training round. As a result, effective client selection is not only important for reducing communication overhead but also for maximizing model performance and ensuring fair participation across a diverse set of clients.

The need for efficient client selection becomes even more apparent in scenarios where clients have limited resources, such as mobile devices or IoT sensors, or when data is highly heterogeneous, leading to non-IID data distributions across the network. Without proper selection strategies, these issues can result in slower convergence, biased models, or inefficiencies in resource utilization.

This chapter explores the mechanisms behind client selection in FL, addressing the challenges posed by heterogeneous data, varying client capabilities, and the need for fairness. Through optimizing client participation, the overall system performance and scalability of FL systems can be significantly improved, ensuring that the global model benefits from high-quality data while minimizing system resource consumption.

5.2 Objectives

This research aims to optimize the client selection process in FL and introduces a multi-criteria client selection mechanism that evaluates clients based on their computational resources, communication bandwidth, data quality, and trustworthiness. The objective is to enhance system performance while reducing training time, addressing the challenges of client heterogeneity and inefficient resource utilization. By selecting clients based on these criteria, the framework seeks to improve system performance and model convergence in real-world environments.

5.3 Challenges

This research faces several challenges in optimizing client selection within FL systems. The primary challenge arises from the heterogeneity of clients, as each device differs significantly in terms of computational power, energy availability, and network conditions. This diversity complicates the selection of clients that can contribute effectively to the training process without introducing delays or inefficiencies. Another challenge is managing the trade-off between performance and training time. Selecting clients with stronger computational capabilities may reduce training time but could compromise the diversity of data contributions, leading to biased or less generalizable global models. Additionally, finding the optimal client selection scheme that dynamically adapts to varying conditions, such as fluctuating network bandwidth and computational resources, adds complexity to the task. Lastly, the need to operate under budget constraints, particularly in resource-constrained environments like mobile and IoT networks, further complicates the selection process. Addressing these challenges is essential for developing a client selection framework that balances system performance and efficiency.

5.4 Contribution

This chapter introduces a novel approach to client selection in FL, addressing the challenges of heterogeneity, performance trade-offs, and resource constraints through a multi-criteria optimization framework. The key contributions of this work are as follows:

- We formulate the client selection problem as a dual optimization task that balances the trade-off between performance and training time. This approach ensures that selected clients provide both high-quality data and efficient computation, optimizing the overall performance of the FL system.

- A multi-criteria client selection framework is proposed, which evaluates clients based on computational resources, data quality, and network conditions. This method adapts dynamically to changes in the environment, ensuring optimal client selection at each communication round.

- The proposed framework incorporates budget constraints into the client selection process. By doing so, the system can operate within practical limits while maintaining high performance, making the framework suitable for deployment in resource-constrained environments such as IoT and mobile devices.

5.5 Problem Formulation

The inherent heterogeneity of edge devices in terms of computational resources, network conditions, and data distributions poses significant performance challenges. Addressing these challenges requires a strategic approach to client selection that balances the need for efficiency and achieving high model accuracy, all under privacy and security constraints.

Formally, let $C = \{c_1, c_2, ..., c_n\}$ represent the set of all available clients. Each client c_i has a local dataset D_i of size $|D_i|$ and characterized by an attribute vector V_i comprised of client's system $[H]$, network $[N]$, data quality $[Q]$, and trustworthiness metrics $[T]$.

$$V_i = [g_j(c_i) \mid j \in \{H, N, Q, T\}]$$

In this context, $g_j(c_i)$ serves as a generic function that, depending on the attribute j being considered, evaluates the corresponding aspect of a client's potential contribution to FL. Specifically:

- $g_H(c_i)$ evaluates the client's system-level metrics such as available CPU, memory, storage, and battery level.

- $g_N(c_i)$ denotes the client's network-level attributes, such as connection status, bandwidth, and latency.

- $g_Q(c_i)$ quantifies data quality measured as a function of local loss and the size of the client's dataset $|D'_i|$.
- $g_T(c_i))$: represents the client's trustworthiness based on behavior metrics and compliance with access control policies.

We elaborate on these criteria in the following sections. Additionally, the inclusion of each client c_i in the FL training process incurs a particular cost, $cost(c_i)$, which accounts for the computational resources expended during local training, the communication overhead involved in transmitting model updates, and the contribution of the client's data to the global model in terms of quality, quantity, and statistical relevance.

To address both performance and training efficiency, we formulate client selection as a dual objective optimization problem: from a given pool of clients C, identify a subset of clients $S \subseteq C$ that maximizes global model utility while minimizing overall training time, under a given budget constraint. Mathematically, the optimization problem can be written as follows:

$$\begin{cases} \max_{S\subseteq\mathcal{C}} & w_{global}^{(t+1)} = \frac{1}{N_S}\sum_{c_i\in S} n_{c_i} w_{c_i}^{(t)} \\ \\ \min_{S\subseteq\mathcal{C}} & T_{\text{total}}(S) = f_T\left(S, \{V_i | c_i \in S\}\right) \end{cases}$$

Subject to:

$$\sum_{c_i\in S} \text{cost}(c_i) \leq B$$

Where:

- $w_{global}^{(t+1)}$ represents the global model's accuracy after updates, calculated through a weighted average of the updates provided by the selected subset of clients, S. Each client's update, $w_{c_i}^{(t)}$, is weighted by the proportion of data samples n_{c_i} they contribute, relative to the total number of samples N_S from all clients in S. This formulation ensures that clients contributing more data have a correspondingly larger influence on the global model's parameters and aligns well with our objective. However, in practice, weighted aggregation can be replaced by any other aggregation mechanism.

- $T_{total}(S)$ denotes the total time required to train the global model using the selected client subset S. This duration is modeled as the aggregate of individual training times across all selected clients in S and integrates the effects of each client's computational and communication capabilities.
- The budgetary constraint stipulates that the cumulative cost associated with engaging the selected clients, $\sum_{c_i \in S} \text{cost}(c_i)$, must not exceed the predefined budget B. This cost includes considerations for computational resources, network usage, and data contributions.

The dual objectives aim to maximize model accuracy by selecting clients with high-quality and diverse data while minimizing total training time by considering computational and network resources. Determining the optimal subset S that satisfies these objectives is computationally challenging. In fact, the problem can be shown to be NP-hard, as it involves combinatorial optimization over the set of all possible client subsets.

Theorem: The multi-criteria optimization problem for client selection in FL, as we formulated above, is NP-hard.

Proof: We demonstrate the NP-hardness of our client selection problem by constructing a reduction from the Knapsack problem, which is a well-known NP-hard problem.

In the Knapsack problem, the goal is to select items to maximize the total value without exceeding a weight limit. We map this problem to our client selection context as follows:

- *Items in Knapsack*: Each item in the Knapsack problem corresponds to a client in the FL network.

- *Weight of Items*: The weight of each item in the Knapsack problem is analogous to the cost associated with selecting a client for training.
- *Value of Items*: The value of an item in the Knapsack problem maps to a combined metric in the FL problem, representing each client's contribution to model accuracy and the efficiency of training. This metric could be a function of the client's data quality and the reciprocal of its expected training time, where shorter training times are more valuable.
- *Weight Capacity*: The weight capacity in the Knapsack problem equates to the total available budget in our problem, encompassing overall resource limitations.

The transformation function T takes an instance of the Knapsack problem (items, weights, values, and capacity) and maps it to an instance of the client selection problem with corresponding clients, resource costs, contributions to accuracy, and training time efficiency under a budget constraint.

Since solving the transformed problem effectively solves the Knapsack problem instance and the Knapsack problem is NP-hard, it follows that the client selection problem, with its dual objectives of maximizing accuracy and minimizing training time under budget constraints, is also NP-hard. This complexity arises from the need to optimize multiple conflicting criteria simultaneously.

5.6 Multi-Criteria Client Selection for FL Using PROMETHEE

The inherent complexity and NP-hardness of multi-criteria client selection in FL make traditional deterministic approaches inefficient. Practical solutions simplify the problem to one solvable in polynomial time, balancing multiple conflicting criteria to optimize accuracy and training time. This section introduces a novel method, using MCDM,

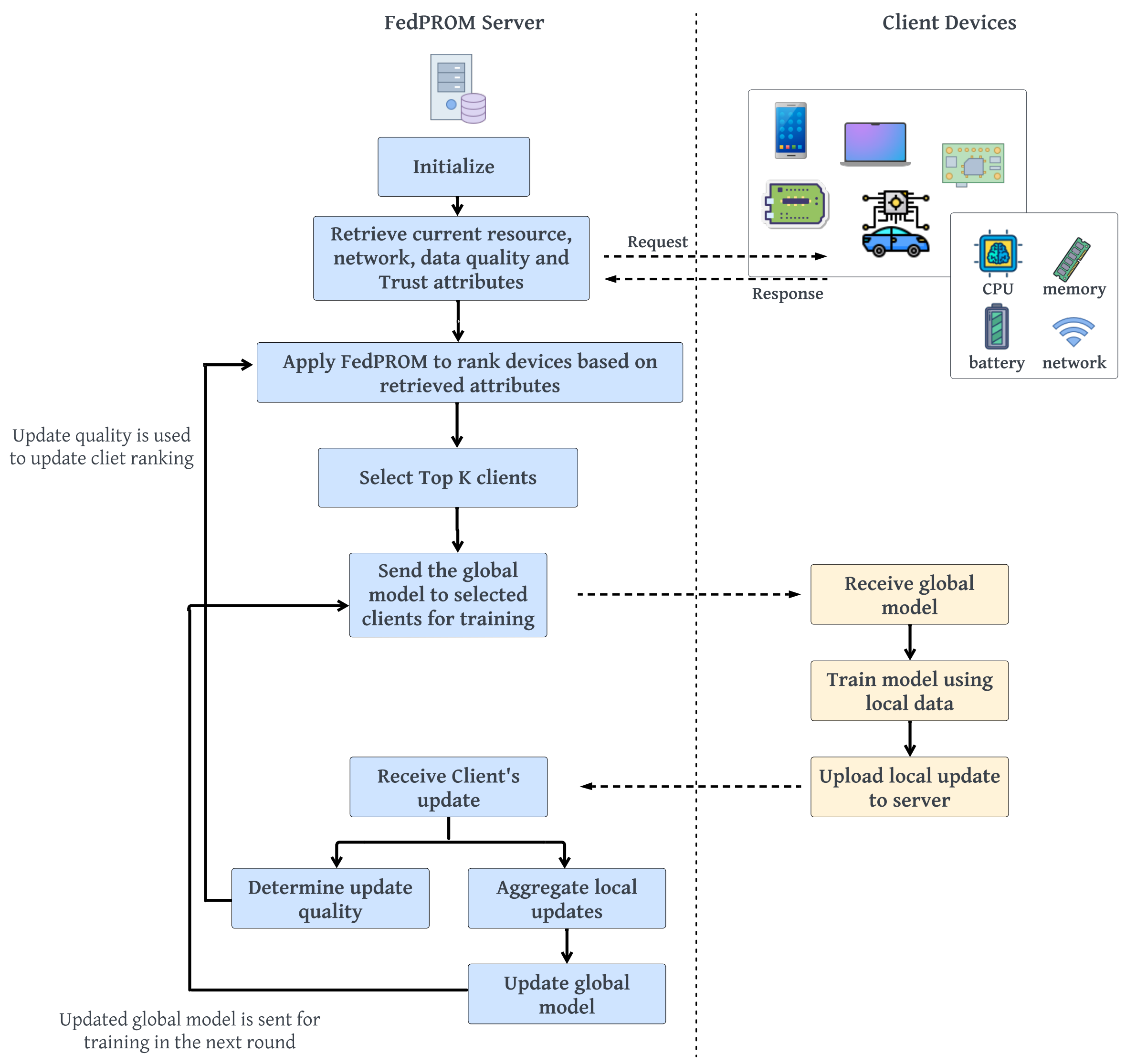

FedPROM Server
Client Devices
Initialize
Retrieve current resource, network, data quality and Trust attributes
Request
Response
CPU
memory
battery
network
Apply FedPROM to rank devices based on retrieved attributes
Update quality is used to update cliet ranking
Select Top K clients
Send the global model to selected clients for training
Receive global model
Train model using local data
Upload local update to server
Receive Client's update
Determine update quality
Aggregate local updates
Update global model
Updated global model is sent for training in the next round

Figure 5.1: **Overview of FedPROM Protocol**. Solid black lines denote computation processes, while dashed lines indicate wireless communications.

of treating FL client selection as a ranking problem. MCDM helps evaluate multiple conflicting criteria systematically, avoiding exhaustive optimal searches by ranking clients based on predetermined criteria and weights. Our approach utilizes the PROMETHEE decision-making process, known for its flexibility and effectiveness in handling qualitative and quantitative assessments. PROMETHEE compares clients pairwise based on criteria, assigning normalized scores to quantify attributes and facilitate effective client selection. The main steps include:

- Define the criteria and assign relative importance (weights) to each criterion.
- Select appropriate preference functions for each criterion, indicating how the decision-maker's preference increases as the performance on that criterion improves.
- Compute the degree of preference for each pair of alternatives based on their performance on each criterion and the corresponding preference functions.
- Aggregate the pairwise preferences to determine the overall ranking of the alternatives.

Figure 5.1 shows the overview of the proposed protocol.

In FedPROM, each client c_i is represented by a set of aggregated attributes $g_j(c_i)$ such as hardware, network, data quality, and trustworthiness. Consolidating individual attributes (e.g., CPU, RAM, latency) into these broader categories allows for a scalable and efficient ranking process as it reduces complexity by evaluating each client's overall capability within the defined category rather than conducting granular comparisons on specific hardware or network metrics. To facilitate client ranking and selection, we quantify these aggregated attributes by defining associated scores using utility functions. These scores are normalized between 0 and 1, where 1 represents the highest possible quality or performance for each criterion. We define the aggregated scores as follows:

- **Hardware Scores ($g_H(c_i)$):** Quantifies a client's computational capabilities for FL tasks, assessed during registration and preliminary training. It incorporates the device's computing power, RAM size, storage capacity, power, etc., into a composite score:

$$\begin{aligned} g_H(c_i) = f(&ComputationUtility, MemoryUtility, \\ &StorageUtility, PowerUtility) \end{aligned} \tag{5.1}$$

Each utility function (e.g., *ComputationUtility*, *MemoryUtility*) is defined as the ratio of the device's available resources to the minimum required for the task. For instance, the *ComputationUtility* function assesses CPU and GPU capabilities based on processing speeds, translating them into a score relative to the task requirements. These functions map the raw attribute data to a normalized score and aggregate them using a weighted average into a composite score, as defined above. Here, the weights of the individual utility are determined based on the FL task requirements. For example, for compute-intensive tasks, *ComputationUtility* and *MemoryUtility* may be assigned higher weights to prioritize clients with better computational resources. In other scenarios, *PowerUtility* may be higher where long-term connectivity is required.

- **Network Scores ($g_N(c_i)$):** Evaluate the client's network attributes like connection status, bandwidth, and latency. These metrics directly affect data transfer rates and the client's responsiveness during training and model updates, hence, are crucial for efficient FL communication. We use the same approach, as defined for hardware scores, to determine the composite score for the network as well:

$$g_N(c_i) = f(\mathit{ConnectionStatus}, \mathit{Bandwidth}, \mathit{Latency}) \tag{5.2}$$

This function assesses the overall network health, prioritizing stable connections with high bandwidth and low latency, and returns a composite score for the network.

- **Data Quality Scores ($g_Q(c_i)$):** Reflects a client's data utility for the federated model, calculated as:

$$g_Q(c_i) = f(\mathit{DataSize}, \mathit{LocalLoss}) \tag{5.3}$$

where f represents the function that assesses the client's data size and the impact of its local loss on learning effectiveness. Prospectively, this formulation could be extended to include additional parameters such as data freshness or the degree of non-IIDness, offering a more comprehensive evaluation of data quality.

- **Trust Scores** ($g_T(c_i)$): Represents the client's trustworthiness based on compliance with security policies, behavior metrics, and trust evaluations. This score is dynamically updated after every training round to reflect the client's ongoing adherence to security and operational standards.

- **Other Scores**: While hardware capabilities, network conditions, data quality, and trust scores are foundational, other contextual factors could also significantly influence client selection in FL environments. For our scenario, we assume all clients are available when needed, focusing on their hardware, network, data quality, and trustworthiness. In any other case, the client's attribute vector can be expanded to include other contextual factors to enable a more comprehensive client evaluation.

These scores are then used to evaluate and rank clients based on their contribution to the FL system, which is determined by the preference function.

For two clients c_i and c_k, the preference P_j of c_i over c_k for a criterion j is determined using a linear preference function that takes into account the difference in criterion values between two clients:

$$P_j(g_j(c_i) - g_j(c_k)) = \begin{cases} 0 & \text{if } d \leq q_j \\ \frac{d-q_j}{p_j-q_j} & \text{if } q_j \leq d \leq p_j \\ 1 & \text{if } d \geq p_j \end{cases} \tag{5.4}$$

Where $d = g_j(c_i) - g_j(c_k)$ represents the difference in criterion values between the two clients. The term q_j here represents an indifference threshold for the criterion j. This parameter is crucial as it signifies a range within which the scores' differences are deemed non-impactful. It effectively states that if the difference between the scores of two alternatives under a given criterion is less than q_j, then these two alternatives are regarded equally preferable with respect to that criterion. Conversely, p_j defines a strict preference zone leading to a preference value of 1.

In the context of FedPROM, we set a low value of the indifference threshold q_j (close to 0) to ensure that even minor differences could play a decisive role in client selection.

These individual criteria preferences are then integrated to generate a global preference score for each client. A global preference score, $\pi(c_i, c_k)$, indicates the degree to which client c_i is preferred over client c_k for m criteria, and is computed as follows:

$$\pi(c_i, c_k) = \sum_{j=1}^{m} W_j P_j(g_j(c_i) - g_j(c_k)) \tag{5.5}$$

Here, W_j represents the weight associated with the criterion j, ensuring the relative importance of the criteria is captured. The purpose of the global preference is to have a comprehensive measure that encapsulates all criteria to compare any two clients directly. The Outranking Flows generalize this idea to provide a broader perspective, analyzing how one client compares to all other clients in the set. For example, the positive outranking flow, $\phi^+(c_i)$, computes the average of the global preference values of the client c_i over all other clients. Essentially, it provides an idea of how preferable client c_i is when compared to every other client in the system. Similarly, the negative outranking flow, $\phi^-(c_i)$, inverts this and computes the average of global preference values of all other clients over the client c_i. It gauges the average extent to which other clients are preferred over the client c_i. These flows are calculated as follows:

$$\phi^+(c_i) = \frac{1}{n-1} \sum_{k=1, k\neq i}^{n} \pi(c_i, c_k) \tag{5.6}$$

$$\phi^-(c_i) = \frac{1}{n-1} \sum_{k=1, k\neq i}^{n} \pi(c_k, c_i) \tag{5.7}$$

Lastly, the final ranking of the clients is determined based on the net outranking flow $\phi(c_i)$.

$$\phi(c_i) = \phi^+(c_i) - \phi^-(c_i) \tag{5.8}$$

A high net flow for a client suggests that, on average, this client is more preferred when compared to most others in the system, making it an optimal candidate for the task. Following these steps, the top k clients can be selected directly.

Algorithm 1 Multi-Criteria Client Selection Using FedPROM

```
1: **Input:** Set of clients $C = \{c_1, c_2, \dots, c_n\}$ with attributes $g_j(c_i)$ for $j \in V_i = \{H_i, N_i, Q_i, T_i\}$, Budget $B$.
2: **Output:** Selected client set $S$.
3: Initialize $S = \emptyset$
4: Initialize $AvailableBudget = B$
5: for each client $c_i \in C$ do
6:     Compute normalized attributes $g'_j(c_i)$
7: end for
8: for each criterion $j$ do
9:     Compute pairwise preference $P_j(g'_j(c_i), g'_j(c_k))$ for all client pairs $(c_i, c_k)$
10: end for
11: for each client $c_i \in C$ do
12:     Calculate global preference score $\pi(c_i, c_k)$
13:     Calculate outranking flows $\phi^+(c_i)$ and $\phi^-(c_i)$
14: end for
15: Rank clients based on net outranking flow $\phi(c_i)$
16: while there are clients in $C$ and $AvailableBudget > 0$ do
17:     Select $c_{best}$ from $C$ with the highest $\phi(c_{best})$ and within budget
18:     if $cost(c_{best}) \leq AvailableBudget$ then
19:         Add $c_{best}$ to $S$
20:         Update $AvailableBudget = AvailableBudget - cost(c_{best})$
21:     end if
22:     Remove $c_{best}$ from $C$
23: end while
24: return $S$
```

Algorithm 1 presents the pseudocode of FedPROM. By transforming the problem into a ranking task, we essentially sidestep the combinatorial complexity inherent in direct subset selection.

5.6.1 Complexity Analysis

In the worst case, the computational complexity of our proposed solution involves:

- The complexity of computing pairwise preferences for m criteria across n clients is $O(m \cdot n^2)$, which, while substantial, is significantly more manageable than the combinatorial explosion associated with NP-hard problems.

- Sorting clients based on their net outranking flows involves an additional computational complexity of $O(n \log n)$ following the computation of preferences.

These steps combine to give the FedPROM protocol an overall computational complexity of $O(m \cdot n^2 + n \log n)$. To further reduce the complexity, we pre-filter the clients based on the required hardware and network capabilities, which significantly decreases the pool of clients, thus reducing the overall complexity. While this complexity is manageable in static environments where client attributes remain relatively stable over time, it poses significant challenges in dynamic, large-scale environments. In scenarios where client attributes such as trustworthiness, data quality, or network conditions frequently change, the necessity to recompute the entire preference matrix for every modification becomes computationally intensive. To enhance scalability and reduce processing overhead, we utilize an optimized version of the PROMETHEE algorithm that leverages a sorting-based approach combined with sliding window optimizations to achieve a more efficient $O(m \cdot n \log n)$ computational complexity [268]. This optimization ensures that client ranking remains both accurate and computationally feasible, even as the client pool grows dynamically.

5.7 Multi-Criteria Client Selection using Sorting-Based PROMETHEE

To reduce the computational complexity further, we utilize a sorting-based PROMETHEE approach [268] that allows efficient ranking of clients even in dynamic environments by only updating the affected rankings when new clients join, leave, or modify their attributes. For each criterion $j \in \{H, N, Q, T\}$, clients are sorted based on their scores, producing four separate, sorted lists. We utilize the same composite score as mentioned earlier in the section. Sorting these scores ensures that comparisons between clients can be limited to those within a specific preference window, significantly reducing the need for full pairwise comparisons across all clients.

For two clients c_i and c_k, the preference P_j of c_i over c_k for a criterion j with score $g_j(c_i)$ and $g_j(c_k)$ is computed using a linear preference function as described in Equation 5.4.

$$P_j(g_j(c_i) - g_j(c_k)) = \begin{cases} 0 & \text{if } d \leq q_j \\ \frac{d-q_j}{p_j-q_j} & \text{if } q_j \leq d \leq p_j \\ 1 & \text{if } d \geq p_j \end{cases}$$

Where $d = g_j(c_i) - g_j(c_k)$ represents the difference in criterion values between the two clients.

To optimize the computational efficiency of the preference calculation, we implement a preference window approach. For each criterion j, the system defines the lower bound l and upper bound u using the thresholds q_j and p_j to create a preference window for each client. Specifically, for a client c_i with criterion score $g_j(c_i)$, the bounds can be defined as follows:

$$l_j = g_j(c_i) - p_j$$

$$u_j = g_j(c_i) - q_j$$

Using these bounds, only clients with scores $g_j(c_k)$ on criterion j falling within a specified range, referred to as the "window" $[l, u]$, are considered comparable to c_i. SecureFed-PROM computes this window incrementally, moving it along the sorted list, adding or removing clients from the window based on their criterion scores relative to c_i.

Based on the partial comparisons within this window, the system computes both the uni-criterion positive flow $\phi_j^+(c_i)$ and negative flow $\phi_j^-(c_i)$ for the client c_i. The uni-criterion positive flow $\phi_j^+(c_i)$ measures the extent to which client c_i is preferred over other clients within the window on a single criterion j. Conversely, the uni-criterion negative flow $\phi_j^-(c_i)$ assesses the degree to which other clients are preferred over c_i on the same criterion j. These flows are computed as follows:

$$\phi_j^+(c_i) = \frac{1}{|\text{window}|} \sum_{c_k \in \text{window}} P_j(g_j(c_i) - g_j(c_k)) \tag{5.9}$$

$$\phi_j^-(c_i) = \frac{1}{|\text{window}|} \sum_{c_k \in \text{window}} P_j(g_j(c_k) - g_j(c_i)) \tag{5.10}$$

Here, $\phi_j^+(c_i)$ represents the degree to which c_i is preferred over other clients in the preference window, while $\phi_j^-(c_i)$ represents the degree to which other clients are preferred over c_i on the same criterion j within this window. Algorithm 2 presents the pseudocode for computing these preferences for a given criterion, known as uni-criterion flows, with a sliding window approach.

These uni-criterion flows for all criteria are then aggregated to compute a comprehensive preference measure for each client. Each client's ranking is derived directly using its positive and negative flows and aggregating them using a weighted sum across all criteria.

$$\begin{aligned} \phi^+(c_i) &= \sum_{j=1}^{m} W_j \phi_j^+(c_i) \\ \phi^-(c_i) &= \sum_{j=1}^{m} W_j \phi_j^-(c_i) \end{aligned} \tag{5.11}$$

Here, W_j represents the criterion weights, which denote the relative importance of each criterion $j \in \{H, N, Q, T\}$ in the overall client ranking. It ensures that the overall ranking reflects the strategic priorities of the FL task, allowing certain criteria to have a greater influence on the final client selection. For example, higher W_j for trust scores may be assigned for model training on sensitive data to prioritize clients that comply with

Algorithm 2 Compute Uni-Criterion Flows with Sliding Window

```
Input: Set of clients $C' = \{c'_1, c'_2, \dots, c'_n\}$ with scores $g_j(c'_i)$ for criterion $j$, thresholds $q_j$ and $p_j$
Output: Positive and negative flows $\phi_j^+(c'_i)$, $\phi_j^-(c'_i)$ for all $c'_i \in C'$
 1: procedure ComputeUniCriterionFlows($C'$, $g_j(c'_i)$, $q_j$, $p_j$)
 2:     Sort $C'$ in ascending order of $g_j(c'_i)$
 3:     $\phi_j^+ \leftarrow$ ComputeFlows($C'$, $g_j(c'_i)$, $q_j$, $p_j$)
 4:     Define $g'_j(c'_i) = -g_j(c'_i)$ for all $c'_i \in C'$
 5:     $\phi_j^- \leftarrow$ ComputeFlows($C'$, $g'_j(c'_i)$, $q_j$, $p_j$)
 6:     return $(\phi_j^+, \phi_j^-)$
 7: end procedure
 8: procedure ComputeFlows($C'$, $g_j(c'_i)$, $q_j$, $p_j$)
 9:     Initialize $\phi_j(c'_i) \leftarrow 0$, $sum\_in\_window \leftarrow 0$, $start \leftarrow 1$, $end \leftarrow 1$
10:     for $i \leftarrow 1$ to $n$ do
11:         $l_j \leftarrow g_j(c'_i) - p_j$, $u_j \leftarrow g_j(c'_i) - q_j$
12:         while $start < i$ and $g_j(C'[start]) < l$ do
13:                                                             ▷ Adjust window
14:             $sum\_in\_window \leftarrow sum\_in\_window - g_j(C'[start])$
15:             $start \leftarrow start + 1$
16:         end while
17:         while $end \leq n$ and $g_j(C'[end]) \leq u$ do
18:                                                             ▷ Expand window
19:             $sum\_in\_window \leftarrow sum\_in\_window + g_j(C'[end])$
20:             $end \leftarrow end + 1$
21:         end while
22:         if $end - start > 0$ then
23:             $\phi_j(c'_i) \leftarrow \frac{(end-start)\cdot(g_j(c'_i)-q_j)-sum\_in\_window}{(p_j-q_j)\cdot(n-1)}$
24:         end if
25:     end for
26:     return $\phi_j(c'_i)$
27: end procedure
```

security policies and exhibit trustworthy behavior. To balance multiple criteria, we start with equal weights for each criterion.

Lastly, the final ranking of the clients is determined based on the net outranking flow $\phi(c_i)$ using Equation 5.8.

$$\phi(c_i) = \phi^+(c_i) - \phi^-(c_i)$$

This net flow score reflects the overall preference of c_i compared to all other clients. A high net score implies that, on average, client c_i is preferable when compared to other clients in the system, making it an optimal candidate for the task.

The clients are then ranked based on their net flow scores, and the top K clients can be selected directly for the FL task. For static environments where client attributes might not change, the top $x\%$ clients can be selected to participate in the subsequent training rounds, further sub-sampling either randomly or subject to budget constraints.

Algorithm 3 Optimized Multi-Criterion Client Selection Using SecureFedPROM

Input:

- Set of clients $C = \{c_1, c_2, \ldots, c_n\}$ with attributes $g_j(c_i)$ for $j \in \{H, N, Q, T\}$
- Weights for each criterion W_j
- Thresholds q_j and p_j for each criterion j
- Budget B

Output:

- Selected client set S

```
1: procedure FedPROM_Optimized($C$, $W$, $q$, $p$, $B$)
2:     Initialize $S \leftarrow \emptyset$
3:     Initialize $AvailableBudget \leftarrow B$
4:     Initialize flow scores $\phi^+(c_i) \leftarrow 0$ and $\phi^-(c_i) \leftarrow 0$ for all $c_i \in C$
                                        ▷ Compute Uni-Criterion Flows for Each Criterion
5:     for each criterion $j \in \{H, N, Q, T\}$ do
6:         $(\phi_j^+, \phi_j^-) \leftarrow$ ComputeUniCriterionFlows($C$, $g_j(c_i)$, $q_j$, $p_j$)
7:         for each $c_i \in C$ do
8:             $\phi^+(c_i) \leftarrow \phi^+(c_i) + W_j \cdot \phi_j^+(c_i)$
9:             $\phi^-(c_i) \leftarrow \phi^-(c_i) + W_j \cdot \phi_j^-(c_i)$
10:        end for
11:    end for
                                        ▷ Calculate Net Outranking Flow for Each Client
12:    for each $c_i \in C$ do
13:        $\phi(c_i) \leftarrow \phi^+(c_i) - \phi^-(c_i)$
14:    end for
                                        ▷ Rank Clients and Select Within Budget
15:    Sort $C$ in descending order based on $\phi(c_i)$
16:    for each $c_i \in sorted(C)$ do
17:        if Cost($c_i$) $\leq AvailableBudget$ then
18:            Add $c_i$ to $S$
19:            $AvailableBudget \leftarrow AvailableBudget -$ Cost($c_i$)
20:        end if
21:        if $AvailableBudget \leq 0$ then
22:            break
23:        end if
24:    end for
25:    return $S$
26: end procedure
```

In dynamic environments where client attributes such as trustworthiness or data quality may change over time or a new client joins, the partial ranking for the clients can be efficiently updated with only localized recalculations for each criterion.

Algorithm 3 presents the complete pseudocode of SecureFedPROM. By transforming the problem into a ranking task, we essentially sidestep the combinatorial complexity inherent in direct subset selection and address the challenges of dual-objective optimization efficiently.

5.7.1 Complexity Analysis

SecureFedPROM leverages a sorting-based PROMETHEE approach to achieve efficient multi-criterion client selection. In the worst case, the computational complexity of our proposed solution involves:

- **Uni-Criterion Flow Computation:** For each criterion, clients are sorted by their attribute scores, resulting in $O(n \log n)$ complexity. The sliding window mechanism then computes flow scores in $O(n)$ for each criterion. With m criteria, the overall complexity for uni-criterion flow computation is $O(m \cdot n \log n)$.

- **Net Outranking Flow Aggregation:** After computing the uni-criterion flows, weighted sums are applied to aggregate these into net flow scores for each client, incurring a complexity of $O(m \cdot n)$.

- **Client Ranking and Selection:** Finally, clients are ranked by sorting the net flow scores, which adds a complexity of $O(n \log n)$.

Thus, the overall complexity of SecureFedPROM is dominated by the uni-criterion flow computation, which remains $O(m \cdot n \log n)$ in the worst case. This efficiency is ideal for dynamic FL environments where clients frequently join or leave the system, as the proposed approach can update the client ranking incrementally, without recomputing the entire preference matrix. Furthermore, the overhead introduced by the SecureFedPROM mainly revolves around preference flow computations and is conveniently handled at the FL server. This balance of computational efficiency and accurate client selection makes SecureFedPROM ideal for large-scale, resource-constrained, and highly dynamic FL environments.

Table 5.1: Hardware Configuration of Simulated Environment

	Low-End	**Mid-Range**	**High-End**	**Excellent**
CPU Count	1 (Single Core)	1 (Dual Core)	2 (Quad Core)	4 (Octa Core)
Cores	1	2	4	8
Frequency (GHz)	1.2	2.5	3.5	3.5+
CPU Utilization (%)	43-100 (High Load)	25-49 (Moderate Load)	12-35 (Low Load)	6-36 (Variable Load)
GPU	Absent	Absent	Absent	Present
RAM (GB)	1-2	2-4	4-8	8-64
Available RAM (GB)	0.5-1 (Minimal)	1-3 (Moderate)	3-7 (Ample)	5-64 (Extensive)
Storage (GB)	1-4 (Minimal)	4-8 (Moderate)	8-64 (Ample)	32-64 (Extensive)

Table 5.2: Network Configuration of Simulated Environment

	Poor	**Average**	**Good**	**Excellent**
Bandwidth (Mb/s)	1-4	4-10	10-100	100-1000
Latency (ms)	20-100	20-80	5-50	1-10

5.8 Results and Discussions

5.8.1 Experimental Setup

This study's experimental design meticulously simulates both statistical and system heterogeneity to evaluate the effectiveness of our multi-criterion client selection strategy under realistic conditions. Clients are categorized based on a spectrum of hardware specifications and network conditions, mirroring the diverse device and connectivity environments encountered in typical FL scenarios. Table 5.1 and Table 5.2 summarize the simulated environment's hardware and network configurations.

All the simulated clients were run on a workstation equipped with a 12th Gen Intel(R) Core(TM) i7-12700K CPU clocking at 3.60 GHz, and 64GB of RAM. Our simulations employ heuristic-based estimates for network delays and training times to account for realistic operational delays. The network delay $T_{network_delay}$ accounts for the time required to download and upload the model and is calculated as:

$$T_{\text{network\_delay}} = \frac{\text{Size of Data Transferred}}{\text{Bandwidth}} + \text{Latency} \tag{5.12}$$

For model training time ($T_{training}$), the heuristic considers computational workload and system capabilities:

$$T_{training} = \frac{comp}{\text{Processing Capacity}} \times \text{Memory Impact Factor} \tag{5.13}$$

Here, *comp* represents the total floating point operations required for the training, adjusted to the system's available processing capacity and the available memory ratio. These heuristics help in accounting for the varying capabilities of devices and network conditions and provide a more realistic simulation of FL environments.

Utilizing benchmark ML datasets, we distribute data among simulated clients under non-IID settings, thereby simulating statistical heterogeneity inherent in real-world settings. Most datasets used in this study are sourced from the LEAF benchmarks [269], which provides a suite of standardized datasets and partitioning strategies designed to evaluate the performance and scalability of FL algorithms under varied conditions. Specifically, we use the following datasets:

1. **Federated Extended MNIST (FEMNIST)** is the federated version of Extended MNIST (EMNIST) [270]. Unlike standard EMNIST, FEMNIST partitions data based on individual writers of the digit/character, creating a highly non-IID distribution where each client has data from only a few users.

2. **CelebA** which partitions the Large-scale CelebFaces Attributes Dataset [271] by individual celebrities.

3. **Reddit**, a federated text dataset containing user-generated posts and comments from the Reddit platform. This dataset is partitioned based on users posting on Reddit and exhibits strong user-specific non-IID characteristics.

4. **Synthetic**, a modified version of the synthetic dataset presented in [272].

5. **CIFAR10** is the only dataset used outside of LEAF to study the non-IID label distribution, where each client received a random subset of 50 to 1,000 samples.

Table 5.3 summarizes the key statistics of the datasets used in this study. We trained the FL model on these datasets using the model architectures defined in LEAF, with TensorFlow employed to build and execute the experimental setup efficiently.

- In our experiments using the FEMNIST dataset, we subsample 10% of the data and use a model with two convolutional layers followed by pooling and a final dense layer with 2048 units. We used a learning rate of 0.004 to train 20 clients per round using a single local epoch for 50 rounds.

- For CelebA, we again subsample 10% of the data and use the same model as described above for FEMNIST. However, we used a small batch size of 5 and trained 20 clients for 50 rounds using a learning rate of 0.001.

- For the Reddit dataset, we train a stacked LSTM model with a batch size of 5 and a learning rate of 5.65, selecting 20 clients per round for 50 training rounds.

- For the experiments with the Synthetic dataset, as described in the LEAF benchmark, we use 1,000 devices and train a perceptron model with sigmoid activations. We train this model over 20 clients per round, each training the model for a single local epoch with a batch size of 5 and a learning rate of 0.1.

- For the CIFAR 10 dataset, we use 255 devices and train a modified version of the pre-trained ResNet-50 model, adjusted with an adaptive average pooling layer and a final fully connected layer outputting ten classes. In each training round, we select 20 devices, each performing 5 local epochs using a learning rate of 0.01.

We compared the achieved accuracy with other state-of-the-art client selection strategies, such as Active selection, Power of Choice (pow-d), Greedy, Resource aware, and Price first. In the following section, we will discuss the results derived from this setup and the implications of our proposed multi-criterion client selection method.

Table 5.3: Statistics of Experimental datasets

Dataset	No. of devices	Total samples	Samples per User		
			mean	std	Skewness
FEMNIST	3,597	817,851	227.37	88.8	0.74
CelebA	9,343	200,288	21.44	7.63	-0.54
Reddit	1660912	56588034	34.07	63.1	1.85
Synthetic	1,000	107,553	107.55	213.22	3.10
CIFAR 10	255	50000	196.07	99.07	-0.12

5.8.2 Evaluation and Results

To evaluate our strategy's effectiveness, we compared its performance against the standard Federated Averaging (FedAvg) algorithm and other state-of-the-art client selection strategies. Here, we provide an in-depth overview of each strategy and the results obtained.

- **Random Selection**: Based on the study [8], this strategy randomly chooses a predefined number of clients without considering specific client attributes or performance characteristics.

- **Active Selection**: This strategy selects clients primarily based on their loss metrics, with a fraction chosen at random for diversity [273].

- **Power of Choice**: This method selects clients by prioritizing those with the highest validation loss, assuming they will contribute the most to global model updates [143].

- **Greedy**: This client selection protocol selects clients based on the ratio of the number of training samples to cost, preferring clients with more samples per cost unit.

- **Resource aware**: We modify the technique presented in [56]. This client selection protocol selects clients based on their computation and communication information with the aim of minimizing the training time.

- **Price first**: It prioritizes the clients with low prices for a given learning task, aiming to select as many clients as possible within the limited budget.
- **FedPROM**: This is our proposed multi-criteria client selection strategy, which considers client-specific hardware capabilities, network conditions, and data quality to ensure optimal client selection.

We thoroughly investigated the performance of each client selection method across a spectrum of evaluation metrics, providing a comprehensive understanding of their impacts on the FL process. A key consideration in our assessment was the efficiency of these methods, which were evaluated by measuring key factors such as the time required to reach specific accuracy level benchmarks.

5.8.3 Performance Evaluation and Discussion

To thoroughly evaluate the performance of these strategies, we considered the following key factors:

1. **Accuracy vs. Number of Rounds**: This metric denotes the accuracy achieved at different training stages, emphasising a strategy's effectiveness as training progresses through multiple rounds.
2. **Time of Arrival at a Desired Accuracy (ToA@x)**: This measure provides a snapshot of efficiency, documenting the time taken to reach a certain accuracy level. The earlier a method reaches the benchmark, the better.
3. **Performance after the final deadline**: This captures the performance of each strategy in terms of accuracy, after 50 rounds of training, providing an overview of long-term effectiveness.

Table 5.4: Results obtained for FEMNIST, CelebA, and Synthetic dataset over non-IID data as described in Table 5.3. ToA@x represents the time (in seconds) required to arrive at a testing classification accuracy of x (the earlier the better). NaN means that the method did not achieve the testing classification accuracy of x in the given 50 rounds of training.

Method	**FEMNIST**				
	ToA@50%	**ToA@60%**	**ToA@70%**	**Training time**	**Final accuracy**
Random selection	536.26s	1523.79s	3775.06s	7930.84s	75.55%
Active selection	2365.46s	2894.20s	4298.21s	4380.21s	70.20%
Power of Choice	312.09s	408.95s	602.3s	4500.28s	61.19%
Greedy selection	718.06s	1023.50s	1938.95s	5097.28s	75.55%
Resource-aware	138.24s	NaN	NaN	432.024s	35.64%
Price-first	489.73s	628.74s	1045.76s	3471.94s	60.13%
FedPROM	56.41s	81.79s	145.15s	627.96s	80.60%
Method	**CelebA**				
	ToA@50%	**ToA@60%**	**ToA@70%**	**Training time**	**Final accuracy**
Random selection	37.87s	95.41s	318.80s	2480.71s	77.09%
Active selection	50.00s	101.38s	419.39s	2599.72s	73.69%
Power of Choice	57.28s	57.28s	NaN	2555.97s	67.98%
Greedy selection	59.32s	177.97s	NaN	2966.18s	62.53%
Resource-aware	15.18s	45.54s	NaN	75.90s	55.92%
Price-first	48.89s	48.89s	NaN	2444.63s	61.10%
FedPROM	15.35s	15.35s	115.91s	818.28s	84.33%
Method	**Synthetic**				
	ToA@50%	**ToA@60%**	**ToA@70%**	**Training time**	**Final accuracy**
Random selection	11.32s	80.05s	244.68s	549.32s	76.79%
Active selection	153.33s	311.09s	NaN	572.46s	63.39%
Power of Choice	9.59s	220.31s	NaN	564.13s	61.88%
Greedy selection	12.04s	NaN	NaN	602.00s	49.11%
Resource-aware	1.28s	NaN	NaN	64.21s	50.74%
Price-first	308.34s	NaN	NaN	592.98s	50.49%
FedPROM	5.85s	39.55s	93.32s	371.52s	81.01

Performance comparison

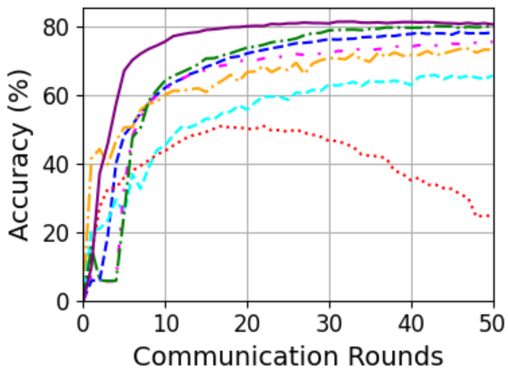
With the configurations mentioned earlier, we then train the model and compare the performance of FedPROM with the benchmark client selection protocols. Particularly, we emulate an FL environment for each learning task, where the features of the candidate clients follow the description in Section 5.8.1. Following the standard, we only select a fraction of clients to participate in each training iteration. Figure 5.2 shows the accuracy of each learning task using different client selection strategies. We can observe that for all learning tasks and FL settings, our proposed strategy, FedPROM, outperforms other state-of-the-art client selection strategies. Specifically, in the FEMNIST and CelebA datasets, FedPROM exhibited a swift convergence to peak accuracy levels, outperforming other client selection strategies that displayed either a slower improvement rate or a lack of stability in convergence. This trend continued in the analysis of the Synthetic and CIFAR 10 datasets, where the FedPROM protocol not only achieved higher final accuracy quickly but also maintained a lead throughout the communication rounds. This indicates that our proposed protocol may reduce the frequency of data transmissions necessary to achieve high accuracy, thereby suggesting a potential reduction in communication overhead and energy consumption.

However, the Reddit dataset (Figure 5.2e) presents a unique challenge due to the sparse and highly skewed data distribution across clients. Here, FedPROM shows a modest improvement in accuracy over other strategies, which indicates that while our method is advantageous, the complexity of text-based, imbalanced data requires additional strategies to enhance performance significantly.

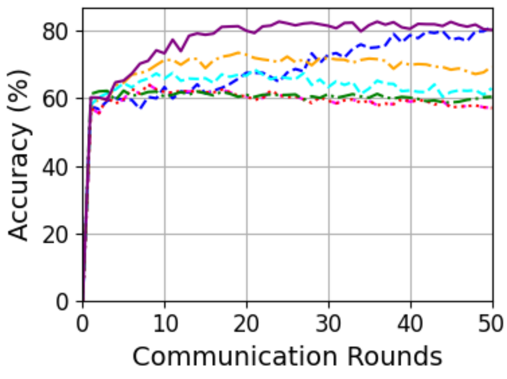
Time of Arrival at a Desired Accuracy (ToA@x)

We observed the changes in the accuracy on testing datasets over time and analyzed when the accuracy reached a certain level for the first time. Specifically, we report Time of Arrival at a Desired Accuracy (ToA@x) for FEMNIST, CelebA, and Synthetic datasets under non-IID data settings for 50%, 60%, and 70% accuracy levels. We summarize these results in Table 5.4.

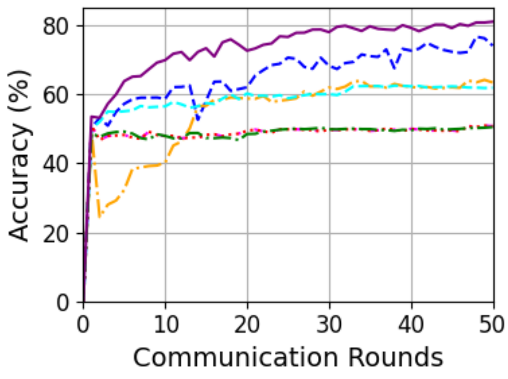
We can observe that FedPROM consistently outperforms the existing client selection strategies regarding both time and rounds required for convergence. For the FEMNIST dataset, FedPROM required only 56.41 seconds and 4 rounds to reach a 50% accuracy level, demonstrating a substantial efficiency gain over other methods. Specifically, it showed an 86.73% reduction in time and a 33.33% reduction in rounds required compared to Random selection, an 81.93% reduction in time and a 50% reduction in rounds compared to Power of Choice, and a 92.13% reduction in time and a 42.86% reduction in rounds over Greedy selection. Similarly, for the CelebA and Synthetic datasets, Fed-



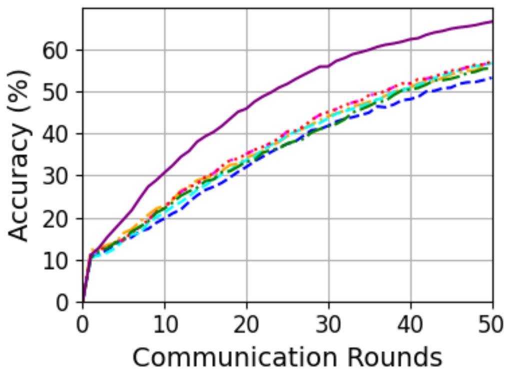
(a) FEMNIST



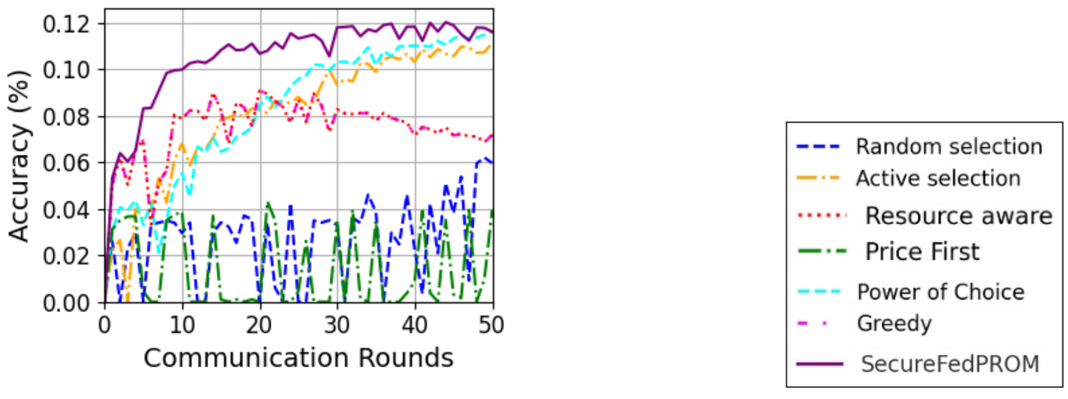
(b) CelebA



(c) Synthetic



(d) CIFAR 10



(e) Reddit

Figure 5.2: Performance comparison of client selection strategies with 20 candidate clients.

PROM achieves 50% accuracy in just 15.35 and 5.85 seconds, demonstrating a 73.20% and 39% reduction compared to Power of Choice and a stark contrast to Greedy and Active selection strategies, which did not reach the same efficiency levels within the specified 50

Table 5.5: Features of selected clients across different client selection strategies when the budget is set to 50.

Client Selection Strategy	Random	Active	Power of Choice	Greedy	Resource Aware	Price First	FedPROM
Avg. no. of clients selected	38	26	32	45	6	50	16
Avg. data size per client	227.15	300	243	242	169	146	294
No. of unique training samples	81118	44023	45918	16786	15148	15423	37775
Final accuracy	66.95%	72.01%	66.03%	69.44%	38.29%	44.79%	75.63%

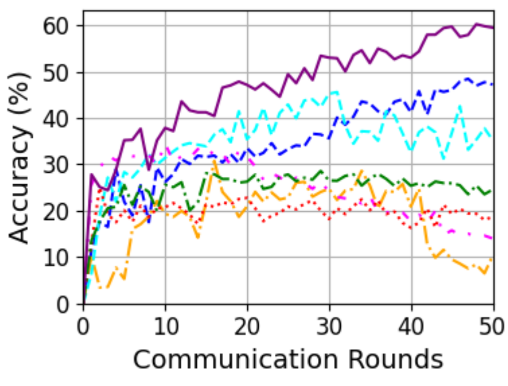
rounds of training.

Moreover, Table 5.5 compares various client selection strategies using different metrics for each strategy, highlighting how each approach performs under a fixed budget scenario. The finding demonstrates how FedPROM outperforms other client selection strategies in terms of final accuracy. This experiment also indicates that despite selecting fewer clients (only 16 on average) and having a relatively high average data size per client (294), FedPROM is more effective in leveraging client data for training, leading to better model performance than the other strategies listed.

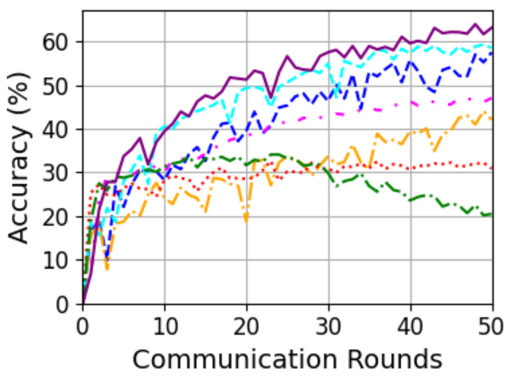
Impact of Budget

In this section, we systematically evaluate the impact of budget constraints on final model accuracy across various client selection strategies. To accurately assess this impact, we first establish a cost model. This model quantifies the computational and communication overhead associated with each client's participation in a learning round and defines cost as a function of resources utilized by the client and the volume of data transmitted during model updates. These metrics capture the resource expenditure of the FL process and are critical for evaluating the efficiency of client selection strategies under varying budget conditions. We varied the budget parameter from 5 to 50 and trained the model to explore its impact. Taking FEMNIST as an example, we summarize these results in Figure 5.3.

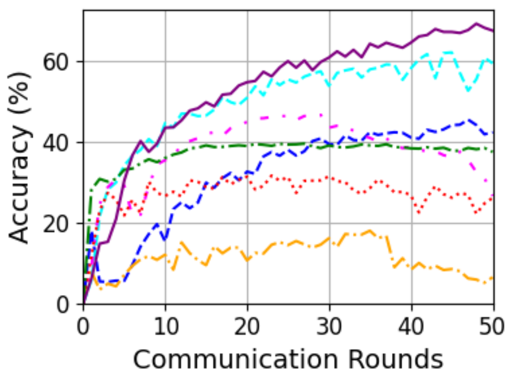
The findings indicate a general trend: increasing the budget correlates with improved accuracy across all client selection strategies. However, the magnitude of this improvement and the efficiency of budget utilization vary significantly among the strategies. With the most stringent budgets, all strategies perform comparably, struggling to achieve even



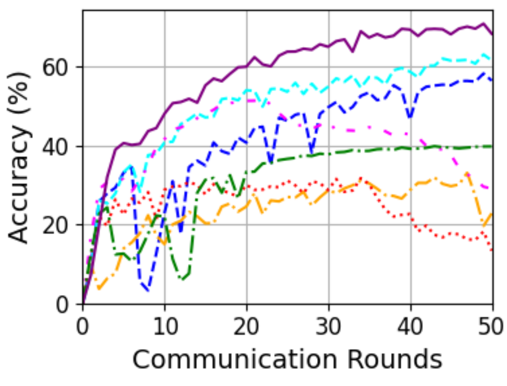
(a) Budget=5



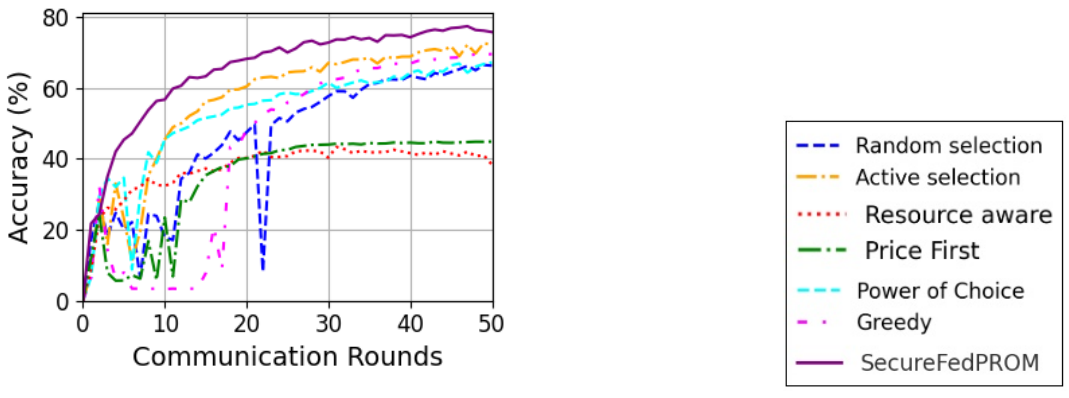
(b) Budget=10



(c) Budget=15



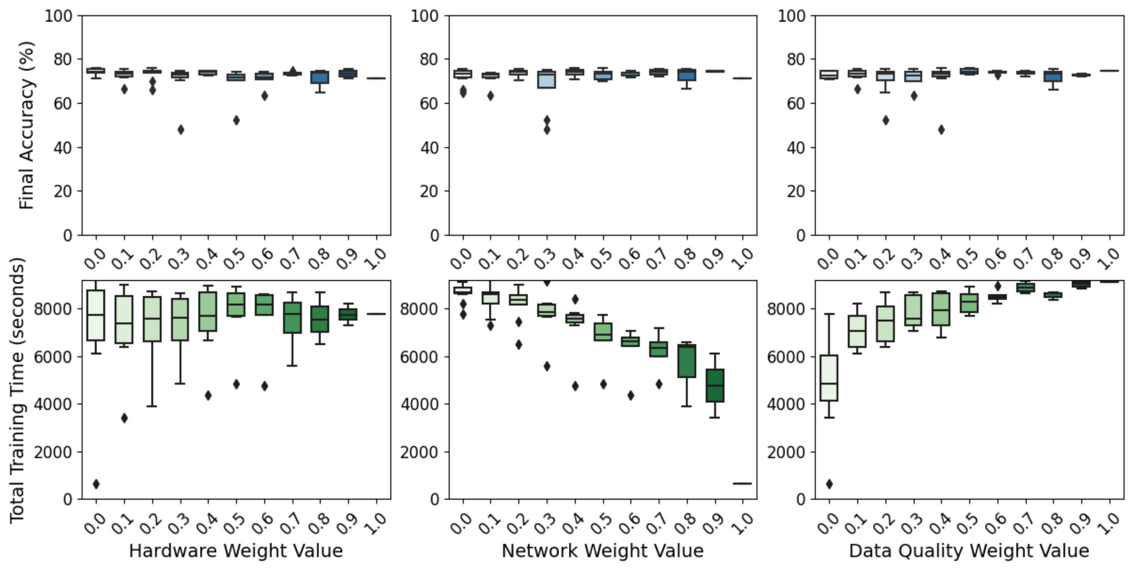
(d) Budget=20



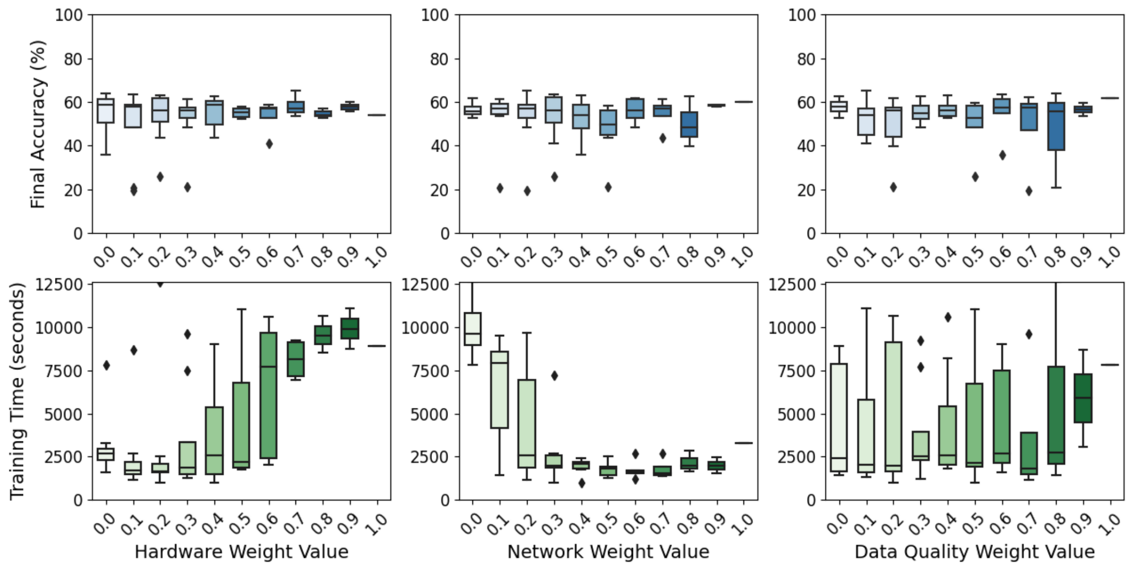
(e) Budget=50

Figure 5.3: Impact of varying budgets on performance

moderate accuracy levels after 50 communication rounds. This implies that overly restrictive budget constraints hamper the learning capability of federated systems. However, it can be seen that FedPROM maintains a relatively stable performance and offers a slight edge compared to other client selection strategies. As the budget increases to 10, strategies like FedPROM and Power of Choice selection begin to differentiate themselves, achieving higher accuracy than others. FedPROM, in particular, shows substantial gains, indicating its effective utilization of additional resources.



(a) FEMNIST



(b) CIFAR10

Figure 5.4: Impact of criteria weighting on global model accuracy and training time

The separation between strategies becomes more explicit at a moderate budget of 15 and 20. FedPROM continues to lead, surpassing the 60% accuracy mark rapidly compared to others. Notably, Power of Choice and Greedy selection also benefit from the increased budget, showing improved performance, yet they do not match the efficiency of FedPROM. Resource-aware and Price-first strategies demonstrate moderate improvements but remain less efficient in utilizing the increased budget. As the budget increases, FedPROM exhibits diminishing returns on accuracy gain, although it still maintains the lead and offers more stable performance. This pattern suggests an optimal budget range for FedPROM, beyond which resource allocation does not directly translate into proportional accuracy improvements; however, it stabilizes the model. Conversely, strategies like Random selection and Active selection show continued, albeit modest, improvements, indicating that they may benefit from larger client pools.

Impact of Criteria Weighting

In FL, client selection significantly influences both training time and model accuracy. Intuitively, hardware and network capabilities could be presumed to significantly influence training time, given their direct relationship with computational efficiency. Meanwhile, data quality is directly tied to model performance, assuming that higher-quality data would boost the model's performance.

In this section, we further aim to quantify the extent of impact each criterion exerts on the FL process. By manipulating the weights of these criteria from 0 to 1, while ensuring their cumulative weight remained constant at 1, we aim to observe and analyze the sensitivity of the training duration and final model accuracy to these factors. We present these results in Figure 5.4.

Using FEMNIST and CIFAR10 as examples, the empirical results reveal distinct sensitivities to the criteria under consideration. When considering hardware weight, the CIFAR10 dataset showed a decreased training time with increased weight. The FEMNIST dataset exhibited a similar trend but more significant reductions in training time, which could be attributed to the dataset's relative simplicity. Yet, this improvement was not mirrored in terms of accuracy, which suggests that while better hardware accelerates computation, it does not inherently enhance the model's learning ability for more complex tasks. We observed a similar trend with the network weight; training times decreased as we increased the network weights. However, it plateaus at higher levels, indicating an optimal network capacity for efficiency gains.

The data quality criterion, however, shows a positive correlation with both the model performance and training efficiency. We observed that increasing the data quality weights translates into accuracy gains across both datasets, thereby confirming the hypothesis that data quality is paramount for model performance. Notably, training time also diminished with higher data quality weights, illustrating that when models train with higher-quality datasets, they converge more rapidly and with greater fidelity.

One of the most noteworthy observations is that no single weight combination universally optimized both model accuracy and training time across all datasets. This suggests that the ideal weight distribution is highly context-dependent. However, it's worth noting that the most significant improvements in accuracy and training time reduction were generally observed when the weights were more balanced across hardware, network, and data quality criteria. Imbalanced weight distributions often led to either high accuracy at the expense of longer training times or vice versa.

An additional observation worth noting is the uniformity of the impact across all three criteria (hardware, network, and data quality) for each dataset. Specifically, the percentage improvements in accuracy and reductions in training time were remarkably consistent when varying any of these weights. For instance, in the FEMNIST dataset, the sensitivity to all three criteria was almost identical, indicating that the FEMNIST dataset is generally less sensitive to the specific criteria and may perform well with various client selection strategies. In contrast, the CIFAR10 dataset showed the most variability in achieving high accuracy and an overall reduction in training time. This suggests that CIFAR10, which is a more complex image classification task, could be more sensitive to the quality of the data and might require a more thoughtful selection of client devices for optimal performance.

This uniformity or variability across criteria can offer insights into the robustness of an FL model against different environmental factors. For simpler tasks like FEMNIST, the model may be more forgiving, allowing for more flexible client selection. However, a more balanced and cautious approach to weight distribution may be crucial for achieving high accuracy and reasonable training times for complex tasks. In the context of FedPROM, these experiments reveal that not only does it offer a more balanced approach to client selection, but it also provides the flexibility to set weight preferences for any criteria, allowing for a tailored optimization strategy based on the task and resource constraints, which is beneficial for its practical usage in various FL scenarios.

5.9 Key Findings

The findings from this chapter demonstrate the effectiveness of a multi-criteria client selection framework in FL environments. Our approach, FedPROM, addresses key challenges associated with client selection by balancing computational power, network conditions, and data quality, resulting in improved system performance and fairness. Optimizing client selection based solely on a single criterion, such as computational resources or data quality, tends to introduce biases that degrade model accuracy and reduce the diversity of the training data. For example, selection strategies focusing on computational capabilities often prioritize high-performing devices, but at the cost of neglecting clients with more diverse or higher-quality data. Similarly, prioritizing data quality alone can lead to longer training times and inefficient resource usage, as clients with better data may not have the necessary computational power to process it quickly.

Our experiments demonstrated that FedPROM significantly reduces training time while maintaining or improving model accuracy. For instance, when applied to the FEMNIST dataset, FedPROM achieved 80.60% accuracy with a training time of just 627.96 seconds, outperforming traditional strategies like Random Selection and Power of Choice. Furthermore, by using a combination of criteria, our method ensures fairer client participation, which leads to better generalization of the global model. This multi-criteria approach also showed resilience against client heterogeneity, ensuring stable convergence rates even under non-IID data distributions, which is a common challenge in FL.

Moreover, our findings suggest that optimizing both performance and training time helps not only in reducing communication overhead but also in ensuring a more equitable distribution of computational load across clients. This is crucial in practical scenarios where devices have limited resources and energy, such as mobile phones or IoT sensors. The adaptability of our client selection framework makes it well-suited for diverse and dynamic environments.

5.10 Summary

This chapter introduced our initial contribution, a multi-criteria client selection framework aimed at optimizing both performance and training time. The proposed approach addresses the fundamental challenges in client selection, such as managing client heterogeneity, reducing communication overhead, and ensuring fairness in client participation. By framing the client selection task as a dual-objective optimization problem, FedPROM balances multiple factors—computational resources, data quality, and network conditions—resulting in improved global model performance and faster convergence.

We demonstrated that FedPROM outperforms traditional single-criterion methods in various experiments, particularly by reducing the time to achieve target accuracies while maintaining high levels of model accuracy. This framework integrates energy and budget constraints, making it a practical solution for resource-constrained environments, such as IoT networks and mobile devices.

Our contributions in this chapter mark a significant step forward in improving the scalability and efficiency of FL systems. By leveraging a multi-criteria approach, FedPROM offers a balanced and flexible solution for selecting clients in diverse, real-world FL scenarios. The next chapter will build on this work by exploring further optimizations in the client selection process, including the incorporation of dynamic client availability and adaptive learning rates, to further enhance the efficiency and effectiveness of FL systems.

Chapter 6

Bayesian Federated Learning using Stochastic Variational Inference

6.1 Introduction

In FL, one of the core challenges is to address the inherent uncertainty and heterogeneity across distributed clients. Traditional FL methods, such as Federated Averaging (FedAvg), treat model updates as deterministic, without accounting for the uncertainty in data distributions or the variances in model updates across clients. This approach often leads to suboptimal performance, particularly in scenarios where data distributions are non-IID or when clients have vastly different amounts of data.

Bayesian Federated Learning (BFL) offers a more flexible and robust alternative by modeling the FL process probabilistically. By formulating the learning task within a Bayesian framework, we can better capture the uncertainty in client updates and make more informed decisions during the aggregation process. This not only helps in dealing with heterogeneous data but also improves model generalization. Through the use of Stochastic Variational Inference (SVI), BFL introduces a scalable approach to approximate posterior distributions at the client side, ensuring that the communication and computational overheads remain manageable.

This chapter explores how Bayesian principles can be effectively integrated into the FL framework, offering a structured method for dealing with the variability in client updates and enhancing model performance under challenging conditions.

6.2 Objectives

The primary objective of this chapter is to introduce and optimize a Bayesian framework for FL that addresses the limitations of traditional deterministic methods like FedAvg. By modeling the uncertainty in client updates through a Bayesian inference approach, we aim to enhance the robustness and generalization of the global model. Specifically, this chapter seeks to leverage SVI to approximate posterior distributions at the client level, enabling efficient handling of non-IID data while maintaining scalability. Additionally, we aim to improve model convergence by incorporating uncertainty into the aggregation process. The ultimate goal is to empirically validate the superior performance of Bayesian approaches compared to traditional aggregation methods.

6.3 Contribution

Our contribution in this chapter is threefold:

- We introduce BayFL-SVI, a novel Bayesian federated learning framework that integrates SVI for scalable and efficient Bayesian inference.
- We provide a rigorous theoretical analysis of our approach, including convergence guarantees, highlighting the effectiveness of the proposed method.
- We present comprehensive empirical results demonstrating significant improvements in convergence rates and model accuracy compared to traditional aggregation protocols in FL.

6.4 Proposed Protocol

This section introduces our proposed protocols, BayFL-SVI, detailing its components and the methodology employed to address the challenges of data variability and efficient aggregation in federated learning. By formulating the FL process as a Bayesian inference problem, our approach allows for integrating prior knowledge and managing uncertainty in model updates. The framework comprises the client-side local training process and server-side hierarchical Bayesian aggregation.

6.4.1 Bayesian Formulation

In a Bayesian framework, model parameters θ are treated as random variables with an associated prior distribution $p(\theta)$. The objective is to update the distribution of these parameters based on the posterior distributions received from clients. For each client i, we denote its local dataset as D_i, and the likelihood of the data given the parameters θ is represented as $p(D_i|\theta)$.

The posterior distribution of the model parameters after observing the data from client i can be determined using Bayes' theorem:

$$p(\theta|D_i) = \frac{p(D_i|\theta)p(\theta)}{p(D_i)} \tag{6.1}$$

where:

- $p(\theta)$ is the prior distribution of the model parameters.
- $p(D_i|\theta)$ is the likelihood of the data given the model parameters.
- $p(D_i)$ is the marginal likelihood or evidence, which is a normalization constant.

In FL, directly computing the posterior $p(\theta|D)$ where $D = \bigcup_i D_i$ is infeasible due to its decentralized nature. Instead, we approximate this posterior using local posterior distributions from each client, $q_i(\theta)$.

6.4.2 Client-Side Training with SVI

Each client in the BayFL-SVI framework performs local training using SVI to approximate the posterior distributions of their model parameters. In variational inference, we approximate the true posterior $p(\theta \mid D_i)$ with a simpler distribution $q_i(\theta \mid \phi_i)$, parameterized by ϕ_i (which includes the mean μ_i and variance σ_i^2). The goal is to minimize the Kullback-Leibler (KL) divergence between $q_i(\theta \mid \phi_i)$ and the true posterior $p(\theta \mid D_i)$. This is equivalent to maximizing the Evidence Lower Bound (ELBO):

$$\mathcal{L}(\phi_i) = \mathbb{E}_{q_i(\theta|\phi_i)}[\log p(D_i \mid \theta)] - \mathrm{KL}(q_i(\theta \mid \phi_i) \| p(\theta)) \tag{6.2}$$

where:

- $\mathbb{E}_{q_i(\theta|\phi_i)}[\log p(D_i \mid \theta)]$ is the expected log-likelihood of the data under the variational distribution.
- $\mathrm{KL}(q_i(\theta \mid \phi_i) \| p(\theta))$ is the Kullback-Leibler divergence between the variational distribution $q_i(\theta \mid \phi_i)$ and the prior $p(\theta)$.

The ELBO balances model fit to the data (first term) and model complexity (second term). Clients optimize the ELBO using stochastic gradient descent. For each mini-batch of local data, the client updates its variational parameters to maximize the ELBO, ensuring that the variational distribution $q_i(\theta \mid \phi_i)$ closely approximates the true posterior. The steps involved are as follows:

1. **Initialization**: Each client receives the initial global model parameters θ_0 from the server and initializes its local variational parameters ϕ_i including μ_i and σ_i^2.
2. **Mini-batch Processing**: Clients process their local data D_i in mini-batches. For each mini-batch, the client computes the ELBO $\mathcal{L}(\phi_i)$ using the current variational parameters as given in Equation 6.2.

3. **Gradient Updates**: The client performs stochastic gradient descent to maximize the ELBO $\mathcal{L}(\phi_i)$, updating the variational parameters ϕ_i to better approximate the true posterior $p(\theta|D_i)$. The update rule can be simply described as:

$$\phi_i \leftarrow \phi_i + \eta \nabla_{\phi_i} \mathcal{L}_i(\phi_i) \tag{6.3}$$

   Here, η is the learning rate.

4. **Local Model Parameter Update**: After processing all mini-batches, the client updates its local model parameters θ_i based on the optimized variational parameters ϕ_i. The optimized variational parameters ϕ_i, which now represent the local posterior distribution, are then sent to the server.

This iterative update ensures that the variational approximation $q_i(\theta \mid \phi_i)$ improves incrementally with each client update.

6.4.3 Server-Side Aggregation with Hierarchical Bayesian Modeling

After clients perform local training using SVI and send their local posterior distributions to the server, the server aggregates them to update the global model parameters. This is done using a Hierarchical Bayesian Modeling approach, which allows us to capture both global trends and client-specific variability in a principled way.

Hierarchical modeling leverages a two-level structure: client-level posteriors are informed by local data, while the global model aggregates these with shared priors, capturing uncertainty at both levels. This is especially useful in non-IID settings, where data distributions differ significantly across clients. Other Bayesian aggregation methods, such as naive averaging of posteriors or simple Bayesian ensembling, lack this expressiveness and tend to underperform in heterogeneous environments. The hierarchical approach offers a more flexible and theoretically grounded way to account for data diversity while maintaining robustness in the aggregation process.

Hierarchical Bayesian Aggregation

In the Hierarchical Bayesian framework, we treat the global model parameters θ as having a prior distribution, and each client's local posterior distribution is used to update this prior to form a global posterior. Let's denote:

- μ_i and σ_i^2 as the mean and variance of the local posterior distribution $q_i(\theta)$ from client i.
- μ_0 and σ_0^2 as the prior mean and variance of the global model parameters.
- N as the total number of clients.

1. **Global Prior and Hyperprior**: Assume a global prior distribution for the model parameters θ:

$$p(\theta \mid \lambda) = \mathcal{N}(\theta \mid \mu_0, \sigma_0^2) \tag{6.4}$$

   where $\lambda = (\mu_0, \sigma_0^2)$ are the hyperparameters of the global prior.

2. **Client Variational Distributions**: Each client i computes a local variational distribution $q_i(\theta \mid \phi_i)$ to approximate the posterior distribution. If the variational distribution is Gaussian:

$$q_i(\theta \mid \phi_i) = \mathcal{N}(\theta \mid \mu_i, \sigma_i^2) \tag{6.5}$$

   Here, $\phi_i = (\mu_i, \sigma_i^2)$ are the variational parameters of client i, the result of the client-side training using SVI, as described in the previous Section 6.4.2.

3. **Aggregating Client Distributions**: To aggregate the variational parameters from clients, we utilize a Bayesian update. Assume that the global posterior $q(\theta)$ is also Gaussian:

$$q(\theta \mid \phi) = \mathcal{N}(\theta \mid \mu, \sigma^2) \tag{6.6}$$

where $\phi = (\mu, \sigma^2)$ are the global variational parameters to be determined.

4. **Update Rules for Global Variational Parameters**:

   To update the global variational parameters, we use the following rules:

$$\frac{1}{\sigma_g^2} = \frac{1}{\sigma_0^2} + \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \tag{6.7}$$

$$\mu_g = \sigma^2 \left(\frac{\mu_0}{\sigma_0^2} + \sum_{i=1}^{N} \frac{\mu_i}{\sigma_i^2} \right) \tag{6.8}$$

   These update rules ensure that the global posterior incorporates information from both the prior and the clients' updates. Once aggregated, the server broadcasts the updated global parameters to all clients for the next round of local training.

By adopting this hierarchical Bayesian approach, the server can effectively combine the local posterior distributions from the clients, taking into account their variability and uncertainty, to construct a robust global model. This method ensures that the global model leverages the diverse data distributions across clients, offering a principled approach to aggregate updates in Federated Learning. The theoretical analysis and proof of convergence are thoroughly explained in the next section.

6.5 Theoretical Convergence Analysis

To demonstrate the convergence of the proposed framework, BayFL-SVI, we show that the aggregated posterior $q(\theta)$ converges in distribution to the true posterior $p(\theta \mid D)$ as the number of clients increases.

6.5.1 Assumptions

To establish the convergence performance of the aggregated model, we adopted the following assumptions.

1. **Smoothness:** The objective function $f(\theta)$ is smooth, i.e., it has Lipschitz continuous gradients:

$$\|\nabla f(\theta_1) - \nabla f(\theta_2)\| \leq L\|\theta_1 - \theta_2\| \quad \text{for some constant } L > 0 \tag{6.9}$$

2. **Bounded Variance:** The stochastic gradients have bounded variance:

$$\mathbb{E}[\|\nabla f_i(\theta) - \nabla f(\theta)\|^2] \leq \sigma^2 \quad \text{for some constant } \sigma > 0 \tag{6.10}$$

3. **Bounded Gradients:** The gradients are bounded:

$$\|\nabla f(\theta)\| \leq G \quad \text{for some constant } G > 0$$

4. **Convexity:** The objective function $f(\theta)$ is convex:

$$f(\theta_1) \geq f(\theta_2) + \nabla f(\theta_2)^T(\theta_1 - \theta_2)$$

5. **Local Exchangeability:** Within each client's dataset, the data points are exchangeable.

6. **Proper Prior:** The global prior $p(\theta \mid \lambda)$ is properly specified.

6.5.2 Variational Inference Convergence

Each client approximates the posterior distribution $p(\theta \mid D_i)$ with a simpler distribution $q_i(\theta \mid \phi_i)$, parameterized by ϕ_i. The goal is to maximize the Evidence Lower Bound (ELBO):

$$\mathcal{L}(\phi_i) = \mathbb{E}_{q_i(\theta|\phi_i)}[\log p(D_i \mid \theta)] - \mathrm{KL}(q_i(\theta \mid \phi_i) \parallel p(\theta))$$

Clients optimize the ELBO using stochastic gradient descent (SGD).

Lemma: Under the assumptions of smoothness and bounded variance, the variational parameters ϕ_i updated using SGD converge to a stationary point of the ELBO:

$$\phi_{t+1} = \phi_t + \eta \nabla_\phi \mathcal{L}(\phi_t)$$

Proof:

Using the smoothness assumption, we have:

$$\mathcal{L}(\phi_{t+1}) \leq \mathcal{L}(\phi_t) + \nabla_\phi \mathcal{L}(\phi_t)^T (\phi_{t+1} - \phi_t) + \frac{L}{2} \|\phi_{t+1} - \phi_t\|^2 \tag{6.11}$$

Substituting the SGD update rule, $\phi_{t+1} = \phi_t + \eta \nabla_\phi \mathcal{L}(\phi_t)$:

$$\mathcal{L}(\phi_{t+1}) \leq \mathcal{L}(\phi_t) + \eta \|\nabla_\phi \mathcal{L}(\phi_t)\|^2 + \frac{L\eta^2}{2} \|\nabla_\phi \mathcal{L}(\phi_t)\|^2 \tag{6.12}$$

Choosing a sufficiently small learning rate η ensures that the ELBO increases monotonically until it converges to a stationary point ϕ^*, where $\nabla_\phi \mathcal{L}(\phi^*) = 0$. Substituting ϕ^* into the gradient equation:

$$\phi^* = \phi_t + \eta \nabla_\phi \mathcal{L}(\phi_t) \implies 0 = \nabla_\phi \mathcal{L}(\phi^*) \tag{6.13}$$

This shows that the gradient of the ELBO at ϕ^* is zero, indicating convergence to a stationary point.

6.5.3 Convergence of the Global Model

Lemma: Under the assumptions of local convergence and proper prior, the global variational distribution $q(\theta) \approx \mathcal{N}(\theta \mid \mu_g, \Sigma_g)$ converges to the true global posterior $p(\theta \mid D)$ as the number of clients N increases.

Proof:

Assume a global hierarchical prior given by:

$$p(\theta \mid \lambda) = \mathcal{N}(\theta \mid \mu_0, \Sigma_0)$$

The objective is to aggregate the local variational parameters received from the clients to form a global posterior distribution where each client's local variational distribution is approximated as $q_{\phi_i}(\theta) \approx \mathcal{N}(\mu_i, \Sigma_i)$.

To aggregate these local distributions, the server updates the global variational parameters, global covariance Σ_g and mean μ_g, using the update rules mentioned in equations 6.7 and 6.8.

$$\frac{1}{\sigma_g^2} = \frac{1}{\sigma_0^2} + \sum_{i=1}^{N} \frac{1}{\sigma_i^2}$$

$$\mu_g = \sigma^2 \left(\frac{\mu_0}{\sigma_0^2} + \sum_{i=1}^{N} \frac{\mu_i}{\sigma_i^2} \right)$$

Here:

- σ_g^2 represents the global uncertainty associated with the parameter estimates.
- $\frac{1}{\sigma_g^2}$ is the inverse of the global variance. It is the sum of the inverse of the prior variance $(\frac{1}{\sigma_0^2})$ and the inverse of the local variances $(\sum_{i=1}^{N} \frac{1}{\sigma_i^2})$.
- μ_g is calculated as a weighted combination of the prior mean (μ_0) and the local means (μ_i), where the weights are given by the respective precisions. As N increases, the influence of the local means μ_i dominates over the prior mean μ_0, assuming the local data is informative.

Given that each client uses SVI, by Lemma 6.5.2, we assume that the variational approximation $q_{\phi_i}(\theta) \approx \mathcal{N}(\mu_i, \Sigma_i)$ converges to the true local posterior $p(\theta \mid D_i)$:

$$q_{\phi_i}(\theta) \to p(\theta \mid D_i)$$

Using the law of large numbers and the central limit theorem, as the number of clients N increases, the aggregated global variational distribution $q(\theta) \approx \mathcal{N}(\mu_g, \Sigma_g)$ should approximate the true global posterior $p(\theta \mid D)$.

- The KL divergence between the aggregated variational distribution $q(\theta) \approx \mathcal{N}(\mu_g, \Sigma_g)$ and the true posterior $p(\theta \mid D)$ decreases as the number of clients increases:
$$\mathrm{KL}(q(\theta) \parallel p(\theta \mid D)) \to 0 \text{ as } N \to \infty$$
- Given local exchangeability and a proper prior, the aggregated posterior $q(\theta)$ converges in distribution to the true posterior $p(\theta \mid D)$ as the number of clients increases.

Lemma: Under the assumptions of smoothness, bounded variance, bounded gradients, and convexity, the BayFL-SVI protocol converges to a stationary point of the global objective.

Proof:

- By Lemma 6.5.2, the variational parameters ϕ at each client converge to a local stationary point of the ELBO.
- By Lemma 6.5.3, the aggregated variational parameters at the server converge to the global variational posterior.
- The convergence of local variational parameters ensures that client updates are unbiased and consistent. The server's hierarchical Bayesian aggregation further refines these updates, leading to global convergence.

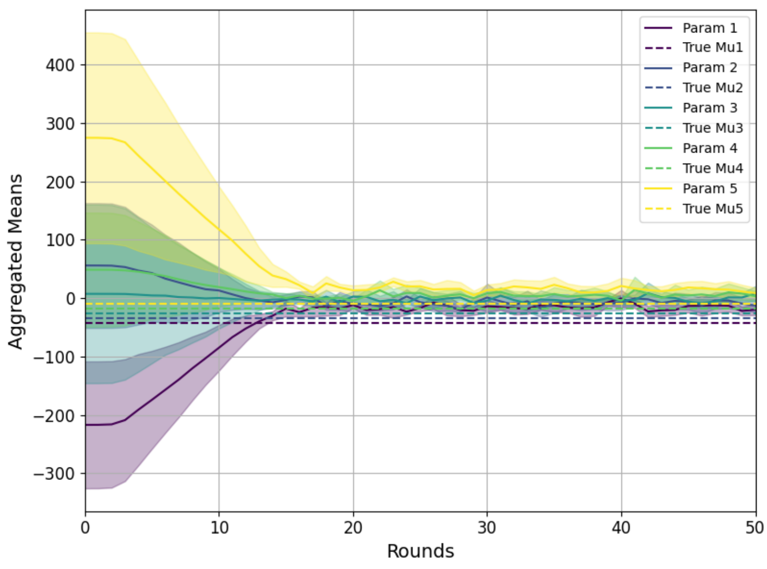


Figure 6.1: Convergence Analysis - Synthetic Data

As $T \to \infty$, the sequence of global model parameters θ_t, converges to the global optimum θ^*. The exchangeability assumption within each client's dataset and the proper hierarchical prior ensure that the aggregated posterior distribution converges to the true global posterior distribution as the number of clients increases. This comprehensive approach highlights the robustness and applicability of Bayesian Federated Learning in the presence of data heterogeneity and privacy constraints.

6.6 Results and Discussion

We conducted several experiments to validate the proposed protocol, BayFL-SVI, demonstrating its convergence behaviour, effectiveness, and robustness in handling non-IID data. Our experiments involved synthetic datasets with known true parameters and real-world datasets to evaluate BayFL-SVI comprehensively.

Firstly, we observe the convergence behavior of the variational parameters of a Bayesian logistic regression on a synthetic dataset of 100 dimensions. Specifically, we distribute 1000 samples among 100 simulated nodes. Each node had different biases added to the true parameter values to simulate non-IID conditions. In each training round, a

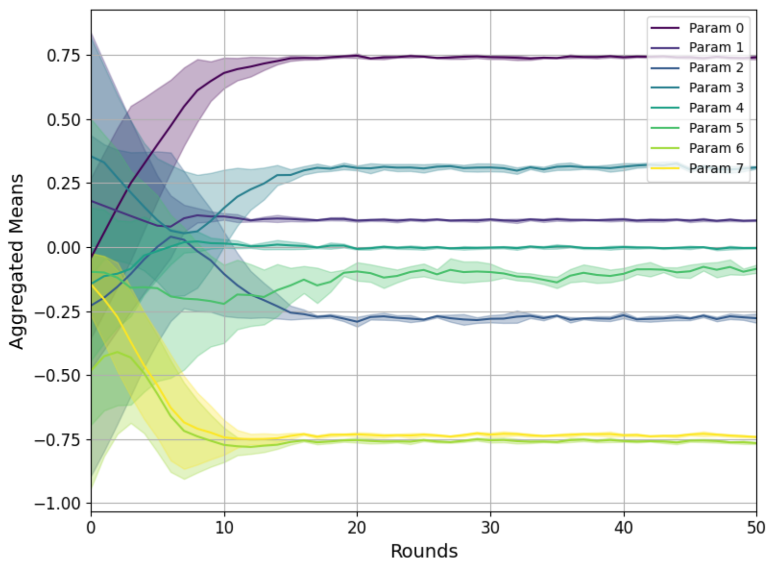


Figure 6.2: Aggregated Means of params using BayFL-SVI over California Housing dataset

subset of 10 nodes was selected. The true mean for each parameter was artificially set to specific values to provide a known ground truth. We implemented a Bayesian Linear Regression model using SVI on each client, aggregating the local posterior distributions using the proposed hierarchical Bayesian approach to update the global model parameters. This process was repeated five times, leading to the convergence of the global mean and the local client means, which were tracked over time. Figure 6.1 shows the convergence of the global mean values for a few sampled parameters, along with confidence intervals and the true mean values, demonstrating that the global model parameters converged towards the true mean values over iterations.

We repeated the same sets of experiments using the California Housing dataset to assess BayFL-SVI in a real-world regression scenario with continuous target values and moderate feature dimensionality. This dataset is widely used for evaluating regression models and offers a practical testbed for analyzing the behavior of Bayesian inference under federated settings. Here, the Bayesian Linear Regression model was implemented using TensorFlow to optimize the local SVI updates. Local posterior distributions from a subset of randomly selected clients were aggregated at each round to update the global model parameters. As illustrated in Figure 6.2, the convergence of the global mean values for all 8 parameters is shown with confidence intervals over multiple runs.

Datasets like the California Housing provide the observed data, such as feature and target values. The relationships between the features and the target variable are not explicitly known. Hence, ground truth parameter values (true μ) cannot be assumed. Instead, we observe the reduction in mean squared error (MSE) loss over iterations to validate the performance of the resulting global posterior distribution. Figure 6.3 illustrates that BayFL-SVI aggregates the local variational parameters, effectively reducing loss iteratively and converging to a stationary point, validating our theoretical analysis.

Lastly, we compare the performance of the BayFL-SVI framework with other state-of-the-art aggregation protocols in FL, such as FedAvg, FedProx [83], Federated Optimization [96], and FedMA [274] over FEMNIST and CelebA datasets. We subsampled 5% of the data for both datasets and selected 10 clients in each training round. For FEMNIST, we train a model with two convolutional layers followed by pooling, and a final dense layer with 2048 units and a learning rate of 0.004. For CelebA, we train a model with three convolutional layers, each followed by max pooling. The output is then flattened and passed through a dense layer to produce a single logit for binary classification. This model is trained using a learning rate of 0.01. Given these settings, Figure 6.4 shows the accuracy distribution for different aggregation mechanisms over 50 training rounds for both datasets.

The results indicate that BayFL-SVI consistently achieves the highest median accuracy across both datasets. For FEMNIST, BayFL-SVI demonstrates a narrow interquartile range (IQR), indicating stable performance despite a few lower-end outliers. On the CelebA dataset, BayFL-SVI continues to outperform other protocols, achieving the highest median accuracy with a narrow interquartile range (IQR) and no significant outliers. This further demonstrates the effectiveness and consistent performance of BayFL-SVI across different datasets. In comparison, other aggregation protocols such as FedMA, FedOpt, FedAvg, and FedProx exhibit lower median accuracies and greater variability.

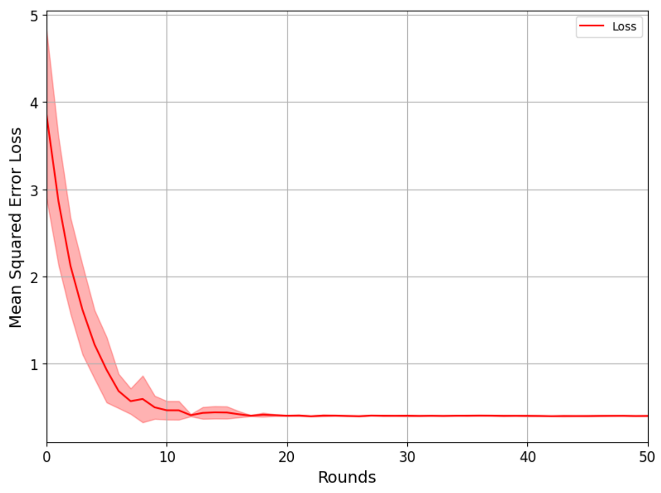
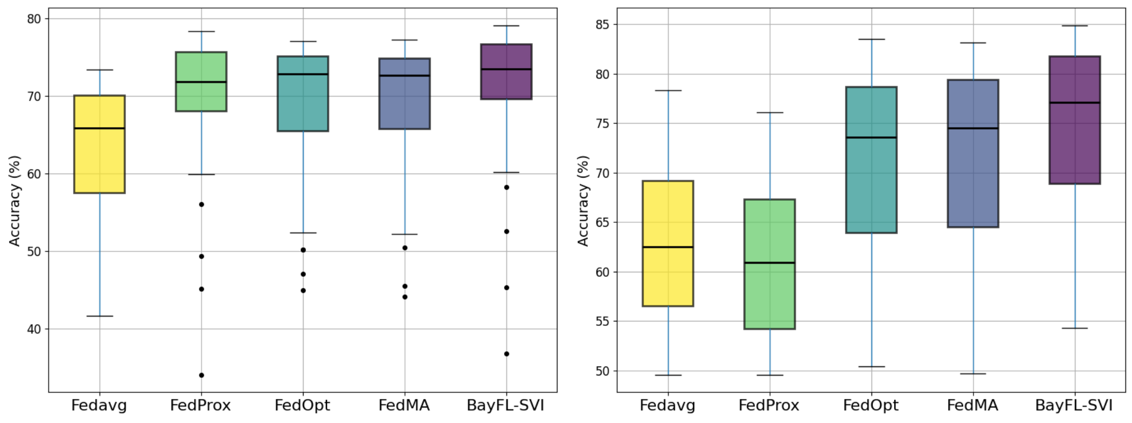


Figure 6.3: Loss Reduction over multiple training rounds using BayFL-SVI over California Housing dataset



(a) FEMNIST (b) CelebA

Figure 6.4: Performance Comparison of different aggregation protocols

Specifically, FedMA shows relatively high median accuracy but still falls short of BayFL-SVI, while FedOpt, FedAvg, and FedProx display broader IQRs and more frequent instances of lower performance.

It is to be noted that all these experiments were conducted using non-IID data. The overall results from these experiments, summarized in Table 6.1, confirm the effectiveness of the BayFL-SVI framework in aggregating local model updates and converging towards accurate global model parameters. The method has shown robustness in handling non-IID data, accounting for client variability, and reducing overall model loss, validating its

Table 6.1: Comparison of different aggregation protocols in FL on FEMNIST and CelebA datasets over non-IID data distribution. The table presents the final accuracy and loss values after 50 training rounds and the accuracy distribution (represented by median and Quartile Range (Q1-Q3)) for each method.

	FEMNIST			
	Final accuracy	**Final loss**	**Median accuracy**	**Q1-Q3**
FedAvg	73.38%	1.03	65.88%	57.47% - 70.08%
FedProx	76.64%	0.77	71.82%	68.02% - 75.70%
FedOpt	75.98%	0.78	72.86%	65.50% - 75.12%
FedMA	76.82%	0.76	72.66%	65.80% - 74.88%
BayFL-SVI	79.28%	0.76	73.69%	70.12% - 76.74%
	CelebA			
	Final accuracy	**Final loss**	**Median accuracy**	**Q1-Q3**
FedAvg	75.42%	0.49	62.53%	56.57% - 69.22%
FedProx	65.6%	0.54	60.93%	54.21% - 67.33%
FedOpt	82.87%	0.36	73.55%	63.92% - 78.71%
FedMA	83.5%	0.36	74.53%	64.54% - 79.38%
BayFL-SVI	84.9%	0.34	77.07%	68.88% - 81.80%

utility.

6.7 Limitations

The proposed Bayesian Federated Learning with Stochastic Variational Inference (BayFL-SVI) framework has some limitations that warrant further investigation. While SVI offers a scalable approximation to full Bayesian inference, it still introduces a non-trivial computational burden, particularly at the client side. Each client must perform variational inference by optimizing both the mean and the variance of the posterior, which involves multiple gradient computations per local update and requires floating-point arithmetic. This is especially challenging for resource-constrained or battery-operated devices that must also manage energy efficiency. The floating-point operations, storage of variational parameters (e.g., means and variances), and repeated gradient updates over multiple mini-batches make the client-side training more computationally intensive than standard methods like FedAvg.

On the server side, the hierarchical Bayesian aggregation involves weighted averaging of client distributions based on their variances. While this remains scalable, the added complexity compared to simple averaging may impact overall runtime in large-scale deployments. As the dataset size and model complexity increase, both client and server-side computational demands scale accordingly, which may limit applicability in some environments.

In addition to computation, communication overhead is another practical limitation. Transmitting posterior distribution increases the size of each client-server update compared to simpler scalar or vector-based aggregation schemes. This added data column can strain bandwidth, especially in systems with limited connectivity or strict communication budgets. Furthermore, the need for frequent and accurate updates to maintain model consistency may compound this overhead.

Another consideration while working with posterior distributions is that they might introduce a smoothing effect in the aggregation process. Since each update integrates prior beliefs, the global model may behave like a low-pass filter, stabilizing the training process by damping abrupt client-specific fluctuations. While this improves robustness, it can also reduce responsiveness to rapid shifts in the data distribution. In applications where the timely detection of rare events or sharp anomalies is critical, this filtering behavior might limit sensitivity and delay adaptation. Exploring alternative formulations or introducing mechanisms to preserve sensitivity to important deviations could help address this trade-off.

While the added computation and communication cost of BayFL-SVI could be justified by improved robustness and accuracy, further optimization, such as model pruning, quantization, or low-rank approximations, may be needed to make it more viable for highly constrained environments. Future work could also explore lighter-weight variational inference methods or hybrid approaches that reduce client-side complexity. Investigating adaptive techniques to selectively apply Bayesian updates based on device capabilities or data characteristics may also improve efficiency. Additionally, studying trade-offs in accuracy and convergence when using alternate approximation methods like Monte Carlo sampling or Laplace approximation could help make the approach more practical in real-world systems.

6.8 Key Findings

The introduction of BFL addresses key challenges in traditional Federated Learning (FL) systems by incorporating uncertainty through Bayesian inference. This probabilistic approach allows us to model the uncertainty in client updates, making the global model more robust to variations in data distributions, particularly in non-IID settings. The use of SVI at the client side ensures that local updates are efficiently approximated as posterior distributions, providing a scalable solution to handle large-scale FL environments. Key observations from the experiments highlight that BFL improves model generalization, enhances convergence rates, and demonstrates better performance compared to traditional FL methods like FedAvg. Additionally, the hierarchical Bayesian aggregation at the server ensures that the model benefits from diverse data while efficiently managing the uncertainty in client contributions.

Bayesian methods also offer a distinct advantage through the use of distributions rather than point estimates. Since clients send variational distributions, not raw model weights or gradients, the information shared is inherently less direct and more abstract. This probabilistic representation adds a natural layer of obfuscation, making it harder for adversaries to reverse-engineer individual data points or reconstruct local models. Unlike techniques such as differential privacy [275], Bayesian approaches diffuse information by design. While this is not a formal privacy guarantee, it reflects a shift in how information is encoded and transmitted in FL. However, further exploration is needed to investigate how much information leakage is reduced in practice, or how this implicit privacy compares empirically to standard mechanisms.

6.9 Summary

This chapter presented a Bayesian approach to Federated Learning, utilizing SVI to manage the uncertainty inherent in distributed and heterogeneous data environments. The proposed BayFL-SVI framework enables each client to approximate posterior distributions locally, which are then aggregated using a hierarchical Bayesian approach at the server. This method not only improves the robustness and generalization of the global model but also enhances its performance in non-IID and heterogeneous data conditions. Our empirical results confirm that BayFL-SVI consistently outperforms traditional FL methods, demonstrating faster convergence and higher accuracy across multiple datasets. Future work will focus on addressing the computational overhead introduced by Bayesian methods and exploring their potential in improving privacy in FL scenarios.

Chapter 7

Zero Trust Federated Learning

7.1 Introduction

While FL offers significant advantages regarding privacy and data locality, its distributed nature also introduces security concerns. The diverse and potentially unsecured nature of client devices that connect and contribute to shared model updates exposes FL systems to a range of vulnerabilities, including unauthorized access, identity spoofing, and data and model poisoning attacks [276, 277]. Malicious clients can disrupt the training process, compromise model integrity, and leak sensitive information, posing significant barriers to the broader adoption of FL. These risks make robust security measures a critical priority for FL systems.

Traditional security models often assume a trusted network perimeter and hence do not adequately address the diverse range of vulnerabilities to which FL systems are often exposed. Addressing these vulnerabilities requires a robust security framework built on zero-trust principles, assuming no device is inherently secure, regardless of its position within or outside the network [278]. In federated settings, where numerous devices request access to participate in collaborative training, such an approach is essential. Integrating a dynamic authentication and authorization mechanism supports this zero-trust model by enabling dynamic and conditional access based on a comprehensive evaluation of device attributes. This ensures that only devices meeting stringent security criteria contribute to the learning process. Additionally, secure aggregation techniques can be layered on top of these mechanisms to mitigate vulnerabilities further, ensuring the system maintains performance even in the presence of malicious clients.

7.2 Objectives

The primary objective of this chapter is to enhance the security of FL systems by implementing dynamic authentication and authorization mechanisms. This involves designing a framework that ensures only trusted and compliant devices can participate in the FL process, thereby protecting the model from vulnerabilities introduced by malicious clients. Another key objective is to incorporate secure aggregation techniques that protect the model updates during the training process, maintaining or even improving the system's performance under adversarial conditions. Ultimately, the chapter aims to demonstrate that with these security enhancements, FL systems can achieve robust performance and reliability, even when faced with potential security threats.

7.3 Contributions

This chapter presents a comprehensive security framework for FL systems, integrating zero-trust principles with robust authentication and secure aggregation techniques. The key contributions of this work are as follows:

- We design an Attribute-Based Access Control (ABAC) model to enforce fine-grained authentication and authorization. This approach ensures that only trusted and compliant edge devices can register and participate in FL tasks.
- We extend the previously proposed Bayesian aggregation methods to include a secure aggregation mechanism, developed to address vulnerabilities such as data and

model poisoning. This method ensures the confidentiality of client updates while maintaining model robustness against adversarial attacks.

- Empirical evaluations demonstrate the effectiveness of the proposed security enhancements. The framework is shown to preserve model accuracy and system performance, even under adversarial conditions, while introducing minimal overhead to the training process.

7.4 Access Control Model for Zero Trust Federated Learning

This section outlines the access control model that supports the registration and participation process for users' devices in our Zero Trust FL environment. We integrate Public Key Infrastructure (PKI) [279] with an adapted attribute-based access control [280, 281] model to provide robust authentication and authorization mechanisms. Devices are authorized based on a comprehensive set of predefined attributes, including device capabilities (e.g., processing power, memory), security posture (e.g., encryption protocols, firewall status), and compliance with organizational policies (e.g., software updates, antivirus status). Upon successful authorization, devices are issued digital certificates via PKI. These certificates are authenticated in the subsequent communication to ensure secure and verifiable communication within the FL network. Only devices that meet these strict criteria and possess valid certificates are granted access, ensuring a secure and reliable client pool for FL tasks. Figure 7.1 represents the abstract ABAC model, while Table 7.1 illustrates the formal definitions of its components.

7.4.1 Model Components

A new edge device requests registration to the client pool (CP), a repository of devices authorized to participate in the FL training process. An edge device can be a laptop,

Table 7.1: ABAC Model Definitions for Zero Trust FL

Basic Sets and Functions:

- C: A finite set of edge devices, often referred to as clients in FL.
- CP: A singleton set {clientPool} where *clientPool* represents the repository of devices authorized to participate in FL tasks.
- A: A finite set of access operations, {register, participate}, where {register} is the operation allowing an edge device to join the client pool, and {participate} permits a device to take part in an FL training task.
- ATT: A finite set of attributes defining the properties, security posture, and compliance state of each device. These attributes are used in evaluating access control policies.
- Range(att): For each attribute $att \in ATT$, Range(att) defines the set of allowable values. For example, RAM and DiskSpace may have numerical ranges, while EncryptionProtocols may have enumerated values (e.g., Active, Inactive).
- attType: A function, {ATT = set, atomic}, that classifies each attribute as either to be set or atomic valued.
- POLICY: A set of authorization policies, each consisting of a rule or set of rules that specify conditions for granting or denying access for the operations defined in A.
- Credentials(c): Cryptographic credentials, such as PKI certificates, issued to an edge device upon successful registration.
- TrustScore(c): A numerical value assigned to each device to reflect its compliance and trustworthiness, which is dynamically updated based on behavior and policy adherence.
- ValidityWindow(c): A time window during which a device's credentials are valid

Authorization function including policies
For each access operation $op \in A$, the authorization function Authorization$_{\mathbf{op}}$(c: C, clientPool: CP) is defined using propositional logic.

- α ::= expr | expr $\wedge$ expr | $\forall x \in$ set.α
- expr ::= atomic atomicAttValCompare atomic | set setAttValCompare set
- atomicAttValCompare ::=$\geq$ | =
- setAttValCompare ::=$\subseteq$ | $\cap \neq \emptyset$
- atomic ::= att(c) | value
  where for each $att \in ATT, c \in C, attType(att) = atomic$
- set ::= att(c) | setValue
  where for each $att \in ATT, c \in C, attType(att) = set$

Authorization Decision Function
A client $c \in C$ is allowed to perform an access operation $op \in A$ on client pool CP, stated as Authorization$_{\mathbf{op}}$(c: C, clientPool: CP) if the required policies needed to allow the operation are evaluated and satisfied. Formally, Authorization$_{\mathbf{op}}$(c: C, clientPool: CP) = **True**.

Raspberry Pi, AWS DeepLens, or any other device that may or may not have minimal computation capabilities for training an ML model. Access (A) represents the operation that the edge device requests to perform on the CP. In this scenario, the ABAC model defines two primary access operations: '**register**' to the authorized client pool and '**participate**'.

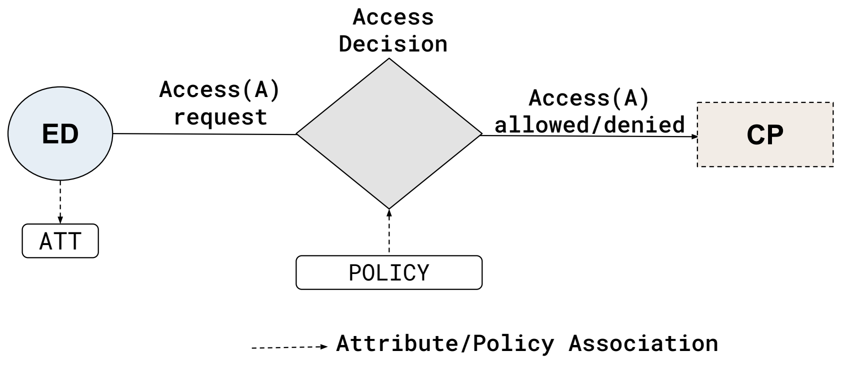


Figure 7.1: A conceptual ABAC model for device registration & participation

The process is initiated when an edge device submits a request to join the CP and seeks authorization to participate in FL training. This request is evaluated at the Access Decision Point (ADP), shown as a diamond-shaped box in Figure 7.1, which determines whether to grant access based on pre-defined security policies. Our illustrated ABAC model utilizes a comprehensive set of attributes (ATT) associated with the client c, including device specifications, security posture, and compliance status, to ensure only qualified devices join the FL network. For example, a device may be allowed to register to the client pool if it has a minimum of 4GB of RAM, 5GB of available disk space, full-disk encryption enabled, and if it connects over secure protocols (e.g., VPN or TLS) to minimize risks associated with data interception. To begin, the client c generates a unique public-private key pair using a secure cryptographic algorithm within PKI. This key pair is used for subsequent communication and authentication within the FL network. The client then generates a Certificate Signing Request (CSR), which includes its public key and the device attributes collected for the registration phase. This CSR represents a formal request for a digital certificate that binds the client's identity to its public key. We assume an honest server that, in this case, serves as a Certificate Authority (CA). The server evaluates the device's attributes in accordance with the defined ABAC policies, determining if the device meets the criteria for network access. Upon satisfying the access control policies, the server signs the CSR with its private key, thereby issuing a digital certificate to the client. This certificate encapsulates the client's public key and authorized attributes, serving as a verifiable credential of its identity. The client c receives the signed digital certificate, which it can use, along with its private key, to authenticate future interactions and participate in the FL training process. Simultaneously, a trust score is assigned to c, representing its trustworthiness based on compliance with the ABAC policies. At the registration stage, c is provided with minimal access rights that include a validity window to limit its lifetime, thereby enforcing periodic re-evaluation. The trust score is also periodically updated based on device compliance and behavior metrics, such as adherence to communication protocols and successful task completion. Devices failing to meet task-specific thresholds are flagged for re-evaluation or removed from the client pool.

When the client c wishes to participate in a specific FL task, it submits a 'participate' operation. The ADP re-evaluates the client's attributes and current trust score against task-specific policies to verify the required clearance. For example, a higher trust score might be required for tasks involving sensitive data or models. The c uses its private key to sign the participation request and any subsequent updates. The server verifies the signature using the client c's public key from its certificate, ensuring the authenticity and integrity of the communications. This approach ensures continuous vetting of devices, both at the point of registration and during participation in FL tasks (in alignment with zero-trust security principles). It minimizes the risk of unauthorized access and potential malicious activity within the FL network.

7.4.2 ABAC Model Definitions

As illustrated in Table 7.1, each edge device has attributes att with values assigned from a set of allowable values, denoted by $Range(att)$, for each attribute $att \in ATT$. In this model, attributes can either be atomic (assigned a single value) or set-valued (assigned multiple values or a range), determined by the function $attType$.

In our model, the set of policies $POLICY$ is specified as the propositional logic using the Backus-Naur Form (BNF) grammar. We use the BNF notations to express the conditions included in the policies enabling the system to evaluate both atomic and set-valued attributes before allowing a device to register or participate in FL tasks. To handle atomic attributes, such as RAM and TrustScore, we follow the rule of `'greater than or equal to'` ($\geq$) to compare the real values against the system-defined threshold values. For set-valued attributes, such as SupportedEncryptionProtocols and AuthenticationMethods, we require that the device's values align with those accepted by the system. For instance, policies require that the SupportedEncryptionProtocols of a device include at least one protocol from the set {AES-256, RSA-2048}. This is enforced by rules such as $\subseteq$ and $\cap \neq \emptyset$, ensuring that the device meets compatibility requirements beyond simple threshold comparisons. The authorization function $\text{Authorization}_{\textbf{op}}$(c: C, clientPool: CP) evaluates the device's attributes against the conditions in the corresponding policy and returns true or false based on the evaluation. If this evaluation returns true, the device is allowed to proceed with the requested operation. If any condition is not met, the function returns False, denying access.

7.4.3 Policy Enforcement and Data Flow Model

Figure 7.2 illustrates the policy enforcement architecture and the data for handling access requests in the FL environment. When an edge device submits a request to register with the client pool or participate in a specific FL task, the policy enforcement system evaluates the request against the defined ABAC policies to determine access permissions. This architecture is designed to handle various client pools, operations, and attribute-based policies. The policy enforcement architecture consists of the following components.

- **Policy Administration Point (PAP)** defines and manages access policies, which it stores in the Policy Repository in a structured format, like a JSON file.
- **Policy Enforcement Point (PEP)** intercepts access requests from edge devices and forwards these to the Policy Decision Point (PDP) along with relevant operation

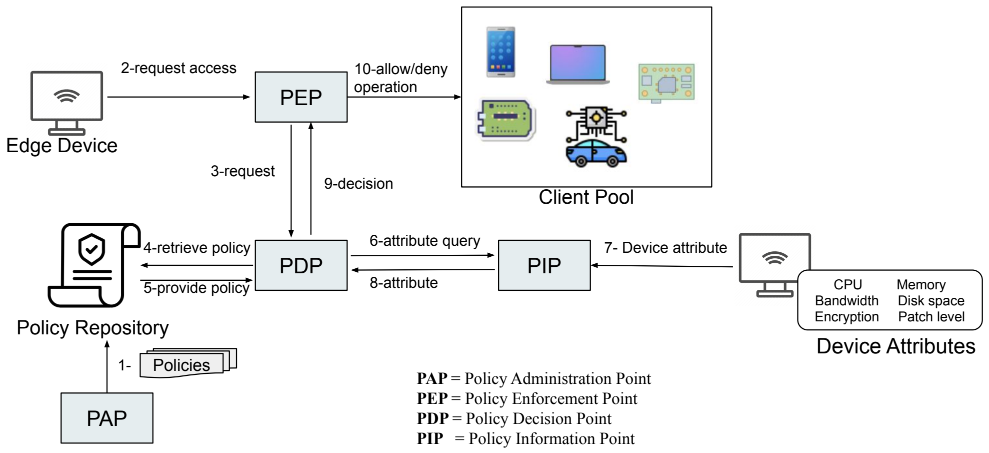


Figure 7.2: ABAC policy enforcement architecture

details (e.g., *register* or *participate*).

- **Policy Decision Point (PDP)** upon receiving a request, retrieves the relevant policy from the Policy Repository. It then queries the Policy Information Point (PIP) for the required device attributes needed for policy evaluation.

- **Policy Information Point (PIP)** collects the necessary attribute values from the requesting device and returns these values to the **PDP** for evaluation when requested.

- The **PDP** evaluates the attributes against the policy conditions and makes a decision on whether the device should be allowed access. If the device meets all policy requirements, the PDP authorizes the request and sends a positive decision to the PEP. If not, the PDP denies the request.

- The **PEP** enforces the decision by allowing or denying the device's access to register with the client pool or participate in the FL task.

For *register* operation, if the PDP authorizes the requesting client, it is issued a digital certificate by the server.

7.4.4 Register Operation: System Setup

In our setup, the digital certificates authenticate the identity and compliance of clients. These certificates bind a device's public key, unique identifier, and attributes, enabling secure participation in FL tasks at a later stage. The process of certificate generation and issuance involves key pair generation for the server and clients, CSR creation, and certificate issuance.

We assume an honest server who also acts as a CA, responsible for issuing and verifying certificates. During the initialization phase, the server generates an Elliptic Curve Cryptography (ECC) key pair using the `secp256r1` curve. The `secp256r1` curve is widely used in cryptography due to its balance between computational efficiency and security. The 256-bit key length provides adequate security against brute-force attacks, while the curve's parameters are well-documented and standardized. The server key pair consists of the following:

- **CA Private Key** (d_{CA}): A randomly selected scalar $d_{\mathrm{CA}} \in_R [1, n-1]$, where n is the order of the curve.
- **CA Public Key** (P_{CA}): Derived as $P_{\mathrm{CA}} = d_{\mathrm{CA}} \cdot G$, where G is the curve's generator point.

The order n is the number of points on the curve that can be generated by repeatedly adding the generator point G to itself. This ensures a large enough keyspace for cryptographic security. The server securely stores d_{CA} and shares its public key P_{CA} to all clients, allowing them to verify the CA's digital signatures. Simultaneously, each client device generates its own ECC key pair using the `secp256r1` curve. This process ensures that the client's cryptographic identity is unique and secure. The key pair includes:

- **Private Key** (d_i): A randomly selected scalar $d_i \in_R [1, n-1]$, kept securely on the device.
- **Public Key** (P_i): Derived as $P_i = d_i \cdot G$, serving as the client's public identity.

After generating the key pair, the client prepares a CSR request to initiate the certificate issuance process. The CSR contains the following components:

- The client's public key (P_i).
- A unique client identifier (ID_i), such as a device ID.
- Device attributes (ATT_i) required for ABAC evaluation, including compliance parameters like encryption protocols.

The CSR is digitally signed by the client using its private key (d_i) to ensure its authenticity and integrity:

$$\text{CSR}_i = \text{Sign}_{d_i}(P_i \parallel ID_i \parallel ATT_i)$$

where $\parallel$ denotes concatenation. The signed CSR is transmitted to the server.

Upon receiving the CSR, the server evaluates the client's attributes against the ABAC policies using the $\text{Authorization}_{\text{register}}(c, CP)$ function. If the device satisfies the policies, the server proceeds to generate a digital certificate (Cert_i):

$$\text{Cert}_i = \{P_i, ID_i, ATT_i, \text{Sig}_{\text{CA}}\}$$

where P_i is the client's public key, ID_i is the unique identifier for the client, ATT_i is the set of device attributes and Sig_{CA} is the CA's signature over the concatenated public key, identifier, and attributes given below:

$$\text{Sig}_{\text{CA}} = \text{ECDSA\_Sign}(d_{\text{CA}}, H(P_i \parallel ID_i \parallel ATT_i))$$

where H is a cryptographic hash function SHA-256, and $\parallel$ denotes concatenation. The server securely transmits the signed certificate back to the client. Each certificate is assigned a validity window ($ValidityWindow$), ensuring that certificates are short-lived. This practice reduces the risk associated with potential key compromises and necessitates periodic renewal, thereby maintaining up-to-date compliance with ABAC policies. The client securely stores Cert_i and d_i for subsequent authentication.

The issuance of a digital certificate marks the completion of the register operation in the ABAC framework. These certificates bind the client's identity to its cryptographic key pair and validated attributes, ensuring that only authorized and authenticated clients are allowed to join the FL client pool. Once a client receives its certificate, it can securely participate in FL tasks.

7.4.5 Participate Operation: Authorization and Secure Communication

Even with strong authorization measures in place, the decentralized nature of FL introduces risks such as eavesdropping, data leakage through shared gradients, or tampering during communication. To tackle these challenges, it is essential to establish secure and encrypted communication channels between clients and the server. At this stage, we leverage the PKI for mutual authentication with the central server and establish a secure communication channel.

After a client completes the *register* operation and obtains a digital certificate, it can request to participate in specific FL tasks. The *participate* operation is designed to dynamically authorize clients for each task, ensuring compliance with task-specific requirements and maintaining a secure communication environment for the transmission of global models. This phase integrates the previously issued digital certificates and ABAC policies into a robust framework for mutual authentication and secure model distribution.

To participate, client c_i initiates a TLS handshake with the server and includes a signed participation request in its communication. The request contains the client's digital certificate (Cert_i) and is signed using the client's private key (d_i):

$$\text{Participate\_Request}_i = \text{Sign}_{d_i}(\text{Cert}_i \parallel \text{ATT}_i)$$

Upon receiving the request, the server first validates the client's certificate Cert_i to verify the client's identity. The certificate verification process uses the CA's public key (P_{CA}) to authenticate the signature of the certificate:

$$\text{ECDSA\_Verify}(P_{\text{CA}}, H(P_i \parallel ID_i \parallel \text{ATT}_i), \text{Sig}_{\text{CA}}) = True$$

where H is a cryptographic hash function SHA-256. If the certificate is invalid or has expired, the server terminates the handshake, and the client's participation request is denied. If the certificate is valid, the server proceeds to evaluate the client's attributes against task-specific ABAC policies. The server uses the following authorization function:

$$\text{Authorization}_{\text{participate}}(C_i, \text{Task}) = \begin{cases} \text{True}, & \text{if ATT}_i \text{ satisfies task policies} \\ \text{False}, & \text{otherwise} \end{cases}$$

This ensures that only clients meeting the task's specific requirements can proceed further. If either the certificate validation or ABAC policy evaluation fails, the server denies the request, and the connection is terminated.

For authorized clients, the server completes the TLS handshake to establish a secure communication channel. The handshake uses ephemeral key pairs generated with the same `secp256r1` curve employed during the registration phase. Each party generates its private and public keys as follows:

- Server Private Key: d_{Server}^{eph}
- Server Public Key: $P_{Server}^{eph} = d_{Server}^{eph} \cdot G$
- Client Private Key: d_i^{eph}
- Client Public Key: $P_i^{eph} = d_i^{eph} \cdot G$

During the handshake, the client and server exchange their public keys (P_i^{eph} and

P^{eph}_{Server}) to compute a shared session key $K_{session}$ using Elliptic Curve Diffie-Hellman (ECDH):

$$K_{session} = H(d^{eph}_i \cdot P^{eph}_{Server}) = H(d^{eph}_{Server} \cdot P^{eph}_i)$$

where H is a key derivation function. This session key is used to encrypt all subsequent communication between the client and server. This phase ensures that only authenticated and authorized clients can securely contribute to the learning process. The use of certificates and task-specific ABAC policies maintains the integrity of the FL process, while the encrypted channel ensures the confidentiality of communications. The next section details how the established channel during the authentication phase can be leveraged for secure communication.

7.5 Secure Aggregation in Zero Trust FL

Once the client's participation request is authorized, the server transmits the global model θ_{global} to the client over the secure TLS channel established during the participation phase. To preserve confidentiality, the global model is encrypted using the session key $K_{session}$ established during the TLS handshake:

$$\text{Enc}(\theta_{\text{global}}, K_{session})$$

Encrypting the model parameters ensures that it remains inaccessible to unauthorized entities even if intercepted during transmission. Upon receiving the model, the client decrypts it using the same session key and prepares it for local training:

$$\theta_{\text{global}} = \text{Dec}(\text{Enc}(\theta_{\text{global}}, K_{session}), K_{session})$$

Using its local dataset (D_i), the client trains the model by employing SVI to approximate the posterior distribution of the model parameters, as explained in Section 6.4.2.

The training process maximizes the ELBO using Equation 6.2 and produces the update:

$$U_i = \{\mu_i, \sigma_i^2\}$$

where μ_i represents the mean and σ_i^2 the variance of the posterior. This compact representation encapsulates the client's local learning while preserving data privacy by avoiding the transmission of raw model parameters. Using its private key d_i, the client generates a digital signature based on the ECDSA algorithm and signs the generated model update U_i.

$$\text{Signature}_i = \text{ECDSA\_Sign}(d_i, H(U_i))$$

where $H(U_i)$ is the cryptographic hash of the update. The client then transmits the signed package $\{U_i, \text{Signature}_i\}$ to the server over the encrypted TLS channel.

Upon receiving the signed update package, the server performs the following steps to verify its authenticity and integrate the update into the global model:

1. The server extracts the client's public key P_i from the client's certificate $Cert_i$, issued during the registration phase.

2. Using P_i, the server verifies the client's signature on the update:

$$\text{ECDSA\_Verify}(P_i, \text{Signature}_i, H(U_i)) = \text{True}$$

   ensuring that the update U_i was indeed generated by the authenticated client and has not been tampered with during transmission.

3. If the signature is valid, the server accepts the client's update $U_i = \{\mu_i, \sigma_i^2\}$ for aggregation.

The updates from all clients are then aggregated using Hierarchical Bayesian principles, as explained in Section 6.4.3. This method models the global posterior distribution

θ_{global} while accounting for the heterogeneity in client data distributions and ensuring that the global model can adapt to diverse data while remaining robust against potential adversarial influences. We can further weigh the client-specific posterior to reflect the reliability and data quality of each client for faster model convergence.

In cases where signature verification fails or a client is found to have violated compliance policies, the server discards the update and logs the event for further analysis. Additionally, the client may be flagged for re-evaluation or revocation to ensure that only reliable and compliant contributions are integrated into the training process.

7.6 Performance Evaluation and Results

This section evaluates the proposed Zero-Trust FL framework, integrating ABAC and secure Bayesian aggregation, under adversarial scenarios. The evaluation focuses on two core security aspects: authentication and authorization (ABAC) and secure aggregation. The results demonstrate the framework's robustness against unauthorized access, impersonation, man-in-the-middle (MITM) attacks, and various poisoning attacks (data and model poisoning). Comparisons are made with standard protocols to establish the efficacy of the proposed solution.

7.6.1 Security Against External Adversaries

To evaluate the security capabilities of our proposed Zero Trust FL protocols against external adversaries, we simulated multiple common attack scenarios in a controlled FL environment: 1: Unauthorized Access, 2: Impersonation, and 3: Man-in-the-Middle (MITM) Attacks. These scenarios were designed to test the framework's ability to detect and prevent malicious attempts during various stages of the FL process, including client registration, participation, and communication.

We simulated various scenarios where adversaries used forged or stolen credentials to simulate unauthorized clients trying to access the system. Moreover, in some scenarios,

Table 7.2: Simulation Results for Zero Trust FL in Detecting Attack Scenarios

Attack Scenario	**Total Attempts**	**Detection Rate (%)**	**False Positive Rate (%)**
Unauthorized Access	50	92.00	4.17
Impersonation	50	88.00	6.38
Man-in-the-Middle	50	62.00	10.87
Overall	**150**	**80.67**	**7.14**

adversaries intercepted communications during model update transfer and tampered with transmitted data. The proposed protocol successfully detected these adversaries and prevented them from accessing the system, maintaining the integrity of the learning process. The detection rates for each scenario were recorded and analyzed. We summarize these results in Table 7.2.

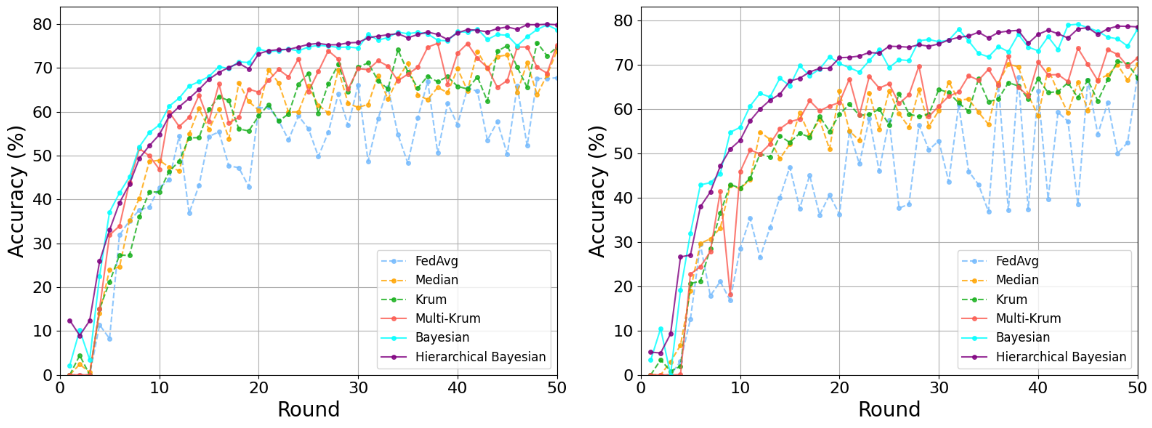
7.6.2 Robustness Against Internal Threats

The framework's secure aggregation component, implemented using Bayesian methods, was evaluated against adversarial attacks targeting the learning process. These included:

- **Label Flip Attacks:** Malicious clients flip labels in their local datasets to poison the global model.
- **Gradient Scaling Attacks:** Adversaries randomly scale their gradients to bias the global model.
- **Random Weights Attacks:** Malicious clients submit random model updates to disrupt training convergence.

The experiments in this section used the FEMNIST dataset with non-IID partitioning and included two varying adversarial client ratios. The proposed protocol was compared to FedAvg, Median, Krum, Multi-Krum, and Bayesian aggregation mechanisms.

In scenarios with 10% malicious clients performing label flipping, the Zero-Trust framework achieved an accuracy of approximately 78% after 30 communication rounds, maintaining stable performance compared to other methods, followed closely by Bayesian aggregation at 75%. Multi-Krum, Krum, and Median showed some resistance but exhibited fluctuating performance, with accuracies ranging between 60% and 65%. FedAvg, with no built-in defenses, struggled to maintain a stable performance. With 20% malicious clients,



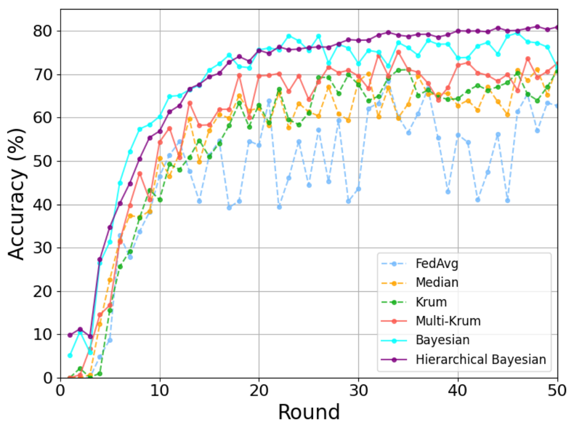
(a) 10% malicious clients (b) 20% malicious clients

Figure 7.3: Test accuracy of algorithms in the first 50 rounds in FEMNIST when adversaries perform a label-flipping attack

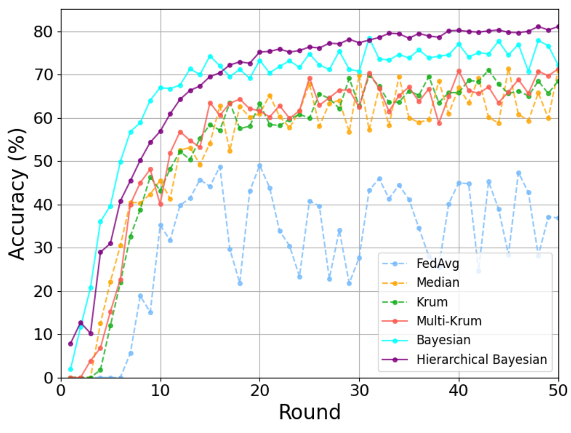
the accuracy of the Zero-Trust framework dropped slightly during the early rounds. However, it maintains its superiority over other methods in the subsequent rounds. Bayesian aggregation followed closely, while Multi-Krum and Median also demonstrated performance drops due to their inability to consistently filter out poisoned updates. We summarize these results in Figure 7.3.

Gradient scaling attacks involve malicious clients scaling their gradients to bias the global model. Under this scenario, the Zero-Trust framework maintained robust performance, achieving around 80% accuracy. As can be seen in Figure 7.4, Bayesian aggregation closely followed, while showing some fluctuations in accuracy over the rounds. Multi-Krum showed moderate robustness but fluctuated significantly under higher adversarial pressure. Median and Krum demonstrated further instability, while FedAvg again struggled to converge due to its lack of defense mechanisms.

Random weight attacks were simulated using a fraction of adversaries submitting random model updates to poison the global training process. The proposed framework demonstrated consistent performance, with an accuracy of 78%. However, in this case,



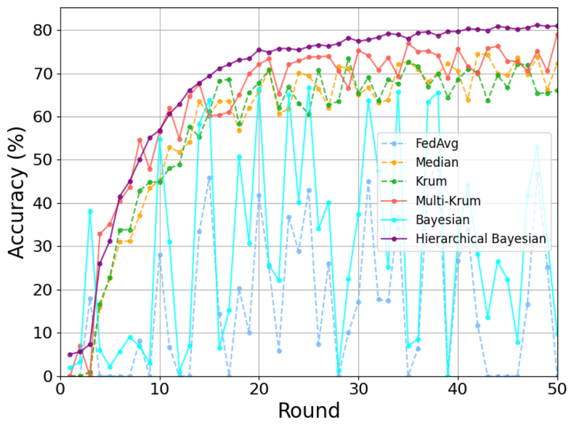
(a) 10% malicious clients



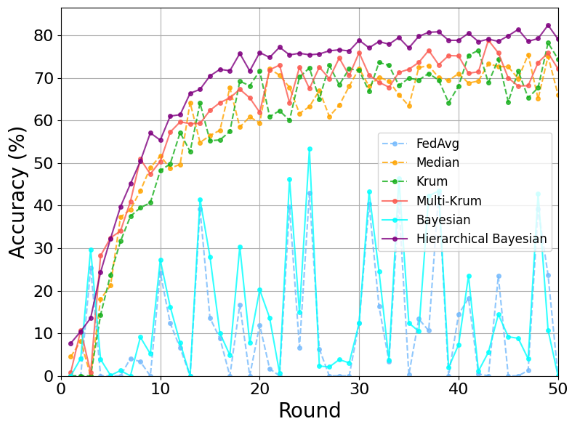
(b) 20% malicious clients

Figure 7.4: Test accuracy of algorithms in the first 50 rounds in FEMNIST when adversaries perform a gradient scaling attack

Bayesian aggregation struggles to maintain a robust aggregation mechanism. Figure 7.5 shows that the global model struggles to converge under Bayesian and FedAvg aggregation mechanisms. While other aggregation mechanisms also show a decrease in accuracy.



(a) 10% malicious clients



(b) 20% malicious clients

Figure 7.5: Test accuracy of algorithms in the first 50 rounds in FEMNIST when adversaries perform a random gradient attack

The results across all scenarios highlight the efficacy of the Zero-Trust FL framework in mitigating both external network threats and internal adversarial attacks. The ABAC mechanism ensures that only legitimate and trusted clients participate in the training process, significantly reducing the risks posed by unauthorized or compromised devices. The secure aggregation further enhances robustness against poisoned updates, maintaining high accuracy and stable performance, even under adversarial conditions.

7.7 Discussion and Limitations

While our experimental results demonstrate significant improvements in training efficiency and model performance compared to existing methods, it is essential to acknowledge the inherent trade-offs and limitations of our approach. The integration of security mechanisms such as ABAC and PKI enhances the security posture of the system by ensuring that only authenticated and authorized clients participate in the training process. However, these security measures introduce additional computational and communication overhead, potentially impacting system efficiency. The overhead arises from the need to perform authentication and authorization checks during client registration and before aggregating model updates. Verifying digital certificates and enforcing access control policies consume resources and may slightly delay the aggregation process, especially in environments with a large number of clients or limited computational capacity. Nevertheless, these trade-offs between accuracy, efficiency, and security reflect the inherent complexities of designing a practical zero-truth FL framework.

As the number of attributes, clients, and access operations grows, managing complex ABAC policies and PKI configurations may become increasingly challenging. These limitations raise important questions about the scalability and maintainability of the framework in large-scale deployments. To address these concerns, a promising direction would be to integrate the proposed protocol with cloud-native FL platforms that already support zero-trust features. Most cloud infrastructures offer built-in services for ABAC, PKI, or certificate management, which could be leveraged to enforce fine-grained access control, manage identities, and handle secure communications more efficiently. This architectural enhancement would help scale the system while reducing management overhead, making it more compatible with diverse FL environments and real-world deployment scenarios.

Another direction could be to incorporate dynamic or real-time trust assessment mechanisms to improve the responsiveness to malicious behavior and support more adaptive security policies. Sophisticated anomaly detection mechanisms could also be incorporated to strengthen the security posture of the framework.

7.8 Summary

This chapter presented a comprehensive framework for Zero-Trust FL that integrates ABAC and secure aggregation protocol to address the inherent vulnerabilities of FL systems. The proposed framework ensures robust authentication and dynamic authorization of clients, preventing unauthorized access, impersonation, and man-in-the-middle attacks. By combining ABAC policies with PKI, the system enforces strict access controls and dynamically evaluates client attributes and trust scores, ensuring that only compliant devices participate in the learning process.

The secure aggregation component, implemented using hierarchical Bayesian principles, enhances the framework's robustness against data and model poisoning attacks, including label flipping, gradient scaling, and random weight injection. Extensive evaluations demonstrated the framework's superiority over existing methods, such as FedAvg, Median, Krum, and Multi-Krum, under varying adversarial conditions. The results highlight the framework's ability to maintain high accuracy and stable convergence, even when up to 20% of participating clients are adversarial. This highlights the applicability of zero-trust FL protocols in diverse and dynamic FL environments where no trust can be assumed. Future work will explore further optimizations to enhance communication efficiency and reduce computational overhead within the zero-trust framework. Another possible direction is designing a more sophisticated and robust security protocol.

Chapter 8

A Fairness-Aware Incentive Mechanism for Crowdsourced Federated Learning Using Shapley Value

8.1 Introduction

In Crowdsourced Federated Learning, clients contribute varying amounts of data and computational resources, making their contributions highly heterogeneous. A key challenge in such systems lies in incentivizing clients effectively while ensuring that rewards are distributed fairly based on their actual contributions to the global model. Without a fair and robust incentive mechanism, participants may lose motivation or attempt to "game the system", leading to reduced model quality and trust in the system. A fair incentive mechanism is essential not only for encouraging genuine participation but also for defending against adversarial behaviors. Free riding and poisoning attacks, where clients provide negligible contributions and attempt to degrade model performance, can severely impact CFL's effectiveness. An effective reward distribution method must address these challenges by objectively evaluating each client's contribution and ensuring proportional compensation.

In our proposed protocol, FedOnDemand, we address these challenges by incorporating the Shapley Value as the foundation of the incentive mechanism. The Shapley Value is a well-established concept in cooperative game theory that offers a systematic framework for measuring contributions by considering the marginal impact of each participant on the overall system. By integrating Shapley Value-based incentives, FedOnDemand can achieve fairness and transparency in reward allocation while discouraging adversarial behaviors. Furthermore, to make the computation feasible in large-scale CFL systems, FedOnDemand employs a Truncated Monte Carlo (TMC) to approximate the Shapley Value efficiently without compromising accuracy. This approach ensures that the incentive mechanism remains scalable and effective, even with a large number of participating clients.

This chapter focuses on leveraging the Shapley Value to design a fairness-aware incentive mechanism for CFL. The proposed mechanism aligns the interests of clients and the server by linking rewards to contributions, ensuring equitable profit distribution while protecting against adversarial threats.

8.2 Objectives

The primary objective of this chapter is to design and evaluate a fairness-aware incentive mechanism for the FedOnDemand protocol in Crowdsourced Federated Learning (CFL). The mechanism aims to address the critical need for equitable reward distribution among clients, ensuring that each participant is compensated based on their actual contribution to the global model. A secondary, but equally important objective, is to enhance the robustness of the learning process against adversarial behaviors where clients attempt to gain rewards with minimal or no meaningful contributions. By integrating fairness into incentive mechanisms, we aim to promote sustained participation, resilience, and foster trust in the FedOnDemand protocol.

8.3 Contributions

This chapter builds upon the established concept of the Shapley Value, leveraging its principles to design an efficient incentive mechanism tailored for the FedOnDemand protocol. The key contributions of this work are as follows:

- We design a fairness-aware incentive mechanism for FedOnDemand that ensures equitable profit allocation among participating clients by evaluating their contributions using the Shapley Value.

- We propose an efficient approximation for Shapley Value computation using TMC sampling to address the computational challenges in large-scale CFL systems.

- We validate the proposed incentive mechanism through theoretical analysis and empirical evaluations, showcasing its effectiveness in promoting fairness, scalability, and robustness against adversarial behaviors.

8.4 Preliminaries

The Shapley Value is a classic game theory solution designed to fairly allocate rewards among players in a cooperative game, which has been applied widely in fields such as management science, economics, and to design incentive mechanisms in classic crowdsourcing. It quantifies the marginal contribution of each player with respect to the total value generated by the coalition.

Consider a set of n players, $C = \{c_1, c_2, \ldots, c_n\}$, engaged in a cooperative game. A value function $v : 2^C \to \mathbb{R}$ is defined, which assigns a real number $v(Z)$ to each subset (or coalition) $Z \subseteq C$. The value $v(Z)$ represents the utility or total payoff that the coalition Z

can achieve collectively. The goal is to distribute the total utility $v(C)$ among all players in C such that each player's reward reflects their contribution to the coalition's success.

As originally shown by Shapley [282], the Shapley Value for a player $c_i \in C$ is defined as:

$$\phi_{c_i}(v) = \sum_{Z \subseteq C \setminus \{c_i\}} \frac{|Z|!(n - |Z| - 1)!}{n!} [v(Z \cup \{c_i\}) - v(Z)] \tag{8.1}$$

where $|Z|$ is the size of the coalition Z, and $n = |C|$. This formulation computes c_i's marginal contribution to every possible coalition Z that excludes c_i, averaged over all such coalitions.

The Shapley Value satisfies several desirable properties that ensure its fairness and effectiveness:

- **Efficiency**: The total value $v(C)$ of the grand coalition C is fully distributed among all players, ensuring that the sum of all individual Shapley Values equals the total utility generated by the entire set of players.:

$$\sum_{i=1}^{n} \phi_{c_i}(v) = v(C) \tag{8.2}$$

- **Symmetry**: If two players c_i and c_j have identical contribution to every coalition $Z \subseteq C \setminus \{c_i, c_j\}$, then their Shapley Values are the same:

$$\text{If } v(Z \cup \{c_i\}) = v(Z \cup \{c_j\}),\ \forall Z \subseteq C \setminus \{c_i, c_j\}, \text{ then } \phi_{c_i}(v) = \phi_{c_j}(v) \tag{8.3}$$

- **Null Player**: If a player c_i does not contribute to any coalition, their Shapley Value is zero:

$$\text{If } v(Z \cup \{c_i\}) = v(Z),\ \forall Z \subseteq C \setminus \{c_i\}, \quad \text{then } \phi_{c_i}(v) = 0 \tag{8.4}$$

- **Additivity**: For two independent games with value functions v_1 and v_2, the Shapley Value for their combined game $v_1 + v_2$ is the sum of the Shapley Values for v_1 and

v_2, ensureing that the Shapley Value respects the linearity of value functions.

These properties collectively establish the fairness and robustness of the Shapley Value for reward allocation in cooperative systems. However, its exact computation requires evaluating the marginal contribution for all 2^n subsets of players, which becomes computationally expensive as n increases. Efficient approximation techniques, such as Truncated Monte Carlo sampling, are therefore essential for applying the Shapley Value in large-scale systems.

8.5 Incentive Mechanism for Crowdsourced Federated Learning

We design the incentive mechanism for the FedOnDemand protocol using the same principles of cooperative game theory outlined above. Each client participating in the learning process is treated as a player in a cooperative game. The total utility or value generated by a coalition of clients corresponds to the improvement in the performance of the global model when the contributions of those clients are aggregated. The objective is to evaluate each client's contribution to the overall utility and allocate rewards fairly based on these contributions.

Let $C = \{c_1, c_2, \ldots, c_n\}$ represent the set of n clients willing to participate in the learning process. For each training round t in FL, only a subset of these clients, $S_t \subseteq C$, is selected and invited to participate in the training. Since only these selected clients contribute to the model in that particular round, it is only logical to treat each training round as a separate cooperative game. The additivity property of the Shapley Value ensures that contributions calculated per round can be summed to determine the total contribution of each client across the entire training process.

In the t-th training round, the value function $v_t(S_t)$ represents the performance improvement of the global model resulting from the contributions of the selected clients S_t.

This value function is defined as:

$$v_t(S_t) = \sum_{c_i \in S_t} x_i + \sum_{\substack{c_i, c_j \in S_t \\ c_i \neq c_j}} \alpha x_i x_j \tag{8.5}$$

where x_i denotes the contribution of client c_i in terms of data and computational resources while $\alpha > 0$ captures the interaction effect between client contributions. For example, a higher α indicates that client contributions complement each other significantly, amplifying the overall value generated by the coalition. Similarly, the marginal contribution of c_i to a coalition $Z \subseteq S_t \setminus \{c_i\}$ is computed using the following equation.

$$\Delta_{c_i}(Z) = v_t(Z \cup \{c_i\}) - v_t(Z) \tag{8.6}$$

The Shapley Value for a client $c_i \in S_t$ in the t-th round is then computed as the weighted sum of all marginal contributions using Equation 8.1

$$\phi_{c_i}^t(v_t) = \sum_{Z \subseteq S_t \setminus \{c_i\}} \frac{|Z|!(|S_t| - |Z| - 1)!}{|S_t|!} \Delta_{c_i}(Z)$$

Here, $\phi_{c_i}^t(v_t)$ represents the average marginal contribution of c_i in t-th training round over all possible coalitions formed from the subset S_t of actively participating clients in that round. After all training rounds are completed, the total Shapley Value for each client c_i across the entire process is computed as:

$$\phi_{c_i}(v) = \sum_{t=1}^{T} \phi_{c_i}^t(v_t) \tag{8.7}$$

where T is the total number of training rounds. This aggregated value reflects c_i's cumulative contribution to the global model and serves as the basis for fair reward allocation. By treating each training round as an independent game, we significantly reduce the computational burden compared to evaluating the contributions of all n clients in every round. This design also aligns with the dynamic nature of CFL, where client availability may vary across rounds, ensuring that rewards are distributed fairly based on actual participation.

We take a simple example to verify the rationale behind this formulation. Consider an FL system with five clients, $C = \{c_1, c_2, c_3, c_4, c_5\}$, participating in the learning. For example, consider a training process that spans only two rounds, where:

- **Round 1:** Clients $S_1 = \{c_1, c_2, c_3\}$ are selected to participate in the training.
- **Round 2:** Clients $S_2 = \{c_2, c_4, c_5\}$ are selected.

During the training process, each client contributes x_i, representing their data and computational resources. The budget R allocated to the task determines the total reward pool available for distribution, while the cost of participation for each client is modeled as $\text{cost}_i(x_i) = \beta x_i^2$, where β is a monotonically increasing parameter that scales with the contribution.

In the first training round, $v_1(S_1)$ measures the improvement in the global model's performance based on the contributions x_1, x_2, x_3 from c_1, c_2 and c_3 respectively, and their interactions using Equation 8.5.

$$v_1(S_1) = x_1 + x_2 + x_3 + \alpha(x_1x_2 + x_1x_3 + x_2x_3)$$

Using the Equation 8.6, the marginal contributions of c_1, c_2, and c_3 are computed by considering their impact on all coalitions within S_1. For instance, the contribution of c_1 would be computed over the following permutations:

$$v_1(\{c_1, c_2\}) - v_1(\{c_2\}), \quad v_1(\{c_1, c_3\}) - v_1(\{c_3\}), \quad v_1(\{c_1, c_2, c_3\}) - v_1(\{c_2, c_3\})$$

The contribution of c_2 and c_3 can be computed similarly. The Shapley Value for clients c_1, c_2, and c_3 is then computed by evaluating their marginal contributions across all permutations of S_1 using Equation 8.1. Specifically, consider the following permutations in S_1:

1. **Permutation 1** $\sigma_1 = [c_1, c_2, c_3]$**:** Client c_1 is the first client in this permutation. Since there are no other clients in the coalition at this point ($Z = \emptyset$), the value of the coalition is $v(Z) = 0$. The marginal contribution of c_1, using Equation 8.6, is:

   $$\Delta_{c_1} = v_1(\{c_1\}) - v_1(\emptyset) = x_1$$

   The client c_2 enters as the second client in this coalition. The coalition before c_2 is $Z = \{c_1\}$, with value $v_1(Z) = x_1$. Thus, the marginal contribution of c_2 can be expressed as follows:

   $$\Delta_{c_2} = v_1(\{c_1, c_2\}) - v_1(\{c_1\}) = x_2 + \alpha x_1 x_2$$

   Lastly, client c_3 is the third and final client in this permutation. The coalition before c_3 is $Z = \{c_1, c_2\}$, with value $v_1(Z) = x_1 + x_2 + \alpha x_1 x_2$. Thus, the marginal contribution of c_3 is:

   $$\Delta_{c_3} = v_1(\{c_1, c_2, c_3\}) - v_1(\{c_1, c_2\}) = x_3 + \alpha(x_1 x_3 + x_2 x_3)$$

2. **Permutation 2** $\sigma_2 = [c_2, c_1, c_3]$**:** In this permutation, client c_2 is the first client. Since $Z = \emptyset$, $v_1(Z) = 0$. The marginal contribution of c_2 is:

   $$\Delta_{c_2} = v_1(\{c_2\}) - v_1(\emptyset) = x_2$$

   Client c_1 is the second client in this permutation. Using the same formulation as established before, the coalition before c_1 is $Z = \{c_2\}$, with value $v_1(Z) = x_2$. The marginal contribution of c_1 is:

   $$\Delta_{c_1} = v_1(\{c_2, c_1\}) - v_1(\{c_2\}) = x_1 + \alpha x_1 x_2$$

Similarly, the coalition before c_3 is $Z = \{c_2, c_1\}$, with value $v_1(Z) = x_2 + x_1 + \alpha x_1 x_2$. The marginal contribution of c_3 is thus, computed as follows:

$$\Delta_{c_3} = v_1(\{c_2, c_1, c_3\}) - v_1(\{c_2, c_1\}) = x_3 + \alpha(x_1 x_3 + x_2 x_3)$$

3. **Permutation 3** $\sigma_3 = [c_3, c_2, c_1]$ This permutation considers c_3 as the first client. Since, for the first client in the permutation, $Z = \emptyset$, $v_1(Z) = 0$, the marginal contribution of c_3 is:

$$\Delta_{c_3} = v_1(\{c_3\}) - v_1(\emptyset) = x_3$$

For c_2, the marginal contribution can be shown as:

$$\Delta_{c_2} = v_1(\{c_3, c_2\}) - v_1(\{c_3\}) = x_2 + \alpha x_2 x_3$$

While for c_1, since there are 2 other players in the coalition already, the coalition before c_1 is $Z = \{c_3, c_2\}$, with value $v_1(Z) = x_3 + x_2 + \alpha x_2 x_3$. The marginal contribution of c_1 is thus given as:

$$\Delta_{c_1} = v_1(\{c_3, c_2, c_1\}) - v_1(\{c_3, c_2\}) = x_1 + \alpha(x_1 x_2 + x_1 x_3).$$

The Shapley Values for each client in Round 1 are then calculated as the weighted average of their marginal contributions across all 3 permutations using Equation 8.1. At the end of Round 1, each participating client receives a Shapley Value $\phi^1_{c_i}(v_1)$ that reflects their contribution. In the next training round, the subset S_2 participates, and the same process is repeated again, The value function $v_2(S_2)$ can be represented as:

$$v_2(S_2) = x_2 + x_4 + x_5 + \alpha(x_2 x_4 + x_2 x_5 + x_4 x_5)$$

Similarly, the marginal contributions of c_2, c_4, and c_5 are calculated for all coalitions within S_2, and the Shapley Values $\phi^2_{c_i}(v_2)$ are determined. Note that c_2 participated in both rounds, so their contributions are evaluated separately in each round.

Once training is completed, the total Shapley Value for each client is computed using Equation 8.7:

$$\phi_{c_i}(v) = \phi^1_{c_i}(v_1) + \phi^2_{c_i}(v_2)$$

For example, since c_2 has contributions in both rounds, its total Shapley Value can be computed as follows:

$$\phi_{c_2}(v) = \phi^1_{c_2}(v_1) + \phi^2_{c_2}(v_2)$$

On the other hand, c_1 only participated in Round 1. Thus:

$$\phi_{c_1}(v) = \phi^1_{c_1}(v_1)$$

The rewards r_i for each client are proportional to their total Shapley Value and can be computed as:

$$r_i = \phi_{c_i}(v) \cdot R \tag{8.8}$$

where R is the budgeted reward pool. Finally, the net profit for each client is:

$$\pi_i = r_i - \text{cost}_i(x_i)$$

where $\text{cost}_i(x_i) = \beta x_i^2$. This mechanism allows us to evaluate client contributions fairly within individual training rounds as well as across rounds to account for varying participation. Since the reward is allocated based on the cumulative Shapley value across rounds, it balances individual contributions with participation costs. However, the computation of Shapley Values, as outlined above, requires evaluating all possible permutations of participating clients within each training round. This leads to a total of $|S_t|!$ permutations for a subset S_t in each round and results in a computational complexity of $O(2^n)$, where n is the number of participants. This can introduce a number of challenges for federated networks, including latency and bottlenecks. To address this challenge, we employ TMC sampling, which provides an efficient approximation to the Shapley Value. The next section explains the TMC sampling, its algorithmic implementation, and its efficiency and accuracy analysis.

8.6 Efficient Shapley Value Estimation via Truncated Monte Carlo

Monte Carlo (MC) methods provide a stochastic approach to approximating the Shapley Value by sampling random coalitions instead of exhaustively enumerating all subsets. The TMC method further optimizes this by terminating the sampling process once the contribution of the client being evaluated can be reasonably estimated. This approach dramatically reduces the number of coalitions that need to be evaluated by limiting the number of sampled permutations and focusing on the most relevant contributions.

The key idea behind TMC sampling is to approximate the marginal contribution of a client c_i by iteratively simulating coalitions in a random order and computing the incremental value added by c_i when added to these coalitions. It operates by iterating over randomly sampled permutations of clients. Instead of traversing the entire sequence of clients in a permutation, it stops after a predefined truncation length L, capturing only the marginal contributions of the first L clients. This truncation is based on the assumption that the most significant contributions to the Shapley Value arise from smaller coalitions and early positions in the permutation. The TMC estimator for the Shapley Value can be defined by:

$$\hat{\phi}_i(v) = \frac{1}{M} \sum_{m=1}^{M} \Delta_i^{(m)} \tag{8.9}$$

where:

- $\hat{\phi}_i(v)$ is the approximate Shapley Value for client c_i

Algorithm 4 Truncated Monte Carlo Sampling for Shapley Value Estimation

```
Input: Value function $v$, Set of clients $C$, Total samples $M$, Truncation length $L$
Output: Approximate Shapley values $\hat{\phi}$ for all clients in $C$
 1: Initialize $\hat{\phi}_i = 0$ for each client $c_i \in C$
 2: for $m = 1$ to $M$ do
 3:     Randomly permute the clients in $C$ to obtain a sequence $\sigma$
 4:     Initialize the coalition $Z = \emptyset$
 5:     for $k = 1$ to $\min(L, |C|)$ do
 6:         Let $c_i = \sigma[k]$ (the $k$-th client in the permutation)
 7:         Compute the marginal contribution $\Delta = v(Z \cup \{c_i\}) - v(Z)$
 8:         Update $\hat{\phi}_i = \hat{\phi}_i + \Delta$
 9:         Update $Z = Z \cup \{c_i\}$
10:     end for
11: end for
12: Normalize $\hat{\phi}_i = \hat{\phi}_i / M$ for all clients $c_i \in C$
13: return $\hat{\phi}$
```

- M is the total number of sampled permutations
- $\Delta_i^{(m)}$ is the marginal contribution of client c_i in the m-th sampled truncated permutation.

During each iteration, a random permutation of the clients is sampled, and contributions are evaluated only up to the truncation length L, beyond which no further coalitions are considered. This significantly reduces computation while maintaining a strong approximation of the true Shapley Value. The parameters M (the number of sampled permutations) and L (the truncation length) balance computational efficiency and approximation accuracy and can be adjusted based on the system's available resources and desired accuracy. Larger M improves precision while L ensures that the truncation captures meaningful contributions without unnecessary overhead. Algorithm 4 presents the pseudocode for the TMC sampling process for approximating the Shapley Value in our framework.

Once the Shapley Values are approximated for each round using TMC sampling, the total Shapley Value for each client is computed by summing their contributions across all training rounds. Using Equation 8.7, the total estimated Shapley Value of client c_i can be represented by:

$$\hat{\phi}_{c_i}(v) = \sum_{t=1}^{T} \hat{\phi}_{c_i}^{t}(v_t) \tag{8.10}$$

The rewards are then distributed proportionally to their total Shapley Values. Algo-

Algorithm 5 Fairness-aware Incentive Mechanism using Shapley Value and Truncated Monte Carlo

Input: Set of clients C, Total training rounds T, Value functions $\{v_t\}_{t=1}^{T}$, Budget R, Parameters M, L for TMC sampling
Output: Rewards $\{r_i\}_{i=1}^{n}$ and net profits $\{\pi_i\}_{i=1}^{n}$ for all clients

```
1: Initialize $\hat{\phi}_{c_i} = 0$ for each client $c_i \in C$
2: for $t = 1$ to $T$ do
3:     Let $S_t \subseteq C$ be the set of participating clients in round $t$
4:     Approximate Shapley Values $\{\hat{\phi}_{c_i}^{t}\}_{c_i \in S_t}$ using Algorithm 4 with inputs $v_t, S_t, M, L$
5:     for $c_i \in S_t$ do
6:         $\hat{\phi}_{c_i} \leftarrow \hat{\phi}_{c_i} + \hat{\phi}_{c_i}^{t}$
7:     end for
8: end for
9: $\hat{\phi}_{c_i} \leftarrow \hat{\phi}_{c_i} / \sum_{j=1}^{n} \hat{\phi}_{c_j}$ for all $c_i \in C$
10: $r_i \leftarrow \hat{\phi}_{c_i} \cdot R$ for all $c_i \in C$
11: $\pi_i \leftarrow r_i - \text{cost}_i(x_i)$ for all $c_i \in C$, where $\text{cost}_i(x_i) = \beta x_i^2$
12: return $\{r_i\}_{i=1}^{n}, \{\pi_i\}_{i=1}^{n}$
```

rithm 5 provides a comprehensive view of the complete incentive mechanism.

8.6.1 Computational Efficiency Analysis of TMC Estimation for Shapley Value

To analyze the efficiency of TMC Estimation for Shapley Value, we extend the same example of an FL system with five clients $C = \{c_1, c_2, c_3, c_4, c_5\}$, where $S_1 = \{c_1, c_2, c_3\}$ participate in Round 1 and $S_2 = \{c_2, c_4, c_5\}$ participate in Round 2. The value functions for the rounds, using Equation 8.5, are given as:

$$v_1(S_1) = x_1 + x_2 + x_3 + \alpha(x_1x_2 + x_1x_3 + x_2x_3)$$

$$v_2(S_2) = x_2 + x_4 + x_5 + \alpha(x_2x_4 + x_2x_5 + x_4x_5)$$

We now demonstrate how TMC approximates the Shapley Value for these clients using $M = 3$ sampled permutations and a truncation length $L = 2$.

In Round 1, the TMC estimator, as given in Equation 8.9, randomly samples three permutations of the participating clients:

1. **Permutation 1 $\sigma_1 = [c_2, c_1, c_3]$:**

   In this permutation, c_2 is the first client in the coalition. Using the same formulations as established in the case of exact Shapley Value computation:

- For the first client, $Z = \{c_2\}, \quad v(Z) = x_2$. Thus the marginal contribution can be computed using Equation 8.6 as $\Delta_{c_2} = x_2$.
- For second client in the permutation, $Z = \{c_2, c_1\}, \quad v(Z) = x_2 + x_1 + \alpha x_1 x_2$ and the marginal contribution is computed as $\Delta_{c_1} = x_1 + \alpha x_1 x_2$.

Since the truncation length is $L = 2$, it limits further computation. Hence, the contributions of c_3 are truncated and not evaluated.

2. **Permutation 2 $\sigma_2 = [c_3, c_1, c_2]$:**

- First client in this permutation is c_3. Thus, $Z = \{c_3\}, \quad v(Z) = x_3$ and the marginal contribution is $\Delta_{c_3} = x_3$.
- Client c_1 is the second player in this coalition. Thus, $Z = \{c_3, c_1\}, \quad v(Z) = x_3 + x_1 + \alpha x_1 x_3$ and marginal contribution is $\Delta_{c_1} = x_1 + \alpha x_1 x_3$.

The contribution evaluation of c_2 are truncated due to $L = 2$.

3. **Permutation 3 $\sigma_3 = [c_1, c_3, c_2]$:**

- Since c_1 is the first client in this permutation, $Z = \{c_1\}, \quad v(Z) = x_1$ and its marginal contribution is computed as $\Delta_{c_1} = x_1$.
- Second client if c_3. Hence, $Z = \{c_1, c_3\}, \quad v(Z) = x_1 + x_3 + \alpha x_1 x_3$ and the marginal contribution of c_3 is $\Delta_{c_3} = x_3 + \alpha x_1 x_3$.

Again, contributions of c_2 are truncated due to truncation length $L = 2$.

The approximate Shapley Values for Round 1 are then calculated by averaging the marginal contributions across the sampled permutations:

$$\hat{\phi}^1_{c_1} = \frac{1}{3}\left((x_1 + \alpha x_1 x_2) + (x_1 + \alpha x_1 x_3) + x_1\right)$$

$$\hat{\phi}^1_{c_2} = \frac{1}{3}\left(x_2 + 0 + 0\right)$$

$$\hat{\phi}^1_{c_3} = \frac{1}{3}\left(0 + x_3 + (x_3 + \alpha x_1 x_3)\right)$$

The Shapley Value for Round 2, $S_2 = \{c_2, c_4, c_5\}$, can be approximated using the same TMC sampling process. We sample three random permutations of the participating clients and evaluate the marginal contributions for the first $L = 2$ clients in each permutation. The approximate Shapley Values for Round 2 can then be shown as:

$$\hat{\phi}^2_{c_2} = \frac{1}{3}\left((x_2 + \alpha x_2 x_4) + x_2 + (x_2 + \alpha x_2 x_5)\right)$$

$$\hat{\phi}^2_{c_4} = \frac{1}{3}\left(x_4 + 0 + 0\right)$$

$$\hat{\phi}^2_{c_5} = \frac{1}{3}\left(0 + (x_5 + \alpha x_2 x_5) + x_5\right)$$

By sampling only 3 permutations and truncating at $L = 2$, TMC significantly reduces computation compared to the exact Shapley Value calculation, which would require evaluating all possible permutations (6 per round). Once the approximate Shapley Values for each client are computed for individual rounds, the total Shapley Value for each client is calculated by summing their contributions using Equation 8.10:

$$\hat{\phi}_{c_i}(v) = \sum_{t=1}^{T} \hat{\phi}^t_{c_i}(v_t) \tag{8.11}$$

Efficiency Gains

In the exact Shapley Value computation, all permutations of S_t must be evaluated which results in:

- Total permutations for S_1: $|S_1|! = 3! = 6$
- Total permutations for S_2: $|S_2|! = 3! = 6$

Each permutation requires computing marginal contributions for all clients, resulting in 18 total computations per round. Across just two rounds evaluated in the example, the exact Shapley Value calculation involves 36 computations. On the other hand, using TMC with $M = 3$ sampled permutations and truncation $L = 2$ involve:

- Total computations for S_1: $M \times L = 6$
- Total computations for S_2: $M \times L = 6$

This results in a total of 12 computations across both rounds, reducing the number of computations by 66.7%. While the efficiency gains depend on the chosen M and L, it can be safely concluded that TMC achieves significant efficiency gains while maintaining an accurate approximation of the Shapley Values. We further present the accuracy analysis of our estimation in the next section.

8.6.2 Accuracy Analysis of TMC Estimation

The accuracy of the TMC estimator, $\hat{\phi}_i(v)$, depends on two critical factors: the number of samples M and the truncation length L. These parameters govern the trade-off between computational efficiency and approximation precision. The goal is to approximate the true Shapley Value $\phi_i(v)$ with high accuracy while ensuring scalability in large-scale systems. Here, the interplay of M and L introduces two key sources of error in the TMC estimator:

- **Bias:** A systematic error arising because truncation limits the evaluation of coalitions to the first L clients in each permutation, excluding potential contributions from larger coalitions.
- **Variance:** A stochastic error resulting from the randomness of sampling, reflecting the variability in marginal contributions across different sampled permutations.

These two factors directly impact the expectation and variance of the TMC estimator, which are key to understanding its accuracy. Here, the expectation of the TMC estimator represents the average value the estimator would converge to over an infinite number of samples and is given by:

$$\mathbb{E}[\hat{\phi}_i(v)] = \phi_i(v) + \text{Bias}(M, L) \tag{8.12}$$

where $\phi_i(v)$ is the true Shapley Value, and $\text{Bias}(M, L)$ is the error introduced by truncation and finite sampling. While increasing L reduces bias by capturing more significant contributions, it also increases computational costs. Therefore, L must be carefully chosen to balance computational efficiency and bias minimization.

Similarly, minimizing the variance introduced by the sampling process ensures the accuracy of the approximation. The variance of the estimator is given by:

$$\text{Var}(\hat{\phi}_i(v)) = \frac{1}{M}\left(\mathbb{E}[\Delta_i^2] - (\mathbb{E}[\Delta_i])^2\right) \tag{8.13}$$

where Δ_i denotes the marginal contribution of client c_i in a sampled coalition. Increasing M reduces this variance, improving the stability and accuracy of the estimator. To establish performance guarantees for the TMC estimator, we analyze its error bounds under the assumption that the value function $v(S)$ is Lipschitz continuous with respect to coalition size.

Theorem 1: Given a truncation length L and a number of samples M, for any tolerance level $\epsilon > 0$ and confidence level $1 - \delta$, there exist sufficiently large M and L such that the TMC estimator satisfies the following bound:

$$\Pr\left(|\hat{\phi}_i(v) - \phi_i(v)| < \epsilon\right) \geq 1 - \delta \tag{8.14}$$

Proof: We combine Hoeffding's concentration inequality with an analysis of truncation bias to bound the deviation of the estimator from the true value. Hoeffding's inequality provides a bound on the probability that the sample mean deviates from its expected value by more than ϵ. For M independent samples of marginal contributions Δ_i:

$$\Pr\left(|\hat{\phi}_i(v) - \mathbb{E}[\hat{\phi}_i(v)]| > \epsilon\right) \leq 2\exp\left(-\frac{2M\epsilon^2}{\text{Range}(\Delta_i)^2}\right) \tag{8.15}$$

where $\text{Range}(\Delta_i) = \max \Delta_i - \min \Delta_i$ is the maximum possible value range of the marginal contributions Δ_i. This inequality shows that as M increases, the probability of large deviations between the estimator $\hat{\phi}_i(v)$ and its expected value $\mathbb{E}[\hat{\phi}_i(v)]$ decreases exponentially.

We have already discussed that the truncation length L introduces bias by excluding contributions from coalitions. This bias depends on the smoothness of the value function $v(S)$. If $v(S)$ is Lipschitz continuous with respect to the coalition size, the contributions of excluded coalitions diminish as L increases. Formally:

$$\text{Bias}(M, L) = \phi_i(v) - \mathbb{E}[\hat{\phi}_i(v)] \tag{8.16}$$

where $\text{Bias}(M, L)$ can be controlled by selecting L large enough to include the most significant contributions. The total error is the combination of the truncation bias and the sampling variance:

$$|\hat{\phi}_i(v) - \phi_i(v)| \leq |\hat{\phi}_i(v) - \mathbb{E}[\hat{\phi}_i(v)]| + |\mathbb{E}[\hat{\phi}_i(v)] - \phi_i(v)| \tag{8.17}$$

Now, using Hoeffding's inequality for the first term and controlling the second term by choosing L appropriately, we can ensure the total error is bounded by ϵ with high probability. Specifically, by choosing M and L such that:

$$\Pr\Big(|\hat{\phi}_i(v) - \mathbb{E}[\hat{\phi}_i(v)]| > \epsilon/2\Big) \leq \delta/2 \tag{8.18}$$

and ensuring the bias satisfies:

$$|\mathbb{E}[\hat{\phi}_i(v)] - \phi_i(v)| \leq \epsilon/2 \tag{8.19}$$

the total probability of exceeding ϵ can be bounded as:

$$\Pr\Big(|\hat{\phi}_i(v) - \phi_i(v)| < \epsilon\Big) \geq 1 - \delta \tag{8.20}$$

Thus, for sufficiently large M and L, the TMC estimator approximates the true Shapley Value $\phi_i(v)$ with high confidence and precision.

8.7 Experimental Evaluation and Analysis

This section evaluates the proposed incentive mechanism using various metrics to validate its effectiveness, efficiency, and fairness. The experiments are designed to compare the accuracy of the proposed TMC approximation with other Shapley Value approximations, assess its payout distribution, and analyze its resilience to free-riding attacks.

The experiments were conducted on a simulated FL environment. Since computing the exact Shapley Value is computationally expensive, we start with a small client pool of five clients. The clients contributed data and computational resources, and their contributions were evaluated using the Shapley Value. To ensure robustness, we introduced free riders—clients that contributed negligible data or computational resources—to test the mechanism's ability to fairly reward benign clients while penalizing malicious actors.

8.7.1 Comparison of Shapley Value Approximations

We compare the cumulative Shapley Values across epochs for five clients using different approximation methods: Exact Shapley, k-subset Shapley, SOR Shapley, TMR Shapley, and the proposed TMC Shapley, with the Exact Shapley Value serving as the baseline. These methods vary in their approach to approximating the Shapley Value:

- **Exact Shapley** computes the true Shapley Value by evaluating the marginal contributions of each client across all possible coalitions, resulting in a computational complexity of $O(2^n)$. This serves as the gold standard for fairness but is computationally infeasible for large-scale systems.
- k-subet Shapley subset Shapley reduces the complexity by sampling k random subsets of clients for each evaluation instead of all possible coalitions. While this method

improves efficiency, it introduces biases that can lead to deviations from the true Shapley Value, particularly in scenarios with high heterogeneity among client contributions.

- **SOR Shapley** uses the Sampling of Random Orders (SOR) approach, where only a subset of client permutations is considered, reducing the computational burden. However, this method often overestimates contributions due to its heuristic-based coalition sampling, especially for clients with high interaction effects.

- **TMR Shapley** employs Truncated Marginal Regression (TMR) to approximate Shapley Values. It iteratively fits marginal contribution curves but struggles to capture higher-order interaction effects among clients, leading to systematic underestimation of contributions.

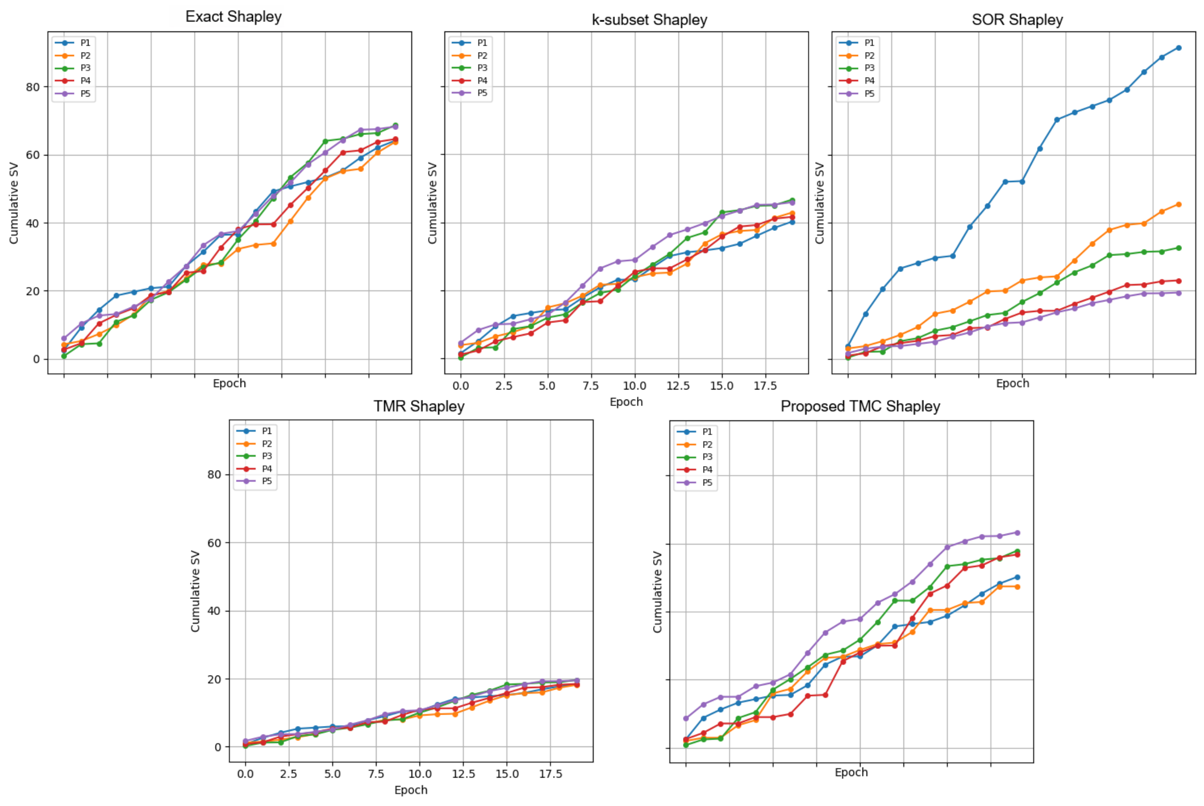


Figure 8.1: Comparison of cumulative Shapley Values across different approximation methods.

As shown in Figure 8.1, the proposed TMC method demonstrates significant improvements over these approximations. The cumulative Shapley Values obtained using TMC closely align with the Exact Shapley, with minimal deviations observed across epochs. This is because the TMC approach leverages truncated sampling of coalitions, focusing computational effort on the most impactful coalitions while maintaining accuracy. In contrast, k-subset Shapley and SOR Shapley deviate considerably due to their reliance on coarse-grained sampling strategies. TMR Shapley, while more stable, underestimates contributions due to its inability to account for higher-order interactions effectively. The results confirm that the TMC method strikes a balance between computational efficiency and accuracy, making it well-suited for large-scale Crowdsourced Federated Learning systems. Its ability to approximate Shapley Values closely to the Exact Shapley ensures fairness in reward distribution while overcoming the computational challenges posed by exhaustive evaluations.

8.7.2 Payout distribution

To validate the fairness of reward allocation using the proposed Truncated Monte Carlo (TMC) method, we compared the payout distribution derived from TMC-based Shapley Values to the distribution based on Exact Shapley Values. The results, depicted in Figure 8.2, demonstrate a high degree of alignment between the two methods. The payouts for each client are proportional to their cumulative Shapley Values, ensuring that clients contributing more to the global model receive higher rewards.

The distribution highlights the robustness of the TMC approach in approximating the true Shapley Value while maintaining fairness. For example, Client 2, which has the highest contribution to the model, receives the largest payout across both Exact and TMC methods, with only a marginal difference of 1.3%. Similarly, the payouts for other clients also closely align, with deviations of less than 2% in all cases. This demonstrates that the TMC method captures the relative contributions of clients accurately, ensuring that the

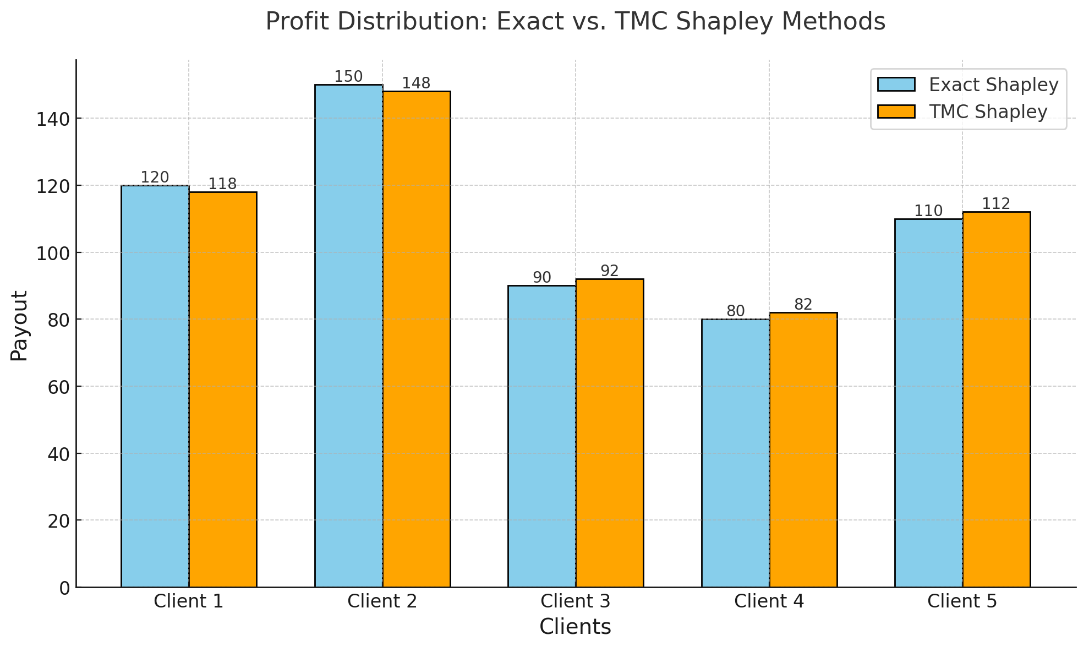


Figure 8.2: Payout comparison of proposed TMC Shapley Value

proportionality in reward allocation is preserved.

Such consistency is critical for fostering trust in FL systems, as it ensures that participants are compensated fairly based on their contributions without introducing significant biases. Moreover, the minimal deviation between Exact and TMC payouts underscores the computational efficiency of the TMC method, which achieves fairness without the prohibitive overhead associated with Exact Shapley Value computation. This makes the TMC method a viable choice for large-scale deployments where scalability and fairness are both paramount.

8.7.3 Resilience Against Free-Riding Attacks

Free-riding attacks pose a significant challenge in FL systems, where some clients may attempt to claim rewards without contributing meaningfully to the global model. To evaluate the resilience of the proposed incentive mechanism against such behavior, we simulated a FL scenario involving both benign clients and free riders. Figure 8.3 illustrates the Shapley Values assigned to benign clients versus free riders over 100 training rounds.

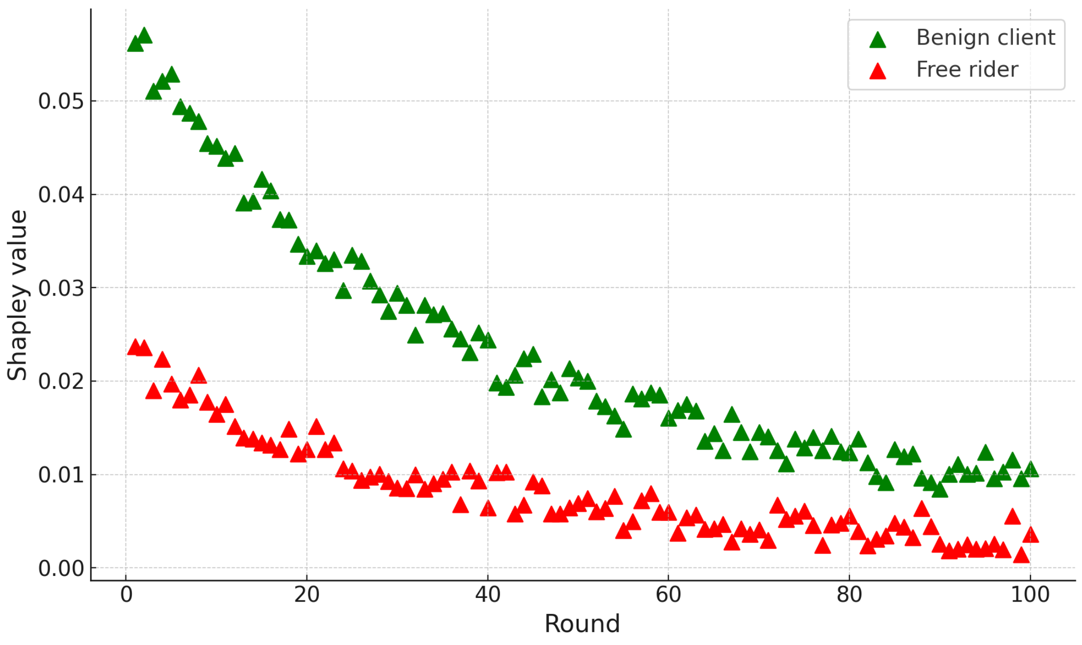


Figure 8.3: Comparison of Shapley Values assigned to Benign clients vs Free riders

The results reveal that benign clients consistently receive higher Shapley Values than free riders, reflecting their substantial contributions to the model's performance. In contrast, the Shapley Values for free riders decline sharply in the early rounds and stabilize near zero, indicating the mechanism's ability to identify and penalize non-contributing participants. This behavior is achieved by leveraging the Shapley Value's marginal contribution framework, which directly ties rewards to tangible improvements in model performance. Since free riders contribute little to no value, their marginal contributions—and hence their Shapley Values—are minimal.

Furthermore, the ability of the incentive mechanism to dynamically evaluate contributions on a per-round basis ensures that free riders are unable to exploit the system over time. As the training progresses, the Shapley Values of benign clients exhibit consistent growth, affirming that the mechanism promotes sustained and meaningful participation. These findings demonstrate the robustness of the proposed TMC-based incentive mechanism in deterring free-riding behavior, ensuring that rewards are distributed equitably among clients who genuinely contribute to the success of the FL process. This resilience enhances the overall reliability and fairness of the system, making it well-suited for real-world CFL deployments.

The experimental results validate the proposed incentive mechanism's effectiveness in addressing key challenges in Crowdsourced FL. It ensures efficient and fair reward distribution while providing resilience by effectively penalizing free riders.

8.8 Limitations and Future Directions

While the proposed TMC-based Shapley Value incentive mechanism offers efficiency, several limitations remain. First, the use of TMC sampling introduces bias compared to exact Shapley Value computation. Since sampling may stop early when a client's marginal contribution is clear, later-positioned clients in the permutation may be underrepresented. This loss of fairness can disproportionately reward clients with unique or early-stage contributions. Although increasing the number of samples or the truncation length can help reduce this bias, it comes at the cost of additional computation.

In its current form, the reward mechanism operates offline, with rewards calculated only after training is complete. A more responsive mechanism could dynamically adjust rewards during training based on real-time system performance and client behavior. This would better align incentives, especially in long-running or streaming FL tasks, and could help discourage free-riding or erratic participation.

Deploying this mechanism in real-world systems involves additional overhead. Even though TMC reduces computation relative to the exact Shapley Value, it still requires tracking contributions per round and computing marginal improvements, which may impact system latency and resource usage. Moreover, system stability under changing client behavior or data quality remains a practical concern. Monitoring and adapting to these changes requires infrastructure that can handle fluctuating client sets and performance metrics.

There is also a broader design space of incentive mechanisms that could be considered. Alternatives such as auction-based strategies, reputation systems, or contract-theoretic approaches may offer different trade-offs in terms of simplicity, scalability, and robustness. Integrating these with Shapley-based methods may yield hybrid solutions that are more adaptable in heterogeneous or adversarial environments.

Future work could also investigate how to adapt the mechanism for diverse FL platforms and resource-constrained settings, while preserving fairness. For example, lightweight approximations or tiered reward models might help scale the mechanism without compromising its core principles.

8.9 Summary

In this chapter, we proposed a fairness-aware incentive mechanism for Crowdsourced FL using the Shapley Value as the foundation for reward allocation. By leveraging the TMC estimator, we addressed the computational inefficiencies of exact Shapley Value computation, making the mechanism scalable for large FL systems. The proposed method was rigorously evaluated against existing approximations, demonstrating competitive accuracy while significantly reducing computational overhead.

Our experiments highlighted the mechanism's ability to fairly distribute rewards among clients based on their contributions, ensuring robustness against free-riding attacks and adversarial behaviors. Benign clients consistently received higher rewards, reflecting their meaningful contributions to the global model. This approach not only incentivizes genuine participation but also fosters trust and sustainability in CFL environments. The findings from this chapter lay a solid foundation for future extensions aimed at improving fairness, scalability, and resilience in FL systems.

Chapter 9

Conclusion

9.1 Summary of Contributions

This dissertation represents a significant step towards transforming Federated Learning (FL) into a dynamic and scalable paradigm, capable of adapting to real-world challenges. The research introduces FedOnDemand, a novel crowdsourced FL framework that emphasizes voluntary participation, scalability, and security, addressing key limitations of traditional FL systems. Below, we outline the major contributions of this work:

First, this research introduces the concept of a dynamic and open FL ecosystem. By incorporating principles of crowdsourcing, the FedOnDemand framework creates an environment where clients can join and leave the learning process based on their availability and willingness to contribute. This dynamic participation ensures that the framework can scale seamlessly to accommodate the variability and diversity of edge devices, which is essential for building an inclusive and robust FL system.

To address the inherent heterogeneity of edge environments, this thesis proposes innovative mechanisms to optimize the FL training process. A multi-criterion client selection protocol was developed to evaluate and select clients based on their computational and communication resources. It also considers the client's data quality and trustworthiness as a critical criterion for client selection. This adaptive approach not only improves the efficiency of the training process but also ensures that the most impactful clients are prioritized, leading to enhanced global model performance. Additionally, the integration of Bayesian Federated Learning introduces uncertainty modeling into the FL pipeline, providing a robust method for managing non-IID data and ensuring reliable model convergence in heterogeneous environments.

Recognizing the critical importance of security in decentralized systems, this research integrates zero-trust principles into FL. By implementing Attribute-Based Access Control (ABAC) and Public Key Infrastructure (PKI) mechanisms, the proposed framework enforces strict access controls and prevents unauthorized client participation. These security measures protect the integrity of the FL process while mitigating risks such as data poisoning and malicious updates.

Another key contribution is the design of a fairness-aware incentive mechanism that leverages Shapley Value-based data valuation to measure client contributions. This mechanism ensures that rewards are distributed equitably based on the value of each client's data and resources. By aligning incentives with contributions, this approach fosters trust and motivates active participation, addressing a critical gap in existing FL frameworks.

Empirical validation through extensive simulations and experiments using benchmark and real-world datasets highlights the effectiveness of the proposed methods. The results demonstrate significant improvements in model accuracy, convergence speed, and system robustness compared to traditional FL approaches. These findings confirm that the integration of dynamic client participation, optimized resource utilization, Bayesian learning techniques, robust security measures, and fair incentivization collectively enhances the operational efficiency and scalability of FL systems. Together, these contributions establish a solid foundation for the next generation of FL systems.

9.2 Key Findings and Insights

This dissertation addresses several fundamental challenges in FL, offering theoretical and empirical advancements in client selection, Bayesian learning, security, and incentivization. While the overarching FedOnDemand framework proposes a theoretical solution for scalability and adaptability in FL, this section emphasizes the practical findings and insights gained from implementing and validating its key components.

- The multi-criterion client selection mechanism demonstrated its effectiveness in dynamically optimizing the selection of clients based on diverse metrics such as computational resources, data quality, and network conditions. Experimental results confirmed that this approach significantly improved global model accuracy and convergence rates, particularly in heterogeneous environments. However, the implementation also revealed inherent trade-offs. For instance, while prioritizing higher-quality clients maximized model performance, it occasionally led to underutilization of other clients, raising questions about fairness in participation. Balancing model optimization with equitable client engagement remains an open area for exploration.

- The integration of Bayesian learning principles addressed critical challenges related to non-IID data distributions. By employing Stochastic Variational Inference (SVI) on the client side and hierarchical Bayesian aggregation on the server side, the proposed framework achieved robust model performance despite high variability in client contributions. These methods not only improved accuracy but also provided uncertainty estimates, making the learning process more interpretable. However, the computational overhead introduced by Bayesian techniques, particularly on resource-constrained devices, posed practical challenges. Future refinements could explore lightweight Bayesian approximations to reduce this overhead.

- The zero-trust security model, incorporating Attribute-Based Access Control (ABAC) and Public Key Infrastructure (PKI), effectively protects the FL process against unauthorized participation and adversarial threats. While this approach provided fine-grained control over client access and reduced the risk of data poisoning and malicious updates, it introduced significant computational and communication overhead. The overhead arises from the need to perform continuous authentication and authorization. Verifying digital certificates and enforcing access control policies consume resources and may introduce latency as well, especially in environments with a large number of clients or limited computational capacity. Nevertheless, these trade-offs between accuracy, efficiency, and security reflect the inherent complexities of designing a practical zero-truth FL framework and highlight the need for scalable authentication protocols in future deployments.

- The fairness-aware incentive mechanism, grounded in Shapley Value calculations, proved effective in promoting sustained client participation. Experimental evaluations showed that clients contributing higher-quality data were rewarded proportionally, addressing concerns of inequity and fostering long-term engagement. However, the computational complexity of Shapley Value estimation, even with efficient approximations like TMC, remains a limiting factor in large-scale deployments. While TMC achieved significant efficiency gains, balancing computational feasibility with accuracy continues to be a critical area for optimization.

While the individual components of FedOnDemand were validated through experiments, the full realization of this crowdsourced FL framework remains theoretical. The integration of dynamic client participation, secure access control, Bayesian learning, and incentivization holds promise for addressing scalability and adaptability in FL. However, implementing such a system at scale would require addressing several challenges, including real-time client orchestration, communication efficiency, and ensuring equitable access to FL tasks across diverse devices. These challenges highlight the complexity of transitioning from theoretical frameworks to practical implementations. However, we believe this dissertation provides fertile ground for future research and would guide future FL system designs.

9.3 Future work

The findings of this dissertation pave the way for several avenues of future research aimed at advancing the practicality and scalability of crowdsourced FL systems. While the proposed FedOnDemand framework outlines a promising direction, its realization at scale requires overcoming numerous technical and operational challenges.

- **Scalable and Adaptive FL Architecture:** One significant area for future work is the integration of real-time client orchestration and adaptive resource allocation mechanisms. One direction is to integrate reinforcement learning (RL) for client and resource orchestration: an RL agent could learn to dynamically schedule clients and tune training configurations based on system state and historical performance. Initial studies have demonstrated that a trust-aware deep RL approach can effectively select appropriate clients by balancing resource consumption and training time. Similarly, the communication strategy between the server and clients can be made adaptive to reduce overhead. For instance, clients might adjust their update frequency or compress model updates based on real-time bandwidth predictions. Such adaptive communication protocols would mitigate communication bottlenecks when many devices participate, addressing current scalability limitations. In summary, enhancing FedOnDemand with learning-based orchestration and communication adaptation promises a more efficient and resilient FL platform.

- **Dynamic Client Selection Strategies:** Our multi-criterion client selection method can be further improved to handle more complex and dynamic scenarios. A key limitation noted was the static weighting of selection criteria, which might not adapt optimally to changing data distributions or client availability. Future research could formulate client selection as a sequential decision problem and apply reinforcement learning or bandit algorithms to continually refine the selection policy. A deep RL agent, for example, could learn which clients to invite in each round to maximize long-term model performance under resource constraints. This would allow the system to automatically discover the best trade-offs between multiple criteria (e.g., accuracy, latency, energy) rather than relying on preset heuristics. Another promising direction is to incorporate additional factors such as model uncertainty or data novelty into the selection process. By prioritizing clients that offer the most informative data (analogous to active learning), the federation could converge faster and generalize better. The selection mechanism could also be made fairness-aware: for instance, ensuring that clients with lower resources or those from underrepresented data groups are not consistently neglected. Finally, scaling the selection process to very large client pools may require hierarchical or decentralized approaches (selection done in stages or by regional aggregators) to reduce complexity. By addressing these issues, the client selection strategy would become more robust against the heterogeneity and unpredictability inherent in crowdsourced FL.

- **Efficient Bayesian Inference in FL:** We noted several limitations in BayFL-SVI, especially the computational and communication overhead that it introduces. Another limitation of the current approach is the approximation error introduced by SVI and the constrained form of the variational posterior. Future work could investigate more exact or flexible Bayesian inference techniques, albeit with careful consideration of their feasibility in a federated setting. For example, one could employ Markov Chain Monte Carlo (MCMC) methods to sample from the posterior distribution of the global model. Recent studies have shown that combining local MCMC sampling on clients with efficient global updates can improve convergence to better optima. While MCMC is computationally intensive, research into federated posterior sampling (e.g., parallel chains on clients or tempered sampling strategies) could make it practical. Another avenue is to explore hybrid approaches that retain the scalability of SVI while borrowing accuracy improvements from MCMC, such as using MCMC periodically to recalibrate the variational approximation. Beyond inference algorithms, future Bayesian FL could consider models with hierarchical or personalized components to better capture client-specific variability and uncertainty. Theoretical work is also needed to understand the convergence properties of Bayesian FL – for instance, studying conditions under which the federated posterior approaches the true posterior as more clients and data participate. Advancing the Bayesian FL component will likely involve a mix of more powerful inference methods (like MCMC), new Bayesian model architectures, and analyses of uncertainty's role in improving federated outcomes.

- **Decentralized and Trust-Aware Security Models:** In the current zero-trust FL design, we assumed a Public Key Infrastructure (PKI) and Attribute-Based Access Control (ABAC) for authenticating clients and restricting their actions. To improve scalability and ease of deployment, future work could explore integrating ABAC with modern Identity and Access Management (IAM) systems. IAM frameworks can simplify client onboarding, centralize policy enforcement, and provide visibility into access patterns across dynamic networks. These systems also allow real-time adaptation by updating trust scores based on observed behavior, such as protocol compliance, model contribution quality, and consistency. Such scores can guide authorization decisions dynamically, enhancing both security and flexibility. Additionally, future directions may include investigating alternative or certificate-less authentication approaches, such as identity-based encryption or lightweight key agreement protocols. These methods reduce the reliance on traditional certificate management, which can be burdensome in edge or resource-constrained environments. Exploring such mechanisms could help streamline authentication while preserving the zero-trust principles required for secure and adaptive federated learning deployments.

- **Fairness and Incentivization at Scale:** Ensuring fairness among heterogeneous clients remains an open challenge that extends beyond contribution-based reward allocation. While this dissertation employs Truncated Monte Carlo (TMC) sampling to approximate the Shapley Value efficiently, future work could explore ways to more tightly integrate this fairness mechanism with the training dynamics. For instance, recent studies have suggested using Shapley-based weights during model aggregation to improve global model quality, especially in imbalanced or non-IID data settings. Building on this, future work could investigate personalizing model updates or final models based on individual contributions, thereby promoting fairness not just in compensation but also in the model's outcomes. Another promising direction involves blending contribution-based incentives with fairness objectives from broader machine learning research. Methods like q-Fair FL, which aim to equalize model performance across clients with differing data quality or device capabilities, could be incorporated to prevent the system from disproportionately benefiting high-resource participants. Additionally, applying tools from cooperative game theory and mechanism design could help analyze how rational clients might adapt their behavior in response to incentive schemes and identify strategies to guide these behaviors toward more equitable outcomes. Ultimately, a more holistic view of fairness—one that accounts for both contribution and utility—will be critical for building robust, inclusive crowdsourced FL systems.

Lastly, the theoretical promise of the FedOnDemand framework needs to be validated through large-scale real-world experiments. This includes evaluating the interplay of the proposed client selection, Bayesian aggregation, security, and incentive mechanisms in diverse environments. Such studies would provide practical insights and uncover unforeseen challenges, ultimately bridging the gap between theoretical advancements and practical deployments. By addressing these challenges, future research can unlock the full potential of crowdsourced FL systems, enabling them to scale effectively, operate securely, and foster equitable participation in real-world applications.

Appendix A

Appendix

A.1 PAC-Bayesian Bound for Sample Complexity Estimation in Federated Learning with Non-IID Data

Theorem. Consider a federated learning setup with K clients, each possessing its own dataset D_k, where the number of samples for client k is denoted as n_k and the total number of samples across all clients is $N = \sum_{k=1}^{K} n_k$. Let the loss function $\ell(h_k(x_k), y_k)$ be bounded by a constant C, i.e., $\ell(h_k(x_k), y_k) \in [0, C]$, and assume that each client holds a prior distribution $p(\theta)$ and a posterior distribution $q_i(\theta \mid \phi_i)$, where $q_i(\theta \mid \phi_i)$ is learned via Stochastic Variational Inference (SVI). Then, the sample complexity N required to achieve a generalization error ϵ with high probability $1 - \delta$ is bounded as:

$$N \geq \frac{8\epsilon^2}{C^2} \left(\sum_{k=1}^{K} \left(\mathrm{KL}(q_i(\theta \mid \phi_i) \| p(\theta)) + \sigma_k^2 \right) + \log \left(\frac{1}{\delta} \right) \right)$$

where σ_k^2 is the variance of the loss function for client k and $\mathrm{KL}(q_i(\theta \mid \phi_i) \| p(\theta))$ is the Kullback-Leibler divergence between the variational distribution $q_i(\theta \mid \phi_i)$ and the prior $p(\theta)$.

Proof. We begin by defining the empirical risk and population risk for each client k. The empirical risk is given by:

$$\hat{L}_k(q_i(\theta \mid \phi_i)) = \mathbb{E}_{h_k \sim q_i(\theta|\phi_i)} \left[\frac{1}{n_k} \sum_{i=1}^{n_k} \ell(h_k(x_k^i), y_k^i) \right]$$

and the population risk is defined as:

$$L_k(q_i(\theta \mid \phi_i)) = \mathbb{E}_{h_k \sim q_i(\theta|\phi_i)} \mathbb{E}_{(x_k,y_k)\sim D_k}[\ell(h_k(x_k), y_k)]$$

The variance of the loss function for client k is:

$$\sigma_k^2 = \mathbb{E}_{(x_k,y_k)\sim D_k}[\ell(h_k(x_k), y_k)^2] - \left(\mathbb{E}_{(x_k,y_k)\sim D_k}[\ell(h_k(x_k), y_k)]\right)^2$$

To bound the deviation between the empirical risk and the population risk, we apply Bernstein's inequality. For each client k, Bernstein's inequality provides:

$$P\left(L_k(q_i(\theta \mid \phi_i)) \leq \hat{L}_k(q_i(\theta \mid \phi_i)) + \sqrt{\frac{2\sigma_k^2 \log(1/\delta')}{n_k}} + \frac{3C \log(1/\delta')}{n_k} \right) \geq 1 - \delta'$$

where δ' is the failure probability for each individual client.

To maintain an overall confidence level of $1 - \delta$ across all K clients, we apply the union bound. By setting $\delta' = \frac{\delta}{K}$, we ensure that the cumulative failure probability does not exceed δ. Thus, with probability at least $1 - \delta$, the population risk across all clients satisfies:

$$L(Q_1, \ldots, Q_K) \leq \hat{L}(Q_1, \ldots, Q_K) + \sum_{k=1}^{K} p(k) \left(\sqrt{\frac{2\sigma_k^2 \log\left(\frac{K}{\delta}\right)}{n_k}} + \frac{3C \log\left(\frac{K}{\delta}\right)}{n_k} \right)$$

where $p(k) = \frac{n_k}{N}$ represents the weight associated with client k's contribution, reflecting data heterogeneity.

Next, we integrate the PAC-Bayesian framework by introducing the Kullback-Leibler

(KL) divergence between the posterior $q_i(\theta \mid \phi_i)$ and the prior $p(\theta)$. The PAC-Bayesian bound provides a trade-off between the empirical risk and the complexity of the posterior relative to the prior. Specifically, for each client k, the PAC-Bayesian inequality states that:

$$L_k(q_i(\theta \mid \phi_i)) \leq \hat{L}_k(q_i(\theta \mid \phi_i)) + \mathrm{KL}(q_i(\theta \mid \phi_i) \| p(\theta)) + \sigma_k^2 + \frac{\log\left(\frac{1}{\delta'}\right)}{2n_k}$$

Summing over all K clients and weighting by $p(k)$, we obtain:

$$L(Q_1, \ldots, Q_K) \leq \hat{L}(Q_1, \ldots, Q_K) + \sum_{k=1}^{K} p(k) \left(\mathrm{KL}(q_i(\theta \mid \phi_i) \| p(\theta)) + \sigma_k^2\right) + \sum_{k=1}^{K} p(k) \frac{\log\left(\frac{K}{\delta}\right)}{2n_k}$$

To further refine this bound, we apply the Gibbs variational inequality, which allows us to express the KL divergence in terms of the difference between the expected losses under the posterior and prior distributions. Specifically, for each client k:

$$\mathbb{E}_{h_k \sim q_i(\theta|\phi_i)}[\ell(h_k(x_k), y_k)] - \mathbb{E}_{h_k \sim p(\theta)}[\ell(h_k(x_k), y_k)] \leq \mathrm{KL}(q_i(\theta \mid \phi_i) \| p(\theta))$$

Substituting this into our PAC-Bayesian bound, we obtain:

$$L(Q_1, \ldots, Q_K) \leq \hat{L}(Q_1, \ldots, Q_K) +$$

$$\sum_{k=1}^{K} p(k) \left(\mathbb{E}_{h_k \sim q_k(\theta|\phi_k)}[\ell(h_k(x_k), y_k)] - \mathbb{E}_{h_k \sim p(\theta)}[\ell(h_k(x_k), y_k)] + \sigma_k^2\right) +$$

$$\sum_{k=1}^{K} p(k) \frac{\log\left(\frac{K}{\delta}\right)}{2n_k}$$

To estimate the total sample size N required to achieve a desired generalization error ϵ, we set the refined PAC-Bayesian bound to be less than or equal to ϵ:

$$\epsilon \geq \sum_{k=1}^{K} p(k) \left(\mathbb{E}_{h_k \sim q_i(\theta|\phi_i)}[\ell(h_k(x_k), y_k)] - \mathbb{E}_{h_k \sim p(\theta)}[\ell(h_k(x_k), y_k)] + \sigma_k^2\right) + \sum_{k=1}^{K} p(k) \frac{\log\left(\frac{K}{\delta}\right)}{2n_k}$$

Given that $p(k) = \frac{n_k}{N}$, where $N = \sum_{k=1}^{K} n_k$ is the total sample size, we substitute and rearrange the terms to solve for N:

$$\epsilon \geq \frac{1}{N} \sum_{k=1}^{K} \left(\mathrm{KL}(q_i(\theta \mid \phi_i) \| p(\theta)) + \sigma_k^2 \right) + \frac{\log\left(\frac{K}{\delta}\right)}{2N}$$

To isolate N, we multiply both sides by N:

$$N \cdot \epsilon \geq \sum_{k=1}^{K} \left(\mathrm{KL}(q_i(\theta \mid \phi_i) \| p(\theta)) + \sigma_k^2 \right) + \frac{\log\left(\frac{K}{\delta}\right)}{2}$$

Rearranging to solve for N, we obtain:

$$N \geq \frac{8\epsilon^2}{C^2} \left(\sum_{k=1}^{K} \left(\mathrm{KL}(q_i(\theta \mid \phi_i) \| p(\theta)) + \sigma_k^2 \right) + \log\left(\frac{1}{\delta}\right) \right)$$

The constant 8 arises from bounding the terms involving C and consolidating the logarithmic terms, ensuring that the sample complexity N sufficiently accommodates both the complexity of the posterior distributions and the variance inherent in the clients' data.

Discussion. The derivation presented above is grounded in several key assumptions that are crucial for its applicability in real-world federated learning scenarios.

Firstly, the boundedness of the loss function $\ell(h_k(x_k), y_k) \in [0, C]$ is a fundamental assumption that facilitates the application of Bernstein's inequality. In practice, this can be achieved by selecting appropriate loss functions or by normalizing the outputs of the model. For instance, in classification tasks using cross-entropy loss, it is standard to bound the loss by setting $C = 1$. This boundedness ensures that the concentration inequalities hold, providing reliable bounds on the generalization error. However, if the loss function inherently produces unbounded values, additional measures such as clipping or normalization must be employed to satisfy this assumption.

Secondly, the selection of priors $p(\theta)$ plays a pivotal role in the PAC-Bayesian framework. Priors can be chosen based on domain knowledge, previous training rounds, or shared information among clients. A well-informed prior can lead to a smaller KL divergence, thereby tightening the generalization bound and reducing the required sample complexity. In federated learning, initializing priors based on historical data or aggregating information from previous iterations can enhance the effectiveness of the bounds.

The variance terms σ_k^2 capture the heterogeneity of data across clients. High variance indicates significant variability in the loss function, necessitating a larger sample size to achieve the desired generalization performance. In practice, σ_k^2 can be estimated empirically from the data or bounded based on theoretical considerations. Accurate estimation of variance is essential, as underestimation can lead to insufficient sample sizes, while overestimation may result in unnecessarily large sample sizes, impacting the efficiency of the federated learning process.

Moreover, the application of the union bound with the confidence adjustment $\delta' = \frac{\delta}{K}$ ensures that the overall confidence level $1 - \delta$ is maintained across all clients. This adjustment is critical in federated learning, where multiple clients contribute to the final model, and cumulative failure probabilities must be controlled to guarantee reliable generalization performance.

In summary, this derivation provides a robust framework for estimating the sample complexity in federated learning settings characterized by non-IID data distributions. By integrating Bernstein's inequality and the Gibbs variational inequality into the PAC-Bayesian bounds, we achieve a more nuanced and tighter estimation that accounts for both variance and divergence among client-specific posteriors. These theoretical advancements are instrumental in designing efficient and reliable federated learning systems, ensuring that sufficient data is allocated to each client to achieve desired performance metrics while accounting for data heterogeneity and model complexity.

Bibliography

[1] Salvatore Vitabile et al. "Medical data processing and analysis for remote health and activities monitoring". In: *High-Performance Modelling and Simulation for Big Data Applications: Selected Results of the COST Action IC1406 cHiPSet*. Springer International Publishing Cham, 2019, pp. 186–220.

[2] Andrea Zanella et al. "Internet of Things for Smart Cities". In: *IEEE Internet of Things Journal* 1.1 (2014), pp. 22–32. DOI: 10.1109/JIOT.2014.2306328.

[3] Weisong Shi et al. "Edge Computing: Vision and Challenges". In: *IEEE Internet of Things Journal* 3.5 (2016), pp. 637–646. DOI: 10.1109/JIOT.2016.2579198.

[4] Pedro Garcia Lopez et al. "Edge-centric Computing". In: *ACM SIGCOMM Computer Communication Review* 45 (Sept. 2015), pp. 37–42. DOI: 10.1145/2831347.2831354. (Visited on 05/22/2020).

[5] Samuel R. Madden et al. "TinyDB: an acquisitional query processing system for sensor networks". In: *ACM Transactions on Database Systems* 30 (Mar. 2005), pp. 122–173. DOI: 10.1145/1061318.1061322. (Visited on 12/15/2019).

[6] Flavio Bonomi et al. "Fog computing and its role in the internet of things". In: *Proceedings of the first edition of the MCC workshop on Mobile cloud computing - MCC '12* (2012). DOI: 10.1145/2342509.2342513.

[7] Tsvi Kuflik, Judy Kay, and Bob Kummerfeld. "Challenges and Solutions of Ubiquitous User Modeling". In: *Ubiquitous Display Environments* (2012), pp. 7–30. DOI: 10.1007/978-3-642-27663-7_2. (Visited on 09/12/2021).

[8] McMahan H Brendan et al. *Communication-Efficient Learning of Deep Networks from Decentralized Data*. arXiv.org, 2016. URL: https://arxiv.org/abs/1602.05629.

[9] Andrew Hard et al. *Federated Learning for Mobile Keyboard Prediction*. 2019. arXiv: 1811.03604 [cs.CL]. URL: https://arxiv.org/abs/1811.03604.

[10] Theodora S. Brisimi et al. "Federated learning of predictive models from federated Electronic Health Records". In: *International Journal of Medical Informatics* 112 (2018), pp. 59–67. ISSN: 1386-5056. DOI: https://doi.org/10.1016/j.ijmedinf.2018.01.007. URL: https://www.sciencedirect.com/science/article/pii/S138650561830008X.

[11] Micah J Sheller et al. "Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data". In: *Scientific reports* 10.1 (2020), p. 12598.

[12] Tao Liu et al. *Efficient and Secure Federated Learning for Financial Applications*. 2023. arXiv: 2303.08355 [cs.LG]. URL: https://arxiv.org/abs/2303.08355.

[13] Muhammad Ammad-ud-din et al. *Federated Collaborative Filtering for Privacy-Preserving Personalized Recommendation System*. 2019. arXiv: 1901.09888 [cs.IR]. URL: https://arxiv.org/abs/1901.09888.

[14] Jeffrey Dean et al. "Large Scale Distributed Deep Networks". In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira et al. Vol. 25. Curran Associates, Inc., 2012. URL: https://proceedings.neurips.cc/paper_files/paper/2012/file/6aca97005c68f1206823815f66102863-Paper.pdf.

[15] Diego Peteiro-Barral and Bertha Guijarro-Berdiñas. "A survey of methods for distributed machine learning". In: *Progress in Artificial Intelligence* 2 (2012), pp. 1–11. URL: https://api.semanticscholar.org/CorpusID:256283562.

[16] JAYAKRUSHNA SAHOO, AKARSH K NAIR, and RICHA SHARMA. "The Evolution of Machine Learning: From Centralized to Distributed". In: *Federated Learning: Principles, Paradigms, and Applications* (2024), p. 1.

[17] Eric P. Xing et al. "Strategies and Principles of Distributed Machine Learning on Big Data". In: *Engineering* 2.2 (2016), pp. 179–195. ISSN: 2095-8099. DOI: https://doi.org/10.1016/J.ENG.2016.02.008. URL: https://www.sciencedirect.com/science/article/pii/S2095809916309468.

[18] Mu Li et al. "Communication Efficient Distributed Machine Learning with the Parameter Server". In: *Advances in Neural Information Processing Systems*. Ed. by Z. Ghahramani et al. Vol. 27. Curran Associates, Inc., 2014. URL: https://proceedings.neurips.cc/paper_files/paper/2014/file/1ff1de774005f8da13f42943881c655f-Paper.pdf.

[19] Jeffrey Dean et al. "Large Scale Distributed Deep Networks". In: *NIPS*. 2012.

[20] Priya Goyal et al. *Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour*. 2018. arXiv: 1706.02677 [cs.CV]. URL: https://arxiv.org/abs/1706.02677.

[21] Reza Shokri and Vitaly Shmatikov. "Privacy-preserving deep learning". In: *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. 2015, pp. 909–910. DOI: 10.1109/ALLERTON.2015.7447103.

[22] Keith Bonawitz et al. "Practical Secure Aggregation for Privacy-Preserving Machine Learning". In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. CCS '17. Dallas, Texas, USA: Association for Computing Machinery, 2017, pp. 1175–1191. ISBN: 9781450349468. DOI: 10.1145/3133956.3133982. URL: https://doi.org/10.1145/3133956.3133982.

[23] Mahadev Satyanarayanan. "The emergence of edge computing". In: *Computer* 50.1 (2017), pp. 30–39.

[24] Li Lin et al. "Computation Offloading Toward Edge Computing". In: *Proceedings of the IEEE* 107.8 (2019), pp. 1584–1607. DOI: 10.1109/JPROC.2019.2922285.

[25] Shaoshan Liu et al. "Edge Computing for Autonomous Driving: Opportunities and Challenges". In: *Proceedings of the IEEE* 107.8 (2019), pp. 1697–1716. DOI: 10.1109/JPROC.2019.2915983.

[26] Bo Yang et al. "Edge Intelligence for Autonomous Driving in 6G Wireless System: Design Challenges and Solutions". In: *IEEE Wireless Communications* 28.2 (2021), pp. 40–47. DOI: 10.1109/MWC.001.2000292.

[27] Tie Qiu et al. "Edge Computing in Industrial Internet of Things: Architecture, Advances and Challenges". In: *IEEE Communications Surveys & Tutorials* 22.4 (2020), pp. 2462–2488. DOI: 10.1109/COMST.2020.3009103.

[28] Partha Pratim Ray, Dinesh Dash, and Debashis De. "Edge computing for Internet of Things: A survey, e-healthcare case study and future direction". In: *Journal of Network and Computer Applications* 140 (2019), pp. 1–22. ISSN: 1084-8045. DOI: https://doi.org/10.1016/j.jnca.2019.05.005. URL: https://www.sciencedirect.com/science/article/pii/S1084804519301651.

[29] Weisong Shi and Schahram Dustdar. "The Promise of Edge Computing". In: *Computer* 49.5 (2016), pp. 78–81. DOI: 10.1109/MC.2016.145.

[30] Daniele Ravi et al. "Deep learning for human activity recognition: A resource efficient implementation on low-power devices". In: *2016 IEEE 13th International Conference on Wearable and Implantable Body Sensor Networks (BSN)*. 2016, pp. 71–76. DOI: 10.1109/BSN.2016.7516235.

[31] Akhil Mathur et al. "DeepEye: Resource Efficient Local Execution of Multiple Deep Vision Models using Wearable Commodity Hardware". In: *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*. MobiSys '17. Niagara Falls, New York, USA: Association for Computing Machin-

ery, 2017, pp. 68–81. ISBN: 9781450349284. DOI: 10.1145/3081333.3081359. URL: https://doi.org/10.1145/3081333.3081359.

[32] Feng Wang et al. "Joint Offloading and Computing Optimization in Wireless Powered Mobile-Edge Computing Systems". In: *IEEE Transactions on Wireless Communications* 17.3 (2018), pp. 1784–1797. DOI: 10.1109/TWC.2017.2785305.

[33] Zhe Yu et al. "Joint Task Offloading and Resource Allocation in UAV-Enabled Mobile Edge Computing". In: *IEEE Internet of Things Journal* 7.4 (2020), pp. 3147–3159. DOI: 10.1109/JIOT.2020.2965898.

[34] Blesson Varghese et al. "Challenges and Opportunities in Edge Computing". In: *2016 IEEE International Conference on Smart Cloud (SmartCloud)*. 2016, pp. 20–26. DOI: 10.1109/SmartCloud.2016.18.

[35] Tian Li et al. "Federated Learning: Challenges, Methods, and Future Directions". In: *IEEE Signal Processing Magazine* 37.3 (May 2020), pp. 50–60. ISSN: 1558-0792. DOI: 10.1109/msp.2020.2975749. URL: http://dx.doi.org/10.1109/MSP.2020.2975749.

[36] Sharnil Pandya et al. "Federated learning for smart cities: A comprehensive survey". In: *Sustainable Energy Technologies and Assessments* 55 (2023), p. 102987. ISSN: 2213-1388. DOI: https://doi.org/10.1016/j.seta.2022.102987. URL: https://www.sciencedirect.com/science/article/pii/S2213138822010359.

[37] Keith Bonawitz et al. *Towards Federated Learning at Scale: System Design*. 2019. arXiv: 1902.01046 [cs.LG]. URL: https://arxiv.org/abs/1902.01046.

[38] Peter Kairouz et al. *Advances and Open Problems in Federated Learning*. 2021. arXiv: 1912.04977 [cs.LG]. URL: https://arxiv.org/abs/1912.04977.

[39] Qiang Yang et al. *Federated Machine Learning: Concept and Applications*. 2019. arXiv: 1902.04885 [cs.AI]. URL: https://arxiv.org/abs/1902.04885.

[40] Jakub Konečný et al. *Federated Optimization: Distributed Machine Learning for On-Device Intelligence*. 2016. arXiv: 1610.02527 [cs.LG]. URL: https://arxiv.org/abs/1610.02527.

[41] Solmaz Niknam, Harpreet S. Dhillon, and Jeffrey H. Reed. "Federated Learning for Wireless Communications: Motivation, Opportunities, and Challenges". In: *IEEE Communications Magazine* 58.6 (2020), pp. 46–51. DOI: 10.1109/MCOM.001.1900461.

[42] Osama Shahid et al. *Communication Efficiency in Federated Learning: Achievements and Challenges*. 2021. arXiv: 2107.10996 [cs.LG]. URL: https://arxiv.org/abs/2107.10996.

[43] Omair Rashed Abdulwareth Almanifi et al. "Communication and computation efficiency in Federated Learning: A survey". In: *Internet of Things* 22 (2023), p. 100742. ISSN: 2542-6605. DOI: https://doi.org/10.1016/j.iot.2023.100742. URL: https://www.sciencedirect.com/science/article/pii/S2542660523000653.

[44] Tian Li et al. *Federated Optimization in Heterogeneous Networks*. 2020. arXiv: 1812.06127 [cs.LG]. URL: https://arxiv.org/abs/1812.06127.

[45] Fan Ang et al. "Robust Federated Learning With Noisy Communication". In: *IEEE Transactions on Communications* 68.6 (2020), pp. 3452–3464. DOI: 10.1109/TCOMM.2020.2979149.

[46] Tien-Ju Yang et al. *Online Model Compression for Federated Learning with Large Models*. 2022. arXiv: 2205.03494 [cs.LG]. URL: https://arxiv.org/abs/2205.03494.

[47] Suhail Mohmad Shah and Vincent K. N. Lau. "Model Compression for Communication Efficient Federated Learning". In: *IEEE Transactions on Neural Networks and Learning Systems* 34.9 (2023), pp. 5937–5951. DOI: 10.1109/TNNLS.2021.3131614.

[48] Kai Liang et al. *Wyner-Ziv Gradient Compression for Federated Learning*. 2021. arXiv: 2111.08277 [cs.LG]. URL: https://arxiv.org/abs/2111.08277.

[49] Amirhossein Reisizadeh et al. "Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization". In: *International conference on artificial intelligence and statistics*. PMLR. 2020, pp. 2021–2031.

[50] Dingzhu Wen, Ki-Jun Jeon, and Kaibin Huang. *Federated Dropout – A Simple Approach for Enabling Federated Learning on Resource Constrained Devices*. 2022. arXiv: 2109.15258 [cs.LG]. URL: https://arxiv.org/abs/2109.15258.

[51] Minsu Kim et al. *SpaFL: Communication-Efficient Federated Learning with Sparse Models and Low computational Overhead*. 2024. arXiv: 2406.00431 [cs.LG]. URL: https://arxiv.org/abs/2406.00431.

[52] Dan Alistarh et al. *QSGD: Communication-Efficient SGD via Gradient Quantization and Encoding*. 2017. arXiv: 1610.02132 [cs.LG]. URL: https://arxiv.org/abs/1610.02132.

[53] Berivan Isik et al. *Sparse Random Networks for Communication-Efficient Federated Learning*. 2023. arXiv: 2209.15328 [cs.LG]. URL: https://arxiv.org/abs/2209.15328.

[54] Mufeng Zhang, Yining Wang, and Tao Luo. "Federated Learning for Arrhythmia Detection of Non-IID ECG". In: *2020 IEEE 6th International Conference on Computer and Communications (ICCC)*. 2020, pp. 1176–1180. DOI: 10.1109/ICCC51575.2020.9344971.

[55] Hangyu Zhu et al. "Federated learning on non-IID data: A survey". In: *Neurocomputing* 465 (2021), pp. 371–390. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2021.07.098. URL: https://www.sciencedirect.com/science/article/pii/S0925231221013254.

[56] Takayuki Nishio and Ryo Yonetani. “Client Selection for Federated Learning with Heterogeneous Resources in Mobile Edge”. In: *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*. IEEE, May 2019. DOI: 10.1109/icc.2019.8761315. URL: http://dx.doi.org/10.1109/ICC.2019.8761315.

[57] Chengxu Yang et al. *Characterizing Impacts of Heterogeneity in Federated Learning upon Large-Scale Smartphone Data*. 2021. arXiv: 2006.06983 [cs.LG]. URL: https://arxiv.org/abs/2006.06983.

[58] Fabio Liberti, Davide Berardi, and Barbara Martini. “Federated Learning in Dynamic and Heterogeneous Environments: Advantages, Performances, and Privacy Problems”. In: *Applied Sciences* 14.18 (2024). ISSN: 2076-3417. DOI: 10.3390/app14188490. URL: https://www.mdpi.com/2076-3417/14/18/8490.

[59] Zhaohui Yang et al. “Energy Efficient Federated Learning Over Wireless Communication Networks”. In: *IEEE Transactions on Wireless Communications* 20.3 (2021), pp. 1935–1949. DOI: 10.1109/TWC.2020.3037554.

[60] Ahmed Imteaj et al. “A Survey on Federated Learning for Resource-Constrained IoT Devices”. In: *IEEE Internet of Things Journal* 9.1 (2022), pp. 1–24. DOI: 10.1109/JIOT.2021.3095077.

[61] Wenjun Qian et al. “DROPFL: Client Dropout Attacks Against Federated Learning Under Communication Constraints”. In: *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2024, pp. 4870–4874. DOI: 10.1109/ICASSP48485.2024.10446609.

[62] Hongsheng Hu et al. “Source Inference Attacks: Beyond Membership Inference Attacks in Federated Learning”. In: *IEEE Transactions on Dependable and Secure Computing* 21.4 (2024), pp. 3012–3029. DOI: 10.1109/TDSC.2023.3321565.

[63] Vale Tolpegin et al. *Data Poisoning Attacks Against Federated Learning Systems*. 2020. arXiv: 2007.08432 [cs.LG]. URL: https://arxiv.org/abs/2007.08432.

[64] Geming Xia et al. "Poisoning Attacks in Federated Learning: A Survey". In: *IEEE Access* 11 (2023), pp. 10708–10722. URL: https://api.semanticscholar.org/CorpusID:256218299.

[65] Clement Fung, Chris J. M. Yoon, and Ivan Beschastnikh. *Mitigating Sybils in Federated Learning Poisoning*. 2020. arXiv: 1808.04866 [cs.LG]. URL: https://arxiv.org/abs/1808.04866.

[66] Yueqi Xie, Minghong Fang, and Neil Zhenqiang Gong. *Model Poisoning Attacks to Federated Learning via Multi-Round Consistency*. 2024. arXiv: 2404.15611 [cs.CR]. URL: https://arxiv.org/abs/2404.15611.

[67] Antonious M. Girgis et al. "Shuffled Model of Federated Learning: Privacy, Accuracy and Communication Trade-Offs". In: *IEEE Journal on Selected Areas in Information Theory* 2.1 (2021), pp. 464–478. DOI: 10.1109/JSAIT.2021.3056102.

[68] Xueting Luo, Zhongyuan Zhao, and Mugen Peng. "Tradeoff between Model Accuracy and Cost for Federated Learning in the Mobile Edge Computing Systems". In: *2021 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*. 2021, pp. 1–6. DOI: 10.1109/WCNCW49093.2021.9419985.

[69] Bing Luo et al. "Cost-Effective Federated Learning Design". In: *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*. 2021, pp. 1–10. DOI: 10.1109/INFOCOM42981.2021.9488679.

[70] Jianyang Ren, Wanli Ni, and Hui Tian. "Toward Communication-Learning Trade-Off for Federated Learning at the Network Edge". In: *IEEE Communications Letters* 26.8 (2022), pp. 1858–1862. DOI: 10.1109/LCOMM.2022.3174295.

[71] Giovanni Paragliola. "Evaluation of the trade-off between performance and communication costs in federated learning scenario". In: *Future Generation Computer Systems* 136 (2022), pp. 282–293. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2022.06.006. URL: https://www.sciencedirect.com/science/article/pii/S0167739X22002175.

[72] Xiaojin Zhang et al. *Trading Off Privacy, Utility and Efficiency in Federated Learning*. 2023. arXiv: 2209.00230 [cs.LG]. URL: https://arxiv.org/abs/2209.00230.

[73] Shiqiang Wang et al. "Adaptive Federated Learning in Resource Constrained Edge Computing Systems". In: *IEEE Journal on Selected Areas in Communications* 37.6 (2019), pp. 1205–1221. DOI: 10.1109/JSAC.2019.2904348.

[74] Divi De Lacour et al. "Towards Scalable Resilient Federated Learning: A Fully Decentralised Approach". In: *2023 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*. 2023, pp. 621–627. DOI: 10.1109/PerComWorkshops56833.2023.10150295.

[75] Han Yu et al. "A Fairness-aware Incentive Scheme for Federated Learning". In: AIES '20. New York, NY, USA: Association for Computing Machinery, 2020, pp. 393–399. ISBN: 9781450371100. DOI: 10.1145/3375627.3375840. URL: https://doi.org/10.1145/3375627.3375840.

[76] Jiawen Kang et al. "Incentive Mechanism for Reliable Federated Learning: A Joint Optimization Approach to Combining Reputation and Contract Theory". In: *IEEE Internet of Things Journal* 6.6 (2019), pp. 10700–10714. DOI: 10.1109/JIOT.2019.2940820.

[77] Mohammed Aledhari et al. "Federated Learning: A Survey on Enabling Technologies, Protocols, and Applications". In: *IEEE Access* 8 (2020), pp. 140699–140725. DOI: 10.1109/ACCESS.2020.3013541.

[78] Priyanka Mary Mammen. *Federated Learning: Opportunities and Challenges*. 2021. arXiv: 2101.05428 [cs.LG]. URL: https://arxiv.org/abs/2101.05428.

[79] K. M. Jawadur Rahman et al. "Challenges, Applications and Design Aspects of Federated Learning: A Survey". In: *IEEE Access* 9 (2021), pp. 124682–124700. DOI: 10.1109/ACCESS.2021.3111118.

[80] Subrato Bharati et al. "Federated learning: Applications, challenges and future directions". In: *International Journal of Hybrid Intelligent Systems* 18.1–2 (May 2022), pp. 19–35. ISSN: 1875-8819. DOI: 10.3233/his-220006. URL: http://dx.doi.org/10.3233/HIS-220006.

[81] Yue Zhao et al. "Federated Learning with Non-IID Data". In: (2018). DOI: 10.48550/ARXIV.1806.00582. URL: https://arxiv.org/abs/1806.00582.

[82] Muhammad Babar, Basit Qureshi, and Anis Koubaa. "Investigating the impact of data heterogeneity on the performance of federated learning algorithm using medical imaging". In: *Plos one* 19.5 (2024), e0302539.

[83] Tian Li et al. *Federated Optimization in Heterogeneous Networks*. 2020. arXiv: 1812.06127 [cs.LG]. URL: https://arxiv.org/abs/1812.06127.

[84] Sai Praneeth Karimireddy et al. "SCAFFOLD: Stochastic Controlled Averaging for Federated Learning". In: *arXiv:1910.06378 [cs, math, stat]* v4 (Apr. 2021). URL: https://arxiv.org/abs/1910.06378 (visited on 12/21/2021).

[85] Hung Nguyen, Peiyuan Wu, and Morris Chang. *Federated Learning for distribution skewed data using sample weights*. 2024. arXiv: 2401.02586 [cs.LG]. URL: https://arxiv.org/abs/2401.02586.

[86] Shuang Zeng et al. *Tackling Data Heterogeneity in Federated Learning via Loss Decomposition*. 2024. arXiv: 2408.12300 [cs.LG]. URL: https://arxiv.org/abs/2408.12300.

[87] Zhenheng Tang et al. *Virtual Homogeneity Learning: Defending against Data Heterogeneity in Federated Learning*. 2022. arXiv: 2206.02465 [cs.LG]. URL: https://arxiv.org/abs/2206.02465.

[88] Fei Chen et al. *Federated Meta-Learning with Fast Convergence and Efficient Communication*. 2019. arXiv: 1802.07876 [cs.LG]. URL: https://arxiv.org/abs/1802.07876.

[89] Sen Lin, Guang Yang, and Junshan Zhang. "A Collaborative Learning Framework via Federated Meta-Learning". In: *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*. 2020, pp. 289–299. DOI: 10.1109/ICDCS47774.2020.00032.

[90] Lei Yang et al. "Personalized Federated Learning on Non-IID Data via Group-based Meta-learning". In: *ACM Trans. Knowl. Discov. Data* 17.4 (Mar. 2023). ISSN: 1556-4681. DOI: 10.1145/3558005. URL: https://doi.org/10.1145/3558005.

[91] Virginia Smith et al. *Federated Multi-Task Learning*. 2018. arXiv: 1705.10467 [cs.LG]. URL: https://arxiv.org/abs/1705.10467.

[92] Yeongwoo Kim et al. "Dynamic Clustering in Federated Learning". In: *ICC 2021 - IEEE International Conference on Communications*. 2021, pp. 1–6. DOI: 10.1109/ICC42927.2021.9500877.

[93] Chengxi Li, Gang Li, and Pramod K. Varshney. "Federated Learning With Soft Clustering". In: *IEEE Internet of Things Journal* 9.10 (2022), pp. 7773–7782. DOI: 10.1109/JIOT.2021.3113927.

[94] Cheng Chen et al. "FedCluster: Boosting the Convergence of Federated Learning via Cluster-Cycling". In: *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, Dec. 2020, pp. 5017–5026. DOI: 10.1109/bigdata50022.2020.9377960. URL: http://dx.doi.org/10.1109/BigData50022.2020.9377960.

[95] Sohei Itahara et al. "Distillation-Based Semi-Supervised Federated Learning for Communication-Efficient Collaborative Training With Non-IID Private Data". In: *IEEE Transactions on Mobile Computing* 22.1 (2023), pp. 191–205. DOI: 10.1109/TMC.2021.3070013.

[96] Sashank Reddi et al. *Adaptive Federated Optimization*. 2021. arXiv: 2003.00295 [cs.LG]. URL: https://arxiv.org/abs/2003.00295.

[97] Li Deng. “The mnist database of handwritten digit images for machine learning research”. In: *IEEE Signal Processing Magazine* 29.6 (2012), pp. 141–142.

[98] Mehreen Tahir and Muhammad Intizar Ali. “On the Performance of Federated Learning Algorithms for IoT”. In: *IoT* 3 (Apr. 2022), pp. 273–284. DOI: 10.3390/iot3020016.

[99] Xinwei Zhang et al. “FedPD: A Federated Learning Framework with Optimal Rates and Adaptivity to Non-IID Data”. In: *arXiv:2005.11418 [cs, stat]* v3 (July 2020). URL: https://arxiv.org/abs/2005.11418.

[100] Canh T. Dinh, Nguyen H. Tran, and Tuan Dung Nguyen. *Personalized Federated Learning with Moreau Envelopes*. 2022. arXiv: 2006.08848 [cs.LG]. URL: https://arxiv.org/abs/2006.08848.

[101] Mang Ye et al. *Heterogeneous Federated Learning: State-of-the-art and Research Challenges*. 2023. arXiv: 2307.10616 [cs.LG]. URL: https://arxiv.org/abs/2307.10616.

[102] Jie Wen et al. “A survey on federated learning: challenges and applications”. In: *International Journal of Machine Learning and Cybernetics* 14.2 (2023), pp. 513–535.

[103] Zheng Chai et al. “Towards taming the resource and data heterogeneity in federated learning”. In: *2019 USENIX conference on operational machine learning (OpML 19)*. 2019, pp. 19–21.

[104] Ahmed M. Abdelmoniem et al. *On the Impact of Device and Behavioral Heterogeneity in Federated Learning*. 2021. arXiv: 2102.07500 [cs.LG]. URL: https://arxiv.org/abs/2102.07500.

[105] Gyudong Kim et al. *HeteroSwitch: Characterizing and Taming System-Induced Data Heterogeneity in Federated Learning*. 2024. arXiv: 2403.04207 [cs.LG]. URL: https://arxiv.org/abs/2403.04207.

[106] Kilian Pfeiffer et al. "Federated Learning for Computationally Constrained Heterogeneous Devices: A Survey". In: *ACM Computing Surveys* 55.14s (July 2023), pp. 1–27. ISSN: 1557-7341. DOI: 10.1145/3596907. URL: http://dx.doi.org/10.1145/3596907.

[107] Jungwuk Park et al. "Handling both stragglers and adversaries for robust federated learning". In: *ICML 2021 Workshop on Federated Learning for User Privacy and Data Confidentiality. ICML Board*. 2021.

[108] Amirhossein Reisizadeh et al. "Straggler-Resilient Federated Learning: Leveraging the Interplay Between Statistical Accuracy and System Heterogeneity". In: *IEEE Journal on Selected Areas in Information Theory* 3.2 (2022), pp. 197–205. DOI: 10.1109/JSAIT.2022.3205475.

[109] Zhilin Wang et al. *Straggler Mitigation and Latency Optimization in Blockchain-based Hierarchical Federated Learning*. 2023. arXiv: 2308.01296 [cs.DC]. URL: https://arxiv.org/abs/2308.01296.

[110] Andrew Hard et al. *Learning from straggler clients in federated learning*. 2024. arXiv: 2403.09086 [cs.LG]. URL: https://arxiv.org/abs/2403.09086.

[111] Natalie Lang, Alejandro Cohen, and Nir Shlezinger. *Stragglers-Aware Low-Latency Synchronous Federated Learning via Layer-Wise Model Updates*. 2024. arXiv: 2403.18375 [cs.LG]. URL: https://arxiv.org/abs/2403.18375.

[112] Ming Chen, Bingcheng Mao, and Tianyi Ma. "Efficient and robust asynchronous federated learning with stragglers". In: *International Conference on Learning Representations*. 2019.

[113] Zhongyu Wang et al. "Asynchronous Federated Learning Over Wireless Communication Networks". In: *IEEE Transactions on Wireless Communications* 21.9 (2022), pp. 6961–6978. DOI: 10.1109/TWC.2022.3153495.

[114] Jianchun Liu et al. “Adaptive Asynchronous Federated Learning in Resource-Constrained Edge Computing”. In: *IEEE Transactions on Mobile Computing* 22.2 (2023), pp. 674–690. DOI: 10.1109/TMC.2021.3096846.

[115] Chenhao Xu et al. “Asynchronous federated learning on heterogeneous devices: A survey”. In: *Computer Science Review* 50 (2023), p. 100595. ISSN: 1574-0137. DOI: https://doi.org/10.1016/j.cosrev.2023.100595. URL: https://www.sciencedirect.com/science/article/pii/S157401372300062X.

[116] Filipe Maciel et al. “Resource Aware Client Selection for Federated Learning in IoT Scenarios”. In: *2023 19th International Conference on Distributed Computing in Smart Systems and the Internet of Things (DCOSS-IoT)*. 2023, pp. 1–8. DOI: 10.1109/DCOSS-IoT58021.2023.00010.

[117] Alaa Awad Abdellatif et al. “Communication-efficient hierarchical federated learning for IoT heterogeneous systems with imbalanced data”. In: *Future Generation Computer Systems* 128 (2022), pp. 406–419. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2021.10.016. URL: https://www.sciencedirect.com/science/article/pii/S0167739X2100412X.

[118] Qiong Wu et al. “HiFlash: Communication-Efficient Hierarchical Federated Learning With Adaptive Staleness Control and Heterogeneity-Aware Client-Edge Association”. In: *IEEE Transactions on Parallel and Distributed Systems* 34.5 (2023), pp. 1560–1579. DOI: 10.1109/TPDS.2023.3238049.

[119] Zhiyuan Wang et al. “Resource-Efficient Federated Learning with Hierarchical Aggregation in Edge Computing”. In: *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*. 2021, pp. 1–10. DOI: 10.1109/INFOCOM42981.2021.9488756.

[120] Zhiyuan Wang et al. “Accelerating Federated Learning With Cluster Construction and Hierarchical Aggregation”. In: *IEEE Transactions on Mobile Computing* 22.7 (2023), pp. 3805–3822. DOI: 10.1109/TMC.2022.3147792.

[121] Durmus Alp Emre Acar et al. *Federated Learning Based on Dynamic Regularization*. 2021. arXiv: 2111.04263 [cs.LG]. URL: https://arxiv.org/abs/2111.04263.

[122] Jie Xu and Heqiang Wang. "Client Selection and Bandwidth Allocation in Wireless Federated Learning Networks: A Long-Term Perspective". In: *IEEE Transactions on Wireless Communications* 20.2 (2021), pp. 1188–1200. DOI: 10.1109/TWC.2020.3031503.

[123] Rong Yu and Peichun Li. "Toward Resource-Efficient Federated Learning in Mobile Edge Computing". In: *IEEE Network* 35.1 (2021), pp. 148–155. DOI: 10.1109/MNET.011.2000295.

[124] Wenqi Shi et al. "Joint Device Scheduling and Resource Allocation for Latency Constrained Wireless Federated Learning". In: *IEEE Transactions on Wireless Communications* 20.1 (2021), pp. 453–467. DOI: 10.1109/TWC.2020.3025446.

[125] Tuo Zhang et al. "TimelyFL: Heterogeneity-Aware Asynchronous Federated Learning With Adaptive Partial Training". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. June 2023, pp. 5064–5073.

[126] Ihab Mohammed et al. "Budgeted Online Selection of Candidate IoT Clients to Participate in Federated Learning". In: *IEEE Internet of Things Journal* 8.7 (2021), pp. 5938–5952. DOI: 10.1109/JIOT.2020.3036157.

[127] Wenchao Xia et al. "Multi-Armed Bandit-Based Client Scheduling for Federated Learning". In: *IEEE Transactions on Wireless Communications* 19.11 (Nov. 2020), pp. 7108–7123. ISSN: 1558-2248. DOI: 10.1109/twc.2020.3008091. URL: http://dx.doi.org/10.1109/TWC.2020.3008091.

[128] Luping WANG, Wei WANG, and Bo LI. "CMFL: Mitigating Communication Overhead for Federated Learning". In: *2019 IEEE 39th International Conference*

on Distributed Computing Systems (ICDCS). 2019, pp. 954–964. DOI: 10.1109/ICDCS.2019.00099.

[129] Felix Sattler et al. *Robust and Communication-Efficient Federated Learning from Non-IID Data*. 2019. arXiv: 1903.02891 [cs.LG].

[130] Wenyu Zhang et al. "Client Selection for Federated Learning With Non-IID Data in Mobile Edge Computing". In: *IEEE Access* 9 (2021), pp. 24462–24474. DOI: 10.1109/ACCESS.2021.3056919.

[131] Aiguo Chen et al. "An emd-based adaptive client selection algorithm for federated learning in heterogeneous data scenarios". In: *Frontiers in Plant Science* 13 (2022), p. 908814.

[132] Fa Xin et al. "Federated Learning Client Selection Mechanism Under System and Data Heterogeneity". In: *2022 IEEE 25th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*. 2022, pp. 1239–1244. DOI: 10.1109/CSCWD54268.2022.9776061.

[133] Yuxuan Zhang et al. *DPP-based Client Selection for Federated Learning with Non-IID Data*. 2023. arXiv: 2303.17358 [cs.LG]. URL: https://arxiv.org/abs/2303.17358.

[134] Xutao Meng et al. "An Optimization Method for Non-IID Federated Learning Based on Deep Reinforcement Learning". In: *Sensors* 23.22 (2023), p. 9226.

[135] Yann Fraboni et al. *Clustered Sampling: Low-Variance and Improved Representativity for Clients Selection in Federated Learning*. 2021. arXiv: 2105.05883 [cs.LG]. URL: https://arxiv.org/abs/2105.05883.

[136] Abdullatif Albaseer et al. "Client Selection Approach in Support of Clustered Federated Learning over Wireless Edge Networks". In: *2021 IEEE Global Communications Conference (GLOBECOM)*. 2021, pp. 1–6. DOI: 10.1109/GLOBECOM46510.2021.9685938.

[137] Sarhad Arisdakessian et al. "Towards Instant Clustering Approach for Federated Learning Client Selection". In: *2023 International Conference on Computing, Networking and Communications (ICNC)*. 2023, pp. 409–413. DOI: 10.1109/ICNC57223.2023.10074237.

[138] Honglan Huang et al. "Active Client Selection for Clustered Federated Learning". In: *IEEE Transactions on Neural Networks and Learning Systems* 35.11 (2024), pp. 16424–16438. DOI: 10.1109/TNNLS.2023.3294295.

[139] Sumit Rai, Arti Kumari, and Dilip K Prasad. "Client selection in federated learning under imperfections in environment". In: *AI* 3.1 (2022), pp. 124–145.

[140] Alramzana Nujum Navaz et al. "Empowering Patient Similarity Networks through Innovative Data-Quality-Aware Federated Profiling". In: *Sensors* 23.14 (2023), p. 6443.

[141] Yongheng Deng et al. "AUCTION: Automated and Quality-Aware Client Selection Framework for Efficient Federated Learning". In: *IEEE Transactions on Parallel and Distributed Systems* 33.8 (2022), pp. 1996–2009. DOI: 10.1109/TPDS.2021.3134647.

[142] Weiwei Lin et al. "Contribution-based Federated Learning client selection". In: *International Journal of Intelligent Systems* 37.10 (2022), pp. 7235–7260.

[143] Yae Jee Cho, Jianyu Wang, and Gauri Joshi. *Client Selection in Federated Learning: Convergence Analysis and Power-of-Choice Selection Strategies*. 2020. arXiv: 2010.01243 [cs.LG].

[144] Shashi Raj Pandey, Lam D Nguyen, and Petar Popovski. "A contribution-based device selection scheme in federated learning". In: *IEEE Communications Letters* 26.9 (2022), pp. 2057–2061.

[145] Liangkun Yu et al. “Jointly Optimizing Client Selection and Resource Management in Wireless Federated Learning for Internet of Things”. In: *IEEE Internet of Things Journal* 9.6 (2022), pp. 4385–4395. DOI: 10.1109/JIOT.2021.3103715.

[146] Junqian Kuang et al. “Client Selection with Bandwidth Allocation in Federated Learning”. In: *2021 IEEE Global Communications Conference (GLOBECOM)*. 2021, pp. 01–06. DOI: 10.1109/GLOBECOM46510.2021.9685090.

[147] Tingting Liu et al. “Uplink and Downlink Decoupled 5G/B5G Vehicular Networks: A Federated Learning Assisted Client Selection Method”. In: *IEEE Transactions on Vehicular Technology* 72.2 (2023), pp. 2280–2292. DOI: 10.1109/TVT.2022.3207916.

[148] Hangjia Zhang et al. “Adaptive Client Selection in Resource Constrained Federated Learning Systems: A Deep Reinforcement Learning Approach”. In: *IEEE Access* 9 (2021), pp. 98423–98432. DOI: 10.1109/ACCESS.2021.3095915.

[149] Muhammad Asad et al. “THF: 3-Way Hierarchical Framework for Efficient Client Selection and Resource Management in Federated Learning”. In: *IEEE Internet of Things Journal* 9.13 (2022), pp. 11085–11097. DOI: 10.1109/JIOT.2021.3126828.

[150] Xiaojing Chen et al. “Two-Phase Deep Reinforcement Learning of Dynamic Resource Allocation and Client Selection for Hierarchical Federated Learning”. In: *2022 IEEE/CIC International Conference on Communications in China (ICCC)*. 2022, pp. 518–523. DOI: 10.1109/ICCC55456.2022.9880724.

[151] Shuo He et al. “Client Selection and Resource Allocation for Federated Learning in Digital-Twin-Enabled Industrial Internet of Things”. In: *2023 IEEE Wireless Communications and Networking Conference (WCNC)*. 2023, pp. 1–6. DOI: 10.1109/WCNC55385.2023.10118927.

[152] Long Luo et al. “Joint Client Selection and Resource Allocation for Federated Learning in Mobile Edge Networks”. In: *2022 IEEE Wireless Communications and*

Networking Conference (WCNC). 2022, pp. 1218–1223. DOI: 10.1109/WCNC51071.2022.9771771.

[153] Shuaijun Chen et al. “Boosting client selection of federated learning under device and data heterogeneity”. In: *arXiv e-prints* (2023), arXiv–2310.

[154] Ziru Niu et al. “FLrce: Resource-Efficient Federated Learning With Early-Stopping Strategy”. In: *IEEE Transactions on Mobile Computing* 23.12 (Dec. 2024), pp. 14514–14529. ISSN: 2161-9875. DOI: 10.1109/tmc.2024.3447000. URL: http://dx.doi.org/10.1109/TMC.2024.3447000.

[155] Zitha Sasindran, Harsha Yelchuri, and T. V. Prabhakar. “Ed-Fed: A generic federated learning framework with resource-aware client selection for edge devices”. In: *2023 International Joint Conference on Neural Networks (IJCNN)*. IEEE, June 2023, pp. 1–8. DOI: 10.1109/ijcnn54540.2023.10191316. URL: http://dx.doi.org/10.1109/IJCNN54540.2023.10191316.

[156] Yae Jee Cho et al. *Bandit-based Communication-Efficient Client Selection Strategies for Federated Learning*. 2020. arXiv: 2012.08009 [cs.LG]. URL: https://arxiv.org/abs/2012.08009.

[157] Monica Ribero and Haris Vikalo. *Communication-Efficient Federated Learning via Optimal Client Sampling*. 2020. arXiv: 2007.15197 [cs.LG]. URL: https://arxiv.org/abs/2007.15197.

[158] Ouiame Marnissi, Hajar El Hammouti, and El Houcine Bergou. *Client Selection in Federated Learning based on Gradients Importance*. 2021. arXiv: 2111.11204 [cs.LG]. URL: https://arxiv.org/abs/2111.11204.

[159] Haneul Ko et al. “Joint Client Selection and Bandwidth Allocation Algorithm for Federated Learning”. In: *IEEE Transactions on Mobile Computing* 22.6 (2023), pp. 3380–3390. DOI: 10.1109/TMC.2021.3136611.

[160] Yuhan Ai et al. “Communication-Efficient Federated Multi-Task Learning with Sparse Sharing”. In: *2023 IEEE 34th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. 2023, pp. 1–6. DOI: 10.1109/PIMRC56721.2023.10293880.

[161] Zhida Jiang et al. “Heterogeneity-Aware Federated Learning with Adaptive Client Selection and Gradient Compression”. In: *IEEE INFOCOM 2023 - IEEE Conference on Computer Communications*. 2023, pp. 1–10. DOI: 10.1109/INFOCOM53939.2023.10229029.

[162] Shanfeng Huang et al. *Accelerating Federated Edge Learning via Topology Optimization*. 2022. arXiv: 2204.00489 [cs.IT]. URL: https://arxiv.org/abs/2204.00489.

[163] Jiajun Wu et al. “Topology-aware Federated Learning in Edge Computing: A Comprehensive Survey”. In: *ACM Computing Surveys* 56.10 (June 2024), pp. 1–41. ISSN: 1557-7341. DOI: 10.1145/3659205. URL: http://dx.doi.org/10.1145/3659205.

[164] Truc Nguyen et al. *Blockchain-based Secure Client Selection in Federated Learning*. 2022. arXiv: 2205.05611 [cs.CR]. URL: https://arxiv.org/abs/2205.05611.

[165] Gaith Rjoub et al. “Trust-augmented deep reinforcement learning for federated learning client selection”. In: *Information Systems Frontiers* 26.4 (2024), pp. 1261–1278.

[166] Osama Wehbi et al. “Towards Mutual Trust-Based Matching For Federated Learning Client Selection”. In: *2023 International Wireless Communications and Mobile Computing (IWCMC)*. 2023, pp. 1112–1117. DOI: 10.1109/IWCMC58020.2023.10182581.

[167] Yuwei Wang and Burak Kantarci. “Reputation-enabled Federated Learning Model Aggregation in Mobile Platforms”. In: *ICC 2021 - IEEE International Conference on Communications*. 2021, pp. 1–6. DOI: 10.1109/ICC42927.2021.9500928.

[168] Attia Qammar et al. *Blockchain-based Optimized Client Selection and Privacy Preserved Framework for Federated Learning*. 2023. arXiv: 2308.04442 [cs.CR]. URL: https://arxiv.org/abs/2308.04442.

[169] Xavier Tan et al. "Reputation-Aware Federated Learning Client Selection Based on Stochastic Integer Programming". In: *IEEE Transactions on Big Data* 10.6 (2024), pp. 953–964. DOI: 10.1109/TBDATA.2022.3191332.

[170] Tiansheng Huang et al. "An Efficiency-Boosting Client Selection Scheme for Federated Learning With Fairness Guarantee". In: *IEEE Transactions on Parallel and Distributed Systems* 32.7 (2021), pp. 1552–1564. DOI: 10.1109/TPDS.2020.3040887.

[171] Yuxin Shi et al. *Fairness-Aware Client Selection for Federated Learning*. 2023. arXiv: 2307.10738 [cs.LG]. URL: https://arxiv.org/abs/2307.10738.

[172] Zhiyuan Zhao and Gauri Joshi. "A Dynamic Reweighting Strategy For Fair Federated Learning". In: *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2022, pp. 8772–8776. DOI: 10.1109/ICASSP43922.2022.9746300.

[173] José Ángel Morell and Enrique Alba. "Dynamic and adaptive fault-tolerant asynchronous federated learning using volunteer edge devices". In: *Future Generation Computer Systems* 133 (2022), pp. 53–67.

[174] Zhe Qu et al. *Context-Aware Online Client Selection for Hierarchical Federated Learning*. 2021. arXiv: 2112.00925 [cs.LG]. URL: https://arxiv.org/abs/2112.00925.

[175] Yongheng Deng et al. "AUCTION: Automated and Quality-Aware Client Selection Framework for Efficient Federated Learning". In: *IEEE Transactions on Parallel and Distributed Systems* 33.8 (2022), pp. 1996–2009. DOI: 10.1109/TPDS.2021.3134647.

[176] Zhipeng Cheng et al. *Learning-Based Client Selection for Federated Learning Services Over Wireless Networks with Constrained Monetary Budgets*. 2022. arXiv: 2208.04322 [cs.LG]. URL: https://arxiv.org/abs/2208.04322.

[177] Sai Qian Zhang, Jieyu Lin, and Qi Zhang. *A Multi-agent Reinforcement Learning Approach for Efficient Client Selection in Federated Learning*. 2022. arXiv: 2201.02932 [cs.LG]. URL: https://arxiv.org/abs/2201.02932.

[178] Osama Wehbi et al. "Towards Bilateral Client Selection in Federated Learning Using Matching Game Theory". In: *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*. 2022, pp. 01–06. DOI: 10.1109/GLOBECOM48099.2022.10001251.

[179] Tong Yin et al. "FedSCS: Client Selection for Federated Learning Under System Heterogeneity and Client Fairness with a Stackelberg Game Approach". In: *2023 IEEE International Conference on Communications Workshops (ICC Workshops)*. 2023, pp. 373–378. DOI: 10.1109/ICCWorkshops57953.2023.10283698.

[180] Miao Hu et al. "Incentive-Aware Autonomous Client Participation in Federated Learning". In: *IEEE Transactions on Parallel and Distributed Systems* 33.10 (2022), pp. 2612–2627. DOI: 10.1109/TPDS.2022.3148113.

[181] Jinlong Pang et al. "An Incentive Auction for Heterogeneous Client Selection in Federated Learning". In: *IEEE Transactions on Mobile Computing* 22.10 (2023), pp. 5733–5750. DOI: 10.1109/TMC.2022.3182876.

[182] Ruonan Dong et al. *An Incentive Mechanism for Federated Learning Based on Multiple Resource Exchange*. 2023. arXiv: 2312.08096 [cs.LG]. URL: https://arxiv.org/abs/2312.08096.

[183] Jianyu Wang et al. "Tackling the Objective Inconsistency Problem in Heterogeneous Federated Optimization". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020,

pp. 7611–7623. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/564127c03caab942e503ee6f810f54fd-Paper.pdf.

[184] Amirhossein Reisizadeh et al. *FedPAQ: A Communication-Efficient Federated Learning Method with Periodic Averaging and Quantization*. 2020. arXiv: 1909.13014 [cs.LG]. URL: https://arxiv.org/abs/1909.13014.

[185] Dong Yin et al. *Byzantine-Robust Distributed Learning: Towards Optimal Statistical Rates*. 2021. arXiv: 1803.01498 [cs.LG]. URL: https://arxiv.org/abs/1803.01498.

[186] Krishna Pillutla, Sham M. Kakade, and Zaid Harchaoui. "Robust Aggregation for Federated Learning". In: *IEEE Transactions on Signal Processing* 70 (2022), pp. 1142–1154. ISSN: 1941-0476. DOI: 10.1109/tsp.2022.3153135. URL: http://dx.doi.org/10.1109/TSP.2022.3153135.

[187] Mustafa Safa Ozdayi, Murat Kantarcioglu, and Rishabh Iyer. *Improving Accuracy of Federated Learning in Non-IID Settings*. 2020. arXiv: 2010.15582 [cs.LG]. URL: https://arxiv.org/abs/2010.15582.

[188] Anusha Lalitha et al. *Decentralized Bayesian Learning over Graphs*. 2019. arXiv: 1905.10466 [stat.ML]. URL: https://arxiv.org/abs/1905.10466.

[189] Adam Thor Thorgeirsson and Frank Gauterin. "Probabilistic predictions with federated learning". In: *Entropy* 23.1 (2020), p. 41.

[190] Mikhail Yurochkin et al. *Bayesian Nonparametric Federated Learning of Neural Networks*. 2019. arXiv: 1905.12022 [stat.ML]. URL: https://arxiv.org/abs/1905.12022.

[191] Longbing Cao et al. *Bayesian Federated Learning: A Survey*. 2023. arXiv: 2304.13267 [cs.LG]. URL: https://arxiv.org/abs/2304.13267.

[192] Xu Zhang et al. *Personalized Federated Learning via Variational Bayesian Inference*. 2022. arXiv: 2206.07977 [cs.LG]. URL: https://arxiv.org/abs/2206.07977.

[193] Hui Chen et al. *Bayesian Personalized Federated Learning with Shared and Personalized Uncertainty Representations*. 2023. arXiv: 2309.15499 [cs.LG]. URL: https://arxiv.org/abs/2309.15499.

[194] Seunghoon Lee et al. *Bayesian Federated Learning over Wireless Networks*. 2020. arXiv: 2012.15486 [eess.SP]. URL: https://arxiv.org/abs/2012.15486.

[195] Luca Barbieri, Stefano Savazzi, and Monica Nicoli. "Compressed Bayesian Federated Learning for Reliable Passive Radio Sensing in Industrial IoT". In: *2024 IEEE Conference on Artificial Intelligence (CAI)*. 2024, pp. 344–349. DOI: 10.1109/CAI59869.2024.00071.

[196] Kavita Kumari et al. "BayBFed: Bayesian Backdoor Defense for Federated Learning". In: *2023 IEEE Symposium on Security and Privacy (SP)*. 2023, pp. 737–754. DOI: 10.1109/SP46215.2023.10179362.

[197] Minyoung Kim and Timothy Hospedales. *FedHB: Hierarchical Bayesian Federated Learning*. 2023. arXiv: 2305.04979 [cs.LG]. URL: https://arxiv.org/abs/2305.04979.

[198] Liangxi Liu et al. "A Bayesian Federated Learning Framework With Online Laplace Approximation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 46.1 (2024), pp. 1–16. DOI: 10.1109/TPAMI.2023.3322743.

[199] Jingjing Xue et al. "FedBIAD: Communication-Efficient and Accuracy-Guaranteed Federated Learning with Bayesian Inference-Based Adaptive Dropout". In: *2023 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. 2023, pp. 489–500. DOI: 10.1109/IPDPS54959.2023.00056.

[200] Peva Blanchard et al. “Machine learning with adversaries: Byzantine tolerant gradient descent”. In: *Advances in neural information processing systems* 30 (2017).

[201] Francesco Colosimo and Floriano De Rango. “Median-Krum: A Joint Distance-Statistical Based Byzantine-Robust Algorithm in Federated Learning”. In: *Proceedings of the Int’l ACM Symposium on Mobility Management and Wireless Access*. MobiWac ’23. Montreal, Quebec, Canada: Association for Computing Machinery, 2023, pp. 61–68. ISBN: 9798400703676. DOI: 10.1145/3616390.3618283. URL: https://doi.org/10.1145/3616390.3618283.

[202] Avishek Ghosh et al. *Robust Federated Learning in a Heterogeneous Environment*. 2019. arXiv: 1906.06629 [cs.LG]. URL: https://arxiv.org/abs/1906.06629.

[203] Kenji Nishimoto et al. “FedATM: Adaptive Trimmed Mean based Federated Learning against Model Poisoning Attacks”. In: *2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring)*. 2023, pp. 1–5. DOI: 10.1109/VTC2023-Spring57618.2023.10200010.

[204] Mojtaba Kaheni et al. “Selective Trimmed Average: A Resilient Federated Learning Algorithm With Deterministic Guarantees on the Optimality Approximation”. In: *IEEE Transactions on Cybernetics* 54.8 (2024), pp. 4402–4415. DOI: 10.1109/TCYB.2024.3350387.

[205] Jinhyun So, Başak Güler, and A. Salman Avestimehr. “Byzantine-Resilient Secure Federated Learning”. In: *IEEE Journal on Selected Areas in Communications* 39.7 (2021), pp. 2168–2181. DOI: 10.1109/JSAC.2020.3041404.

[206] Lingchen Zhao et al. “SEAR: Secure and Efficient Aggregation for Byzantine-Robust Federated Learning”. In: *IEEE Transactions on Dependable and Secure Computing* 19.5 (2022), pp. 3329–3342. DOI: 10.1109/TDSC.2021.3093711.

[207] Xu Ma et al. “Privacy-preserving Byzantine-robust federated learning”. In: *Computer Standards & Interfaces* 80 (2022), p. 103561. ISSN: 0920-5489. DOI: https:

//doi.org/10.1016/j.csi.2021.103561. URL: https://www.sciencedirect.com/science/article/pii/S0920548921000568.

[208] Arjun Nitin Bhagoji et al. *Analyzing Federated Learning through an Adversarial Lens*. 2019. arXiv: 1811.12470 [cs.LG]. URL: https://arxiv.org/abs/1811.12470.

[209] Luca Melis et al. *Exploiting Unintended Feature Leakage in Collaborative Learning*. 2018. arXiv: 1805.04049 [cs.CR]. URL: https://arxiv.org/abs/1805.04049.

[210] Li Bai et al. "Membership Inference Attacks and Defenses in Federated Learning: A Survey". In: *ACM Comput. Surv.* (Nov. 2024). Just Accepted. ISSN: 0360-0300. DOI: 10.1145/3704633. URL: https://doi.org/10.1145/3704633.

[211] Monique Ogburn, Claude Turner, and Pushkar Dahal. "Homomorphic encryption". In: *Procedia Computer Science* 20 (2013), pp. 502–509.

[212] Oded Goldreich. "Secure multi-party computation". In: *Manuscript. Preliminary version* 78.110 (1998), pp. 1–108.

[213] Jaehyoung Park and Hyuk Lim. "Privacy-Preserving Federated Learning Using Homomorphic Encryption". In: *Applied Sciences* 12.2 (2022). ISSN: 2076-3417. DOI: 10.3390/app12020734. URL: https://www.mdpi.com/2076-3417/12/2/734.

[214] Abbass Madi et al. "A Secure Federated Learning framework using Homomorphic Encryption and Verifiable Computing". In: *2021 Reconciling Data Analytics, Automation, Privacy, and Security: A Big Data Challenge (RDAAPS)*. 2021, pp. 1–8. DOI: 10.1109/RDAAPS48126.2021.9452005.

[215] Weizhao Jin et al. *FedML-HE: An Efficient Homomorphic-Encryption-Based Privacy-Preserving Federated Learning System*. 2024. arXiv: 2303.10837 [cs.LG]. URL: https://arxiv.org/abs/2303.10837.

[216] Zao Chen. "Federated Learning based on Homomorphic Encryption and Digital Signatures". In: *Highlights in Science, Engineering and Technology* 39 (2023), pp. 595–603.

[217] Meng Hao et al. "Efficient, Private and Robust Federated Learning". In: *Proceedings of the 37th Annual Computer Security Applications Conference*. ACSAC '21. Virtual Event, USA: Association for Computing Machinery, 2021, pp. 45–60. ISBN: 9781450385794. DOI: 10.1145/3485832.3488014. URL: https://doi.org/10.1145/3485832.3488014.

[218] Bin Liu and Eric B. Blancaflor. "Research on federal learning privacy protection based on secure multi-party computing". In: *Proceedings of the 2024 3rd International Conference on Cyber Security, Artificial Intelligence and Digital Economy*. CSAIDE '24. Nanjing, China: Association for Computing Machinery, 2024, pp. 142–147. ISBN: 9798400718212. DOI: 10.1145/3672919.3672947. URL: https://doi.org/10.1145/3672919.3672947.

[219] Sagar Pathak and Dipankar Dasgupta. "Federated Learning with Authenticated Clients". In: *2024 IEEE 15th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*. 2024, pp. 0771–0780. DOI: 10.1109/UEMCON62879.2024.10754714.

[220] Gaolei Li, Yuanyuan Zhao, and Yi Li. "CATFL: Certificateless Authentication-based Trustworthy Federated Learning for 6G Semantic Communications". In: *2023 IEEE Wireless Communications and Networking Conference (WCNC)*. 2023, pp. 1–6. DOI: 10.1109/WCNC55385.2023.10118885.

[221] Xiaohan Yuan et al. "FedComm: A Privacy-Enhanced and Efficient Authentication Protocol for Federated Learning in Vehicular Ad-Hoc Networks". In: *IEEE Transactions on Information Forensics and Security* 19 (2024), pp. 777–792. DOI: 10.1109/TIFS.2023.3324747.

[222] Mochan Fan et al. "Blockchain-Based Decentralized and Lightweight Anonymous Authentication for Federated Learning". In: *IEEE Transactions on Vehicular Technology* 72.9 (2023), pp. 12075–12086. DOI: 10.1109/TVT.2023.3265366.

[223] Shan Ji et al. "LAFED: A lightweight authentication mechanism for blockchain-enabled federated learning system". In: *Future Generation Computer Systems* 145 (2023), pp. 56–67. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2023.03.014. URL: https://www.sciencedirect.com/science/article/pii/S0167739X23000912.

[224] Scott Rose et al. "Zero Trust Architecture". In: *Zero Trust Architecture* 800-207 (Aug. 2020). DOI: 10.6028/nist.sp.800-207. URL: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-207.pdf.

[225] Jiuru Wang et al. "Attribute and User Trust Score-Based Zero Trust Access Control Model in IoV". In: *Electronics* 12 (Jan. 2023), p. 4825. DOI: 10.3390/electronics12234825. URL: https://www.mdpi.com/2079-9292/12/23/4825 (visited on 04/09/2024).

[226] Eranga Bandara et al. "Skunk — A Blockchain and Zero Trust Security Enabled Federated Learning Platform for 5G/6G Network Slicing". In: *2022 19th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. 2022, pp. 109–117. DOI: 10.1109/SECON55815.2022.9918536.

[227] Muhammad Asad and Safa Otoum. "Integrative Federated Learning and Zero-Trust Approach for Secure Wireless Communications". In: *IEEE Wireless Communications* 31.2 (2024), pp. 14–20. DOI: 10.1109/MWC.001.2300355.

[228] Victor R Kebande et al. "A blockchain-based multi-factor authentication model for a cloud-enabled internet of vehicles". In: *Sensors* 21.18 (2021), p. 6018.

[229] Fahed Alkhabbas et al. "Assert: A blockchain-based architectural approach for engineering secure self-adaptive iot systems". In: *Sensors* 22.18 (2022), p. 6842.

[230] Maanak Gupta et al. “An attribute-based access control for cloud enabled industrial smart vehicles”. In: *IEEE Transactions on Industrial Informatics* 17.6 (2020), pp. 4288–4297.

[231] Feras M Awaysheh et al. “Next-generation big data federation access control: A reference model”. In: *Future Generation Computer Systems* 108 (2020), pp. 726–741.

[232] Vincent C Hu et al. “Guide to attribute based access control (abac) definition and considerations (draft)”. In: *NIST special publication* 800.162 (2013), pp. 1–54.

[233] Larissa Hammon and Hajo Hippner. “Crowdsourcing”. In: *Business & Information systems engineering* 4 (2012), pp. 163–166.

[234] Daren C. Brabham. “Crowdsourcing as a Model for Problem Solving: An Introduction and Cases”. In: *Convergence* 14.1 (2008), pp. 75–90. DOI: 10.1177/1354856507084420. eprint: https://doi.org/10.1177/1354856507084420. URL: https://doi.org/10.1177/1354856507084420.

[235] Kevin Crowston. “Amazon Mechanical Turk: A Research Tool for Organizations and Information Systems Scholars”. In: *Shaping the Future of ICT Research. Methods and Approaches*. Ed. by Anol Bhattacherjee and Brian Fitzgerald. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 210–221. ISBN: 978-3-642-35142-6.

[236] Chris Callison-Burch. “Fast, Cheap, and Creative: Evaluating Translation Quality Using Amazon’s Mechanical Turk”. In: *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*. Ed. by Philipp Koehn and Rada Mihalcea. Singapore: Association for Computational Linguistics, Aug. 2009, pp. 286–295. URL: https://aclanthology.org/D09-1030.

[237] Aniket Kittur et al. “CrowdForge: crowdsourcing complex work”. In: *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*. UIST ’11. Santa Barbara, California, USA: Association for Computing Machinery,

2011, pp. 43–52. ISBN: 9781450307161. DOI: 10.1145/2047196.2047202. URL: https://doi.org/10.1145/2047196.2047202.

[238] Jennifer Wortman Vaughan. "Making Better Use of the Crowd: How Crowdsourcing Can Advance Machine Learning Research". In: *Journal of Machine Learning Research* 18.193 (2018), pp. 1–46. URL: http://jmlr.org/papers/v18/17-234.html.

[239] Abhigna B.S., Nitasha Soni, and Shilpa Dixit. "Crowdsourcing – A Step Towards Advanced Machine Learning". In: *Procedia Computer Science* 132 (2018). International Conference on Computational Intelligence and Data Science, pp. 632–642. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2018.05.062. URL: https://www.sciencedirect.com/science/article/pii/S1877050918307944.

[240] Alexander Ratner et al. "Snorkel: rapid training data creation with weak supervision". In: *Proceedings of the VLDB Endowment* 11.3 (Nov. 2017), pp. 269–282. ISSN: 2150-8097. DOI: 10.14778/3157794.3157797. URL: http://dx.doi.org/10.14778/3157794.3157797.

[241] Ye Shi, Shao-Yuan Li, and Sheng-Jun Huang. *Learning from Crowds with Sparse and Imbalanced Annotations*. 2021. arXiv: 2107.05039 [cs.LG]. URL: https://arxiv.org/abs/2107.05039.

[242] Haitao Xu et al. "Blockchain-Based Crowdsourcing Makes Training Dataset of Machine Learning No Longer Be in Short Supply". In: *Wireless Communications and Mobile Computing* 2022.1 (2022), p. 7033626.

[243] Kai Ye et al. "Enabling Social Network: An Exploration of Crowdsourcing System". In: *Proceedings of the 12th Chinese Conference on Computer Supported Cooperative Work and Social Computing*. ChineseCSCW '17. Chongqing, China: Association for Computing Machinery, 2017, pp. 249–250. ISBN: 9781450353526. DOI: 10.1145/3127404.3127460. URL: https://doi.org/10.1145/3127404.3127460.

[244] Christian Ulrik Søttrup and Jakob Gregor Pedersen. "Developing distributed computing solutions combining grid computing and public computing". In: *Disponıvel em https://uimon.cern.ch/twiki/pub/LHCAtHome/LinksAndDocs/ChristianSoettrupBOINCThesis.pdf, Setembro* (2005).

[245] Zhongli Dong, Young Choon Lee, and Albert Y. Zomaya. "Crowdware: A Framework for GPU-Based Public-Resource Computing with Energy-Aware Incentive Mechanism". In: *2015 IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom)*. 2015, pp. 266–273. DOI: 10.1109/CloudCom.2015.73.

[246] Mahmood Hosseini et al. "Crowdcloud: a crowdsourced system for cloud infrastructure". In: *Cluster Computing* 22.2 (June 2019), pp. 455–470. ISSN: 1386-7857. DOI: 10.1007/s10586-018-2843-2. URL: https://doi.org/10.1007/s10586-018-2843-2.

[247] Aniket Kittur, Ed H. Chi, and Bongwon Suh. "Crowdsourcing user studies with Mechanical Turk". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '08. Florence, Italy: Association for Computing Machinery, 2008, pp. 453–456. ISBN: 9781605580111. DOI: 10.1145/1357054.1357127. URL: https://doi.org/10.1145/1357054.1357127.

[248] Hong Xie et al. "Incentive mechanism and protocol design for crowdsourcing systems". In: *2014 52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. 2014, pp. 140–147. DOI: 10.1109/ALLERTON.2014.7028448.

[249] Nur Kaynar and Auyon Siddiq. "Estimating effects of incentive contracts in online labor platforms". In: *Management Science* 69.4 (2023), pp. 2106–2126.

[250] Jiawen Kang et al. "Incentive Mechanism for Reliable Federated Learning: A Joint Optimization Approach to Combining Reputation and Contract Theory". In: *IEEE*

Internet of Things Journal 6.6 (2019), pp. 10700–10714. DOI: 10.1109/JIOT.2019.2940820.

[251] Diwakar Gupta, Aditya Vedantam, and Justin Azadivar. “Optimal contract mechanism design for performance-based contracts”. In: (2011).

[252] Chien-Ju Ho et al. “Incentivizing High Quality Crowdwork”. In: *Proceedings of the 24th International Conference on World Wide Web*. WWW ’15. International World Wide Web Conferences Steering Committee, May 2015. DOI: 10.1145/2736277.2741102. URL: http://dx.doi.org/10.1145/2736277.2741102.

[253] Yutian Wen et al. “Quality-Driven Auction-Based Incentive Mechanism for Mobile Crowd Sensing”. In: *IEEE Transactions on Vehicular Technology* 64.9 (2015), pp. 4203–4214. DOI: 10.1109/TVT.2014.2363842.

[254] Yingjie Wang et al. “An Optimization and Auction-Based Incentive Mechanism to Maximize Social Welfare for Mobile Crowdsourcing”. In: *IEEE Transactions on Computational Social Systems* 6.3 (2019), pp. 414–429. DOI: 10.1109/TCSS.2019.2907059.

[255] Bowen Zhou, Satish Narayana Srirama, and Rajkumar Buyya. “An auction-based incentive mechanism for heterogeneous mobile clouds”. In: *Journal of Systems and Software* 152 (2019), pp. 151–164. ISSN: 0164-1212. DOI: https://doi.org/10.1016/j.jss.2019.03.003. URL: https://www.sciencedirect.com/science/article/pii/S0164121219300548.

[256] Jia Xu et al. “Online Incentive Mechanism for Mobile Crowdsourcing Based on Two-Tiered Social Crowdsourcing Architecture”. In: *2018 15th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. 2018, pp. 1–9. DOI: 10.1109/SAHCN.2018.8397102.

[257] Jiangang Pang and Zhiying Liu. “Motivation system of crowdsourcing community from a supply chain perspective”. In: *Mathematical Problems in Engineering* 2016.1 (2016), p. 8306950.

[258] Yufeng Zhan et al. “A Learning-Based Incentive Mechanism for Federated Learning”. In: *IEEE Internet of Things Journal* 7.7 (2020), pp. 6360–6368. DOI: 10.1109/JIOT.2020.2967772.

[259] Mingshu Cong et al. “A Game-Theoretic Framework for Incentive Mechanism Design in Federated Learning”. In: *Federated Learning: Privacy and Incentive*. Ed. by Qiang Yang, Lixin Fan, and Han Yu. Cham: Springer International Publishing, 2020, pp. 205–222. ISBN: 978-3-030-63076-8. DOI: 10.1007/978-3-030-63076-8_15. URL: https://doi.org/10.1007/978-3-030-63076-8_15.

[260] Rongfei Zeng et al. “Incentive Mechanisms in Federated Learning and A Game-Theoretical Approach”. In: *IEEE Network* 36.6 (2022), pp. 229–235. DOI: 10.1109/MNET.112.2100706.

[261] Rongfei Zeng et al. *A Comprehensive Survey of Incentive Mechanism for Federated Learning*. 2021. arXiv: 2106.15406 [cs.LG]. URL: https://arxiv.org/abs/2106.15406.

[262] Han Yu et al. “A fairness-aware incentive scheme for federated learning”. In: *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. 2020, pp. 393–399.

[263] Xun Yang et al. “Federated learning incentive mechanism design via enhanced shapley value method”. In: *Wireless Communications and Mobile Computing* 2022.1 (2022), p. 9690657.

[264] Zhuan Shi et al. “Fedfaim: A model performance-based fair incentive mechanism for federated learning”. In: *IEEE Transactions on Big Data* (2022).

[265] J. P. Brans and Ph. Vincke. “A Preference Ranking Organisation Method: (The PROMETHEE Method for Multiple Criteria Decision-Making)”. In: *Management Science* 31 (1985), pp. 647–656. URL: https://www.jstor.org/stable/2631441 (visited on 06/25/2021).

[266] Dan Peng, Fan Wu, and Guihai Chen. “Data Quality Guided Incentive Mechanism Design for Crowdsensing”. In: *IEEE Transactions on Mobile Computing* 17 (Feb. 2018), pp. 307–319. DOI: 10.1109/tmc.2017.2714668. (Visited on 11/07/2020).

[267] Xingqiu He et al. *A Shapley Value-Based Incentive Mechanism in Collaborative Edge Computing*. IEEE Xplore, Dec. 2021. DOI: 10.1109/GLOBECOM46510.2021.9685706. URL: https://ieeexplore.ieee.org/abstract/document/9685706 (visited on 08/25/2023).

[268] Toon Calders and Dimitri Van Assche. *PROMETHEE is Not Quadratic: An O(qn log(n)) Algorithm*. 2016. arXiv: 1603.00091 [cs.DS]. URL: https://arxiv.org/abs/1603.00091.

[269] Sebastian Caldas et al. *LEAF: A Benchmark for Federated Settings*. 2019. arXiv: 1812.01097 [cs.LG]. URL: https://arxiv.org/abs/1812.01097.

[270] Gregory Cohen et al. *EMNIST: an extension of MNIST to handwritten letters*. 2017. arXiv: 1702.05373 [cs.CV]. URL: https://arxiv.org/abs/1702.05373.

[271] Ziwei Liu et al. *Deep Learning Face Attributes in the Wild*. 2015. arXiv: 1411.7766 [cs.CV]. URL: https://arxiv.org/abs/1411.7766.

[272] Tian Li et al. *Fair Resource Allocation in Federated Learning*. 2020. arXiv: 1905.10497 [cs.LG]. URL: https://arxiv.org/abs/1905.10497.

[273] Jack Goetz et al. *Active Federated Learning*. 2019. arXiv: 1909.12641 [cs.LG].

[274] Hongyi Wang et al. *Federated Learning with Matched Averaging*. 2020. arXiv: 2002.06440 [cs.LG]. URL: https://arxiv.org/abs/2002.06440.

[275] Kang Wei et al. *Federated Learning with Differential Privacy: Algorithms and Performance Analysis*. 2019. arXiv: 1911.00222 [cs.LG]. URL: https://arxiv.org/abs/1911.00222.

[276] Yuchen Li et al. *Privacy Threats Analysis to Secure Federated Learning*. 2021. arXiv: 2106.13076 [cs.LG].

[277] Lei Shi et al. “Data poisoning attacks on federated learning by using adversarial samples”. In: *2022 International Conference on Computer Engineering and Artificial Intelligence (ICCEAI)*. 2022, pp. 158–162. DOI: 10.1109/ICCEAI55464.2022.00041.

[278] Ioan-Alexandru Dumitru. “Zero Trust Security”. In: *Proceedings of the International Conference on Cybersecurity and Cybercrime (IC3)* (2022). URL: https://api.semanticscholar.org/CorpusID:248578977.

[279] Carlisle Adams and Steve Lloyd. *Understanding PKI : concepts, standards, and deployment considerations*. Addison-Wesley, Cop, 2003.

[280] Xin Jin, Ram Krishnan, and Ravi Sandhu. “A unified attribute-based access control model covering DAC, MAC and RBAC”. In: *Data and Applications Security and Privacy XXVI: 26th Annual IFIP WG 11.3 Conference, DBSec 2012, Paris, France, July 11-13, 2012. Proceedings 26*. Springer. 2012, pp. 41–55.

[281] Maanak Gupta et al. “Reachability analysis for attributes in ABAC with group hierarchy”. In: *IEEE Transactions on Dependable and Secure Computing* 20.1 (2022), pp. 841–858.

[282] Lloyd S Shapley. “A value for n-person games”. In: *Contribution to the Theory of Games* 2 (1953).